



開發人員指南

Amazon CloudFront



Amazon CloudFront: 開發人員指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能隸屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

Amazon CloudFront 介紹	1
如何設定 CloudFront 交付內容	2
選擇標準分佈或多租用戶分佈	3
定價	4
使用 CloudFront 的方式	4
加速靜態網站內容交付	4
提供隨需視訊或即時串流視訊	5
在整個系統處理過程中加密特定欄位	5
在邊緣進行自訂	5
使用 Lambda@Edge 自訂項目提供私有內容	6
CloudFront 如何交付內容	6
CloudFront 如何將內容交付給您的使用者	6
CloudFront 使用區域節點快取的方法	7
CloudFront 邊緣伺服器	9
使用 CloudFront 受管字首清單	10
使用 AWS SDK	10
CloudFront 技術資源	11
開始使用	13
設定您的 AWS 帳戶	13
註冊 AWS 帳戶	13
建立具有管理存取權的使用者	14
選擇如何存取 CloudFront	15
開始使用標準分佈	15
先決條件	16
建立儲存貯體	16
上傳內容	17
建立分佈	17
存取內容	18
清除	19
增強基本分佈	19
開始使用 (AWS CLI)	19
先決條件	20
建立 Amazon S3 儲存貯體	20
將內容上傳至儲存貯體	20

建立原始存取控制 (OAC)	21
建立標準分佈	21
更新您的 S3 儲存貯體政策	22
確認分佈部署	23
透過 CloudFront 存取內容	24
清除	24
開始使用安全的靜態網站	26
解決方案概觀	26
部署解決方案	27
CloudFront 固定費率定價計劃	32
CloudFront 固定費率定價計劃的優點	32
依定價計劃方案的功能	33
定價計劃功能	34
每月用量限額	52
根據歷史用量的資格	53
您的計劃涵蓋的成本	53
Route 53 DNS 管理和您的計劃	53
使用定價計劃降低整體 AWS 成本	55
管理您的固定費率定價計劃	55
訂閱定價計劃的新分佈	55
訂閱現有的 分佈至定價計劃	56
升級定價計劃	56
降級定價計劃	57
取消定價計劃	57
取消待定計劃變更	57
刪除具有定價計劃的分佈	58
許可	58
固定費率定價計劃配額	58
不支援的功能	59
不支援的功能	59
不支援的關聯	61
帳戶層級限制條件	61
資源層級限制條件	61
可能影響定價計劃的其他功能	62
定價計劃與pay-as-you-go定價	63
設定分佈	64

了解多租用戶分佈的運作方式	65
運作方式	66
條款	68
不支援的功能	69
分佈租用戶自訂	70
請求憑證 (分佈租用戶)	73
建立自訂連線群組 (選用)	80
移轉至多租用戶分佈	81
建立分發	82
在主控台中建立 CloudFront 分佈	84
顯示的值	88
其他連結	89
將網域新增至您的 CloudFront 標準分佈	90
預先設定的分佈設定	92
Amazon S3 原始伺服器	92
API Gateway 原始伺服器	93
自訂原始伺服器和 EC2 執行個體	94
ELB 原始伺服器	96
MediaPackage v1 原始伺服器	98
MediaPackage v2 原始伺服器	99
MediaTailor 原始伺服器	100
所有分佈設定	101
原始設定	101
快取行為設定	111
分佈設定	123
自訂錯誤頁面和錯誤快取	131
地理限制	132
測試分佈	133
建立物件的連結	133
更新分佈	134
在主控台中更新分佈	134
標記分佈	136
標籤限制	137
新增、編輯和刪除分佈的標籤	137
程式設計標記	138
刪除 分發	138

使用各種原始伺服器	140
使用 Amazon S3 儲存貯體	141
使用 MediaStore 容器或 MediaPackage 頻道	152
使用 Application Load Balancer	153
使用 Network Load Balancer	153
使用 Lambda 函數 URL	153
使用 Amazon EC2 (或其他自訂原始伺服器)	154
使用 CloudFront 原始伺服器群組	156
使用 Amazon API Gateway	156
啟用 IPv6	156
IPv6 檢視器請求	157
IPv6 原始伺服器請求	158
使用持續部署以安全地測試變更	158
CloudFront 持續部署工作流程	160
使用臨時分佈和持續部署政策	161
監控臨時分佈	169
了解持續部署的運作方式	170
持續部署的配額和其他考量	172
使用自訂 URL	173
使用備用網域名稱的需求	173
使用備用網域名稱的限制	175
新增替代網域名稱	177
移動替代網域名稱	180
移除替代網域名稱	190
在備用網域名稱中使用萬用字元	191
使用 WebSocket	192
WebSocket 通訊協定的運作方式	192
WebSocket 要求	193
建議的 WebSocket 標頭	193
請求用於列入允許清單的 Anycast 靜態 IP	194
先決條件	194
請求 Anycast 靜態 IP 清單	194
建立 Anycast 靜態 IP 清單	195
將 Anycast 靜態 IP 清單與現有分佈建立關聯	195
將 Anycast 靜態 IP 清單與新分佈建立關聯	196
將 Anycast 靜態 IP 清單與連線群組建立關聯	197

更新 Anycast 靜態 IP 清單	197
使用 IPAM 在 CloudFront 使用自有 IP	198
使用 gRPC	201
gRPC 如何在 CloudFront 中運作	202
在 CloudFront 中使用共用資源	204
共用資源的先決條件	204
共用 VPC 原始伺服器	205
使用共用 VPC 原始伺服器	207
識別共用 VPC 原始伺服器	208
取消共用共用的 VPC 原始伺服器	208
共用 VPC 原始伺服器的責任和許可	209
擁有者的許可	209
消費者的許可	209
AWSRAMDefaultPermissionCloudFront	209
計費和計量	209
共用資源配額	210
快取和可用性	211
提升您的快取命中率	211
指定 CloudFront 快取物件的時間長短	212
使用 Origin Shield	212
根據查詢字串參數快取	212
根據 Cookie 值快取	213
根據請求標頭快取	214
不需要壓縮時，移除 Accept-Encoding 標頭	214
透過 HTTP 提供媒體內容	215
使用 Origin Shield	215
Origin Shield 的使用案例	216
選擇 Origin Shield AWS 的區域	220
啟用 Origin Shield	222
預估 Origin Shield 成本	224
Origin Shield 高可用性	225
Origin Shield 與其他 CloudFront 功能如何互動	225
使用原始伺服器容錯移轉增加可用性	226
建立原始伺服器群組	227
控制原始伺服器逾時和嘗試次數	228
使用原始伺服器容錯移轉與 Lambda@Edge 函數搭配	229

搭配原始伺服器容錯移轉使用自訂錯誤頁面	230
管理快取過期	231
使用標頭來控制個別物件的快取持續時間	232
提供過時 (過期) 內容	233
指定 CloudFront 快取物件的時間	235
使用 Amazon S3 主控台新增標頭到物件	240
快取和查詢字串參數	241
查詢字串轉送和快取的主控台和 API 設定	243
最佳化快取	243
查詢字串參數和 CloudFront 標準記錄檔 (存取日誌)	244
根據 Cookie 快取內容	244
根據請求標頭快取內容	247
標頭和分佈 - 概觀	248
選取快取時所依據的標頭	249
設定 CloudFront 以遵循 CORS 設定	250
根據裝置類型設定快取	250
根據檢視器語言設定快取	251
根據檢視器位置設定快取	251
根據請求的通訊協定設定快取	251
設定壓縮檔案的快取	251
快取如何根據標頭影響效能	251
標頭大小寫和標頭值如何影響快取	251
CloudFront 傳回給檢視器的標頭	252
使用政策控制快取金鑰	253
了解快取政策	254
政策資訊	254
存活期 (TTL) 設定	254
快取金鑰設定	255
建立快取政策	260
使用受管快取政策	263
Amplify	264
CachingDisabled	265
CachingOptimized	266
CachingOptimizedForUncompressedObjects	267
Elemental-MediaPackage	267
UseOriginCacheControlHeaders	268

UseOriginCacheControlHeaders-QueryStrings	269
了解快取金鑰	270
預設快取金鑰	270
自訂快取金鑰	272
使用政策控制原始伺服器請求	274
了解原始伺服器請求政策	275
政策資訊	275
原始伺服器請求設定	275
建立原始伺服器請求政策	277
使用受管原始伺服器請求政策	281
AllViewer	281
AllViewerAndCloudFrontHeaders-2022-06	282
AllViewerExceptHostHeader	283
CORS-CustomOrigin	284
CORS-S3Origin	284
Elemental-MediaTailor-PersonalizedManifests	285
HostHeaderOnly	285
UserAgentRefererHeaders	286
新增 CloudFront 請求標頭	286
裝置類型標頭	287
檢視器位置標頭	287
用於判斷檢視器標頭結構的標頭	289
TLS 相關標頭	289
其他 CloudFront 標題	290
了解原始伺服器請求政策和快取政策如何協同運作	290
使用政策新增或移除回應標頭	293
了解回應標頭政策	294
政策詳細資訊 (中繼資料)	294
CORS 標頭	294
安全性標頭	298
自訂標頭	300
移除標頭	300
Server-Timing 標頭	302
建立回應標頭政策	306
使用受管回應標頭政策	312
CORS-and-SecurityHeadersPolicy	312

CORS-With-Preflight	313
CORS-with-preflight-and-SecurityHeadersPolicy	314
SecurityHeadersPolicy	315
SimpleCORS	316
請求和回應行為	317
CloudFront 如何處理 HTTP 和 HTTPS 的請求	317
Amazon S3 原始伺服器之請求和回應行為	318
CloudFront 如何處理和轉送請求至您的 Amazon S3 原始伺服器	318
CloudFront 如何處理來自您 Amazon S3 原始伺服器的回應	324
為自訂原始伺服器之請求和回應行為	326
CloudFront 如何處理和轉送請求至您的自訂原始伺服器	326
CloudFront 如何處理來自您的自訂原始伺服器的回應	341
原始伺服器群組的請求和回應行為	345
將自訂標頭新增到原始伺服器請求	346
使用案例	346
設定 CloudFront 以將自訂標頭新增到原始伺服器請求	347
CloudFront 無法新增到原始伺服器請求的自訂標頭	347
設定 CloudFront 以轉送 Authorization 標頭	348
CloudFront 如何處理範圍 GET	349
使用範圍請求快取大物件	350
CloudFront 如何處理來自原始伺服器的 HTTP 3xx 狀態碼。	350
CloudFront 如何處理來自原始伺服器的 HTTP 4xx 和 5xx 狀態碼	351
當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤	352
您尚未設定自訂錯誤頁面時 CloudFront 如何處理錯誤	354
CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼	355
產生自訂錯誤回應	357
設定錯誤回應行為	358
針對特定的 HTTP 狀態碼建立自訂錯誤頁面	359
將物件和自訂錯誤頁面存放在不同位置	361
變更 CloudFront 傳回的回應碼	361
控制 CloudFront 快取錯誤的時間長度	362
新增、移除或取代內容	364
新增和存取內容	364
使用檔案版本控制更新或移除現有內容	364
使用版本控制的檔案名稱更新現有檔案	365
移除內容，使 CloudFront 不會對其進行分配	365

自訂檔案 URL	365
使用您自己的網域名稱 (example.com)	366
在 URL 中使用結尾斜線 (/)	366
建立適用於受限制內容的簽署 URL	367
指定預設根物件	367
如何指定預設根物件	367
預設根物件的運作方式	369
未定義根物件的情況下 CloudFront 的運作方式	370
使檔案失效以移除內容	371
選擇要使檔案失效或者使用版本控制的檔案名稱	371
確定要失效的檔案	372
讓檔案失效時的注意事項	372
使檔案失效	376
並行失效請求上限	379
支付檔案失效的費用	379
提供壓縮檔案	380
設定 CloudFront 來壓縮物件	380
CloudFront 壓縮如何運作	381
壓縮條件	382
CloudFront 壓縮的檔案類型	383
ETag 標頭轉換	385
使用 AWS WAF 保護	386
為分佈啟用 AWS WAF	387
為新分佈啟用 AWS WAF	387
使用現有的 Web ACL	388
啟用機器人控制功能	388
依機器人類別設定保護	389
管理 CloudFront 的 AWS WAF 安全性保護	390
必要條件	390
啟用 AWS WAF 日誌	391
設定速率限制	391
停用 AWS WAF 安全保護	392
設定安全存取和限制對內容的存取	394
透過 CloudFront 使用 HTTPS	395
檢視器和 CloudFront 之間需使用 HTTPS	396
需要 HTTPS 到自訂原始伺服器	398

需要 HTTPS 與 Amazon S3 原始伺服器通訊	400
檢視器和 CloudFront 之間支援的通訊協定和密碼	402
CloudFront 和原始伺服器之間支援的通訊協定和密碼	410
使用備用網域名稱和 HTTPS	413
選擇 CloudFront 提供 HTTPS 請求的方式	413
與 CloudFront 搭配使用 SSL/TLS 憑證的要求	416
搭配 CloudFront 使用 SSL/TLS 憑證的配額 (僅限檢視器與 CloudFront 之間的 HTTPS)	421
設定備用網域名稱和 HTTPS	422
判斷 SSL/TLS RSA 憑證中公有金鑰的大小	426
增加 SSL/TLS 憑證的配額	426
輪換 SSL/TLS 憑證	427
從自訂 SSL/TLS 憑證還原到預設 CloudFront 憑證	428
若要從具有專用 IP 位址的自訂 SSL/TLS 憑證切換到 SNI	429
相互 TLS (mTLS) 檢視器	430
運作方式	430
使用案例	430
信任存放區和憑證管理	431
為 CloudFront 分佈啟用交互 TLS	437
關聯 CloudFront 連線函數	441
設定其他設定	446
快取政策並轉送至原始伺服器的檢視器 mTLS 標頭	448
使用 CloudFront Connection Function 和 KVS 撤銷	451
使用連線日誌的可觀測性	455
限制已簽署 URL 和已簽署 Cookie 的內容	460
如何提供私有內容	460
限制檔案存取	461
指定可信簽署者	463
決定使用已簽署 URL 或已簽署 Cookie	471
使用已簽署 URL	472
使用已簽署 Cookie	491
用於 Base64 編碼和加密的 Linux 命令和 OpenSSL	516
已簽署 URL 程式碼範例	517
限制對 AWS 原始伺服器的存取	545
限制對 AWS Elemental MediaPackage v2 原始伺服器的存取	546
限制對 AWS Elemental MediaStore 原始伺服器的存取	552
限制對 AWS Lambda 函數 URL 原始伺服器的存取	559

限制對 Amazon S3 原始伺服器的存取	568
使用 VPC 原始伺服器限制存取	582
限制對 Application Load Balancers 的存取	589
設定 CloudFront 將自訂 HTTP 標頭新增至請求	590
將 Application Load Balancer 設定為僅轉寄包含特定標頭的請求	592
(選用) 改善此解決方案的安全性	593
(選用) 使用 CloudFront AWS 的受管字首清單限制對原始伺服器的存取	594
地理限制	594
使用 CloudFront 地理限制	594
使用第三方地理位置服務。	596
使用欄位層級加密來協助保護敏感資料	597
欄位層級加密概觀	599
設定欄位層級加密	599
在您的原始伺服器解密資料欄位	605
隨需視訊和即時串流視訊	608
關於串流視訊	608
傳遞隨需視訊	609
為 Microsoft Smooth Streaming 設定隨需視訊	609
傳遞視訊串流	611
使用 AWS Elemental MediaStore 做為原始伺服器來提供影片	612
提供以 AWS Elemental MediaPackage 格式化的即時視訊	613
使用 AWS Elemental MediaPackage 提供隨選視訊內容	618
媒體品質感知彈性	623
MQAR 日誌欄位	625
使用函數在邊緣自訂	626
CloudFront Functions 與 Lambda@Edge 之間的差異	626
使用 CloudFront Functions 進行自訂	629
教學課程：建立簡單的 CloudFront 函數	630
教學課程：建立使用鍵值的 CloudFront 函數	633
撰寫函數程式碼	635
建立函數	726
測試函數	729
更新函數	734
發佈函數	736
將函數與分佈相關聯	738
CloudFront KeyValueStore	741

使用 CloudFront Connection Functions 自訂	760
概觀和工作流程	761
組態和限制	762
建立 CloudFront 連線函數以進行交互 TLS (檢視器) 驗證	763
寫入 CloudFront 連線函數程式碼以進行交互 TLS (檢視器) 驗證	766
在部署之前測試 CloudFront 連線函數	774
將連線函數與分佈建立關聯	775
使用 CloudFront Functions 和 KeyValueCollection 實作交互 TLS (檢視器) 的憑證撤銷	776
使用 Lambda@Edge 進行自訂	783
Lambda@Edge 如何處理請求和回應	784
使用 Lambda@Edge 的方式	784
開始使用 Lambda@Edge	785
設定 IAM 許可權限和角色	792
撰寫及建立 Lambda@Edge 函數	798
為 Lambda@Edge 函數新增觸發條件	802
測試和偵錯	808
刪除函數與複本	815
事件結構	816
使用請求和回應	832
函數範例	837
對邊緣函數的限制	875
對所有邊緣函數的限制	875
對 CloudFront Functions 的限制	880
對 Lambda@Edge 的限制	882
報告、指標和日誌	887
AWS CloudFront 的帳單和用量報告	887
檢視 CloudFront 的 AWS 帳單報告	888
檢視 CloudFront 的 AWS 用量報告	889
解譯 CloudFront 的 AWS 帳單和用量報告	890
檢視 CloudFront 主控台報告	896
檢視 CloudFront 快取統計資料報告	896
檢視 CloudFront 熱門物件報告	902
檢視 CloudFront 最佳推薦網站報告	907
檢視 CloudFront 用量報告	910
檢視 CloudFront 檢視器報告	917
使用 Amazon CloudWatch 監控 CloudFront 指標	927

檢視 CloudFront 和邊緣函數指標	928
建立 警示	934
下載指標資料	935
CloudFront 指標	938
CloudFront 和邊緣函數記錄	943
記錄請求	944
記錄邊緣函數	945
記錄服務活動	945
存取日誌 (標準日誌)	945
使用即時存取日誌	987
邊緣函數日誌	1006
AWS CloudTrail 日誌	1010
使用 追蹤組態變更 AWS Config	1022
AWS Config 使用 CloudFront 設定	1023
檢視 CloudFront 組態歷史記錄	1024
使用 AWS Config 規則評估 CloudFront 組態	1024
安全	1025
資料保護	1025
傳輸中加密	1027
靜態加密	1027
限制存取內容	1027
身分和存取權管理	1028
目標對象	1029
使用身分驗證	1029
使用政策管理存取權	1030
Amazon CloudFront 如何與 IAM 搭配運作	1031
身分型政策範例	1036
AWS 受管政策	1046
使用服務連結角色	1053
對 CloudFront 身分與存取進行疑難排解	1057
日誌記錄和監控	1058
法規遵循驗證	1059
CloudFront 合規最佳實務	1060
恢復能力	1061
CloudFront 原始伺服器容錯移轉	1061
基礎架構安全	1061

疑難排解	1063
故障診斷分佈問題	1063
CloudFront 傳回 Access Denied 錯誤	1063
當我嘗試新增替代網域名稱時，CloudFront 傳回 InvalidViewerCertificate 錯誤	1065
當我嘗試新增 CNAME 時，CloudFront 會傳回 DNS 記錄設定不正確的錯誤	1066
我無法在我的分佈中檢視檔案	1067
錯誤訊息：Certificate: <certificate-id> is being used by CloudFront	1068
對錯誤回應狀態碼進行疑難排解	1068
HTTP 400 狀態碼 (錯誤的請求)	1069
HTTP 401 狀態碼 (未授權)	1070
HTTP 403 狀態碼 (方法無效)	1071
HTTP 403 狀態碼 (許可遭拒)	1071
HTTP 404 狀態碼 (找不到)	1073
HTTP 412 狀態碼 (先決條件失敗)	1074
HTTP 500 狀態碼 (內部伺服器錯誤)	1074
HTTP 502 狀態碼 (無效的閘道)	1075
HTTP 503 狀態碼 (服務無法使用)	1079
HTTP 504 狀態碼 (閘道逾時)	1081
負載測試 CloudFront	1085
配額	1087
一般配額	1088
分佈的一般配額	1088
政策的一般配額	1091
mTLS 和信任存放區的配額	1092
CloudFront Functions 上的配額	1093
連線函數的配額	1093
鍵值存放區的配額	1094
Lambda@Edge 的配額	1094
SSL 憑證的配額	1096
失效的配額	1097
金鑰群組的配額	1097
WebSocket 連線的配額	1098
欄位層級加密的配額	1098
Cookie 的配額 (舊版快取設定)	1099
查詢字串的配額 (舊版快取設定)	1099
標頭的配額	1099

多租用戶配額	1100
相關資訊	1101
程式碼範例	1102
基本概念	1103
動作	1104
案例	1170
建立多租用戶分佈和分佈租用戶	1170
刪除簽署資源	1181
開始使用 CloudFront	1184
簽署 URL 和 Cookie	1192
CloudFront Functions 範例	1195
新增 HTTP 安全標頭	1196
新增 CORS 標頭	1197
新增快取控制標頭	1198
新增真正的用戶端 IP 標頭	1199
新增原始伺服器標頭	1199
將 index.html 新增至請求 URL	1200
標準化查詢字串參數	1201
重新導向到新的 URL	1202
重寫請求 URI	1204
選取更接近檢視器的原始伺服器	1206
使用鍵值對	1208
驗證簡易字符	1209
文件歷史紀錄	1213
.....	mccxxxiv

Amazon CloudFront 介紹

Amazon CloudFront 是一種 Web 服務，可將 .html、.css、.js 和影像檔案等靜態與動態內容加速發佈給使用者。CloudFront 透過透過稱為節點的資料中心全球網路交付您的內容。當使用者請求您使用 CloudFront 提供的內容時，請求會被路由到可提供最低延遲 (時間延遲) 的節點，以便能以最佳的效能發佈內容。

- 如果最低延遲的節點中有該內容，CloudFront 就會立即交付該內容。
- 如果該節點中沒有此內容，CloudFront 就會從您已定義的原始伺服器中擷取，這些伺服器可能是您識別為內容最終版本來源的 Amazon S3 儲存貯體、MediaPackage 頻道或 HTTP 伺服器 (例如 Web 伺服器)。

例如，假設您是從傳統的 Web 伺服器提供影像，而非從 CloudFront。例如，您可以使用 URL `https://example.com/sunsetphoto.png` 來提供影像 `sunsetphoto.png`。

您的使用者可輕鬆瀏覽至此 URL 並看到影像。但是他們可能不知道他們的請求是從一個網路路由到另一個網路，透過組成網際網路的複雜互連網路集合，直到找到映像為止。

CloudFront 透過 AWS 骨幹網路將每個使用者請求路由到能以最佳方式提供內容的節點，以加速發佈內容。一般而言，這是能最快將內容交付給瀏覽者的 CloudFront 節點伺服器。使用 AWS 網路大幅減少了使用者請求必須通過的網路數量，從而提升了效能。使用者可以獲得更低的延遲 - 載入檔案的第一個位元所需的時間 - 以及更高的資料傳輸速率。

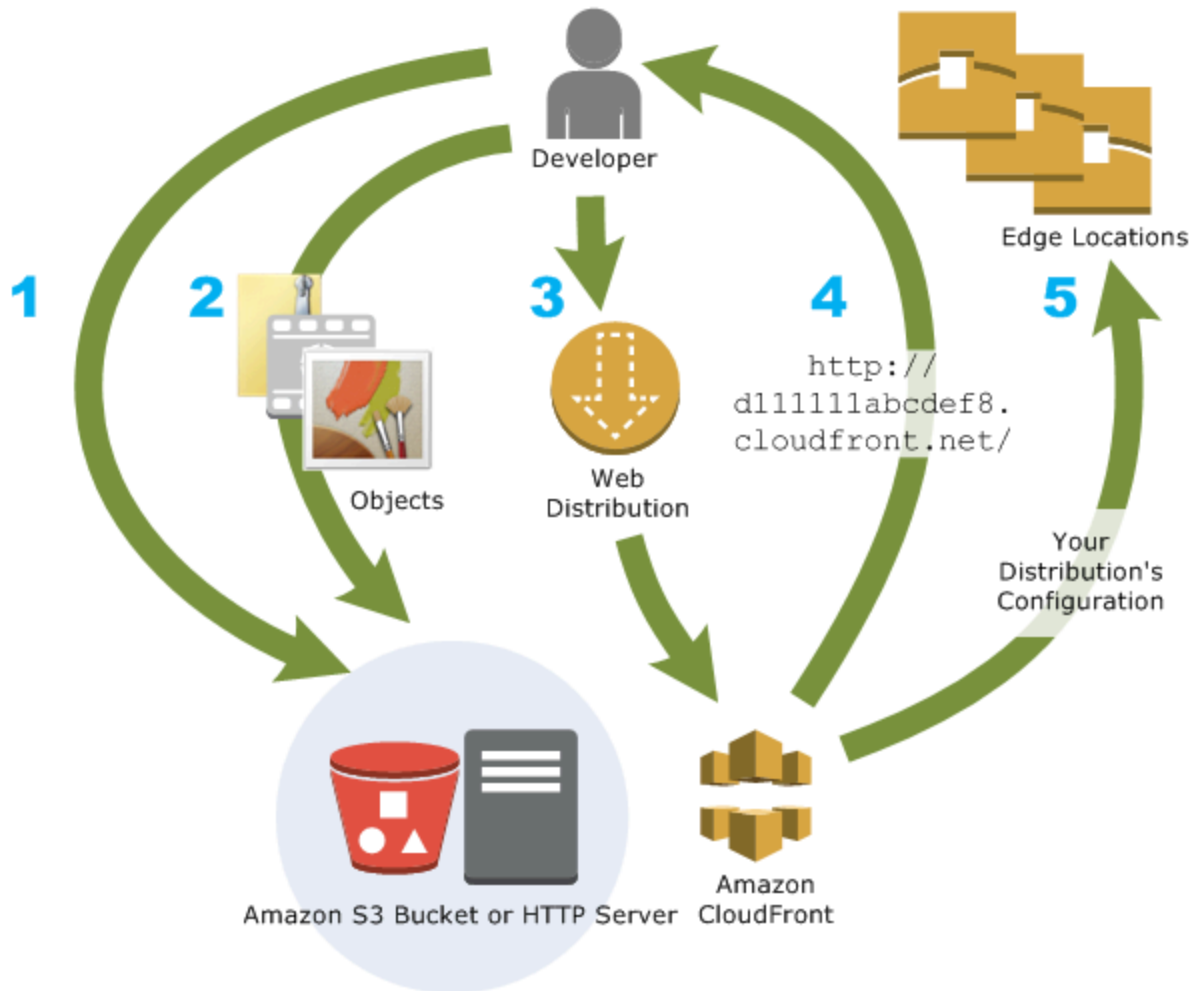
您也能獲得更高的可靠性和可用性，因為檔案 (也稱為物件) 的副本現在保留 (或快取) 在世界各地的多個節點。

主題

- [如何設定 CloudFront 交付內容](#)
- [選擇標準分佈或多租用戶分佈](#)
- [定價](#)
- [使用 CloudFront 的方式](#)
- [CloudFront 如何交付內容](#)
- [CloudFront 邊緣伺服器的位置和 IP 位址範圍](#)
- [以 AWS SDK 使用 CloudFront](#)
- [CloudFront 技術資源](#)

如何設定 CloudFront 交付內容

您建立 CloudFront 發佈告知 CloudFront 交付的內容出自何處，以及如何追蹤及管理內容交付的詳細資訊。然後 CloudFront 使用接近瀏覽者的電腦 (節點伺服器)，在某人想要查看或使用內容時，快速交付該內容。



如何設定 CloudFront 交付內容

1. 您可以指定「原始伺服器」，像是 Amazon S3 儲存貯體或您自己的 HTTP 伺服器，CloudFront 會取得其中的檔案，再從 CloudFront 節點發佈到世界各地。

原始伺服器儲存物件的原始、最終版本。如果您是透過 HTTP 提供內容，則您的原始伺服器會是 Amazon S3 儲存貯體或 HTTP 伺服器，例如 Web 伺服器。您的 HTTP 伺服器可以在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或您管理的伺服器上執行，這些伺服器也稱為「自訂的原始伺服器」。

2. 您將檔案上傳到原始伺服器。您的檔案 (也稱為物件) 通常包含網頁、影像和媒體檔案，但也可以是能透過 HTTP 提供的任何內容。

如果您使用 Amazon S3 儲存貯體做為原始伺服器，可以將儲存貯體中的物件變成可公開讀取，讓所有知道物件 CloudFront URL 的人都能存取物件。您還可以選擇保持物件的隱私以及控制誰能對其作存取。請參閱[使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。

3. 當使用者透過您的網站或應用程式請求檔案時，您建立的 CloudFront 「發佈」會告訴 CloudFront 要從哪些原始伺服器取得您的檔案。同時，您可以指定詳細資訊，例如是否希望 CloudFront 記錄所有請求，以及是否希望一建立發佈即啟用。
4. CloudFront 會將網域名稱指派給新發佈，此發佈可在 CloudFront 主控台中查看，或在對程式設計請求 (例如 API 請求) 的回應中傳回。如果想要，您可以改為新增要使用的替代網域名稱。
5. CloudFront 會將您的發佈組態 (不是內容) 傳送至其所有「節點」或「連接點」(POP)，這些是分散在各地之資料中心的伺服器集合，CloudFront 會在這裡快取檔案的副本。

開發網站或應用程式時，您可以使用 CloudFront 提供給 URL 的網域名稱。例如，如果 CloudFront 傳回 `d111111abcdef8.cloudfront.net` 做為發佈的網域名稱，則在 Amazon S3 儲存貯體 (或 HTTP 伺服器根目錄) 中的 `logo.jpg` URL 就會是 `https://d111111abcdef8.cloudfront.net/logo.jpg`。

或者，您可以設定 CloudFront 使用您自己的發佈網域名稱。在這種情況下，URL 可能是 `https://www.example.com/logo.jpg`。

或者，您可以設定原始伺服器在檔案中新增標頭，指出您希望檔案在 CloudFront 節點的快取中停留多長時間。在預設情況下，每個檔案在到期前的 24 小時內保持在節點。最短到期時間是 0 秒；沒有最大到期時間。如需更多詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

選擇標準分佈或多租用戶分佈

CloudFront 為單一網站或應用程式以及多租用戶情境提供分佈選項。

標準分佈

專為每個網站或應用程式的獨特組態而設計。請在下列使用案例中選擇此選項：

- 您需要單獨的 CloudFront 分佈
- 每個網站或應用程式都需要自己的自訂設定

大多數人會從標準分佈開始使用。

多租用戶分佈和分佈租用戶 (CloudFront SaaS Manager)

專為 SaaS 供應商和多租用戶情境而設計。請在下列使用案例中選擇此選項：

- 您正在建置一個可服務多個客戶網站或應用程式的 SaaS 平台
- 您需要有效率地管理多個類似的分佈
- 您需要集中控制共用的組態

如需更多詳細資訊，請參閱 [了解多租用戶分佈的運作方式](#)。

定價

CloudFront 會收取從其邊緣節點傳出的資料傳輸費用，以及 HTTP 或 HTTPS 請求的費用。定價會因使用類型、地理區域和功能選擇而有所不同。

使用 Amazon Simple Storage Service (Amazon S3)、Elastic Load Balancing 或 Amazon API Gateway 等 AWS 原始伺服器時，一律可免費將資料從原始伺服器傳輸至 CloudFront。在使用 AWS 原始伺服器時，只有從 CloudFront 傳出資料傳輸到檢視器時才會向您收費。

如需詳細資訊，請參閱 [CloudFront 定價](#) 和 Billing and Savings Bundle [常見問題](#)。

使用 CloudFront 的方式

使用 CloudFront 可協助您達成各種目標。本節僅摘列其中幾項，另提供詳細資訊的連結以讓您了解更多可能性。

主題

- [加速靜態網站內容交付](#)
- [提供隨需視訊或即時串流視訊](#)
- [在整個系統處理過程中加密特定欄位](#)
- [在邊緣進行自訂](#)
- [使用 Lambda@Edge 自訂項目提供私有內容](#)

加速靜態網站內容交付

CloudFront 可加速將靜態內容 (如影像、樣式表、JavaScript 等) 交付予全球各地的瀏覽者。您可以使用 CloudFront，利用 AWS 骨幹網路和 CloudFront 節點伺服器的優勢，在檢視者瀏覽您的網站時為其提供快速又安全可靠的體驗。

簡單存放及交付靜態內容的方法是使用 Amazon S3 儲存貯體。使用 S3 結合 CloudFront 有許多優點，包括可選擇使用[原始存取控制](#)，輕鬆限制對您 Amazon S3 內容的存取。

如需有關使用 Amazon S3 搭配 CloudFront 的詳細資訊，包括可協助您快速上手的 CloudFormation 範本，請參閱[開始使用安全的靜態網站](#)。

提供隨需視訊或即時串流視訊

CloudFront 為全球各地的瀏覽者提供多種媒體串流選項，包括預錄檔案和直播活動。

- 對於隨選影片 (VOD) 串流，您可使用 CloudFront 以常見的格式，如 MPEG DASH、Apple HLS、Microsoft Smooth Streaming 和 CMAF 等，將內容串流至任何裝置。
- 對於廣播即時串流，您可在節點快取媒體片段，以便將按正確順序傳輸各片段的資訊清單檔案的多個請求相組合，從而減輕原始伺服器的負擔。

如需如何使用 CloudFront 交付串流內容的詳細資訊，請參閱[使用 CloudFront 提供隨需視訊和即時串流視訊](#)。

在整個系統處理過程中加密特定欄位

設定 HTTPS 搭配 CloudFront 時，即已獲得原始伺服器的安全端對端連線。若增設欄位層級加密，您便可在整個系統處理過程中保護特定資料並實施 HTTPS 安全性，從而只有原始伺服器端的某些應用程式才能看到該資料。

若要設定欄位層級加密，請在 CloudFront 中新增公有金鑰，然後指定您想要使用該金鑰加密的一組欄位。如需詳細資訊，請參閱[使用欄位層級加密來協助保護敏感資料](#)。

在邊緣進行自訂

透過在節點執行無伺服器程式碼，開啟了讓您為瀏覽者自訂內容與體驗的諸多可能性，同時還可減少延遲。例如，您可在原始伺服器停機進行維護時傳回自訂錯誤訊息，以免瀏覽者看到的是一般 HTTP 錯誤訊息。或者，您可以在 CloudFront 將請求轉送到原始伺服器之前，使用函數協助授權使用者並控制對內容的存取。

使用 Lambda @Edge 搭配 CloudFront 可讓您使用多種方式來自訂 CloudFront 交付的內容。若要進一步了解 Lambda@Edge 以及如何使用 CloudFront 建立與部署函數，請參閱[使用 Lambda@Edge 在邊緣自訂](#)。如欲查看可供您針對自身解決方案進行自訂的若干程式碼範例，請參閱[Lambda@Edge 範例函數](#)。

使用 Lambda@Edge 自訂項目提供私有內容

除了使用已簽章 URL 或已簽章 Cookie，使用 Lambda@Edge 也可協助您設定 CloudFront 分佈從您的自訂原始伺服器提供私有內容。

若要使用 CloudFront 提供私有內容，請執行以下操作：

- 要求使用者 (瀏覽者) 使用 [已簽章的 URL 或已簽章的 Cookie](#) 來存取內容。
- 限制針對您原始伺服器的存取，使該伺服器僅可透過 CloudFront 面對原始伺服器使用。您可通過下列其中之一的方式執行此操作：
 - 對於 Amazon S3 原始伺服器，您可[使用原始存取控制 \(OAC\)](#)。
 - 對於自訂的原始伺服器，您可以執行下列操作：
 - 如果自訂原始伺服器受到 Amazon VPC 安全群組或 AWS Firewall Manager 保護，則您可[使用 CloudFront 受管字首清單](#)，僅允許 CloudFront 面對原始伺服器 IP 位址向您原始伺服器的傳入流量。
 - 使用自訂 HTTP 標頭來將存取限制為僅供 CloudFront 的請求使用。如需詳細資訊，請參閱[the section called “在自訂原始伺服器上限制存取檔案”](#)及 [the section called “將自訂標頭新增到原始伺服器請求”](#)。如需使用自訂標頭來限制 Application Load Balancer 原始伺服器的存取，請參閱[the section called “限制對 Application Load Balancers 的存取”](#)。
 - 如果自訂原始伺服器需要自訂的存取控制邏輯，您可以使用 Lambda@Edge 來實作該邏輯，如此部落格文章中所述：[使用 Amazon CloudFront 與 Lambda@Edge 提供私有內容](#)。

CloudFront 如何交付內容

經過一些初步設定後，CloudFront 會與您的網站或應用程式合作，加速交付您的內容。本節說明 CloudFront 如何在收到瀏覽者請求後向其提供內容。

主題

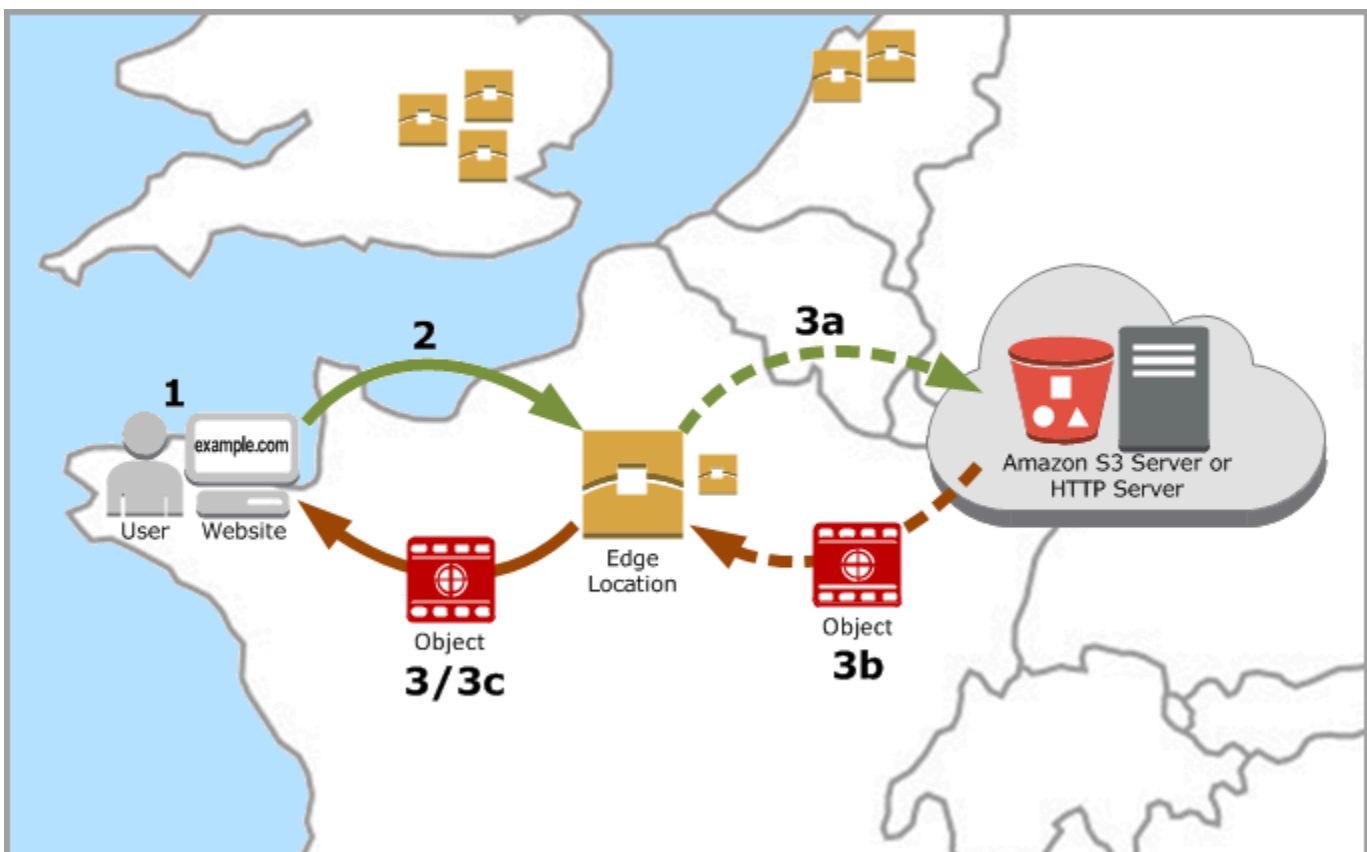
- [CloudFront 如何將內容交付給您的使用者](#)
- [CloudFront 使用區域節點快取的方法](#)

CloudFront 如何將內容交付給您的使用者

在您設定 CloudFront 以交付內容之後，以下是使用者請求物件時會發生的情況：

1. 使用者存取您的網站或應用程式並傳送一或多個物件的請求，如影像檔案或 HTML 檔案。

2. DNS 將請求路由到最適合請求的 CloudFront POP (邊緣節點)，通常會為了延遲選擇距離最近的 CloudFront POP。
3. CloudFront 會檢查快取中是否有所請求的物件。如果物件在快取中，CloudFront 會將物件傳回給使用者。如果物件不在快取中，CloudFront 會執行下列作業：
 - a. CloudFront 會比較請求與分佈中的規格，將請求轉送至原始伺服器以取得對應的物件類型，例如，轉送至 Amazon S3 儲存貯體或 HTTP 伺服器。
 - b. 原始伺服器會將物件傳送回節點。
 - c. 來自原始伺服器的第一個位元組一抵達，CloudFront 就會開始向使用者轉送物件。下次有人請求物件時，CloudFront 還會將物件新增到快取中。



CloudFront 使用區域節點快取的方法

CloudFront 連接點 (也稱為 POP 或節點) 確保能將熱門內容快速提供給瀏覽者。即使內容的熱門度不足以留在 POP，CloudFront 也有可將更多內容帶給瀏覽者的「區域節點快取」，有助於提升該內容的效能。

區域節點快取有助於處理所有內容類型，特別是隨著時間推移漸漸不熱門的內容。範例包含使用者產生的內容，例如影片、照片或圖案；電子商務資產，例如產品照片和影片；以及可能因新聞和事件而知名度突然暴增的相關內容。

區域快取的工作方式

區域節點快取是部署在全球各地接近瀏覽者的 CloudFront 位置。位於原始伺服器和 POP (直接向瀏覽者提供內容的全球節點) 之間。隨著物件變得較不熱門，個別 POP 可能會移除這些物件，為更熱門的內容騰出空間。區域節點快取擁有的快取比個別 POP 大，因此物件在最近的區域節點快取位置保留在快取中的時間會比較久。這有助於將更多內容放在距離瀏覽者更近的地方，降低 CloudFront 返回原始伺服器的需求，並提高整體的瀏覽者效能。

當瀏覽者在您的網站上或透過您的應用程式發出請求時，DNS 會將請求路由到最能滿足使用者請求的 POP。此位置通常是與延遲有關的最近 CloudFront 節點。在 POP 中，CloudFront 會檢查快取中是否有所請求的物件。如果物件在快取中，CloudFront 會將物件傳回給使用者。如果物件不在快取中，POP 通常將移至最近的區域節點快取，以擷取物件。如需 POP 何時略過區域節點快取，並直接進入原始伺服器的相關資訊，請參閱以下備註。

在區域節點快取位置中，CloudFront 會再次檢查其快取中是否有請求的物件。如果檔案在快取中，CloudFront 會將其轉送至提出請求的 POP。來自區域節點快取位置的第一個位元組一抵達，CloudFront 就會開始向使用者轉送物件。下次有人請求物件時，CloudFront 還會將物件新增到 POP 的快取中。

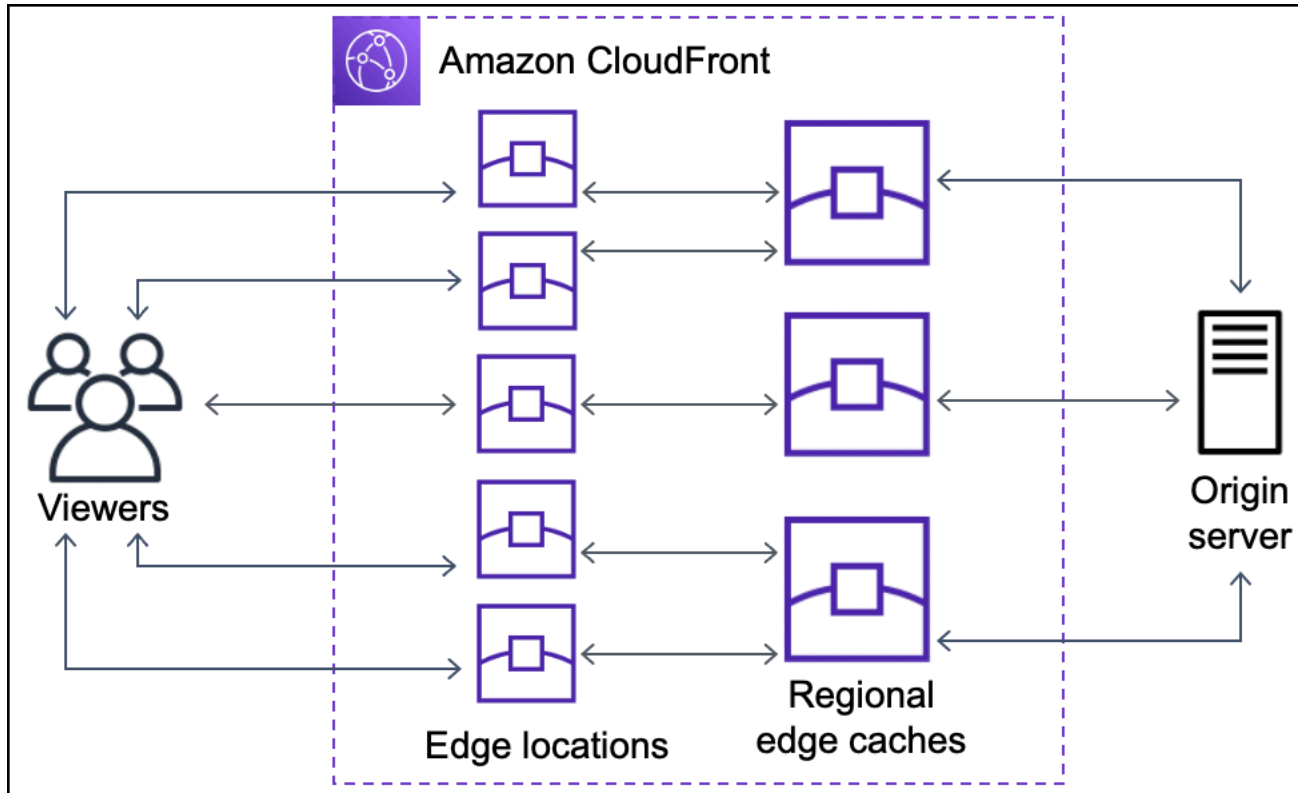
至於未在 POP 和區域節點快取位置中進行快取的物件，CloudFront 會比較請求和分佈規格，並將請求轉送到原始伺服器。在原始伺服器將物件傳送回區域節點快取位置後，會將其轉送到 POP，由 CloudFront 轉送給使用者。在這種情況下，CloudFront 除了將物件新增到 POP 之外，還會新增到區域節點快取位置的快取，以供瀏覽者下次請求物件時擷取。如此可確保某個區域中的所有 POP 都共用一份本機快取，不會向原始伺服器提出多個請求。CloudFront 也會與原始伺服器保持永久連線，以便儘快從原始伺服器擷取物件。

Note

- 區域節點快取具有與 POP 相同的功能。例如，快取失效請求在到期之前從 POP 快取和區域節點快取中移除物件。下次瀏覽者請求該物件時，CloudFront 會回到原始伺服器擷取物件的最新版本。
- Proxy HTTP 方法 (PUT、POST、PATCH、OPTIONS 和 DELETE) 直接從 POP 進入原始伺服器，而且不透過區域節點快取進行代理。
- 在請求時決定的動態請求不會流經區域邊緣快取，而是直接移至原始伺服器。

- 當原始伺服器為 Amazon S3 儲存貯體，而請求的最佳區域節點快取與 S3 儲存貯體位於相同 AWS 區域時，POP 會略過區域節點快取並直接進入 S3 儲存貯體中。

下列圖表說明請求和回應經 CloudFront 節點和區域節點快取的處理方式。



CloudFront 邊緣伺服器的位置和 IP 位址範圍

如需 CloudFront 邊緣伺服器位置的清單，請參閱 [Amazon CloudFront 全球邊緣網路](#) 頁面。

Amazon Web Services (AWS) 會以 JSON 格式發佈目前的 IP 位址範圍。若要檢視目前範圍，請下載 [ip-ranges.json](#)。如需詳細資訊，請參閱 Amazon Web Services 一般參考中的 [AWS IP 位址範圍](#)。

若要尋找與 CloudFront 邊緣伺服器相關聯的 IP 位址範圍，請在 ip-ranges.json 中搜尋下列字串：

```
"region": "GLOBAL",  
"service": "CLOUDFRONT"
```

或者，您可以在 <https://d7uri8nf7uskq.cloudfront.net/tools/list-cloudfront-ips> 僅檢視 CloudFront IP 範圍。

使用 CloudFront 受管字首清單

CloudFront 受管字首清單包含所有 CloudFront 全球面向原始伺服器的伺服器 IP 位址範圍。如果您的原始伺服器是在 AWS 上託管且受到 Amazon VPC [安全群組](#) 的保護，便可使用 CloudFront 受管字首清單將傳入您原始伺服器的流量來源設為僅允許 CloudFront 面向原始伺服器的伺服器，藉此避免任何非 CloudFront 流量連至您的原始伺服器。CloudFront 會維護受管字首清單，讓清單上所有 CloudFront 全球面向原始伺服器的伺服器 IP 位址始終維持在最新狀態。有了 CloudFront 受管字首清單，您就不必親自閱讀或維護 IP 位址範圍清單。

例如，假設您的原始伺服器是位於歐洲 (倫敦) 區域 (eu-west-2) 的 Amazon EC2 執行個體。如果執行個體位於 VPC 中，您可以建立安全群組規則，允許來自 CloudFront 受管字首清單的傳入 HTTPS 存取。如此可讓所有 CloudFront 全球面向原始伺服器的伺服器連至該執行個體。如果您移除了所有其他來自安全群組的傳入規則，則會阻止任何非 CloudFront 流量連至執行個體。

CloudFront 受管字首清單如下所示：

- `com.amazonaws.global.cloudfront.origin-facing` (IPv4)
- `com.amazonaws.global.ipv6.cloudfront.origin-facing` (IPv6)

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的「[使用 AWS 受管字首清單](#)」。

Important

CloudFront 受管字首清單在應用至 Amazon VPC 配額的方式上，有其獨到之處。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的「[AWS 受管字首清單權重](#)」。

以 AWS SDK 使用 CloudFront

AWS 軟體開發套件 (SDK) 適用於許多常用的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	程式碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例

SDK 文件	程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

可用性範例

找不到所需的內容嗎？請使用本頁面底部的提供意見回饋連結申請程式碼範例。

CloudFront 技術資源

使用下列資源取得有關 CloudFront 技術問題的解答：

- [AWS re:Post](#) - 社群式問答網站，讓開發人員討論與 CloudFront 相關的技術問題。
- [支援中心](#)：此網站包含您最近支援案例和 AWS Trusted Advisor 及運作狀態檢查結果的相關資訊。其中還提供論壇、技術常見問題、服務運作狀態儀表板和支援計劃相關資訊的連結。

- [AWS Premium Support](#)：瞭解 AWS Premium Support 這項一對一的快速回應支援服務管道，如何協助您在 AWS 上建構和執行應用程式。
- [AWS IQ](#)：取得 AWS 認證專業人員和專家的協助。

開始使用 CloudFront

本節主題說明如何開始使用 Amazon CloudFront 交付您的內容。

[設定您的 AWS 帳戶](#) 主題說明下列教學課程的先決條件，例如建立 AWS 帳戶 和建立具有管理存取權的使用者。

基本分佈教學課程說明如何設定原始存取控制 (OAC)，將已驗證的請求傳送至 Amazon S3 原始伺服器。

安全靜態網站教學課程說明如何使用 OAC 搭配 Amazon S3 原始伺服器為您的網域名稱建立安全靜態網站。本教學課程使用 Amazon CloudFront (CloudFront) 範本進行設定和部署。

主題

- [設定您的 AWS 帳戶](#)
- [開始使用 CloudFront 標準分佈](#)
- [開始使用標準分佈 \(AWS CLI\)](#)
- [開始使用安全的靜態網站](#)

設定您的 AWS 帳戶

此主題說明初步的步驟，例如建立 AWS 帳戶帳戶，以讓您準備使用 Amazon CloudFront。

主題

- [註冊 AWS 帳戶](#)
- [建立具有管理存取權的使用者](#)
- [選擇如何存取 CloudFront](#)

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

1. 開啟 <https://portal.aws.amazon.com/billing/signup>。

2. 請遵循線上指示進行。

部分註冊程序需接收來電或簡訊，並在電話鍵盤輸入驗證碼。

當您註冊時 AWS 帳戶，AWS 帳戶根使用者會建立。根使用者有權存取該帳戶中的所有 AWS 服務和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行[需要根使用者存取權的任務](#)。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 <https://aws.amazon.com/> 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立具有管理存取權的使用者

註冊後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center 和建立管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者[AWS 管理主控台](#)身分登入。在下一頁中，輸入您的密碼。

如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的[以根使用者身分登入](#)。

2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

如需說明，請參閱《IAM 使用者指南》中的[為您的 AWS 帳戶 根使用者（主控台）啟用虛擬 MFA 裝置](#)。

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[啟用 AWS IAM Identity Center](#)。

2. 在 IAM Identity Center 中，將管理存取權授予使用者。

如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱 AWS IAM Identity Center 《使用者指南》中的[使用預設值設定使用者存取 IAM Identity Center 目錄](#)。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《使用者指南》中的[登入 AWS 存取入口網站](#)。

指派存取權給其他使用者

- 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[建立許可集](#)。

- 將使用者指派至群組，然後對該群組指派單一登入存取權。

如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的[新增群組](#)。

選擇如何存取 CloudFront

您可以通過以下方式存取 Amazon CloudFront：

- AWS 管理主控台 – 本指南中的程序說明如何使用 AWS 管理主控台 來執行任務。
- AWS SDKs – 如果您使用的是提供 開發套件的 AWS 程式設計語言，您可以使用 開發套件來存取 CloudFront。開發套件會簡化身分驗證、與開發環境輕鬆整合，並讓您存取 CloudFront 命令。如需詳細資訊，請參閱[以 AWS SDK 使用 CloudFront](#)。
- CloudFront API – 如果您使用沒有適用開發套件的程式設計語言，請參閱 [Amazon CloudFront API 參考](#)，以取得 API 動作以及如何提出 API 請求的相關資訊。
- AWS CLI – The AWS Command Line Interface (AWS CLI) 是用於管理的統一工具 AWS 服務。如需如何安裝和設定 AWS CLI的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的[安裝或更新為 AWS CLI的最新版本](#)。
- Tools for Windows PowerShell：如果您有使用 Windows PowerShell 的經驗，您可能會偏好使用 AWS Tools for Windows PowerShell。如需詳細資訊，請參閱「AWS Tools for PowerShell 使用者指南」中的[安裝 AWS Tools for Windows PowerShell](#)。

開始使用 CloudFront 標準分佈

本節中的程序示範如何使用 CloudFront 來設定可執行下列動作的標準分佈：

- 建立 S3 儲存貯體以用作分佈來源。
- 將物件原始版本存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。
- 使用原始存取控制 (OAC) 將已驗證的請求傳送至您的 Amazon S3 原始伺服器。OAC 透過 CloudFront 傳送請求，以防止檢視器直接存取您的 S3 儲存貯體。如需 OAC 的詳細資訊，請參閱 [限制對 Amazon S3 原始伺服器的存取](#)。
- 在物件 URL 中使用 CloudFront 網域名稱 (例如 `https://d1111111abcdef8.cloudfront.net/index.html`)。
- 將您的物件儲存在 CloudFront 邊緣節點達 24 小時預設持續時間 (最短持續時間為 0 秒)。

您建立 CloudFront 分佈時，以上作業大部分是由系統自動為您設定。

主題

- [先決條件](#)
- [建立 Amazon S3 儲存貯體](#)
- [將內容上傳至儲存貯體](#)
- [建立使用 Amazon S3 原始伺服器搭配 OAC 的 CloudFront 分佈](#)
- [透過 CloudFront 存取內容](#)
- [清除](#)
- [增強基本分佈](#)

先決條件

開始之前，請確定您已完成 [設定您的 AWS 帳戶](#) 所述的步驟。

建立 Amazon S3 儲存貯體

Amazon S3 儲存貯體是檔案 (物件) 或資料夾的容器。CloudFront 使用 S3 儲存貯體做為來源時，能夠自動分佈幾乎任何類型的檔案。例如，CloudFront 可以分發文字、影像和影片。您可以在 Amazon S3 上存放的資料量沒有上限。

在本教學課程中，您會使用建立基本網頁的範例 hello world 檔案來建立 S3 儲存貯體。

建立儲存貯體

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/> 開啟 Amazon S3 主控台。

2. 我們建議您使用我們的 Hello World 範例來進行此「入門」操作。下載 Hello World 網頁：[hello-world-html.zip](#)。將其解壓縮，並將 css 資料夾和 index 檔案儲存在方便的位置，例如執行瀏覽器的桌面。
3. 選擇建立儲存貯體。
4. 輸入符合《Amazon Simple Storage Service 使用者指南》中[一般用途儲存貯體命名規則](#)的唯一儲存貯體名稱。
5. 對於 區域，我們建議選擇接近您地理位置的 AWS 區域。(這可降低延遲和成本。)
 - 不過選擇不同區域也沒問題。您可以這樣做來因應法規要求。
6. 將所有其他設定保留為預設值，然後選擇 Create bucket (建立儲存貯體)。

將內容上傳至儲存貯體

您建立 Amazon S3 儲存貯體後，將解壓縮 hello world 檔案的內容上傳至其中。(您已下載並在 [建立 Amazon S3 儲存貯體](#) 中解壓縮此檔案。)

將內容上傳至 Amazon S3

1. 在一般用途儲存貯體區段中，選擇新儲存貯體的名稱。
2. 選擇上傳。
3. 在上傳頁面上，將 css 資料夾和 index 檔案拖曳到放置區域。
4. 將所有其他設定保留為預設值，然後選擇上傳。

建立使用 Amazon S3 原始伺服器搭配 OAC 的 CloudFront 分佈

在本教學課程中，您將建立 CloudFront 分佈，該分佈使用 Amazon S3 原始伺服器搭配原始存取控制 (OAC)。OAC 可協助您將已驗證的請求安全地傳送至 Amazon S3 原始伺服器。如需 OAC 的詳細資訊，請參閱 [限制對 Amazon S3 原始伺服器的存取](#)。

以使用 OAC 的 Amazon S3 原始伺服器建立 CloudFront 分佈

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇 Create Distribution (建立分佈)。
3. 輸入標準分佈的分佈名稱。名稱將顯示為 Name 鍵值做為標籤。您可以稍後變更此值。您最多可以為標準分佈新增 50 個標籤。如需詳細資訊，請參閱[標記分佈](#)。
4. 選擇單一網站或應用程式、下一步。

5. 選擇下一步。
6. 對於來源類型頁面，選取 Amazon S3。
7. 對於 S3 原始伺服器，請選擇瀏覽 S3，然後選取您為此教學課程建立的 S3 儲存貯體。
8. 對於設定，請選擇使用建議的原始伺服器設定。CloudFront 將使用 Amazon S3 原始伺服器預設的建議快取和原始伺服器設定，包括設定原始存取控制 (OAC)。如需建議設定的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。
9. 選擇下一步。
10. 在啟用安全性保護頁面上，選擇是否啟用 AWS WAF 安全性保護。
11. 選擇下一步。
12. 選擇 Create Distribution (建立分佈)。CloudFront 會為您更新 S3 儲存貯體政策。
13. 檢閱新分佈的詳細資訊區段。您的分佈完成部署時，上次修改欄位會從正在部署變更為日期和時間。
14. 記錄 CloudFront 指派給您的分佈的網域名稱。它看起來類似下列：`d111111abcdef8.cloudfront.net`。

在生產環境中使用本教學課程中的分佈和 S3 儲存貯體之前，請務必進行設定使其滿足您的特定需求。如需在生產環境中設定存取權限的詳細資訊，請參閱 [設定安全存取和限制對內容的存取](#)。

透過 CloudFront 存取內容

若要透過 CloudFront 存取您的內容，請將您的 CloudFront 分佈的網域名稱與內容主頁面合併。(您已在 [建立使用 Amazon S3 原始伺服器搭配 OAC 的 CloudFront 分佈](#) 中記錄您的分佈網域名稱。)

- 您的分佈網域名稱可能會像這樣：`d111111abcdef8.cloudfront.net`。
- 網站主頁的路徑通常是 `/index.html`。

因此，透過 CloudFront 存取您的內容的 URL 可能像這樣：

`https://d111111abcdef8.cloudfront.net/index.html`

如果您遵循上述步驟並使用 Hello World 網頁，您應該會看到顯示 Hello world! 的網頁。

當您上傳更多內容到此 S3 儲存貯體時，藉由將 CloudFront 分佈網域名稱與 S3 儲存貯體中物件的路徑合併，即可透過 CloudFront 存取內容。例如，如果您上傳名為 `new-page.html` 的新檔案到 S3 儲存貯體的根目錄，則 URL 會像這樣：

`https://d1111111abcdef8.cloudfront.net/new-page.html.`

清除

如果您只將分佈和 S3 儲存貯體建立為學習練習，請將其刪除，這樣就不會再產生費用。先刪除分佈。如需詳細資訊，請參閱下列連結：

- [刪除分發](#)
- [刪除儲存貯體](#)

增強基本分佈

本入門教學課程提供建立分佈的最小架構。我們建議您進一步探索下列增強功能：

- 您可以使用 CloudFront 私有內容功能來限制對 Amazon S3 儲存貯體中內容的存取。如需有關分佈私有內容的詳細資訊，請參閱[使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。
- 您可以將 CloudFront 分佈設定為使用自訂網域名稱 (例如，`www.example.com` 而非 `d1111111abcdef8.cloudfront.net`)。如需詳細資訊，請參閱[使用自訂 URL](#)。
- 本教學課程使用具有原始存取控制 (OAC) 的 Amazon S3 原始伺服器。不過如果您的原始伺服器是設定為網站端點的 S3 儲存貯體，則無法使用 OAC。如果是這種情況，您必須使用 CloudFront 設定儲存貯體做為自訂原始伺服器。如需詳細資訊，請參閱[使用設定為網站端點的 Amazon S3 儲存貯體](#)。如需 OAC 的詳細資訊，請參閱[限制對 Amazon S3 原始伺服器的存取](#)。

開始使用標準分佈 (AWS CLI)

本節中的程序說明如何 AWS CLI 搭配 CloudFront 使用 來設定涉及下列項目的基本組態：

- 建立 Amazon S3 儲存貯體以用作分佈來源。
- 在 S3 儲存貯體中儲存物件的原始版本。
- 使用原始存取控制 (OAC) 將已驗證的請求傳送至您的 Amazon S3 原始伺服器。OAC 透過 CloudFront 傳送請求，以防止檢視器直接存取您的 S3 儲存貯體。如需 OAC 的詳細資訊，請參閱[限制對 Amazon S3 原始伺服器的存取](#)。
- 在物件 URL 中使用 CloudFront 網域名稱 (例如 `https://d1111111abcdef8.cloudfront.net/index.html`)。
- 將您的物件儲存在 CloudFront 邊緣節點達 24 小時預設持續時間 (最短持續時間為 0 秒)。

其中的大多數選項均可供自訂。如需如何自訂 CloudFront 分佈選項的相關資訊，請參閱[建立分發](#)。

先決條件

開始之前，請確定您已完成 [設定您的 AWS 帳戶](#) 所述的步驟。

安裝 AWS CLI 並使用 登入資料進行設定。如需詳細資訊，請參閱 AWS CLI 使用者指南中的 [AWS CLI 入門](#)。

建立 Amazon S3 儲存貯體

Amazon S3 儲存貯體是檔案 (物件) 或資料夾的容器。CloudFront 使用 S3 儲存貯體做為來源時，能夠自動分佈幾乎任何類型的檔案。例如，CloudFront 可以分發文字、影像和影片。您可以在 Amazon S3 上存放的資料量沒有上限。

在本教學課程中，您會建立 S3 儲存貯體並上傳用來建立基本網頁的 HTML 檔案。

```
aws s3 mb s3://amzn-s3-demo-bucket/ --region us-east-1
```

以全域唯一儲存貯體名稱取代 *amzn-s3-demo-bucket*。對於 AWS 區域，我們建議您選擇地理位置接近的區域。這可降低延遲和成本，但選擇不同的區域也可以。例如您可以這樣做來因應法規要求。

將內容上傳至儲存貯體

在本教學課程中，請下載及擷取基本「Hello World」網頁的範例內容檔案。

```
# Create a temporary directory
mkdir -p ~/cloudfront-demo

# Download the sample Hello World files
curl -o ~/cloudfront-demo/hello-world-html.zip https://docs.aws.amazon.com/
AmazonCloudFront/latest/DeveloperGuide/samples/hello-world-html.zip

# Extract the zip file
unzip ~/cloudfront-demo/hello-world-html.zip -d ~/cloudfront-demo/hello-world
```

這會建立具有 index.html 檔案和 css 資料夾的目錄。將這些檔案上傳至 S3 儲存貯體。

```
aws s3 cp ~/cloudfront-demo/hello-world/ s3://amzn-s3-demo-bucket/ --recursive
```

建立原始存取控制 (OAC)

在本教學課程中，您將建立原始存取控制 (OAC)。OAC 可協助您將已驗證的請求安全地傳送至 Amazon S3 原始伺服器。如需 OAC 的詳細資訊，請參閱 [限制對 Amazon S3 原始伺服器的存取](#)。

```
aws cloudfront create-origin-access-control \  
  --origin-access-control-config Name="oac-for-  
s3",SigningProtocol=sigv4,SigningBehavior=always,OriginAccessControlOriginType=s3
```

將輸出中的 OAC ID 儲存為環境變數。使用您自己的 OAC ID 取代範例值。下一個步驟會用到此值。

```
OAC_ID="E1ABCD2EFGHIJ"
```

建立標準分佈

建立一個名為 `distribution-config.json` 的分佈組態檔案。將範例儲存貯體名稱取代為 `Id`、`DomainName` 和 `TargetOriginId` 值的儲存貯體名稱。

```
cat > distribution-config.json << EOF  
{  
  "CallerReference": "cli-example-$(date +%s)",  
  "Origins": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "Id": "S3-amzn-s3-demo-bucket",  
        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",  
        "S3OriginConfig": {  
          "OriginAccessIdentity": ""  
        },  
        "OriginAccessControlId": "$OAC_ID"  
      }  
    ]  
  },  
  "DefaultCacheBehavior": {  
    "TargetOriginId": "S3-amzn-s3-demo-bucket",  
    "ViewerProtocolPolicy": "redirect-to-https",  
    "AllowedMethods": {  
      "Quantity": 2,  
      "Items": ["GET", "HEAD"],  
      "CachedMethods": {
```

```
        "Quantity": 2,
        "Items": ["GET", "HEAD"]
      },
    },
    "DefaultTTL": 86400,
    "MinTTL": 0,
    "MaxTTL": 31536000,
    "Compress": true,
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      }
    }
  },
  "Comment": "CloudFront distribution for S3 bucket",
  "Enabled": true
}
EOF
```

建立標準分佈。

```
aws cloudfront create-distribution --distribution-config file://distribution-
config.json
```

將輸出的分佈 ID 和網域名稱儲存為環境變數。使用自己的取代範例值。您將在本教學課程稍後使用這些值。

```
DISTRIBUTION_ID="EABCD1234XMPL"
DOMAIN_NAME="d111111abcdef8.cloudfront.net"
```

在生產環境中使用本教學課程中的分佈和 S3 儲存貯體之前，請務必進行設定使其滿足您的特定需求。如需在生產環境中設定存取權限的詳細資訊，請參閱 [設定安全存取和限制對內容的存取](#)。

更新您的 S3 儲存貯體政策

更新您的 S3 儲存貯體政策，以允許 CloudFront 存取物件。以您的儲存貯體名稱取代範例儲存貯體名稱。

```
# Get your AWS account ID
```

```
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)

# Create the bucket policy
cat > bucket-policy.json << EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipal",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn": "arn:aws:cloudfront::$ACCOUNT_ID:distribution/
$DISTRIBUTION_ID"
        }
      }
    ]
  ]
}
EOF

# Apply the bucket policy
aws s3api put-bucket-policy \
  --bucket amzn-s3-demo-bucket \
  --policy file://bucket-policy.json
```

確認分佈部署

您建立分佈後，需要一些時間才能完成部署。分佈狀態從 `InProgress` 變更為 `Deployed` 時，請繼續下一個步驟。

```
aws cloudfront get-distribution --id $DISTRIBUTION_ID --query 'Distribution.Status'
```

或者您可以使用 `wait` 命令來等待分佈部署。

```
aws cloudfront wait distribution-deployed --id $DISTRIBUTION_ID
```

透過 CloudFront 存取內容

若要透過 CloudFront 存取您的內容，請將您的 CloudFront 分佈的網域名稱與內容主頁面合併。將範例 CloudFront 網域名稱取代為您自己的網域名稱。

```
https://d1111111abcdef8.cloudfront.net/index.html
```

如果您遵循上述步驟並建立 HTML 檔案，您應該會看到顯示 Hello world! 的網頁。

當您上傳更多內容到此 S3 儲存貯體時，藉由將 CloudFront 分佈網域名稱與 S3 儲存貯體中物件的路徑合併，即可透過 CloudFront 存取內容。例如，如果您上傳名為 new-page.html 的新檔案到 S3 儲存貯體的根目錄，則 URL 會像這樣：

```
https://d1111111abcdef8.cloudfront.net/new-page.html.
```

清除

如果您只將分佈和 S3 儲存貯體建立為學習練習，請將其刪除，這樣就不會再產生費用。先停用及刪除分佈。

停用並刪除標準分佈 (AWS CLI)

1. 首先請停用分佈。

```
# Get the current configuration and ETag
ETAG=$(aws cloudfront get-distribution-config --id $DISTRIBUTION_ID --query 'ETag'
--output text)

# Create a modified configuration with Enabled=false
aws cloudfront get-distribution-config --id $DISTRIBUTION_ID | \
jq '.DistributionConfig.Enabled = false' > temp_disabled_config.json

# Update the distribution to disable it
aws cloudfront update-distribution \
  --id $DISTRIBUTION_ID \
  --distribution-config file:///$(jq '.DistributionConfig'
temp_disabled_config.json) \
  --if-match $ETAG
```

2. 等待停用分佈。

```
aws cloudfront wait distribution-deployed --id $DISTRIBUTION_ID
```

3. 刪除分佈。

```
# Get the current ETag
ETAG=$(aws cloudfront get-distribution-config --id $DISTRIBUTION_ID --query 'ETag'
--output text)

# Delete the distribution
aws cloudfront delete-distribution --id $DISTRIBUTION_ID --if-match $ETAG
```

刪除 S3 儲存貯體 (AWS CLI)

- 刪除 S3 儲存貯體及其內容。用您自己的儲存貯體名稱取代範例儲存貯體名稱。

```
# Delete the bucket contents
aws s3 rm s3://amzn-s3-demo-bucket --recursive

# Delete the bucket
aws s3 rb s3://amzn-s3-demo-bucket
```

若要清除為此教學課程建立的本機檔案，請執行下列命令：

```
# Clean up local files
rm -f distribution-config.json bucket-policy.json temp_disabled_config.json
rm -rf ~/cloudfront-demo
```

或者您可以刪除為此教學課程建立的 OAC。

```
# Get the OAC ETag
OAC_ETAG=$(aws cloudfront get-origin-access-control --id $OAC_ID --query 'ETag' --
output text)

# Delete the OAC
aws cloudfront delete-origin-access-control --id $OAC_ID --if-match $OAC_ETAG
```

開始使用安全的靜態網站

您可以使用本主題中所述的解決方案，為您的網域名稱建立安全的靜態網站，以開始使用 Amazon CloudFront。「靜態網站」只會使用靜態檔案 (例如 HTML、CSS、JavaScript、影像和視訊)，而不需要伺服器或伺服器端處理。有了這個解決方案，您的網站可以獲得以下好處：

- 使用 [Amazon 簡易儲存服務 \(Amazon S3\)](#) 的耐久儲存 – 此解決方案會建立 Amazon S3 儲存貯體來託管靜態網站的內容。要更新您的網站，只需將您的新檔案上傳到 S3 儲存貯體即可。
- 透過 Amazon CloudFront 內容傳遞網路加快速度 – 此解決方案會建立 CloudFront 分佈，以低延遲的方式將您的網站提供給檢視器。分佈是以[原始存取控制](#) (OAC) 進行設定，以確保網站只能透過 CloudFront 存取，而不是直接從 S3 存取。
- 受 HTTPS 和安全標頭保護：此解決方案會在 [AWS Certificate Manager \(ACM\)](#) 中建立 SSL/TLS 憑證，並將其附加到 CloudFront 分佈中。此憑證可讓您透過 HTTPS 安全地為您網域的網站提供分佈。
- 已使用 [AWS CloudFormation](#) 設定及部署 – 此解決方案使用 CloudFormation 範本來設定所有元件，讓您可以更專注於網站內容，而花更少時間在設定元件上。

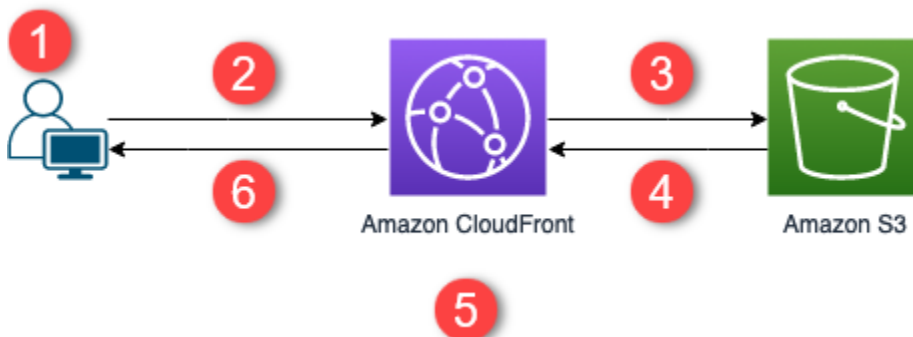
這個解決方案在 GitHub 上提供開放原始碼。若要檢視程式碼、提交提取請求或開立問題單，請前往 <https://github.com/aws-samples/amazon-cloudfront-secure-static-site>。

主題

- [解決方案概觀](#)
- [部署解決方案](#)

解決方案概觀

下圖顯示此靜態網站解決方案運作方式的概觀：



1. 瀏覽者在 `www.example.com` 請求網站。
2. 如果請求的物件已快取，CloudFront 會將物件從其快取傳回給檢視器。
3. 如果物件不在 CloudFront 的快取中，則 CloudFront 會從原始伺服器 (S3 儲存貯體) 請求物件。
4. S3 會將物件傳回 CloudFront。
5. CloudFront 會快取物件。
6. 物件會傳回給檢視器。對後續來到相同 CloudFront 節點之物件的請求，會從 CloudFront 快取中提供。

部署解決方案

若要部署此安全靜態網站解決方案，您可以選擇下列其中一個選項：

- 使用 CloudFormation 主控台以預設內容部署解決方案，然後將您的網站內容上傳到 Amazon S3。
- 將解決方案複製到您的電腦，以新增您的網站內容。然後，使用 AWS Command Line Interface (AWS CLI) 部署解決方案。

Note

您必須使用美國東部 (維吉尼亞北部) 區域來部署 CloudFormation 範本。

主題

- [必要條件](#)
- [使用 CloudFormation 主控台](#)
- [在本機複製解決方案](#)
- [尋找存取日誌](#)

必要條件

若要使用此解決方案，您必須具備下列先決條件：

- 指向 Amazon Route 53 託管區域的已註冊網域名稱 (例如 `example.com`)。託管區域必須位於您部署此解決方案的相同 AWS 帳戶 中。如果您沒有已註冊的網域名稱，可以使用 [Route 53 註冊一個網域名稱](#)。如果您有已註冊的網域名稱，但未指向 Route 53 託管區域，請將 [Route 53 配置為您的 DNS 服務](#)。

- AWS Identity and Access Management (IAM) 許可，以啟動建立 IAM 角色的 CloudFormation 範本，以及在解決方案中建立所有 AWS 資源的許可。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的[使用 AWS Identity and Access Management 控制存取權限](#)。

您必須自行負責使用此解決方案時所產生的費用。如需費用的詳細資訊，請參閱[各個 AWS 服務的定價頁面](#)。

使用 CloudFormation 主控台

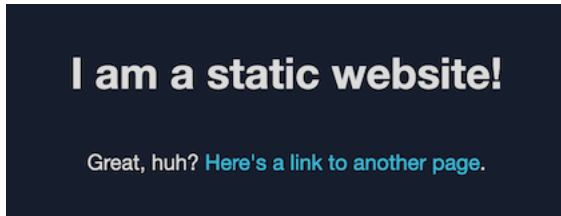
使用 CloudFormation 主控台進行部署

1. [在 CloudFormation 主控台中啟動此解決方案](#)。如有必要，請登入您的 AWS 帳戶 帳戶。
2. 建立堆疊精靈會在 CloudFormation 主控台中開啟，其中的預先填入欄位會指定此解決方案 CloudFormation 範本。

請選擇頁面最下方的 Next (下一頁)。

3. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，輸入下列欄位的數值：
 - SubDomain – 輸入為您網站所用的子網域名稱。例如，如果子網域名稱為 www，表示您的網站可以在 [www.example.com](#) 使用。(請將 example.com 取代為您的網域名稱，如下列項目符號所述)。
 - DomainName – 輸入您的網域名稱，例如 [example.com](#)。此網域必須指向 Route 53 託管區域。
 - HostedZoneId：網域名稱的 Route 53 託管區域 ID。
 - CreateApex：(選用) 在 CloudFront 組態中建立網域頂點的別名 (example.com)。
4. 完成時，請選擇 Next (下一步)。
5. (選用) 在 配置堆疊選項頁面上，[新增標籤和其他堆疊選項](#)。
6. 完成時，請選擇 Next (下一步)。
7. 在 Review (檢閱) 頁面上，捲動至頁面底部，然後選取 Capabilities (功能) 區段中的兩個方塊。這些功能允許 CloudFormation 建立 IAM 角色，以允許存取堆疊的資源，並以動態方式命名資源。
8. 選擇 Create Stack (建立堆疊)。
9. 等待堆疊完成建立。堆疊會建立一些巢狀堆疊，而且可能需要幾分鐘才能完成。完成時，Status (狀態) 會變更為 CREATE_COMPLETE。

當狀態為 `CREATE_COMPLETE` 時，請前往 <https://www.example.com> 檢視您的網站 (將 `www.example.com` 取代為您步驟 3 中指定的子網域和網域名稱)。您應該會看到網站的預設內容：



將網站的預設內容取代為您自己的內容

1. 在以下網址開啟 Amazon S3 主控台：<https://console.aws.amazon.com/s3/>。
2. 選擇名稱以 `amazon-cloudfront-secure-static-site-s3bucketroot-` 開頭的儲存貯體。

 Note

確定選擇名稱中包含 `s3bucketroot` 的儲存貯體，而不是 `s3bucketlog`。名稱中含有 `s3bucketroot` 儲存貯體包含網站內容。帶有 `s3bucketlog` 儲存貯體只包含日誌檔案。

3. 刪除網站的預設內容，然後上傳您自己的內容。

 Note

如果您使用此解決方案的預設內容檢視您的網站，則某些預設內容可能會快取在 CloudFront 節點。若要確保檢視器能看到您更新的網站內容，請使檔案失效，以便從 CloudFront 節點移除快取的複本。如需更多詳細資訊，請參閱 [使檔案失效以移除內容](#)。

在本機複製解決方案

先決條件

要在部署此解決方案之前新增您的網站內容，您必須在本機封裝解決方案的成品，這需要 Node.js 和 npm。如需更多詳細資訊，請參閱 <https://www.npmjs.com/get-npm>。

新增您的網站內容並部署解決方案

1. 從 複製或下載解決方案<https://github.com/aws-samples/amazon-cloudfront-secure-static-site> 複製或下載之後，請開啟命令提示字元或終端機，然後瀏覽至 `amazon-cloudfront-secure-static-site` 資料夾。

2. 執行下列命令以安裝並封裝解決方案的成品：

```
make package-static
```

3. 將網站的內容複製到 `www` 資料夾中，覆寫預設的網站內容。
4. 執行下列 AWS CLI 命令以建立 Amazon S3 儲存貯體來儲存解決方案的成品。用您自己的儲存貯體名稱取代 *amzn-s3-demo-bucket-for-artifacts*。

```
aws s3 mb s3://amzn-s3-demo-bucket-for-artifacts --region us-east-1
```

5. 執行下列 AWS CLI 命令，將解決方案成品封裝為 CloudFormation 範本。以您在上一個步驟中建立的儲存貯體名稱取代 *amzn-s3-demo-bucket-for-artifacts*。

```
aws cloudformation package \  
  --region us-east-1 \  
  --template-file templates/main.yaml \  
  --s3-bucket amzn-s3-demo-bucket-for-artifacts \  
  --output-template-file packaged.template
```

6. 執行下列命令以使用 CloudFormation 部署解決方案，並取代下列值：

- *your-CloudFormation-stack-name*：以 CloudFormation 堆疊的名稱取代。
- *example.com* – 取代為您的網域名稱。此網域必須指向相同 AWS 帳戶 中的 Route 53 託管區域。
- *www* – 取代為您的網站所使用的子網域名稱。例如，如果子網域名稱為 `www`，表示您的網站可以在 `www.example.com` 使用。
- *hosted-zone-ID*：取代為您網域名稱的 Route 53 託管區域 ID。

```
aws cloudformation deploy \  
  --region us-east-1 \  
  --stack-name your-CloudFormation-stack-name \  
  --template-file packaged.template \  
  --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \  
  --
```

```
--parameter-overrides DomainName=example.com SubDomain=www HostedZoneId=hosted-zone-ID
```

- (選用) 若要使用網域頂點部署堆疊，請改為執行下列命令。

```
aws --region us-east-1 cloudformation deploy \  
  --stack-name your-CloudFormation-stack-name \  
  --template-file packaged.template \  
  --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \  
  --parameter-overrides DomainName=example.com SubDomain=www \  
  HostedZoneId=hosted-zone-ID CreateApex=yes
```

7. 等待 CloudFormation 堆疊完成建立。堆疊會建立一些巢狀堆疊，而且可能需要幾分鐘才能完成。完成時，Status (狀態) 會變更為 CREATE_COMPLETE。

當狀態變更為 CREATE_COMPLETE 時，請前往 <https://www.example.com> 檢視您的網站 (將 www.example.com 取代為您在上一個步驟中指定的子網域和網域名稱)。您應該會看到您網站的內容。

尋找存取日誌

此解決方案會啟用 CloudFront 分佈的[存取日誌](#)。請完成以下步驟來尋找分佈的存取日誌。

尋找分佈的存取日誌

1. 在以下網址開啟 Amazon S3 主控台：<https://console.aws.amazon.com/s3/>。
2. 選擇名稱以 amazon-cloudfront-secure-static-site-s3bucketlogs- 開頭的儲存貯體。

Note

確定選擇名稱中包含 s3bucketlogs 的儲存貯體，而不是 s3bucketroot。名稱中含有 s3bucketlog 的儲存貯體包含日誌檔案。帶有 s3bucketroot 的儲存貯體包含網站內容。

3. 名為 cdn 的資料夾包含 CloudFront 存取記錄。

CloudFront 固定費率定價計劃

CloudFront 固定費率定價計劃將 Amazon CloudFront 全球內容交付網路 (CDN) 與多個 AWS 服務 和功能結合為每月價格，而不會產生超額費用，無論流量峰值或攻擊。

固定費率定價計劃包含下列功能，每月價格簡單：

- CloudFront CDN
- AWS WAF 和 DDoS 保護
- 機器人管理和分析
- Amazon Route 53 DNS
- Amazon CloudWatch Logs 擷取
- TLS 憑證
- 無伺服器邊緣運算
- 每月 Amazon S3 儲存點數

計劃提供免費、專業、商業和高級方案，以符合應用程式的需求。計劃不需要年度承諾，即可獲得最佳的可用費率。從 免費計劃開始並升級，以存取更多功能和更大的用量限額。

主題

- [CloudFront 固定費率定價計劃的優點](#)
- [依定價計劃方案的功能](#)
- [每月用量限額](#)
- [您的計劃涵蓋的成本](#)
- [使用定價計劃降低整體 AWS 成本](#)
- [管理您的固定費率定價計劃](#)
- [許可](#)
- [固定費率定價計劃配額](#)
- [不支援的功能](#)

CloudFront 固定費率定價計劃的優點

CloudFront 定價計劃提供幾個主要優點：

- 合併服務和定價

以單一固定費率將多個 AWS 服務 和 功能合併為單一計劃。旨在消除個別的服務購買和預付定價計算。

- 沒有超額

無論流量尖峰或攻擊為何，都不會產生超額費用。

- 清除用量限額

每個計劃都包含已發佈的使用額度，專為該方案的最佳效能而設計。監控您的用量、接收主動通知，並根據應用程式的需求進行升級，無需長期承諾。

- 防範 DDoS 攻擊

CloudFront 和 會在攻擊到達您的基礎設施之前加以 AWS WAF 吸收和封鎖。僅針對合法流量保留您的運算、資料庫和基礎設施使用率。封鎖的 DDoS 攻擊和遭 封鎖的請求 AWS WAF 永遠不會計入您的用量限額。

- 降低整體 AWS 成本

從 Amazon S3、AWS Application Load Balancer (ALB) 或 Amazon API Gateway 等服務上執行 AWS 的應用程式到 CloudFront 的資料傳輸仍然免費。當您透過 CloudFront 而非直接透過網際網路提供 AWS 應用程式時，您的固定費率計劃會以簡單的每月價格涵蓋應用程式與瀏覽者之間的資料傳輸成本，而無需擔心超額。到達原始伺服器的請求較少，也會降低根據用量收費的服務成本。

依定價計劃方案的功能

每個定價計劃涵蓋一個 CloudFront 分佈，最多一個頂點（根）網域，將基本功能和服務合併為每月價格。每個計劃也包含額外的 S3 儲存點數。

較高層級的計劃包含來自較低層級計劃的所有功能，以及其他功能。

- 免費 – 適用於業餘愛好者、學習者和開發人員入門。
- Pro – 啟動和成長小型網站、部落格和應用程式。
- 商業 – 保護和加速商業應用程式。
- Premium – 擴展和保護業務和關鍵任務應用程式。

選取包含應用程式所需功能和組態的計劃方案。請參閱下列每個定價計劃的功能。

定價計劃功能

下表顯示每個定價計劃方案中包含的 CloudFront、AWS WAF DDoS、Amazon Route 53、Amazon CloudWatch 和 Amazon S3 功能。

效能和交付	免費	pro	商業	高級
全域 CDN	是	是	是	是
使用 CloudFront 超過 750 個全球節點做為 Web 應用程式的大型分散式單一進入點。加速靜態、動態和不可快取的應用程式。				
內容快取	是	是	是	是
將您的內容副本存放在全球 750 個以上的 CloudFront 節點，從最近的位置將其交付給使用者。減少載入時間、保護您的應用程式免於流量遽增，並透過在本機提供重複的請求來節省成本，而不是從您的應用程式伺服器。				
快速快取失效	是	是	是	是

效能和交付	免費	pro	商業	高級
在幾秒鐘內移除或更新所有節點的快取內容。				
智慧路由	是	是	是	是
使用即時網路資料，以智慧方式將使用者路由到最佳節點，並透過 AWS 私有網路連接到您的 AWS 原始伺服器，以獲得更好的效能。				
分層快取	是	是	是	是
區域節點快取 位於節點和應用程式之間，以存放更長時間的內容、減少應用程式負載並維持快速交付。				
預設快取規則	是	是	是	是
做出有效的快取決策，在沒有自訂組態的情況下快取大多數 Web 應用程式。				

效能和交付	免費	pro	商業	高級
自訂快取規則 透過指定要使用的請求值、使用快取政策最佳化應用程式的效能、個人化和新鮮度需求，來控制 CloudFront 如何快取內容。 ??? 高速原始路由 使用 Origin Shield，動態請求會使用 CloudFront 的私有網路從節點路由到您的原始伺服器，以實現到原始伺服器的高效能路徑。 原始負載減少 使用 Origin Shield 在 Web 應用程式附近新增額外的快取層。Origin Shield 會合併來自所有節點的請求，減少應用程式負載，特別是在流量激增期間。			是 	是 
				是 
				是 

效能和交付	免費	pro	商業	高級
自動原始伺服器容錯移轉 如果您的主要原始伺服器故障，則會自動將流量路由到備份原始伺服器，在不中斷使用者的情況下維持高可用性。				是 
預設原始伺服器請求規則 使用針對常見案例最佳化的AWS 受管原始伺服器請求政策，控制來自檢視器請求的資訊會自動包含在對原始伺服器的請求中。	是 	是 	是 	是 
預設回應標頭規則 使用AWS 受管回應標頭政策，在對檢視器的回應中新增或移除 HTTP 標頭，針對常見的安全標頭、CORS 設定和其他標準使用案例預先設定。	是 	是 	是 	是 

效能和交付	免費	pro	商業	高級
自訂原始伺服器 請求規則			是	是
建立您自己的 原始伺服器請求政策 ，以確切指定 哪些 URL 查詢 字串、標頭和 Cookie 會轉送 到您的原始伺服器，進而啟用自 訂分析和請求處 理。				
自訂回應標頭規 則			是	是
建立您自己的 回應標頭政策 ，以 控制 CloudFront 在回應檢視器 時新增或移除哪 些 HTTP 標頭， 例如安全標頭 、內容安全政策 (CSP)、CORS 設定和自訂應用 程式標頭。				

效能和交付	免費	pro	商業	高級
快取行為的數量	5	10	50	100

設定[快取行為](#)以控制 CloudFront 如何處理特定 URL 模式的請求，包括提供內容的原始伺服器、快取內容的方式，以及是否需要 HTTPS 或簽章URLs。

安全性與保護

永遠在線的
DDoS 保護

是

是

是

是



防範以您的網站
或應用程式為目
標的 DDoS 攻
擊。

進階 DDoS 保護

是

是



使用 DDoS AMR. AWS learn 可在幾秒鐘內識別和封鎖 DDoS 攻擊。會學習您的唯一應用程式模式，以區分攻擊和自然激增與合法使用者。

[AntiDDoS](#)

效能和交付	免費	pro	商業	高級
Web Application Firewall (WAF)	是	是	是	是
根據 Amazon 內部威脅情報，防範常見的應用程式漏洞和潛在威脅。請求會在到達您的伺服器之前遭到封鎖。				
WAF 規則的數量	5	25	50	75
您可以在 WAF 組態中建立和啟用的安全規則總數，包括自訂規則和 AWS 受管規則。				
WordPress、PHP 和 SQL 資料庫的保護		是	是	是
使用案例型安全規則來保護常見的應用程式和作業系統，例如 WordPress、PHP、SQL 資料庫、Linux 和 Windows。				

效能和交付	免費	pro	商業	高級
IP 型速率限制	是	是	是	是
在 5 分鐘內自動封鎖超過可設定請求數量的 IP 地址，以防止 HTTP 洪水攻擊和拒絕服務 (DoS) 嘗試。				
地理流量封鎖	是	是	是	是
封鎖來自所選國家或地區的請求。				
標頭型威脅篩選		是	是	是
建立 WAF 安全規則，根據 HTTP 請求標頭篩選威脅。				
Regex 型威脅篩選			是	是
使用規則表達式建立 WAF 安全規則，以符合 URI 路徑和 HTTP 請求屬性。				

效能和交付	免費	pro	商業	高級
JavaScript 挑戰			是	是
要求瀏覽器完成驗證合法使用者的 JavaScript 挑戰，以封鎖自動化威脅。				
機器人管理和分析			是	是
使用常見機器人的 AWS WAF Bot Control 偵測和分析機器人流量。提供控制項來封鎖、挑戰或允許未驗證的機器人，同時識別和區分已驗證的機器人，例如搜尋引擎。				
自訂 WAF 回應		是	是	是
當規則封鎖請求時，設定特定 HTTP 狀態碼和選用的自訂 HTML、純文字或 JSON 回應。				

效能和交付	免費	pro	商業	高級
標頭插入	是	是	是	是
將自訂 HTTP 標頭新增至通過 WAF 檢查的請求，讓下游應用程式以不同的方式處理請求，或標記這些請求以進行分析。				
請求內文檢查	16 KB	16 KB	64 KB	64 KB
HTTP 請求內文內容的大小上限，可由檢查 AWS WAF 威脅和惡意模式。				
VPC 內的私有原始伺服器			是	是
將應用程式保留在 VPC 私有子網路中，只能透過 CloudFront 分佈存取，並使用 VPC 原始 伺服器從公有網際網路隱藏，以增強安全性。				

效能和交付	免費	pro	商業	高級
原始存取控制 (OAC)	是	是	是	是
維護私有 S3 儲存貯體，並僅允許透過您指定的 CloudFront 分佈進行存取，確保您的內容受到您 CloudFront 分佈中設定的 WAF 規則、速率限制和其他安全控制所保護。				
免費 TLS 憑證	是	是	是	是
透過 自動續約網域的免費 TLS 憑證 AWS Certificate Manager。				
Signed URLs (簽署的 URL)	是	是	是	是
建立安全 URLs，為特定使用者提供私有內容的暫時存取權。通常用於與授權使用者共用私有文件，或在付款驗證後授予受保護內容的安全存取權。				

效能和交付	免費	pro	商業	高級
相互 TLS (mTLS)				是
使用 mTLS 身分驗證限制對應用程式的存取，確保只有具有有效憑證的受信任用戶端才能連線。				
邊緣運算				
無伺服器邊緣運算	是	是	是	是
在邊緣執行輕量型 JavaScript，使用 CloudFront Functions 以毫秒為單位修改 URLs、HTTP 標頭和請求/回應元素。				
Edge 鍵值存放區		是	是	是
使用 KeyValueStore 將資料存放在邊緣，透過 CloudFront Functions 進行快速且動態的內容自訂。				
網路和通訊協定支援				

效能和交付	免費	pro	商業	高級
IPv6	是	是	是	是
透過現代 IPv6 和傳統 IPv4 連線，將內容從 CloudFront 交付給檢視器和原始伺服器。為您的應用程式啟用 end-to-end IPv6 支援。				
HTTP/2	是	是	是	是
透過多工、標頭壓縮和串流優先順序等現代通訊協定功能，實現更快的頁面載入速度。在瀏覽器和用戶端支援時自動使用。				
HTTP/3	是	是	是	是
使用 QUIC 將內容交付至支援內容的瀏覽器和用戶端，以實現更快的連線並改善效能。特別有利於行動使用者，並在網路條件變更時維持連線。				

效能和交付	免費	pro	商業	高級
TLS 1.3	是	是	是	是
透過交握程序提供更快的 HTTPS 連線，相較於 TLS 1.2 中的兩個，需要一次往返。相較於先前的 TLS 版本，第一個位元組延遲最多可減少 33%。為您的應用程式啟用 end-to-end。				
WebSockets	是	是	是	是
啟用瀏覽器和伺服器之間的即時、持久性雙向通訊。非常適合 AI 聊天應用程式、多玩家遊戲、協作工作空間和金融交易平台等即時資料饋送。				
記錄和監控				

效能和交付	免費	pro	商業	高級
存取日誌		是	是	是
使用 Amazon CloudWatch Logs 擷取存取詳細的 CloudFront 請求日誌 以了解安全性和交付流量模式，無需額外費用。				
WAF 請求日誌		是	是	是
存取詳細的 AWS WAF 請求日誌，以了解安全性和交付流量模式。Amazon CloudWatch Logs 擷取包含在內，無需額外費用。				

效能和交付	免費	pro	商業	高級
安全性儀表板	是	是	是	是
監控安全事件、調查威脅，並使用視覺化分析立即採取封鎖動作，而無需撰寫安全規則。Pro 和以上版本包含視覺化日誌分析器，可在不查詢日誌的情況下快速了解流量模式。				
DNS				
Amazon Route 53 DNS	是	是	是	是
使用 Route 53 快速、可靠的公有授權 DNS 服務。				
每個託管區域的記錄	50	100	1000	5000
託管區域中允許的 DNS 記錄數目上限。				

效能和交付	免費	pro	商業	高級
DNSSEC	是	是	是	是
保護您的網域免受 DNS 詐騙和 man-in-the-middle 攻擊，攻擊者會攔截 DNS 查詢並將訪客重新導向至仿造網站。透過密碼編譯簽署您的 DNS 記錄來保護 DNS 流量。				
儲存				
Amazon S3 儲存體	5 GB	50 GB	1 TB	5 TB
抵銷 中任何 S3 標準儲存成本的 Amazon S3 儲存額度 AWS 帳戶。S3 不限於 CloudFront 內容或受限於計劃用量限額。				
支援與可靠性				

效能和交付	免費	pro	商業	高級
全年無休的帳戶和帳單支援	是	是	是	是
對帳戶和帳單問題的One-on-one回應。				
如果您有付費的支援計劃，則有資格取得所有固定費率計劃的支援。				
文件和 AWS 支援 論壇	是	是	是	是
存取產品文件、技術文件、最佳實務指南、AWS re:Post 社群論壇和服務運作狀態資訊，以協助您規劃和疑難排解。				

效能和交付	免費	pro	商業	高級
執行時間 SLA			是	是
Amazon CloudFront、Amazon Route 53 和 AWS WAF Amazon CloudWatch 的服務水準協議 (SLA) 提供服務可用性承諾。如果 AWS 不符合相關聯的 SLA 承諾，您將有資格收到服務點數。				

每月用量限額

每個固定費率計劃都包含每月用量額度，專為該方案的最佳效能而設計。您可以隨時在 CloudFront 主控台中追蹤您的用量限額。當您達到限額的 50%、80% 和 100% 時，您也會收到自動電子郵件通知，但通知可能會延遲。

選取每月用量限額在請求和資料傳輸上容納基準流量的計劃。如果您超過限額，則不會產生任何超額費用。這可讓您操作應用程式，而不必擔心意外流量激增或攻擊的成本。如果您超出計劃的功能或基準流量有所變更，請升級至下一層，以存取更多功能並提高每月用量額度。如果您的用量超過 CloudFront 固定費率定價計劃中的限額，AWS 可能會採取適當的動作，其中可能包括降低效能（例如，從較少或更遠的節點提供流量、降低輸送量或限流），或需要變更定價結構。

如果您應用程式的基準用量超過 5 億個請求或每月 50 TB，[請聯絡我們](#)以取得自訂定價。

每個計劃方案的每月用量限額

	免費	pro	商業	高級
要求	1 M	1,000 萬	1.25 公尺	5 億

	免費	pro	商業	高級
資料傳輸	100 GB	50 TB	50 TB	50 TB

Note

封鎖的 DDoS 攻擊和遭封鎖的請求 AWS WAF 永遠不會計入您的用量限額。

根據歷史用量的資格

您的歷史 CloudFront 用量可能會影響您註冊或降級至特定計劃方案的資格。如果您最近的用量超過計劃方案的使用額度，您可能需要選取更符合您工作負載的更高方案。

您的計劃涵蓋的成本

您的計劃涵蓋下列項目的費用：

- 您的 CloudFront 分佈
- 與您的分佈相關聯的 AWS WAF Web ACL
- 分發 CloudFront 存取日誌和相關聯 WAF 日誌的 CloudWatch CloudWatch Logs 擷取
- 連接到分發計劃的 Route 53 託管區域、DNS 記錄和 DNS 查詢

無論是否使用 S3 儲存貯體做為 CloudFront 分佈的原始伺服器，您也會收到 S3 點數來抵銷付款人帳戶中的 S3 標準儲存用量。

Route 53 DNS 管理和您的計劃

如果您將 Route 53 用於 DNS 並將區域連接到您的計劃，您的固定費率計劃可以包含您的 Route 53 託管區域成本。您可以在 CloudFront 分發的管理計劃區段中，將區域連接至您的計劃。當您的區域連接到計劃時，您的計劃會涵蓋託管區域的標準成本，包括每月託管區域費用、DNS 記錄和 DNS 查詢費用，受以下各層的個別限額約束。託管區域必須符合下列要求：

- 存在於與 CloudFront 分佈相同的 AWS 帳戶中
- 維護您計劃方案每個託管區域的允許記錄數目
- 涵蓋 CloudFront 分佈使用的網域

如果您的託管區域未連接到您的計劃，它將保持pay-as-you-go定價，您在其中負責所有標準 Route 53 成本。

了解每月 DNS 查詢額度

當您的託管區域連接到您的計劃時，您會收到：

1. 指向 CloudFront 分佈 [和其他支援的 AWS 服務](#) ALIAS 記錄的 DNS 查詢
2. 其他 DNS 記錄類型的每月額外限額

	免費	pro	商業	高級
每月對 ALIAS 記錄 (CloudFront 和其他支援的 AWS 服務) 的 DNS 查詢	沒有限制	沒有限制	沒有限制	沒有限制
每月額外 DNS 查詢額度	1 M	5 M	20 M	1 億

Note

若要最大化您的計劃利益，請使用 ALIAS 記錄來指向您的 CloudFront 分佈。指向 CloudFront [和其他支援的 AWS 服務](#) ALIAS 記錄不會計入您的每月 DNS 查詢額度。所有其他 DNS 查詢，包括 CloudFront 的 CNAME 記錄，都會計入您的 DNS 查詢額度。

超過 DNS 查詢額度

如果您的 DNS 查詢用量超過計劃的每月限額，AWS 可能會通知您。此時，您可以在 CloudFront 分佈的管理計劃區段中將託管區域與計劃分離，以將託管區域傳回pay-as-you-go定價。如果您在收到此通知後未分離託管區域，AWS 可能會自動將託管區域轉換為pay-as-you-go定價。當託管區域移至pay-as-you-go定價時，您必須負責支付所有標準 Route 53 成本。您的 CloudFront 分佈和所有其他計劃利益保持不變。

使用定價計劃降低整體 AWS 成本

CloudFront 固定費率定價計劃可以透過三種方式降低整體 AWS 成本：

首先，會自動免除 CloudFront 與在 Amazon S3、AWS Application Load Balancer (ALB) 或 Amazon API Gateway 等服務上執行 AWS 的應用程式之間的資料傳輸成本。當您透過 CloudFront 而非直接透過網際網路提供 AWS 應用程式時，您的固定費率計劃會以簡單的每月價格涵蓋應用程式與瀏覽者之間的資料傳輸成本，而無需擔心超額。

其次，CloudFront 透過保護您的應用程式基礎設施並減少到達原始伺服器的請求數量，來降低您的運算和資料庫成本。它提供來自節點或區域節點快取的快取內容、收合重複的請求，並在到達後端服務之前封鎖惡意和不需要的流量。這表示對應用程式伺服器、資料庫和其他根據用量收費 AWS 服務的請求較少，可降低成本。

最後，每個計劃都包含 Amazon S3 Standard 儲存點數，以抵銷的儲存用量 AWS 帳戶。

若要最大限度地節省這些成本，請將 AWS 原始伺服器設定為僅接受來自 CloudFront 的流量。對於 S3，搭配私有儲存貯體使用[原始存取控制 OAC](#)，以授予您指定 CloudFront 分佈的存取權。對於私有子網路中的 Application Load Balancer、Network Load Balancer 和 Amazon EC2 執行個體，[請使用 VPC Origins 限制對指定 CloudFront 分佈的存取](#)。

管理您的固定費率定價計劃

請遵循 CloudFront 主控台內的這些程序來訂閱、升級、降級或取消分佈的定價計劃。

訂閱定價計劃的新分佈

當您建立新的分佈時，您可以訂閱定價計劃。

訂閱定價計劃的新分佈

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後依照步驟建立分佈。
3. 選擇分發的定價計劃。請注意，某些功能不適用於每個定價計劃方案。檢閱每個計劃的功能，並選擇您應用程式所需的定價計劃。
4. 完成[建立分佈](#)的步驟。

訂閱現有的 分佈至定價計劃

當您更新分佈時，您可以訂閱定價計劃。在選擇定價計劃之前，請確定您的分佈組態與您想要的計劃相容。

Tip

如果您目前的分佈使用任何**不支援的功能**，您必須先停用這些功能，才能訂閱定價計劃。這包括停用 Lambda@Edge 或即時存取日誌等功能。

一旦分佈組態相容，您可以在更新分佈時選擇所需的定價計劃。

訂閱現有的 分佈至定價計劃

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後依照步驟更新現有的分佈。
3. 選擇分發的定價計劃。請注意，某些功能不適用於每個定價計劃方案。檢閱每個計劃的功能，並選擇您應用程式所需的定價計劃。
4. 完成步驟以[更新您的分佈](#)。

升級定價計劃

如果您即將到來或已超過每月用量限額，或想要啟用下一層可用的功能，建議您升級計劃。

當您升級至更高的計劃方案時，變更會立即生效。您的價格和用量額度會按比例分配。您的分發和相關聯的資源將可存取新計劃的可用功能和更高的用量限額。

升級定價計劃

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇訂閱現有定價計畫的分佈。
4. 按照提示升級分發的定價計劃。
5. 完成步驟以[更新現有的分佈](#)。

降級定價計劃

如果您不需要現有方案上的其他功能，我們建議您降級至較低的計劃方案。例如，如果您預期應用程式流量較低，可能會降級。

如果您降級至較低的方案，您的帳單變更將在下一個帳單週期開始時生效。

如果您的分佈目前超過計劃的用量額度，一旦用量在所需方案的用量額度內，您就可以降級。為了避免在下一個帳單週期向您收取現有計劃方案的費用，請在月底之前降級。

降級定價計劃

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇訂閱現有定價計畫的分佈。
4. 依照提示來降級分發的定價計劃。如果您有不支援的功能，您必須從分發中移除該功能或資源。
5. 完成步驟以[更新現有的分佈](#)。

取消定價計劃

當您取消定價計劃時，您將維持固定費率價格，直到目前的帳單週期結束為止。然後，您的分佈和所有相關計劃資源將在下一個計費週期開始時切換到pay-as-you-go定價。

若要取消定價計劃

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇訂閱現有定價計畫的分佈。
4. 依照提示取消分發的定價計劃。如果您有不支援的功能，您必須從分發中移除該功能或資源。
5. 完成步驟以[更新現有的分佈](#)。

取消待定計劃變更

如果您降級或取消固定費率定價計劃，您必須等到目前帳單週期結束，變更才會生效。若要保留現有的固定費率定價計劃、升級或再次降級定價計劃，您必須先取消待定計劃變更。

取消待定定價計畫變更

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇訂閱現有定價計畫的分佈。
4. 按照提示取消分發的待定計畫變更。
5. 選擇您想要用於分佈的定價計畫。
6. 完成步驟以更新現有的分佈。

刪除具有定價計畫的分佈

您無法刪除訂閱定價計畫的分佈。您必須先取消定價計畫，然後在目前的帳單週期之後刪除分佈。

刪除具有定價計畫的分佈

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 請依照上述步驟取消分佈的定價計畫。
4. 依照步驟[刪除分佈](#)。

Note

您可以停用訂閱定價計畫的分佈，但您仍需支付該計畫的費用。若要停止您的計畫產生費用，您必須先取消它。

許可

若要檢視或管理 CloudFront 分佈的定價計畫訂閱，您必須擁有必要的許可。如需詳細資訊，請參閱[AWS 受管政策：CloudFrontFullAccess](#)及[AWS 受管政策：CloudFrontReadOnlyAccess](#)。

固定費率定價計畫配額

下表顯示 CloudFront 固定費率定價計畫的配額和限制。

Note

您的 無法增加這些配額 AWS 帳戶。

帳戶層級配額	配額
每個 的定價計劃 AWS 帳戶	100
每個 的免費計劃 AWS 帳戶	3
每個計畫的 Apex 層級網域	1

不支援的功能

您必須先確保停用特定功能並移除關聯，才能將分佈與定價計劃建立關聯。

備註

- 如果您的分佈或帳戶具有任何這些限制，您必須先解決這些限制，才能使用定價計劃。變更分佈後，請等待變更傳播到所有節點。
- 如果您使用的是定價計劃，則必須有與分佈相關聯的 AWS WAF Web ACL。除非您切換到該分佈的 pay-as-you-go 否則無法移除或取消此資源與分佈的關聯。

不支援的功能

如果定價方案的組態包含下列不支援的功能，則您無法訂閱其分佈。您可以停用不支援的功能，並使用替代選項，或為您的分佈保留 pay-as-you-go。

不支援的功能	替代選項	AWS 服務
多租戶分佈	使用 標準分佈 或 pay-as-you-go 定價	CloudFront
持續部署 和 預備分佈	使用 pay-as-you-go 定價	CloudFront

不支援的功能	替代選項	AWS 服務
Anycast IP 清單 組態	使用pay-as-you-go定價	CloudFront
即時存取日誌	使用 標準存取日誌 或pay-as-you-go定價	CloudFront
Lambda@Edge 函數	使用 CloudFront Functions pay-as-you-go定價	CloudFront
目標機器人	使用常見的機器人或pay-as-you-go定價	AWS WAF
CAPTCHA	使用挑戰或pay-as-you-go定價	AWS WAF
合作夥伴受管規則	使用pay-as-you-go定價	AWS WAF
帳戶建立詐騙預防	使用pay-as-you-go定價	AWS WAF
帳戶接管保護	使用pay-as-you-go定價	AWS WAF
規則群組	建立個別規則（規則群組是可套用至 Web ACL 的共用 AWS WAF 規則，類似於 CloudFront 上的政策）	AWS WAF
舊版功能		
ForwardedValues 組態	使用 原始伺服器請求政策	CloudFront
專用 IP/SSL	使用pay-as-you-go定價	CloudFront
欄位層級加密	使用pay-as-you-go定價	CloudFront

不支援的功能	替代選項	AWS 服務
AWS Identity and Access Management (IAM) 伺服器憑證	Use AWS Certificate Manager (ACM) 憑證	CloudFront
原始存取身分 (OAI)	使用 原始存取控制 (OAC)	CloudFront
舊版快取設定	使用 快取政策和原始伺服器請求政策 。	CloudFront

不支援的關聯

如果分佈已與下列任何已與其他分佈相關聯的資源建立關聯，則您無法訂閱定價計劃的分佈。與訂閱定價計劃之分佈相關聯的資源只能用於該分佈。例如，如果您有使用金鑰值存放區的 CloudFront 函數，則無法針對定價計劃中的分佈共用該函數和金鑰值存放區。

- CloudFront Functions
- 與鍵值存放區相關聯的 CloudFront Functions
- AWS WAF Web ACLs

若要訂閱定價計劃的分佈，請移除相關聯的資源，或將其取代為另一個資源。

帳戶層級限制條件

AWS 帳戶 如果符合下列任何條件，則不符合定價計劃的資格：

- 您已達到允許的訂閱數量上限。請參閱 [固定費率定價計劃配額](#)。
- 您的帳戶正在使用 AWS 免費方案。

資源層級限制條件

如果分佈符合下列任一條件，則不符合定價計劃的資格：

- 您的分佈已啟用 AWS Shield Advanced

- 您的分佈已啟用 Web ACL 的 [Firewall Manager Service](#)。Firewall Manager 不會在定價計劃中管理 CloudFront 分發的 WebACL。

可能影響定價計劃的其他功能

固定費率定價計劃可讓您為您的 CloudFront 分佈和上述同時包含在計劃中並與 CloudFront 分佈相關聯的功能支付固定費率。所有其他功能可能會產生額外費用，包括但不限於下列項目：

Route 53

- Route 53 DNSSEC 需 AWS KMS 付費
- Route 53 IP (CIDR) 區塊（每個區塊的前 1,000 個免費 AWS 帳戶）
- Route 53 運作狀態檢查（前 50 個是每個免費 AWS 帳戶）

記錄功能

- Route 53 DNS 查詢日誌、CloudFront Functions 日誌和 CloudFront Connection Function Logs
- AWS WAF 日誌交付至 Amazon S3
- CloudFront 或 AWS WAF 日誌交付至 Amazon Data Firehose
- CloudFront 的其他 CloudWatch 指標
- Parquet 格式的 CloudFront 存取日誌

Note

您的計劃包含 CloudFront 標準日誌（存取日誌）的 Amazon CloudWatch Logs 擷取和 WAF 日誌，無需額外費用。您的計劃不涵蓋所有其他 CloudWatch 成本，例如儲存和查詢。所有其他日誌也會另外計費。

Note

您的計劃包含 Route 53 的公有授權 DNS。當您的 Route 53 託管區域連接到您的定價計劃時，您的計劃會涵蓋託管區域的標準成本，包括每月託管區域費用、DNS 記錄和 DNS 查詢費

用，視每個方案的個別限額而定。Route 53 用量的所有其他成本和未列在上述計畫中的功能，都不屬於您的計畫。

定價計劃與pay-as-you-go定價

固定費率方案和pay-as-you-go定價可根據您的需求提供不同的優勢。使用固定費率計劃時，您支付一個價格，其中包含多個 AWS 服務，例如 CloudFront AWS WAF、Route 53 和 CloudWatch Logs 擷取，即使在流量激增或攻擊期間也不會面臨超額費用。

使用pay-as-you-go定價時，會根據實際用量分別針對每個服務和功能向您收費。雖然這可提供完整的服務選擇和組態彈性，但您的成本可能會根據流量模式每月變化，而且您需要監控多個服務的用量來管理成本。

如果您想要合併每月帳單、簡化的服務組態和內建安全功能，而無需擔心超額費用，則固定費率計劃是理想的選擇。如果您需要完全控制個別服務功能、自訂組態、存取固定費率計劃中未提供的功能，或者如果您預期處理大型、可預測的流量尖峰，Pay-as-you-go定價是更好的選擇。Amazon CloudFront 固定費率定價計劃不得與任何其他優惠、促銷或折扣合併使用。

設定分佈

您建立 Amazon CloudFront 分佈，以告知 CloudFront 交付的內容出自何處，以及如何追蹤及管理內容交付的詳細資訊。

當您建立 CloudFront 分佈時，CloudFront 會根據您的內容原始伺服器類型設定大部分的分佈設定。如需上述預先設定項目的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。或者，您可以選擇手動編輯分佈設定。如需詳細資訊，請參閱[所有分佈設定參考](#)。

下列設定可以進行調整：

- 您的內容原始伺服器：Amazon S3 儲存貯體、AWS Elemental MediaPackage 頻道、AWS Elemental MediaStore 容器、Elastic Load Balancing 負載平衡器或 HTTP 伺服器，CloudFront 會從中取得要分發的檔案。您最多可以針對單一分佈指定 25 個原始伺服器的任意組合。
- 存取權限 - 您希望每個人皆可存取檔案，或者僅限部分使用者存取。
- 安全性 - 您是否要啟用 AWS WAF 保護，並請求使用者使用 HTTPS 存取您的內容。對於多租戶分佈，僅支援 AWS WAF V2 Web 存取控制清單 ACLs)。
- 快取金鑰 - 您想要包括在快取金鑰的值 (如果有的話)。快取金鑰可唯一識別指定分佈快取中的每個檔案。
- 原始伺服器請求設定 - 您是否要 CloudFront 在傳送到原始伺服器的請求中包含 HTTP 標頭、Cookie 或查詢字串。
- 地理限制 - 您是否希望 CloudFront 禁止選取的國家/地區的使用者存取您的內容。
- 日誌 - 您希望 CloudFront 建立標準日誌或顯示檢視器活動的即時存取日誌。

如需詳細資訊，請參閱[所有分佈設定參考](#)。

如需您可以為每個建立的分佈數量上限 AWS 帳戶，請參閱 [分佈的一般配額](#)。您可為每個分佈提供的檔案數量沒有上限。

您可以使用分佈透過 HTTP 或 HTTPS 提供以下內容：

- 使用 HTTP 或 HTTPS 的靜態和動態下載內容，例如 HTML、CSS、JavaScript 和影像檔案。
- 各類格式的隨需視訊，如 Apple HTTP Live Streaming (HLS) 和 Microsoft Smooth Streaming。(對於多租用戶分佈，不支援 Smooth Streaming。) 如需詳細資訊，請參閱[使用 CloudFront 傳遞隨需視訊](#)。

- 即時事件，例如即時會議、即時討論會，或即時音樂會。對於即時串流，您可以使用 CloudFormation 堆疊自動建立分佈。如需詳細資訊，請參閱[使用 CloudFront 和 AWS Media Services 交付影片串流](#)。

下列主題提供 CloudFront 分佈的詳細資訊，以及如何進行設定以符合您的業務需求。如需有關建立分佈的詳細資訊，請參閱 [建立分發](#)。

主題

- [了解多租用戶分佈的運作方式](#)
- [建立分發](#)
- [預先設定的分佈設定參考](#)
- [所有分佈設定參考](#)
- [測試分佈](#)
- [更新分佈](#)
- [標記分佈](#)
- [刪除 分發](#)
- [將各種原始伺服器與 CloudFront 分佈搭配使用](#)
- [為 CloudFront 分佈啟用 IPv6](#)
- [使用 CloudFront 持續部署，安全地測試 CDN 組態變更](#)
- [透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#)
- [使用 WebSockets 搭配 CloudFront 分佈](#)
- [請求用於列入允許清單的 Anycast 靜態 IP](#)
- [使用 gRPC 搭配 CloudFront 分佈](#)

了解多租用戶分佈的運作方式

您可以使用可在多個分佈租用戶間重複使用的設定來建立 CloudFront 多租用戶分佈。透過多租用戶分佈，您可讓 CloudFront 根據您的內容原始伺服器類型設定您的分佈設定。如需上述預先設定項目的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。

使用多租用戶分佈而非標準分佈的優勢包括：

- 減少營運負擔。

- 適用於 Web 管理員和軟體供應商的可重複使用組態，會用於管理多個 Web 應用程式的 CloudFront 分佈，這些 Web 應用程式會將內容交付給最終使用者。
- 增強與其他 的整合 AWS 服務，以大規模提供自動化憑證管理、統一的安全控制和輕鬆的組態控制。
- 在整個實作中維持一致的資源模式。定義必須共用的設定，然後為要覆寫的設定指定自訂項目。
- 可自訂的原始伺服器 and 安全性設定，以滿足分佈租用戶層級的特定需求。
- 將您的分佈租用戶組織到不同的層。例如，如果某些分佈租用戶需要 Origin Shield 而有些不需要，您可以將分佈租用戶分組為不同的多租用戶分佈。
- 在多個網域之間共用通用 DNS 組態。

與標準分佈不同，多租用戶分佈無法直接存取，因為它沒有 CloudFront 路由端點。因此，它必須與連線群組和一或多個分佈租用戶搭配使用。雖然標準分佈有自己的 CloudFront 端點，且可供最終使用者直接存取，但無法用作其他分佈的範本。

如需多租用戶分佈配額的相關資訊，請參閱 [多租用戶配額](#)。

主題

- [運作方式](#)
- [條款](#)
- [不支援的功能](#)
- [分佈租用戶自訂](#)
- [為您的 CloudFront 分佈租用戶請求憑證](#)
- [建立自訂連線群組 \(選用\)](#)
- [移轉至多租用戶分佈](#)

運作方式

在標準分佈中，分佈包含您想要為網站或應用程式啟用的所有設定，例如原始伺服器組態、快取行為和安全設定。如果您想要建立單獨的網站並使用許多相同的設定，則每次都需要建立新的分佈。

CloudFront 的多租用戶分佈有所不同，因為您可以建立一個初始的多租用戶分佈。對於每個新網站，您建立的分佈租用戶會自動繼承其來源分佈的定義值。然後，您可以自訂分佈租用戶的特定設定。

概觀

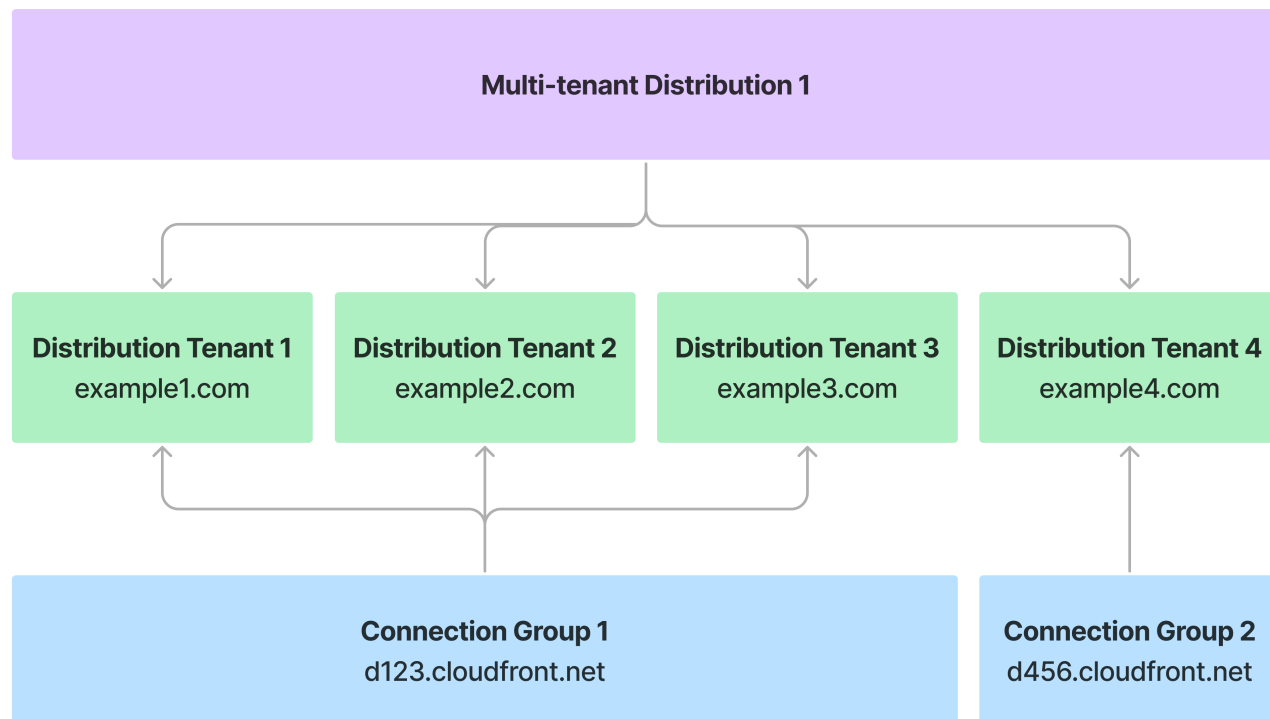
1. 若要開始使用，請先建立多租用戶分佈。CloudFront 會根據內容原始伺服器類型，為您設定分佈設定。您可以自訂 VPC 原始伺服器以外所有原始伺服器的設定。VPC 原始伺服器設定會在 VPC 原始伺服器資源本身上自訂。如需您可自訂之多租用戶分佈設定的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。
 - 您用於多租用戶分佈的 TLS 憑證可由您的分佈租用戶繼承。多租用戶分佈本身無法路由，因此不會有與其相關聯的網域名稱。
2. 根據預設，CloudFront 會為您建立連線群組。連線群組會控制檢視器對內容的請求如何連接到 CloudFront。您可以在連線群組中自訂一些路由設定。

您可以手動建立自己的連線群組來變更此設定。如需詳細資訊，請參閱[建立自訂連線群組 \(選用\)](#)。
3. 然後，您可以建立一或多個分佈租用戶。分佈租用戶是檢視器存取內容的「前門」。每個分佈租用戶都會參考多租用戶分佈，並自動與 CloudFront 為您建立的連線群組建立關聯。分佈租用戶支援個別網域或子網域。
4. 然後，您可以自訂一些分佈租用戶設定，例如虛設網域和原始伺服器路徑。如需詳細資訊，請參閱[分佈租用戶自訂](#)。
5. 最後，您必須更新 DNS 主機中的 DNS 記錄，將流量路由到分佈租用戶。若要執行此操作，請從連線群組取得 CloudFront 端點值，並建立指向 CloudFront 端點的 CNAME 記錄。

Example 範例

下圖示範多租用戶分佈、分佈租用戶和連線群組如何協同運作，為多個網域的檢視器提供內容。

1. 多租用戶分佈會定義每個分佈租用戶的繼承設定。您可以使用多租用戶分佈做為範本。
2. 從多租用戶分佈建立的每個分佈租用戶都有自己的網域。
3. 在您建立多租用戶分佈時，會自動將分佈租用戶加入 CloudFront 為您建立的連線群組中。連線群組控制檢視器請求如何連線到 CloudFront 網路。



如需多租用戶分佈建立說明的詳細資訊，請參閱 [在主控台中建立 CloudFront 分佈](#)。

條款

下列概念說明多租用戶分佈的元件：

多租用戶分佈

藍圖、多租用戶分佈，指定任何分佈租用戶的所有共用組態設定，包括快取行為、安全保護和原始伺服器。多租用戶分佈無法直接提供流量。它們必須與連線群組和分佈租用戶搭配使用。

標準分佈

沒有多租用戶功能的分佈。這些分佈最適合支援單一網站或應用程式。

分佈租用戶

分佈租用戶會繼承多租用戶分佈組態。某些組態設定可以在分佈租用戶層級自訂。分佈租用戶必須擁有有效的 TLS 憑證，只要涵蓋分佈租用戶網域或子網域，就可以從多租用戶分佈進行繼承。

分佈租用戶都必須與一個連線群組建立關聯。CloudFront 會在您建立分佈租用戶時為您建立連接群組，並自動將任何分佈租用戶指派給該連接群組。

多租用

您可以使用多租用戶分佈跨多個網域提供內容，同時共用組態和基礎設施。這種方法允許不同的網域 (稱為租用戶) 從多租用戶分佈共用常見設定，同時維護自己的自訂。

連線群組

提供為檢視器提供內容的 CloudFront 路由端點。您必須將每個分佈租用戶與連線群組建立關聯，以取得您為分佈租用戶網域或子網域建立的 CNAME 記錄對應的 CloudFront 路由端點。連線群組可以在多個分佈租用戶之間共用。連線群組會管理分佈租用戶的路由設定，例如 IPv6 和 Anycast IP 清單設定。

Parameters

預留位置值的索引鍵/值對清單，例如原始伺服器路徑和網域名稱。您可以在多租用戶分佈中定義參數，並在分佈租用戶層級提供這些參數的值。您可以選擇是否需要為分佈租用戶輸入參數值。

如果您未在分佈租用戶中提供選用參數的值，則會使用多租用戶分佈的預設值作為值。

CloudFront 路由端點

連線群組的正式 DNS，例如 `d123.cloudfront.net`。用於分佈租用戶網域或子網域的 CNAME 記錄中。

自訂

您可以自訂分佈租用戶，使其使用與多租用戶分佈不同的設定。對於每個分佈租用戶，您可以指定不同的 AWS WAF Web 存取控制清單 (ACL)、TLS 憑證和地理限制。

不支援的功能

下列功能無法與多租用戶分佈搭配使用。如果您想使用與標準分佈相同的設定建立新的多租用戶分佈，請注意某些設定不可用。

備註

- 目前，AWS Firewall Manager 政策僅適用於您的標準分佈。Firewall Manager 將在未來版本中新增對多租用戶分佈的支援。

- 與標準分佈不同，您可以在分佈租用戶層級指定網域名稱 (別名)。如需詳細資訊，請參閱 [為您的 CloudFront 分佈租用戶請求憑證](#) 和 [CreateDistributionTenant](#) API 操作。

- [持續部署](#)
- [原始存取身分 \(OAI\)](#) – 請改用 [原始存取控制 \(OAC\)](#)。
- [專用 IP 自訂 SSL 支援](#) – 僅支援 sni-only 方法。
- [AWS WAF Classic \(V1\) Web ACL](#) – 僅支援 AWS WAF V2 Web ACLs。
- [標準記錄 \(舊版\)](#)
- [最小 TTL](#)
- [預設 TTL](#)
- [最大 TTL](#)
- [ForwardedValues](#)
- [PriceClass](#)
- [可信簽署者](#)
- [Smooth Streaming](#)
- [AWS Identity and Access Management \(IAM\) 伺服器憑證](#)
- [專用 IP 位址](#)
- [最低通訊協定版本 SSLv3](#)

下列設定無法在多租用戶分佈或分佈租用戶中進行設定。反之，請在連線群組中設定您想要的值。連線群組中關聯的所有分佈租用戶都會使用這些設定。如需詳細資訊，請參閱 [建立自訂連線群組 \(選用\)](#)。

- [啟用 IPv6 \(檢視器請求\)](#)
- [Anycast 靜態 IP 清單](#)

分佈租用戶自訂

使用多租用戶分佈時，您的分佈租用戶會繼承多租用戶分佈組態。不過，您可以在分佈租用戶層級自訂一些設定。

您可以自訂下列項目：

- 參數 – 參數是可用於原始網域或原始路徑的鍵/值對。請參閱 [參數如何與分佈租用戶搭配使用](#)。

- **AWS WAF Web ACL (V2)** – 您可以為分佈租用戶指定單獨的 Web ACL，這會覆寫用於多租用戶分佈的 Web ACL。您也可以針對特定分佈租用戶停用此設定，這表示分佈租用戶不會繼承多租用戶分佈的 Web ACL 保護。如需詳細資訊，請參閱[AWS WAF Web ACL](#)。
- **地理限制** – 您為分佈租用戶指定的地理限制會覆寫多租用戶分佈的任何地理限制。例如，如果您在多租用戶分佈中封鎖德國 (DE)，則所有關聯的分佈租用戶也會封鎖 DE。不過，如果您允許特定分佈租用戶的 DE，該分佈租用戶設定將覆寫多租用戶分佈的設定。如需詳細資訊，請參閱[限制您內容的地理分佈](#)。
- **失效路徑** – 指定您要讓分佈租用戶失效之內容的檔案路徑。如需詳細資訊，請參閱[使檔案失效](#)。
- **自訂 TLS 憑證** – AWS Certificate Manager (ACM) 您為分佈租用戶指定的憑證是多租用戶分佈中提供的憑證的補充。不過，如果多租用戶分佈和分佈租用戶憑證都涵蓋相同的網域，則會使用租用戶憑證。如需詳細資訊，請參閱[為您的 CloudFront 分佈租用戶請求憑證](#)。
- **網域名稱** – 您必須為每個分佈租用戶指定至少一個網域名稱。

參數如何與分佈租用戶搭配使用

參數是可用於預留位置值的鍵/值對。定義您想要在多租用戶分佈中使用的參數，並指定是否需要這些參數。

在多租用戶分佈中定義參數時，您要選擇在分佈租用戶層級中是否需要輸入那些參數。

- 如果您在多租用戶分佈中根據需要定義參數，則必須在分佈租用戶層級輸入這些參數。(它們不會被繼承)。
- 如果不需要這些參數，您可以在分佈租用戶繼承的多租用戶分佈中提供預設值。

您可以使用以下屬性中的參數：

- 原始網域名稱
- 原始伺服器路徑

在多租用戶分佈中，您可以為每個上述屬性定義最多兩個參數。

範例參數

請參閱下列範例，以使用網域名稱和原始路徑的參數。

網域名稱參數

在多租用戶分佈組態中，您可以定義原始網域名稱的參數，如下列範例所示：

Amazon S3

- `{{parameter1}}.amzn-s3-demo-logging-bucket.s3.us-east-1.amazonaws.com`
- `{{parameter1}}-amzn-s3-demo-logging-bucket.s3.us-east-1.amazonaws.com`

自訂原始伺服器

- `{{parameter1}}.lambda-url.us-east-1.on.aws`
- `{{parameter1}}.mediapackagev2.ap-south-1.amazonaws.com`

當您建立分佈租用戶時，請指定要用於 *parameter1* 的值。

```
"Parameters": [  
  {  
    "Name": "parameter1",  
    "Value": "mycompany-website"  
  }  
]
```

使用在多租用戶分佈中指定的先前範例，分佈租用戶的原始網域名稱解析如下：

- `mycompany-website.amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com`
- `mycompany-website-amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com`
- `mycompany-website.lambda-url.us-east-1.on.aws`
- `mycompany-website.mediapackagev2.ap-south-1.amazonaws.com`

原始伺服器路徑參數

同樣地，您可以在多租用戶分佈中定義原始伺服器路徑的參數，如下列範例所示：

- `/{{parameter2}}`
- `/{{parameter2}}/test`
- `/public/{{parameter2}}/test`
- `/search?name={{parameter2}}`

當您建立分佈租用戶時，請指定要用於 *parameter2* 的值。

```
"Parameters": [  
  {  
    "Name": "parameter2",  
    "Value": "myBrand"  
  }  
]
```

使用在多租用戶分佈中指定的先前範例，分佈租用戶的原始伺服器路徑解析如下：

- */myBrand*
- */myBrand/test*
- */public/myBrand/test*
- */search?name=myBrand*

Example 範例

您想要為您的客戶建立多個網站 (租用戶)，而且您需要確保每個分佈租用戶資源都使用正確的值。

1. 您可以建立多租用戶分佈，並包含分佈租用戶組態的兩個參數。
2. 對於原始網域名稱，您可以建立名為 *customer-name* 的參數，並指定這是必要的。您會在 S3 儲存貯體之前輸入參數，使其顯示為：

```
{{customer-name}}.amzn-s3-demo-bucket3.s3.us-east-1.amazonaws.com.
```

3. 對於原始伺服器路徑，您可以建立名為 *my-theme* 的第二個參數，並指定它是選用的，預設值為 *#*。您的原始路徑顯示為：*/{{my-theme}}*
4. 當您建立分佈租用戶時：
 - 對於網域名稱，您必須指定 *customer-name* 的值，因為它在多租用戶分佈中標記為必要。
 - 對於原始路徑，您可以選擇指定 *my-theme* 的值或使用預設值。

為您的 CloudFront 分佈租用戶請求憑證

當您建立分佈租用戶時，租用戶會從多租用戶分佈繼承共用 AWS Certificate Manager (ACM) 憑證。此共用憑證會為與多租用戶分佈相關聯的所有租用戶提供 HTTPS。

當您建立或更新 CloudFront 分佈租用戶以新增網域時，您可以從 ACM 新增受管 CloudFront 憑證。然後 CloudFront 會代表您從 ACM 取得 HTTP 驗證的憑證。您可以將此租用戶層級 ACM 憑證用於自訂網域組態。CloudFront 可簡化續約工作流程，以協助讓憑證保持最新，並且確保內容交付不中斷。

Note

您雖擁有憑證，但它只能與 CloudFront 資源搭配使用，且無法匯出私有金鑰。

您可以在建立或更新分佈租用戶時請求憑證。

主題

- [新增網域和憑證 \(分佈租用戶\)](#)
- [完成網域設定](#)
- [將網域指向 CloudFront](#)
- [網域考量 \(分佈租用戶\)](#)
- [萬用字元網域 \(分佈租用戶\)](#)

新增網域和憑證 (分佈租用戶)

下列程序說明如何新增網域並更新分佈租用戶的憑證。

新增網域和憑證 (分佈租用戶)

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 SaaS 下，選擇分佈租用戶。
3. 搜尋分佈租用戶。使用搜尋列中的下拉式功能表，並且依網域、名稱、分佈 ID、憑證 ID、連線群組 ID 或 Web ACL ID 進行篩選。
4. 選擇分佈租用戶的名稱。
5. 針對網域，選擇管理網域。
6. 針對憑證，選擇您是否要為分佈租用戶建立自訂 TLS 憑證。憑證會驗證您是否獲授權使用網域名稱。憑證必須存在於美國東部 (維吉尼亞北部) 區域。
7. 針對網域，選擇新增網域，然後輸入網域名稱。根據您的網域，以下訊息會顯示在您輸入的網域名稱下。

- 此網域由憑證涵蓋。
- 此網域由憑證涵蓋，待驗證。
- 憑證未涵蓋此網域。(這表示您必須驗證網域擁有權。)

8. 選擇更新分佈租用戶。

在租用戶詳細資訊頁面的網域下，您可以看到下列欄位：

- 網域擁有權 – 網域擁有權的狀態。您必須先使用 TLS 憑證驗證來驗證您的網域擁有權，CloudFront 才能提供內容。
 - DNS 狀態 – 網域的 DNS 記錄必須指向 CloudFront 才能正確路由流量。
9. 如果您的網域擁有權未經過驗證，請在租用戶詳細資訊頁面的網域下，選擇完成網域設定，然後完成下列程序，以將 DNS 記錄指向您的 CloudFront 網域名稱。

完成網域設定

請依照這些程序來驗證您是否擁有分佈租用戶的網域。根據您的網域，選擇下列其中一個程序。

Note

如果您的網域已指向具有 Amazon Route 53 別名記錄的 CloudFront，則必須在網域名稱前面使用 `_cf-challenge`。新增 DNS TXT 記錄。此 TXT 記錄會驗證您的網域名稱是否已連結至 CloudFront。針對每個網域重複此步驟。以下說明如何更新您的 TXT 記錄：

- 記錄名稱：`_cf-challenge.DomainName`
- 記錄類型：TXT
- 記錄值：`CloudFrontRoutingEndpoint`

例如，您的 TXT 記錄可能如下所示：`_cf-challenge.example.com` TXT
`d111111abcdef8.cloudfront.net`

您可以在分佈租用戶詳細資訊頁面主控台中找到您的 CloudFront 路由端點，或使用《Amazon CloudFront API 參考》中的 [ListConnectionGroups](#) API 動作來尋找它。

i Tip

如果您是 SaaS 供應商，而且想要允許憑證發行，而不需要您的客戶 (租用戶) 將 TXT 記錄直接新增至其 DNS，請執行下列動作：

1. 如果您擁有網域 `example-saas-provider.com`，請將子網域指派給租用戶，例如 `customer-123.example-saas-provider.com`
2. 在您的 DNS 中，將 `_cf-challenge.customer-123.example-saas-provider.com` TXT `d1111111abcdef8.cloudfront.net` TXT 記錄新增至您的 DNS 組態。
3. 接下來，您的客戶 (租用戶) 可以更新自己的 DNS 記錄，將他們的網域名稱映射到您提供的子網域。

```
www.customer-domain.com CNAME customer-123.example-saas-provider.com
```

I have existing traffic

如果您的網域無法容忍停機時間，請選取此選項。您必須能夠存取您的原始伺服器/網路伺服器。使用下列程序來驗證網域擁有權。

在擁有現有流量時完成網域設定

1. 針對指定您的 Web 流量，選擇我有現有流量，然後選擇下一步。
2. 針對驗證網域擁有權，選擇下列其中一個選項：
 - 使用現有憑證 – 搜尋現有的 ACM 憑證或輸入涵蓋所列網域的憑證 ARN。
 - 手動檔案上傳 – 選擇您是否具有將檔案上傳到 Web 伺服器的直接存取權。

針對每個網域，建立包含來自字符位置之驗證字符的純文字檔案，並將其上傳至現有伺服器上指定檔案路徑的原始伺服器。此檔案的路徑可能如以下範例所示：`/.well-known/pki-validation/acm_9c2a7b2ec0524d09fa6013efb73ad123.txt`。完成該步驟後，ACM 會驗證字符，然後發出網域的 TLS 憑證。

- HTTP 重新導向 – 如果您沒有將檔案上傳至 Web 伺服器的直接存取權，或者您使用 CDN 或代理服務，請選擇此選項。

針對每個網域，在現有伺服器上建立 301 重新導向。複製重新導向自下的已知路徑，並指向重新導向至下的指定憑證端點。您的重新導向看起來可能與以下範例相似：

```
If the URL matches: example.com/.well-known/pki-validation/  
leabe938a4fe077b31e1ff62b781c123.txt  
Then the settings are:Forwarding URL  
Then 301 Permanent Redirect:To validation.us-east-1.acm-  
validations.aws/123456789012/.well-known/pki-validation/  
leabe938a4fe077b31e1ff62b781c123.txt
```

 Note

您可以選擇檢查憑證狀態，以驗證 ACM 何時發出網域的憑證。

3. 選擇下一步。
4. 完成 [將網域指向 CloudFront](#) 的步驟。

I don't have traffic

如果您要新增網域，請選取此選項。CloudFront 會為您管理憑證驗證。

如果您沒有流量，請完成網域設定

1. 針對指定您的 Web 流量，選擇我還沒有流量。
2. 針對每個網域名稱，完成 [將網域指向 CloudFront](#) 的步驟。
3. 更新每個網域名稱的 DNS 記錄後，請選擇下一步。
4. 等待憑證發出。

 Note

您可以選擇檢查憑證狀態，以驗證 ACM 何時發出網域的憑證。

5. 選擇提交。

將網域指向 CloudFront

更新您的 DNS 記錄，將流量從每個網域路由到 CloudFront 路由端點。您可以有多個網域名稱，但必須全部解析為此端點。

將網域指向至 CloudFront

1. 複製 CloudFront 路由端點值，例如 d1111111abcdef8.cloudfront.net。
2. 更新您的 DNS 記錄，將流量從每個網域路由到 CloudFront 路由端點。
 1. 登入您的網域註冊商或 DNS 提供者管理主控台。
 2. 導覽至網域的 DNS 管理區段。
 - 對於子網域 – 建立 CNAME 記錄。例如：
 - 名稱 – 您的子網域 (例如 www 或 app)
 - 值/目標 – CloudFront 路由端點
 - 記錄類型 – CNAME
 - TTL – 3600 (或任何適合您的使用案例)
 - 對於 apex/root 網域 – 這需要唯一的 DNS 組態，因為無法在根網域或 apex 網域層級使用標準 CNAME 記錄。由於大多數 DNS 供應商不支援 ALIAS 記錄，因此建議您在 Route 53 中建立 ALIAS 記錄。例如：
 - 名稱 – 您的 Apex 網域 (例如 example.com)
 - 記錄類型 – A
 - 別名 – 是
 - 別名目標 – 您的 CloudFront 路由端點
 - 路由政策 – 簡單 (或任何適合您的使用案例)
 3. 確認 DNS 變更已傳播。(這通常發生在 TTL 過期時，有時可能需要 24-48 小時。) 使用 dig 或 nslookup 等工具。

```
dig www.example.com
# Should eventually return a CNAME pointing to your CloudFront routing endpoint
```
3. 返回 CloudFront 主控台，然後選擇提交。當您的網域處於作用中狀態時，CloudFront 會更新網域狀態，以指出您的網域已準備好為流量提供服務。

如需詳細資訊，請參閱您的 DNS 提供者的文件：

- [Cloudflare](#)
- [ClouDNS](#)
- [DNSimple](#)

請求憑證 (分佈租用戶)

- [Gandi.net](#)
- [GoDaddy](#)
- [Google 雲端 DNS](#)
- [名稱表](#)

網域考量 (分佈租用戶)

當網域處於作用中狀態時，網域控制便已建立，且 CloudFront 會回應此網域的所有檢視器請求。一旦啟用，網域就無法停用或變更為非作用中狀態。當網域已在使用時，便無法與另一個 CloudFront 資源建立關聯。若要將網域與另一個分佈建立關聯，請使用 [UpdateDomainAssociation](#) 請求將網域從一個 CloudFront 資源移至另一個資源。

當網域處於非作用中狀態時，CloudFront 不會回應檢視器對網域的請求。當網域處於非作用中狀態時，請注意下列事項：

- 如果您有待定的憑證請求，CloudFront 會回應已知路徑的請求。當請求擱置時，網域便無法與任何其他 CloudFront 資源建立關聯。
- 如果您沒有待定的憑證請求，CloudFront 便不會回應網域的請求。您可以將網域與其他 CloudFront 資源建立關聯。
- 每個分佈租用戶只能有一個待定憑證請求。您必須先取消現有的待定請求，才能為其他網域請求另一個憑證。取消現有的憑證請求不會刪除相關聯的 ACM 憑證。您可使用 ACM API 來刪除該憑證。
- 如果您將新憑證套用至分佈租用戶，這將取消與先前憑證的關聯。您可以重複使用憑證來涵蓋另一個分佈租用戶的網域。

如同 DNS 驗證憑證的續約，當憑證續約成功時，您將會收到通知。不過，您不需要執行任何其他動作。CloudFront 會自動管理網域的憑證續約。

Note

您不需要呼叫 ACM API 操作來建立或更新憑證資源。您可以使用 [CreateDistributionTenant](#) 和 [UpdateDistributionTenant](#) API 操作來指定受管憑證請求的詳細資訊，以管理您的憑證。

萬用字元網域 (分佈租用戶)

在下列情況中，分佈租用戶支援萬用字元網域：

- 當萬用字元包含在從上層多租用戶分佈繼承的共用憑證中時
- 當您為分佈租用戶使用有效的現有自訂 TLS 憑證時

建立自訂連線群組 (選用)

根據預設，CloudFront 會在您建立多租用戶分佈時為您建立連線群組。連線群組會控制檢視器對內容的請求如何連接到 CloudFront。

我們建議您使用預設連線群組。不過，如果您需要隔離企業應用程式或分別管理分佈租用戶群組，您可以選擇建立自訂連線群組。例如，如果分佈租用戶遇到 DDoS 攻擊，您可能需要將分佈租用戶移至單獨的連線群組。如此一來，您就可以保護其他分佈租用戶免受影響。

建立自訂連線群組 (選用)

或者，您可以選擇為分佈租用戶建立自訂連線群組。

建立自訂連線群組 (選用)

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇設定。
3. 開啟連線群組設定。
4. 在導覽窗格中，選擇連線群組，然後選擇建立連線群組。
5. 針對連線群組名稱，輸入連線群組的名稱。建立連線群組後，您就無法更新此名稱。
6. 針對 IPv6，指定您是否要啟用此 IP 通訊協定。如需詳細資訊，請參閱[啟用 IPv6 \(檢視器請求\)](#)。
7. 針對 Anycast 靜態 IP 清單，指定您是否要從一組 IP 位址將流量交付給您的分佈租用戶。如需詳細資訊，請參閱[Anycast 靜態 IP 清單](#)。
8. (選用) 將標籤新增至您的連線群組。
9. 選擇建立連線群組。

連線群組建立後，您可以找到指定的設定，以及 ARN 和端點。

- ARN 看起來會如以下範例：arn:aws:cloudfront::123456789012:connection-group/cg_2uVbA9KeWaADTbKzhj91cKDoM25
- 端點看起來會像下列範例：d1111111abcdef8.cloudfront.net

您可以在建立自訂連線群組之後對其進行編輯或刪除。您必須先刪除所有相關聯的分佈租用戶，才能刪除連線群組。您無法刪除 CloudFront 在您建立多租用戶分佈時為您建立的預設連線群組。

Important

如果您變更分佈租用戶的連線群組，CloudFront 將繼續承載分佈租用戶的流量，但延遲會增加。我們建議您更新分佈租用戶的 DNS 記錄，以使用來自新連線群組的 CloudFront 路由端點。

在您更新 DNS 紀錄之前，CloudFront 會根據目前網站在 DNS 中所指向的路由端點設定進行路由。例如，假設您的預設連線群組不使用 Anycast 靜態 IP，但您的新自訂連線群組會使用。您必須先更新 DNS 記錄，CloudFront 才會將 Anycast 靜態 IP 用於您自訂連線群組中的分佈租用戶。

移轉至多租用戶分佈

如果您有 CloudFront 標準分佈，並且想要移轉到多租用戶分佈，請遵循以下步驟。

從標準分佈移轉到多租用戶分佈

1. 請參閱[不支援的功能](#)。
2. 建立與標準分佈具有相同組態的多租用戶分佈，減去不支援的功能。如需詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。
3. 建立分佈租用戶，並新增您擁有的替代網域名稱。

Warning

請勿使用與標準分佈相關聯的目前網域名稱。請改為新增預留位置網域。稍後您將移動您的網域。如需有關建立分佈租用戶的詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。

4. 為分佈租用戶網域提供現有憑證。這是將涵蓋預留位置網域和您要移動之網域的憑證。
5. 從主控台的分佈租用戶詳細資訊頁面複製 CloudFront 路由端點。或者，您也可以使用《Amazon CloudFront API 參考》中的 [ListConnectionGroups](#) API 動作來尋找它。
6. 若要驗證網域擁有權，請使用底線 (_) 字首建立 DCV TXT 記錄，該字首指向分佈租用戶的 CloudFront 路由端點。如需詳細資訊，請參閱[將網域指向 CloudFront](#)。
7. 當您的變更傳播完成後，請更新您的分佈租用戶，以使用您先前用於標準分佈的網域。

- 主控台 – 如需詳細說明，請參閱 [新增網域和憑證 \(分佈租用戶\)](#)。
- API – 使用《Amazon CloudFront API 參考》中的 [UpdateDomainAssociation](#) API 動作。

Important

這會重設內容的快取金鑰。之後，CloudFront 會開始使用新的快取金鑰快取您的內容。如需詳細資訊，請參閱 [了解快取金鑰](#)。

8. 更新您的 DNS 記錄，將網域指向分佈租用戶的 CloudFront 路由端點。完成此步驟後，您的網域將準備好將流量提供給分佈租用戶。如需詳細資訊，請參閱 [將網域指向 CloudFront](#)。
9. (選用) 在成功將網域移轉至分佈租用戶後，您可以使用涵蓋分佈租用戶網域名稱的不同 CloudFront 受管憑證。若要請求受管憑證，請建立個別的 TXT 記錄來發行憑證，並遵循 [完成網域設定](#) 中的步驟。

建立分發

此主題說明如何使用 CloudFront 主控台建立分佈。

概觀

1. 建立一或多個 Amazon S3 儲存貯體或將 HTTP 伺服器設定為原始伺服器。原始伺服器是您存放內容原始版本的位置。當 CloudFront 取得檔案請求時，它會移至原始伺服器以取得在節點分配的檔案。您可以使用 Amazon S3 儲存貯體和 HTTP 伺服器的任意組合做為您的原始伺服器。
 - 如果您使用的是 Amazon S3，請注意您的儲存貯體名稱必須全部小寫和不能包含空格。
 - 如果您使用的是 Amazon EC2 伺服器或其他自訂原始伺服器，請檢閱 [使用 Amazon EC2 \(或其他自訂原始伺服器\)](#)。
 - 如需您可為分佈建立之原始伺服器數量的目前上限，或是有關請求更高配額的詳細資訊，請參閱 [分佈的一般配額](#)。
2. 將內容上傳到原始伺服器。您使物件可公開讀取，或使用 CloudFront 已簽署的 URL 來限制對您的內容進行存取。

⚠ Important

您負責確保原始伺服器的安全。您必須確保 CloudFront 具有存取伺服器的許可，並且該安全設定可以保護您的內容。

3. 建立您的 CloudFront 分佈：

- 如需在 CloudFront 主控台中建立分佈的詳細程序，請參閱 [在主控台中建立 CloudFront 分佈](#)。
 - 如需使用 CloudFront API 建立分佈的資訊，請參閱《Amazon CloudFront API 參考》中的 [CreateDistribution](#)。
4. (選用) 如果您使用 CloudFront 主控台建立分佈，請為您的分佈建立更多快取行為或原始伺服器。如需行為和來源的詳細資訊，請參閱 [更新多租用戶分佈](#)。
5. 測試您的分佈。如需測試的詳細資訊，請參閱 [測試分佈](#)。
6. 開發您的網站或應用程式，使用 CloudFront 在您在步驟 3 中建立分佈之後所傳回的網域名稱來存取您的內容。例如，如果 CloudFront 傳回 d111111abcdef8.cloudfront.net 作為您分佈的網域名稱，則 Amazon S3 儲存貯體中或 HTTP 伺服器上根目錄中檔案 image.jpg 的 URL 將是 https://d111111abcdef8.cloudfront.net/image.jpg。

如果您在建立您的分佈時指定了一或多個備用網域名稱 (CNAME)，則可以使用自己的網域名稱。在這種情況下，image.jpg 的 URL 可能 https://www.example.com/image.jpg。

注意下列事項：

- 如果您想要使用簽章的 URL 來限制對內容的存取的詳細資訊，請參閱 [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。
- 如果您想要提供壓縮內容的詳細資訊，請參閱 [提供壓縮檔案](#)。
- 如需 Amazon S3 和自訂原始伺服器的 CloudFront 請求和回應行為的資訊，請參閱 [請求和回應行為](#)。

主題

- [在主控台中建立 CloudFront 分佈](#)
- [CloudFront 在主控台中顯示的值](#)
- [其他連結](#)
- [將網域新增至您的 CloudFront 標準分佈](#)

在主控台中建立 CloudFront 分佈

當您建立分佈時，CloudFront 會根據您的內容原始伺服器類型設定分佈設定。如需上述預先設定項目的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。您也可以使用可在多個分佈租用戶間重複使用的設定來建立多租用戶分佈。如需詳細資訊，請參閱 [了解多租用戶分佈的運作方式](#)。或者，您可以手動設定自己的分佈設定。

Multi-tenant

建立多租用戶分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後選擇建立分佈。
3. 選擇多租用戶架構，下一步。
4. 輸入多租用戶分佈的分佈名稱。該名稱將顯示為 Name 鍵值做為標籤。您可以稍後變更此值。您最多可以為多租用戶分佈新增 50 個標籤。如需詳細資訊，請參閱 [標記分佈](#)。
5. (選用) 對於萬用字元憑證，請選擇將涵蓋根網域下所有子網域的 AWS Certificate Manager (ACM) 憑證，例如 **.example.com*。憑證必須位於美國東部 (維吉尼亞北部) 區域。
6. 選擇下一步。
7. 在指定原始伺服器頁面上，選取 CloudFront 將從中取得內容的原始伺服器類型。CloudFront 將針對多租用戶分佈使用該原始伺服器類型的建議設定。如需建議設定的詳細資訊，請參閱 [預先設定的分佈設定參考](#)。
8. 對於原始伺服器，在您選取的原始伺服器類型下，選擇或輸入要使用的原始伺服器。
9. 對於原始伺服器路徑，輸入斜線 (/) 字元，後面接著原始伺服器路徑。
10. (選用) 若要新增參數，請選擇為原始伺服器網域名稱或原始伺服器路徑插入參數。您可以為每個欄位輸入最多兩個參數。
 - a. 選擇建立新參數。
 - b. 在建立新參數對話方塊中，針對參數名稱輸入參數的唯一名稱，並選擇性輸入描述。
 - c. 針對必要參數，選取核取方塊，讓此參數值在分佈租用戶層級為必要。如果不需要，請輸入分佈租用戶將繼承的預設值。
 - d. 選擇 Create parameter (建立參數)。此參數會出現在對應的欄位中。
11. 針對選項，請選擇下列其中一種選項：

- 使用建議的原始伺服器設定 – 針對您選取的原始伺服器類型使用預設建議的快取和原始伺服器設定。
- 自訂原始伺服器設定 – 自訂快取和原始伺服器設定。如果您選擇此選項，請指定出現的您自己的值。

12. 選擇下一步。

13. 在啟用安全性保護頁面上，選擇是否啟用 AWS WAF 安全性保護。您可以稍後針對特定分佈租用戶自訂 Web ACL。如需詳細資訊，請參閱[為新分佈啟用 AWS WAF](#)。

14. 選擇下一步、建立分佈。

15. 在分佈頁面上，您的多租用戶分佈會出現在資源清單中。您可以選擇所有分佈下拉式清單，依標準分佈或多租用戶分佈進行篩選。您也可以選擇要依標準或多租用戶分佈篩選的類型欄。

根據預設，CloudFront 會為您建立連線群組。連線群組會控制檢視器對內容的請求如何連接到 CloudFront。您可以在連線群組中自訂一些路由設定。如需詳細資訊，請參閱[了解多租用戶分佈的運作方式](#)。

您可以使用多租用戶分佈作為範本來建立其他分佈租用戶。

建立分佈租用戶

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，執行下列任一操作：
 - 選擇分佈，選擇多租用戶分佈，然後選擇建立租用戶。
 - 選擇分佈租用戶，然後選擇 建立租用戶。
3. 針對分佈租用戶名稱，輸入名稱。名稱在您的 中必須是唯一的 AWS 帳戶，而且在建立之後無法變更。
4. 對於範本分佈，從清單中選擇多租用戶分佈 ID。
5. 針對管理標籤，請為分佈租用戶新增最多 50 個鍵/值對。如需詳細資訊，請參閱[標記分佈](#)。
6. 選擇下一步。
7. 在新增網域頁面上，針對憑證，選擇您的分佈租用戶是否想要自訂 TLS 憑證。憑證會驗證您是否獲授權使用網域名稱。憑證必須存在於美國東部 (維吉尼亞北部) 區域。
8. 針對網域，輸入您的自訂網域名稱。

 Note

如果您輸入的網域名稱未涵蓋在憑證內，則需要驗證您擁有該網域。您現在仍然可以建立分佈租用戶，並在稍後驗證網域擁有權。如需詳細資訊，請參閱[為您的 CloudFront 分佈租用戶請求憑證](#)。

9. 選擇下一步。
10. 在定義參數頁面中，您在多租用戶散佈中所指定的參數會顯示出來。針對必要參數，輸入參數名稱旁的值，並儲存您的變更。
11. 若要新增另一個參數，請選擇新增參數，然後輸入名稱和值。
12. 選擇下一步。
13. (選用) 針對安全性自訂，如果您選擇覆寫分佈設定，請為您的使用案例選取選項。
14. (選用) 對於地理限制自訂，如果您選擇覆寫分佈設定，請為分佈租用戶選取適當的限制類型和國家/地區。如需詳細資訊，請參閱[限制您內容的地理分佈](#)。
15. 選擇下一步。
16. 選擇建立分佈租用戶。

您可以在分佈租用戶頁面上找到所有分佈租用戶。您可以依以下項目篩選：

關聯

- 分佈 ID
- 憑證 ID
- 連線群組 ID
- Web ACL ID

Properties

- Name
- 網域

您可以編輯分佈租用戶來自訂特定設定。如需詳細資訊，請參閱[分佈租用戶自訂](#)。

Standard

建立標準分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後選擇建立分佈。
3. 輸入標準分佈的分佈名稱。名稱將顯示為 Name 鍵值做為標籤。您可以稍後變更此值。您最多可以為標準分佈新增 50 個標籤。如需詳細資訊，請參閱[標記分佈](#)。
4. 選擇單一網站或應用程式、下一步。
5. (選用) 對於網域設定，輸入已在 中向 Route 53 註冊的網域 AWS 帳戶，或註冊新的網域。完成設定步驟。
 - 若您的網域使用的 DNS 提供者不是 Route 53，您仍可新增該網域，但必須在建立散佈之後再進行。立即略過網域設定，以繼續建立分佈。您稍後必須手動設定網域和 TLS 憑證。如需詳細資訊，請參閱[將網域新增至您的 CloudFront 標準分佈](#)。
6. 選擇下一步。
7. 在指定原始伺服器頁面上，選取 CloudFront 將從中取得內容的原始伺服器類型。CloudFront 會針對分佈的原始伺服器類型使用建議的設定。如需建議設定的詳細資訊，請參閱[預先設定的分佈設定參考](#)。
8. 針對原始伺服器，選擇或輸入您的原始伺服器。
9. 針對設定，請選擇下列其中一種選項：
 - 使用建議的原始伺服器設定 – 針對您選取的原始伺服器類型使用預設建議的快取和原始伺服器設定。
 - 自訂原始伺服器設定 – 自訂快取和原始伺服器設定。如果您選擇此選項，請指定您自己的值。
10. 選擇下一步。
11. 在啟用安全性保護頁面上，選擇是否啟用 AWS WAF 安全性保護。
12. 選擇下一步。
13. (選用) 如果您為網域使用 Route 53，您將會看到 TLS 憑證頁面。如果 CloudFront 在 AWS 帳戶的 中找不到網域的現有 AWS Certificate Manager (ACM) 憑證us-east-1 AWS 區域，您可以選擇自動建立憑證或手動建立憑證。建立憑證之後，請選擇下一步。
14. 檢閱您的分佈詳細資訊，然後選擇建立分佈。

15. 當 CloudFront 建立您的分佈後，該分佈在狀態欄中的值會從部署中變更為該分佈完成部署的日期與時間。

CloudFront 指派到您的分佈的網域名稱，將顯示在分佈清單中。(它也顯示在所選用分佈的 General (一般) 索引標籤上。)

 Tip

您可以請依照 [透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#) 中的步驟使用備用網域名稱，而不是由 CloudFront 名稱指派名稱給您。

16. 當部署分佈時，請確認您可以使用新 CloudFront URL (d1111111abcdef8.cloudfront.net) 或 CNAME 存取您的內容。如需詳細資訊，請參閱[測試分佈](#)。
17. 當您準備好將流量傳送至分佈時，請務必更新您的 DNS 記錄以指向 CloudFront。如需詳細資訊，請參閱[將網域指向至 CloudFront \(標準分佈\)](#)。

CloudFront 在主控台中顯示的值

當您建立新的分佈或更新現有的分佈，CloudFront 會在 CloudFront 主控台顯示以下資訊。

 Note

作用中的信任簽署者，AWS 帳戶 具有作用中 CloudFront 金鑰對且可用來建立有效簽章 URLs 的，目前在 CloudFront 主控台中看不到。

分佈 ID

當您使用 CloudFront API 對分發執行動作時，將透過分發 ID 指定欲使用的分發，例如 EDFDVBD6EXAMPLE。分佈的分佈 ID 不得變更。

部署和狀態

部署分佈時，您會在上次修改的資料欄下看到部署中狀態。等待分佈完成部署，並確認狀態欄顯示已啟用。如需詳細資訊，請參閱[分佈狀態](#)。

上次修改

上次修改的分佈日期和時間，使用 ISO 8601 格式，例如 2012-05-19T19:37:58Z。如需詳細資訊，請參閱<https://www.w3.org/TR/NOTE-datetime>。

網域名稱

您在指向物件的連結中使用分佈的網域名稱。例如，如果分佈的網域名稱是 `d111111abcdef8.cloudfront.net`，則 `/images/image.jpg` 的連結將是 `https://d111111abcdef8.cloudfront.net/images/image.jpg`。您無法為您的分佈變更 CloudFront 網域名稱。如需連結至您物件的 CloudFront URL 的詳細資訊，請參閱[自訂 CloudFront 中檔案的 URL 格式](#)。

如果您指定的一或多個備用網域名稱 (CNAME)，則可以使用自己的網域名稱來連結到您的物件，而不使用 CloudFront 網域名稱。如需 CNAME 的詳細資訊，請參閱[備用網域名稱 \(CNAME\)](#)。

Note

CloudFront 網域名稱是唯一的。您的分佈的網域名稱從未用於之前的分佈，並且未來將永遠不會重複使用於另一個分佈。

其他連結

如需有關建立分佈的詳細資訊，請參閱以下連結。

- 若要了解如何建立使用 Amazon Simple Storage Service (Amazon S3) 儲存貯體原始伺服器搭配原始存取控制 (OAC) 的分佈，請參閱 [開始使用 CloudFront 標準分佈](#)。
- 如需使用 CloudFront API 建立分佈的資訊，請參閱《Amazon CloudFront API 參考》中的 [CreateDistribution](#)。
- 如需更新分佈的相關資訊 (例如，將快取行為新增至標準分佈，或自訂分佈租用戶)，請參閱 [更新分佈](#)。
- 若要查看您可為每個 AWS 帳戶建立之分佈數量的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱 [分佈的一般配額](#)。

將網域新增至您的 CloudFront 標準分佈

建立新的 CloudFront 標準分佈之後，您可以將網域新增至該分佈。或者，您可以在建立標準分佈時為標準分佈設定 Amazon Route 53 網域。如需詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。

將網域新增至您的現有標準分佈

將網域新增至您的標準分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後選擇分佈 ID。
3. 在設定、替代網域名稱下，選擇新增網域。
4. 輸入最多五個要服務的網域。
5. 選擇下一步。
6. 對於 TLS 憑證，如果 CloudFront 在 AWS 帳戶的中找不到網域的現有 AWS Certificate Manager (ACM) 憑證us-east-1 AWS 區域，您可以建立一個憑證。
 - 如果您使用的是 Amazon Route 53 (Route 53)，CloudFront 會自動為您建立憑證。
7. 佈建憑證時，您必須使用 DNS 提供者更新您的 DNS 記錄，以證明網域擁有權。然後，選擇驗證憑證。如需詳細資訊，請參閱[將網域指向至 CloudFront \(標準分佈\)](#)。
 - 如果您使用的是 Route 53，CloudFront 會為您更新 DNS 記錄。
8. 選擇下一步。
9. 檢閱您的變更，然後選擇新增網域。
10. 將流量傳送到分佈之前，請務必更新您的 DNS 記錄以指向 CloudFront。如需詳細資訊，請在分佈詳細資訊頁面的設定區段中選擇路由網域到 CloudFront。
 - 如果您使用的是 Route 53，您可以讓 CloudFront 自動為您設定 DNS 路由。

將網域指向至 CloudFront (標準分佈)

更新您的 DNS 記錄，將流量從每個網域路由到 CloudFront 主機名稱。您可以有多個網域名稱，但必須全部解析為此主機名稱。

將網域指向至 CloudFront

1. 複製 CloudFront 主機名稱值，例如 d111111abcdef8.cloudfront.net。
2. 更新您的 DNS 記錄，將流量從每個網域路由到 CloudFront 主機名稱。
 1. 登入您的網域註冊商或 DNS 提供者管理主控台。
 2. 導覽至網域的 DNS 管理區段。
 - 對於子網域 – 建立 CNAME 記錄。例如：
 - 名稱 – 您的子網域 (例如 www 或 app)
 - 值/目標 – 您的 CloudFront 主機名稱
 - 記錄類型 – CNAME
 - TTL – 3600 (或任何適合您的使用案例)
 - 對於 apex/root 網域 – 這需要唯一的 DNS 組態，因為無法在根網域或 apex 網域層級使用標準 CNAME 記錄。由於大多數 DNS 供應商不支援 ALIAS 記錄，因此建議您在 Route 53 中建立 ALIAS 記錄。例如：
 - 名稱 – 您的 Apex 網域 (例如 example.com)
 - 記錄類型 – A
 - 別名 – 是
 - 別名目標 – 您的 CloudFront 主機名稱
 - 路由政策 – 簡單 (或任何適合您的使用案例)
 3. 確認 DNS 變更已傳播。(這通常發生在 TTL 過期時，有時可能需要 24-48 小時。) 使用 dig 或 nslookup 等工具。

```
dig www.example.com
# Should eventually return a CNAME pointing to your CloudFront hostname
```

3. 返回 CloudFront 主控台，然後選擇提交。當您的網域處於作用中狀態時，CloudFront 會更新網域狀態，以指出您的網域已準備好為流量提供服務。

如需詳細資訊，請參閱您的 DNS 提供者的文件：

- [Cloudflare](#)
- [ClouDNS](#)
- [DNSimple](#)

- [Gandi.net](#)
- [GoDaddy](#)
- [Google 雲端 DNS](#)
- [名稱表](#)

預先設定的分佈設定參考

當您建立 CloudFront 分佈時，CloudFront 會根據您的內容原始伺服器類型設定大部分的分佈設定。或者，您可以選擇手動編輯分佈設定。如需詳細資訊，請參閱[所有分佈設定參考](#)。

下列各節說明分佈的預設預先組態設定，以及您可以自訂的設定。

Amazon S3 原始伺服器

以下是 CloudFront 在多租用戶分佈中為您的 Amazon S3 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 原始存取控制 (僅限主控台) – CloudFront 會為您設定此項目。CloudFront 會嘗試為標準分佈和多租用戶分佈新增 S3 儲存貯體政策，而原始網域中不使用任何參數。
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3

以下是 CloudFront 在多租用戶分佈中為您的 Amazon S3 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD
- 限制檢視器存取 – 否
- 快取政策 – CachingOptimized
- 原始伺服器請求政策 – 無

- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

以下是您可以在多租用戶分佈中為 Amazon S3 原始伺服器自訂的設定。

可自訂的設定

- S3 存取 – CloudFront 會根據您的 S3 儲存貯體設定為您設定：
 - 如果您的儲存貯體為公有 – 不需要原始存取控制 (OAC) 政策。
 - 如果您的儲存貯體是私有 – 您可以選擇或建立要使用的 OAC 政策。
- 啟用 Origin Shield – 否
- 自動壓縮物件 – 是
 - 如果您選擇是，則會使用 CachingOptimized 快取政策。
 - 如果您選擇否，則會使用 CachingOptimizedForUncompressedObjects 快取政策。

API Gateway 原始伺服器

以下是 CloudFront 在多租用戶分佈中為您的 API Gateway 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 通訊協定 – 僅限 HTTPS
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始伺服器路徑 – 無
- 原始存取控制 (僅限主控台) – CloudFront 會為您設定此項目
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30

- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為您的 API Gateway 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取政策 – CachingDisabled (可能的值：UseOriginCacheControlHeaders、UseOriginCacheControlHeaders-QueryStrings)
- 原始伺服器請求政策 – AllViewerExceptHostHeader (可能值：AllViewer、AllViewerandCloudFrontHeaders-2022-06)
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

以下是您可以在多租用戶分佈中為 API Gateway 原始伺服器自訂的設定。

可自訂的設定

- 啟用 Origin Shield – (預設：否)
- 自動壓縮物件 – (預設：是)

自訂原始伺服器和 EC2 執行個體

以下是 CloudFront 在多租用戶分佈中為您的自訂原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 通訊協定 – 比對檢視器
- HTTP 連接埠 – 80
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始伺服器路徑 – 無
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30
- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為您的自訂原始伺服器和 EC2 執行個體預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取政策 – UseOriginCacheControlHeaders (可能的值：UseOriginCacheControlHeaders-QueryStrings、CachingDisabled、CacheOptimized、CachingOptimizedForUncompressedObjects)
- 原始伺服器請求政策 – AllViewer (可能的值：AllViewerExceptHostHeader、AllViewerandCloudFrontHeaders-2022-06)
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

以下是您可以在多租用戶分佈中為自訂原始伺服器 and EC2 執行個體自訂的設定。

可自訂的設定

- 啟用 Origin Shield – (預設：否)
- 自動壓縮物件 – (預設：是)
- 快取 – (預設：Cache by Default)
 - 如果選取 Cache by Default，則會使用 UseOriginCacheControlHeaders 快取政策。
 - 如果選取 Do Not Cache by Default，則會使用 CachingDisabled 快取政策。
- 在快取中包含查詢字串 – (預設：是，如果 Cache by Default 已選取)
 - 如果已選取 Do Not Cache by Default，然後選擇在快取中包含查詢字串，則會使用 UseOriginCacheControlHeaders-QueryString 快取政策。

ELB 原始伺服器

以下是 CloudFront 在多租用戶分佈中為您的 ELB 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 通訊協定 – 僅限 HTTPS
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始伺服器路徑 – 無
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30
- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為您的 ELB 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是

- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取 – (預設 : Cache by Default)
 - 如果選取 Cache by Default , 則會使用 UseOriginCacheControlHeaders 快取政策。
 - 如果選取 Do Not Cache by Default , 則會使用 CachingDisabled 快取政策。
- 在快取中包含查詢字串 – (預設 : 是 , 如果 Cache by Default 已選取)
 - 如果已選取 Do Not Cache by Default , 然後選擇在快取中包含查詢字串 , 則會使用 UseOriginCacheControlHeaders-QueryStrings 快取政策。
- 原始伺服器請求政策 – All Viewer (可能值 : AllViewerExceptHostHeader、AllViewerandCloudFrontHeaders-2022-06)
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

以下是您可以在多租用戶分佈中為 ELB 原始伺服器自訂的設定。

可自訂的設定

- 啟用 Origin Shield – (預設 : 否)
- 自動壓縮物件 – (預設 : 是)
- 快取 – (預設 : Cache by Default)
 - 如果選取 Cache by Default , 則會使用 UseOriginCacheControlHeaders 快取政策。
 - 如果選取 Do Not Cache by Default , 則會使用 CachingDisabled 快取政策。
- 在快取中包含查詢字串 – (預設 : 是 , 如果 Cache by Default 已選取)
 - 如果已選取 Do Not Cache by Default , 然後選擇在快取中包含查詢字串 , 則會使用 UseOriginCacheControlHeaders-QueryStrings 快取政策。

MediaPackage v1 原始伺服器

以下是 CloudFront 在多租用戶分佈中為 MediaPackage v1 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 通訊協定 – 僅限 HTTPS
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始路徑 – 您可以輸入 MediaPackage URL 來提供此資訊。
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30
- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為 MediaPackage v1 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取政策 – Elemental-MediaPackage
- 原始伺服器請求政策 – 無
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

MediaPackage v2 原始伺服器

以下是 CloudFront 在多租用戶分佈中為 MediaPackage v2 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 原始存取控制 – CloudFront 會為您設定此項目並新增政策
- 通訊協定 – 僅限 HTTPS
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始伺服器路徑 – 無
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30
- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為 MediaPackage v2 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取政策 – Elemental-MediaPackage
- 原始伺服器請求政策 – 無
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

MediaTailor 原始伺服器

以下是 CloudFront 在多租用戶分佈中為您的 MediaTailor 原始伺服器預先設定的原始伺服器設定。

原始伺服器設定 (預先設定)

- 通訊協定 – 僅限 HTTPS
- HTTPS 連接埠 – 443
- 最低來源 SSL 通訊協定 – TLSv1.2
- 原始路徑 – 您可以輸入 MediaPackage URL 來提供此資訊。
- 新增自訂標頭 – 無
- 啟用 Origin Shield – 否
- 連線嘗試 – 3
- 回應逾時 – 30
- 保持連線逾時 – 5

以下是 CloudFront 在多租用戶分佈中為您的 MediaTailor 原始伺服器預先設定的快取設定。

快取設定 (預先設定)

- 自動壓縮物件 – 是
- 檢視器通訊協定政策 – 重新導向至 HTTPS
- 允許的 HTTP 方法 – GET, HEAD
- 快取 HTTP 方法 – 否
- 允許透過 HTTP/2 進行 gRPC 請求 – 否
- 限制檢視器存取 – 否
- 快取政策 – 無
- 原始伺服器請求政策 – Elemental-MediaTailor-PersonalizedManifests
- 回應標頭政策 – 無
- Smooth Streaming – 否
- 欄位層級加密 – 否
- 啟用即時存取日誌 – 否
- 函數 – 否

所有分佈設定參考

建立或更新分佈時，您可以選擇手動編輯 CloudFront 分佈設定。以下是您可以編輯的設定。

不過，CloudFront 會根據您的內容原始伺服器類型，為您設定大多數分佈設定。如需詳細資訊，請參閱[預先設定的分佈設定參考](#)。

如需使用 CloudFront 主控台建立或更新分發的相關資訊，請參閱[the section called “建立分發”](#)或[the section called “更新分佈”](#)。

主題

- [原始設定](#)
- [快取行為設定](#)
- [分佈設定](#)
- [自訂錯誤頁面和錯誤快取](#)
- [地理限制](#)

原始設定

使用 CloudFront 主控台建立或更新分佈時，您可以提供有關一或多個位置 (稱為原始伺服器) 的資訊，您可以在其中存放 Web 內容的原始版本。CloudFront 從您的原始伺服器原始伺服器取得 Web 內容，並透過全球網路邊緣伺服器將其提供給檢視器。

如需您可為分佈建立之原始伺服器數量的目前上限，或是有關請求更高配額的詳細資訊，請參閱 [the section called “分佈的一般配額”](#)。

如果您想要刪除原始來源，則必須先編輯或刪除與該原始來源相關聯的快取行為。

Important

如果您要刪除原始來源，請確認之前由該原始來源提供的檔案，可在另一個原始來源中使用，而且您的快取行為現在正將這些檔案的請求，路由傳送到新的原始來源。

當您建立或更新分佈時，請為每個原始來源指定以下值。

主題

- [原始網域](#)
- [通訊協定 \(僅限自訂原始伺服器\)](#)
- [原始伺服器路徑](#)
- [Name](#)
- [原始存取 \(僅限 Amazon S3 原始伺服器\)](#)
- [新增自訂標頭](#)
- [啟用 Origin Shield](#)
- [連線嘗試](#)
- [連線逾時。](#)
- [回應逾時](#)
- [回應完成逾時](#)
- [保持連線逾時 \(僅限自訂與 VPC 原始伺服器\)](#)
- [回應和保持連線逾時配額](#)

原始網域

原始伺服器網域是 CloudFront 取得原始伺服器物件資源之 DNS 網域名稱，例如 Amazon S3 儲存貯體或 HTTP 伺服器。例如：

- Amazon S3 儲存貯體 – *amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com*

Note

如果您最近建立了 S3 儲存貯體，CloudFront 分佈可能會傳回 HTTP 307 Temporary Redirect 回應 (最多需要 24 小時)。最多可能需要 24 小時的時間，才能將 S3 儲存貯體名稱傳播到所有 AWS 區域。傳播完成後，分佈會自動停止傳送這些重新引導回應；您不需要執行任何動作。如需詳細資訊，請參閱[為何會收到來自 Amazon S3 的 HTTP 307 暫時重新引導回應？](#)和[暫時請求重新引導](#)。

- 設定為網站的 Amazon S3 儲存貯體 – *amzn-s3-demo-bucket.s3-website.us-west-2.amazonaws.com*
- MediaStore 容器 – *examplemediastore.data.mediastore.us-west-1.amazonaws.com*
- MediaPackage 端點 – *examplemediapackage.mediapackage.us-west-1.amazonaws.com*
- Amazon EC2 執行個體 – *ec2-203-0-113-25.compute-1.amazonaws.com*

- ELB 負載平衡器 – *example-load-balancer-1234567890.us-west-2.elb.amazonaws.com*
- 您自己的 Web 伺服器 – *www.example.com*

在 Origin domain (原始網域) 欄位中選擇網域名稱，或是輸入名稱。來自選擇加入區域的資源必須手動輸入。網域名稱不區分大小寫。您的原始伺服器網域必須具有可公開解析的 DNS 名稱，透過網際網路將請求從用戶端路由到目標。

若您將 CloudFront 設定為透過 HTTPS 連線至原始伺服器，則憑證中的其中一個網域名稱必須與您在原始伺服器網域名稱中指定的網域名稱相符。如果沒有網域名稱相符，CloudFront 會將 HTTP 狀態碼 502 (無效的閘道) 傳回到檢視器。如需詳細資訊，請參閱[CloudFront 分佈和憑證中的網域名稱](#)及[CloudFront 與自訂原始伺服器間的 SSL/TLS 溝通失敗](#)。

 Note

若您使用的原始伺服器請求政策會將檢視器的主機標頭轉送至原始伺服器，則該原始伺服器必須回應一個與檢視器主機標頭相符的憑證。如需詳細資訊，請參閱[新增 CloudFront 請求標頭](#)。

如果您的原始伺服器是 Amazon S3 儲存貯體，請注意下列事項：

- 若儲存貯體已設為網站，請輸入您儲存貯體的 Amazon S3 靜態網站託管端點；請不要從 Origin domain (原始網域) 欄位中的清單選取儲存貯體名稱。靜態網站託管端點顯示在 Amazon S3 主控台中，在 Static website hosting (靜態網站託管) 下的 Properties (屬性) 頁面上。如需詳細資訊，請參閱[the section called “使用設定為網站端點的 Amazon S3 儲存貯體”](#)。
- 如果您為儲存貯體配置了 Amazon S3 Transfer Acceleration，請不要為 Origin domain (原始網域) 指定 s3-accelerate 端點。
- 如果您使用不同 AWS 帳戶的儲存貯體，而且儲存貯體未設定為網站，請使用下列格式輸入名稱：

bucket-name.s3.region.amazonaws.com

如果您的儲存貯體位於美國區域，而且您希望 Amazon S3 將請求路由到維吉尼亞北部的一個設施，請使用以下格式：

bucket-name.s3.us-east-1.amazonaws.com

- 除非使用 CloudFront 原始存取控制來保護 Amazon S3 中的內容，否則這些檔案必須是可公開讀取。如需存取控制的詳細資訊，請參閱 [the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

⚠ Important

如果該原始伺服器是 Amazon S3 儲存貯體時，則儲存貯體名稱必須符合 DNS 命名請求。如需詳細資訊，請參閱 [Amazon Simple Storage Service 使用者指南](#) 中的儲存貯體限制與局限。

當您變更原始伺服器的 Origin domain (原始網域) 的值時，CloudFront 會立即開始將變更複製到 CloudFront 節點。在特定節點更新分佈組態之前，CloudFront 會繼續將請求轉送至先前原始伺服器。只要在該節點更新分佈組態，CloudFront 就會開始將請求轉送到新的原始伺服器。

變更原始伺服器不需要使用 CloudFront 將來自新原始伺服器的物件來重新填入邊緣快取。只要您的應用程式中的檢視器請求尚未變更，則 CloudFront 會繼續為已存在於邊緣快取中的物件提供服務，直到每個物件上的 TTL 到期，或者直到很少未被請求的物件被逐出。

通訊協定 (僅限自訂原始伺服器)

i Note

這僅適用於自訂原始伺服器。

在從原始伺服器擷取物件時，您想要 CloudFront 使用的通訊協定政策。

請選擇下列其中一個值：

- HTTP only (僅限 HTTP)：CloudFront 只會使用 HTTP 來存取原始伺服器。

⚠ Important

當原始伺服器是 Amazon S3 靜態網站託管端點時，HTTP only (僅限 HTTP) 是預設設定，因為 Amazon S3 不支援 HTTPS 連線的靜態網站託管端點。CloudFront 主控台不支援變更 Amazon S3 靜態網站託管端點的此設定。

- HTTPS only (僅限 HTTPS)：CloudFront 僅使用 HTTPS 存取原始伺服器。
- Match viewer (比對檢視器)：CloudFront 使用 HTTP 或 HTTPS 與您的原始伺服器通訊，這取決於檢視器請求的通訊協定。請注意 CloudFront 僅會在檢視器使用 HTTP 和 HTTPS 協定發出請求時，快取物件一次。

⚠ Important

對於 CloudFront 轉送至此原始伺服器的 HTTPS 檢視器請求，原始伺服器上的 SSL/TLS 憑證中的網域名稱之一，必須符合您為 Origin domain (原始網域) 指定的網域名稱。否則，CloudFront 將使用 HTTP 狀態碼 502 (無效閘道) 來回應檢視器請求，而非傳回請求的物件。如需詳細資訊，請參閱[the section called “與 CloudFront 搭配使用 SSL/TLS 憑證的要求”](#)。

主題

- [HTTP 連接埠](#)
- [HTTPS 連接埠](#)
- [最低來源 SSL 通訊協定](#)

HTTP 連接埠**i Note**

這僅適用於自訂原始伺服器。

(選擇性) 您可以指定自訂原始伺服器偵聽的 HTTP 連接埠。有效值包括連接埠 80、443，以及 1024 到 65535 之間。預設值為連接埠 80。

⚠ Important

當原始伺服器是 Amazon S3 靜態網站託管端點時，連線埠 80 是預設設定，因為 Amazon S3 僅支援連線埠 80 的靜態網站託管端點。CloudFront 主控台不支援變更 Amazon S3 靜態網站託管端點的此設定。

HTTPS 連接埠**i Note**

這僅適用於自訂原始伺服器。

(選擇性) 您可以指定自訂原始伺服器偵聽的 HTTPS 連接埠。有效值包括連接埠 80、443，以及 1024 到 65535 之間。預設值為連接埠 443。當 Protocol (通訊協定) 設定為 HTTP only (僅限 HTTP)，您無法指定 HTTPS port (HTTPS 連接埠) 值。

最低來源 SSL 通訊協定

Note

這僅適用於自訂原始伺服器。

選擇 CloudFront 在建立與您原始伺服器的 HTTPS 連線時可使用的最低 TLS/SSL 通訊協定。由於較低的 TLS 通訊協定安全性較低，因此我們建議您選擇您來源支援的最新 TLS 通訊協定。當 Protocol (通訊協定) 設定為 HTTP only (僅限 HTTP)，您無法指定 Minimum origin SSL protocol (最低來源 SSL 通訊協定) 值。

如果您使用 CloudFront API 來設定用於 CloudFront 的 TLS/SSL 通訊協定，則無法設定最低通訊協定。反之，您可指定所有 TLS/SSL 協定，以供 CloudFront 與您的原始伺服器搭配使用。如需詳細資訊，請參閱 Amazon CloudFront API 參考中的 [OriginSslProtocols](#)。

原始伺服器路徑

如果您希望 CloudFront 從原始伺服器的目錄中請求內容，請輸入以斜線 (/) 開頭的目錄路徑。CloudFront 會將目錄路徑附加到 Origin domain (原始網域) 的值中，例如：**cf-origin.example.com/production/images**。請勿在路徑的尾端加上斜線 (/)。

例如，假設您已為您的分佈指定以下值：

- Origin domain (原始網域) – 名為 **amzn-s3-demo-bucket** 的 Amazon S3 儲存貯體
- Origin path (原始伺服器路徑) - **/production**
- Alternate domain names (CNAME) (備用網域名稱 (CNAME)) – **example.com**

當使用者在瀏覽器中輸入 `example.com/index.html` 時，CloudFront 會針對 `amzn-s3-demo-bucket/production/index.html` 將請求傳送到 Amazon S3。

當使用者在瀏覽器中輸入 `example.com/acme/index.html` 時，CloudFront 會針對 `amzn-s3-demo-bucket/production/acme/index.html` 將請求傳送到 Amazon S3。

Name

名稱是一個可唯一識別該分佈中此原始伺服器的字串。如果您除了預設快取行為外，還建立快取行為，則可以使用您在此處指定的名稱，以識別您所希望 CloudFront 將請求路由至其中的原始伺服器，前提是該請求需與該快取行為的路徑模式相符。

原始存取 (僅限 Amazon S3 原始伺服器)

Note

這僅適用於 Amazon S3 儲存貯體原始伺服器 (那些不使用 S3 靜態網站端點的原始伺服器)。

如果您想將對 Amazon S3 儲存貯體的存取限制為僅限特定 CloudFront 分佈，請選擇 Origin access control settings (recommended) (原始伺服器存取控制設定 (建議使用))。

如果 Amazon S3 儲存貯體可公開存取，請選擇 public (公有)。

如需詳細資訊，請參閱[the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

如需如何請求使用者只使用 CloudFront URL 來存取自訂原始伺服器上的物件的詳細資訊，請參閱[the section called “在自訂原始伺服器上限制存取檔案”](#)。

新增自訂標頭

如果您希望 CloudFront 在將請求傳送到您的原始伺服器時新增自訂標頭，請指定標頭名稱和值。如需詳細資訊，請參閱[the section called “將自訂標頭新增到原始伺服器請求”](#)。

如需可新增自訂標頭數量上限的目前上限、自訂標頭的名稱和值的最大長度，以及所有標頭名稱和值的總長度上限的詳細資訊，請參閱 [配額](#)。

啟用 Origin Shield

選擇 Yes (是) 以啟用 CloudFront Origin Shield。如需 Origin Shield 的詳細資訊，請參閱 [the section called “使用 Origin Shield”](#)。

連線嘗試

您可以設定 CloudFront 嘗試連線至原始伺服器的次數。您可以指定 1、2 或 3 做為嘗試次數。預設數字 (如果您未另外指定) 是 3。

將此設定與 Connection timeout (連線逾時) 搭配使用，可指定在嘗試連線至次要原始伺服器或將錯誤回應傳回給檢視器之前 CloudFront 等待的時間長度。依預設，CloudFront 會等待 30 秒 (嘗試 3 次，每次 10 秒)，然後再嘗試連線至次要原始伺服器或傳回錯誤回應。您可以指定較少的嘗試次數、較短的連線逾時或兩者，以縮短此時間。

如果指定的連線嘗試次數失敗，CloudFront 會執行下列其中一項操作：

- 如果原始伺服器是原始伺服器群組的一部分，則 CloudFront 會嘗試連線至次要原始伺服器。如果對次要原始伺服器指定的連線嘗試次數失敗，則 CloudFront 會傳回錯誤回應給檢視器。
- 如果原始伺服器不是原始伺服器群組的一部分，則 CloudFront 會傳回錯誤回應給檢視器。

對於自訂原始伺服器 (包括使用靜態網站託管所配置的 Amazon S3 儲存貯體)，此配置也會指定 CloudFront 嘗試從原始伺服器取得回應的次數。如需詳細資訊，請參閱[the section called “回應逾時”](#)。

連線逾時。

連線逾時是嘗試建立與原始伺服器的連線時 CloudFront 所等待的秒數。您可以指定介於 1 到 10 之間的秒數 (含)。預設逾時 (如果您另外未指定) 是 10 秒。

將此設定與 Connection attempts (連線嘗試次數) 搭配使用，可指定在嘗試連線至次要原始伺服器之前或將錯誤回應傳回給檢視器之前 CloudFront 等待的時間長短。依預設，CloudFront 會等待 30 秒 (嘗試 3 次，每次 10 秒)，然後再嘗試連線至次要原始伺服器或傳回錯誤回應。您可以指定較少的嘗試次數、較短的連線逾時或兩者，以縮短此時間。

如果 CloudFront 未在指定的秒數內建立與原始伺服器的連線，CloudFront 會執行下列其中一項操作：

- 如果指定的 Connection attempts (連線嘗試次數) 超過 1，CloudFront 會再次嘗試建立連線。CloudFront 最多可嘗試 3 次，由 Connection attempts (連線嘗試次數) 的值決定。
- 如果所有連線嘗試失敗，且原始伺服器是原始伺服器群組的一部分，則 CloudFront 會嘗試連線至次要原始伺服器。如果對次要原始伺服器指定的連線嘗試次數失敗，則 CloudFront 會傳回錯誤回應給檢視器。
- 如果所有連線嘗試失敗，且原始伺服器不是原始伺服器群組的一部分，則 CloudFront 會傳回錯誤回應給檢視器。

回應逾時

原始伺服器回應逾時，也稱為原始伺服器讀取逾時或原始伺服器請求逾時，適用於以下兩個值：

- 在將請求轉送到自訂原始伺服器之後，CloudFront 等待回應的時間 (以秒為單位)。

- CloudFront 在收到來自原始伺服器的回應封包後，並在接收下一個封包前，等待的時間長短 (以秒為單位)

Tip

如果因為檢視器遇到 HTTP 504 狀態碼錯誤，而您想要增加逾時值，請考慮在變更逾時值之前先探索其他方式來消除這些錯誤。請查看在 [the section called “HTTP 504 狀態碼 \(閘道逾時\)”](#) 中的故障診斷建議。

CloudFront 行為取決於檢視器請求中的 HTTP 方法：

- GET 與 HEAD 請求 – 如果原始伺服器在回應逾時期間未回應或是停止回應，CloudFront 會結束連線。CloudFront 會根據 [the section called “連線嘗試”](#) 的值再次嘗試連線。
- DELETE、OPTIONS、PATCH、PUT 及 POST 請求 – 如果在讀取逾時期間原始伺服器未回應，CloudFront 將會結束連線，並且不再嘗試聯絡原始伺服器。用戶端可以視需要重新提交請求。

回應完成逾時

Note

回應完成逾時不支援[持續部署](#)功能。

從 CloudFront 到原始伺服器的請求可以保持開啟並等待回應的時間 (以秒為單位)。如果此時未從原始伺服器收到完整回應，CloudFront 會結束連線。

不同於回應逾時，這是個別回應封包的等待時間，回應完成逾時是 CloudFront 等待回應完成的允許時間上限。您可以使用此設定，以確保 CloudFront 不會無限期待緩慢或沒有回應的原始伺服器，即使其他逾時設定允許更長的等待時間。

此最大逾時包括您為其他逾時設定指定的項目，以及每次重試的連線嘗試次數。您可以同時使用這些設定來指定 CloudFront 等待完整請求的時間，以及何時結束請求，無論是否完成。

例如，如果您設定下列設定：

- 連線嘗試為 3
- 連線逾時為 10 秒

- 回應逾時為 30 秒
- 回應完成逾時為 60 秒

這表示 CloudFront 會嘗試連線到原始伺服器 (最多總共 3 次嘗試)，每次連線嘗試會在 10 秒逾時。連線後，CloudFront 最多會等待 30 秒，讓原始伺服器回應請求，直到收到回應的最後一個封包為止。

無論連線嘗試次數或回應逾時，如果來自原始伺服器的完整回應需要超過 60 秒才能完成，CloudFront 都會結束連線。然後，如果您指定了 [the section called “HTTP 504 狀態碼 \(閘道逾時\)”](#) 錯誤回應或自訂錯誤回應，CloudFront 將傳回給檢視器。

備註

- 如果您設定回應完成逾時的值，則值必須等於或大於[回應逾時 \(原始伺服器讀取逾時\)](#) 的值。
- 如果您未設定回應完成逾時的值，CloudFront 不會強制執行最大值。

保持連線逾時 (僅限自訂與 VPC 原始伺服器)

保持連線逾時是在取得上次封包回應之後，CloudFront 嘗試與您的自訂原始伺服器保持連線的時間 (以秒為單位)。維護持久性連線可節省重新建立 TCP 連線所需的時間，並為後續請求執行另一個 TLS 交換。增加「保持活動逾時」有助於改善分佈的「每個連線請求指標」。

Note

為了讓 Keep-alive Timeout (保持連線逾時) 值有效，必須將原始伺服器設定為允許持久連線。

回應和保持連線逾時配額

- 對於[回應逾時](#)，預設為 30 秒。
- 對於[保持連線逾時](#)，預設值為 5 秒。

如果您請求增加的逾時 AWS 帳戶，請更新您的分佈原始伺服器，使其具有您想要的回應逾時和持續作用逾時值。您帳戶的配額增加不會自動更新您的原始伺服器。例如，如果您使用 Lambda@Edge 函數來設定 90 秒的保持連線逾時，您的原始伺服器必須已有 90 秒以上的保持連線逾時。否則，您的 Lambda@Edge 函數可能無法執行。

如需有關分佈配額的詳細資訊，包括如何請求提高配額，請參閱 [分佈的一般配額](#)。

快取行為設定

快取行為可讓您為網站上的檔案的指定 URL 路徑模式配置各種 CloudFront 功能。例如，一個快取行為可能會套用到 Web 伺服器上的 images 目錄中的所有 .jpg 檔案，您正在使用它做為 CloudFront 的原始伺服器。您可以為每個快取行為設定的功能包括：

- 路徑模式
- 如果您為 CloudFront 分佈設定了多個原始伺服器，您希望 CloudFront 將您的請求轉送至哪個原始伺服器
- 是否將查詢字串轉發到您的原始伺服器。
- 存取指定的檔案是否需要已簽署的 URL
- 是否要求使用者使用 HTTPS 來存取這些檔案
- 這些檔案在 CloudFront 快取中停留的最短時間，此時間與原始伺服器新增至檔案中的任何 Cache-Control 標頭值無關

在建立新的分佈，您可以指定預設快取行為的設定，該設定會自動將所有請求轉發到您在建立分佈時指定的原始伺服器。在建立分發後，您可以建立額外的快取行為，以定義 CloudFront 在收到與路徑模式符合的物件的請求時如何回應，例如，*.jpg。如果您建立額外的快取行為，則預設的快取行為一律是最後要處理的。其他快取行為按其在 CloudFront 主控台中所列的順序進行處理，或者如果您使用的是 CloudFront API，它們在分發的 DistributionConfig 元素中列出順序。如需詳細資訊，請參閱[路徑模式](#)。

當建立快取行為時，您可以指定一個您希望 CloudFront 從中取得物件的原始伺服器。因此，如果您希望 CloudFront 從您的所有原始伺服器分配物件，則必須至少擁有的快取行為儘可能與該您的原始伺服器一樣多 (包括預設的快取行為)。例如，如果您有兩部原始伺服器，且只有預設的快取行為，則預設快取行為會導致 CloudFront 從其中一部原始伺服器取得物件，但永遠不會使用其他原始伺服器。

如需您可為分發新增之快取行為數量的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱[分佈的一般配額](#)。

主題

- [路徑模式](#)
- [來源或來源群組](#)
- [檢視器通訊協定政策](#)

- [Allowed HTTP methods \(允許的 HTTP 方法\)](#)
- [欄位層級加密 Config](#)
- [快取的 HTTP 方法](#)
- [允許透過 HTTP/2 進行 gRPC 請求](#)
- [根據選取請求標頭的快取](#)
- [允許清單標頭](#)
- [物件快取](#)
- [最短 TTL](#)
- [最長 TTL](#)
- [預設 TTL](#)
- [轉送 Cookie](#)
- [允許清單 Cookie](#)
- [查詢字串轉送和快取](#)
- [查詢字串允許清單](#)
- [Smooth Streaming](#)
- [限制檢視器存取 \(使用已簽章的 URL 或已簽章的 Cookie\)](#)
- [可信簽署者](#)
- [AWS 帳戶 數字](#)
- [自動壓縮物件](#)
- [CloudFront 事件](#)
- [Lambda 函數 ARN](#)
- [包含內文](#)

路徑模式

路徑模式 (例如, `images/*.jpg`) 指定您希望將此快取行為套用到哪些請求。當 CloudFront 收到檢視器請求時, 會依快取行為在分佈中的排列順序, 比較請求的路徑與路徑模式。第一種符合決定哪個快取行為套用到該請求。例如, 假設您在以下三種路徑模式下有三個快取行為模式, 並按此順序排列:

- `images/*.jpg`
- `images/*`

- *.gif

Note

您可以選擇在路徑模式的開頭包括斜線 (/)，例如 /images/*.jpg。無論是否有前導斜線，CloudFront 的行為都一樣。如果您未在路徑開頭指定 /，則會自動隱含此字元；CloudFront 會以相同方式處理路徑，無論是否使用前置 /。例如，CloudFront 會將相同 /*product.jpg 視為 *product.jpg

對檔案 images/sample.gif 的請求無法滿足第一個路徑模式，所以關聯的快取行為不能套用到該請求。該檔案確實滿足第二個路徑模式，因此即使該請求也與第三個路徑模式相符合，也會套用與第二個路徑模式相關聯的快取行為。

Note

當您建立新的分佈時，預設快取行為的 Path Pattern (路徑模式) 值將設定為 * (所有檔案)，而且無法變更。這個值會導致 CloudFront 將您的物件的所有請求轉送到在 [原始網域](#) 欄位中指定的原始伺服器。如果對物件的請求與任何其他快取行為的路徑模式不符合，CloudFront 將套用您在預設快取行為中指定的行為。

Important

請謹慎定義路徑模式及其序列，否則可能會讓使用者對您的內容作出無法預料的存取。例如，假設請求與兩個快取行為的路徑模式相符合。第一個快取行為不需要簽章的 URL 和第二個快取行為確實需要簽章的 URL。使用者能夠在不使用簽署的 URL 的情況下存取物件，因為 CloudFront 處理與第一個符合相關聯的快取行為。

如果您正在使用 MediaPackage 頻道，您必須針對您為原始伺服器的端點類型定義的快取行為納入特定的路徑模式。例如，如果是 DASH 端點，請在 路徑模式 中，輸入 *.mpd。如需詳細資訊和特定說明，請參閱[提供以 AWS Elemental MediaPackage 格式化的即時視訊](#)。

您所指定的路徑套用到指定目錄中所有檔案和指定目錄下子目錄中所有檔案的請求。CloudFront 在評估路徑模式時不考慮查詢字串或 Cookie。例如，如果 images 目錄包含 product1 和 product2 子目錄，則路徑模式 images/*.jpg 適用於 images、images/product1 和 images/product2 目

錄中任何 .jpg 檔案的請求。如果要對 images/product1 目錄中的檔案套用與 images 和 images/product2 目錄中的套用不同的快取行為，請為 images/product1 建立一個單獨的快取行為，並將該快取行為移至 images 目錄快取行為上方 (之前) 的位置。

您可以在路徑模式中使用以下萬用字元：

- * 符合 0 或更多字元。
- ? 符合完全 1 個字元。

以下範例說明萬用字元的工作方式：

路徑模式	與路徑模式相符的檔案
*.jpg	所有 .jpg 檔案。
images/*.jpg	位於 images 目錄和 images 目錄下子目錄中的所有 .jpg 檔案。
a*.jpg	• 檔案名稱以 a 開頭的所有 .jpg 檔案，例如 apple.jpg 和 appalachian_trail_2012_05_21.jpg。 • 檔案路徑以 a 開頭的所有 .jpg 檔案，例如 abra/cadabra/magic.jpg。
a??.jpg	檔案名稱以 a 開頭的所有 .jpg 檔案，後面緊接恰好兩個其他字元，例如 ant.jpg 和 abe.jpg。
.doc	檔案名稱副檔名以 .doc 開頭的所有檔案，例如 .doc、.docx 和 .docm 檔案。在這種情況下，您不能使用路徑模式 *.doc?，因為該路徑模式不適用於 .doc 檔案的請求；? 萬用字元僅取代一個字元。

路徑模式 的長度上限為 255 個字元。這個值可以包含以下任何字元：

- A-Z、a-z

途徑模式會區分大小寫，因此路徑模式 *.jpg 不適用於檔案 LOGO.JPG

- 0-9
- _ - . * \$ / ~ " ' @ : +
- & 通過並以 & 傳回

路徑標準化

CloudFront 會依照 [RFC 3986](#) 的規範來標準化 URI 路徑，然後將路徑與正確的快取行為比對。快取行為相符後，CloudFront 會將原始 URI 路徑傳送至原始伺服器。如果不相符，請求會改為符合您的預設快取行為。

有些字元會標準化並從路徑中移除，例如多個斜線 (//) 或句點 (. .)。這可能會改變 CloudFront 用來比對預期快取行為的 URL。

Example 範例

您可以指定快取行為的 /a/b* 和 /a* 路徑。

- 傳送 /a/b?c=1 路徑的檢視器將符合 /a/b* 快取行為。
- 傳送 /a/b/..?c=1 路徑的檢視器將符合 /a* 快取行為。

若要解決正被標準化的路徑，您可以更新請求路徑或快取行為的路徑模式。

來源或來源群組

僅當您建立或更新現有分佈的快取行為時才適用此設定。

輸入現有來源或來源群組的值。這會識別當請求 (例如 https://example.com/logo.jpg) 與快取行為的路徑模式 (例如 *.jpg) 或預設快取行為的路徑模式 (*) 相符時，您要 CloudFront 將請求路由到的原始伺服器或原始伺服器群組。

檢視器通訊協定政策

選擇您想要檢視器在 CloudFront 節點存取您的內容時使用的通訊協定政策：

- HTTP and HTTPS (HTTP 和 HTTPS)：檢視器可使用這兩種通訊協定。

- Redirect HTTP to HTTPS (將 HTTP 重新引導到 HTTPS)：檢視器可使用這兩種通訊協定，但是 HTTP 請求會自動重新引導到 HTTPS 請求。
- HTTPS Only (僅限 HTTPS)：檢視器只能在使用 HTTPS 的情況下存取您的內容。

如需詳細資訊，請參閱[檢視器和 CloudFront 之間的通訊需使用 HTTPS](#)。

Allowed HTTP methods (允許的 HTTP 方法)

指定您希望 CloudFront 處理並轉送到您的原始伺服器的 HTTP 方法。

- GET、HEAD：您只能使用 CloudFront 來從您的原始伺服器取得物件或物件標頭。
- GET、HEAD、OPTIONS：您只能使用 CloudFront 來從您的原始伺服器取得物件，取得物件標頭或擷取原始伺服器支援的選項清單。
- GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE：您可以使用 CloudFront 來取得、新增、更新和刪除物件，以及取得物件標頭。此外，您可以執行其他 POST 操作，例如從 Web 表單提交資料。

Note

如果您在工作負載中使用 gRPC，則必須選擇

GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE。gRPC 工作負載需要 POST 方法。如需詳細資訊，請參閱[使用 gRPC 搭配 CloudFront 分佈](#)。

CloudFront 快取對 GET 和 HEAD 請求的回應，以及可選的 OPTIONS 請求的回應。OPTIONS 請求的回應與 GET 和 HEAD 請求的回應分開快取 (OPTIONS 方法包含在適用於 OPTIONS 請求的[快取金鑰](#)中)。CloudFront 不會快取對使用其他方法之請求的回應。

Important

如果您選擇 GET、HEAD、OPTIONS 或

GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE，則可能需要限制對 Amazon S3 儲存貯體或自訂原始伺服器原始伺服器的存取，以防止使用者執行不希望它們執行的操作。下列範例說明如何限制存取：

- 如果您正在使用 Amazon S3 做為分佈的原始伺服器：建立 CloudFront 原始存取控制以限制對 Amazon S3 內容的存取，並為原始存取控制授予許可。例如，若您將 CloudFront 配置為

接受並轉送這些方法，僅因為您要使用 PUT，則仍然必須配置 Amazon S3 儲存貯體原則以正確處理 DELETE 請求。如需詳細資訊，請參閱[限制對 Amazon S3 原始伺服器的存取](#)。

- 如果您使用自訂原始伺服器：請設定您的原始伺服器以處理所有方法。例如，如果您將 CloudFront 配置為接受並轉送這些方法，僅因為您要使用 POST，則仍然必須配置您的原始伺服器以正確處理 DELETE 請求。

欄位層級加密 Config

如果想要針對特定資料欄位強制執行欄位層級加密，請在下拉式清單中選擇欄位層級加密組態。

如需詳細資訊，請參閱[使用欄位層級加密來協助保護敏感資料](#)。

快取的 HTTP 方法

指定當檢視器提交 OPTIONS 請求，您是否希望 CloudFront 快取原始伺服器原始伺服器的回應。CloudFront 一律快取 GET 和 HEAD 請求的回應。

允許透過 HTTP/2 進行 gRPC 請求

指定您是否希望分佈允許 gRPC 請求。若要啟用 gRPC，請選取下列設定：

- 針對[允許的 HTTP 方法](#)，選取 GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE 方法。gRPC 需要 POST 方法。
- 選取在您選取 POST 方法後出現的 gRPC 核取方塊。
- 針對[支援的 HTTP 版本](#)，選取 HTTP/2。

如需詳細資訊，請參閱[使用 gRPC 搭配 CloudFront 分佈](#)。

根據選取請求標頭的快取

指定是否希望 CloudFront 根據特定標頭的值來快取物件：

- 無 (提升快取) – CloudFront 不會根據標頭值快取物件。
- 允許清單 – CloudFront 僅依據指定標頭的值來快取物件。使用允許清單標頭選擇您希望 CloudFront 進行快取時所依據的標頭。
- 所有 – CloudFront 不會快取與此快取行為相關聯的物件。相反地，CloudFront 會將每個請求傳送給原始伺服器。(不建議用於 Amazon S3 原始伺服器。)

無論您選擇的選項為何，CloudFront 皆會轉送特定標頭到原始伺服器，並根據您轉送的標頭來採取特定動作。如需 CloudFront 處理標頭轉送的詳細資訊，請參閱[HTTP 請求標頭和 CloudFront 行為 \(自訂和 Amazon S3 原始伺服器\)](#)。

如需如何使用請求標頭以在 CloudFront 配置快取的詳細資訊，請參閱[根據請求標頭快取內容](#)。

允許清單標頭

這些設定只有當您針對根據選取的請求標頭的快取選擇允許清單時才適用。

指定希望 CloudFront 在快取物件時考慮的標頭。從可用標頭清單中選擇標頭，然後選擇 Add (新增)。若要轉送自訂標頭，請在欄位中輸入標頭的名稱，然後選擇 Add Custom (新增自訂)。

如需您可為每個快取行為列入允許清單之標頭數量的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱[標頭的配額](#)。

物件快取

如果您的原始伺服器向您的物件新增 Cache-Control 標頭以控制物件在 CloudFront 快取中的停留時間，而且您不想變更 Cache-Control 值，請選擇使用原始伺服器快取標頭。

若要指定物件停留在 CloudFront 快取中的最短和最長時間，而不考慮 Cache-Control 標頭，以及當物件中的 Cache-Control 標頭時遺失時，物件停留在 CloudFront 快取中的預設時間，請選擇自訂。然後，在 Minimum TTL (最短 TTL)、Default TTL (預設 TTL) 及 Maximum TTL (最長 TTL) 欄位中指定值。

如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

最短 TTL

指定時間的長度下限 (以秒為單位)，此時間是您希望在 CloudFront 傳送另一個請求至原始伺服器前，物件要停留在 CloudFront 快取中以判定物件是否已更新的時間。

Warning

如果您的最小 TTL 大於 0，則即使原始伺服器標頭中存在 Cache-Control: no-cache、no-store 或 private 指令，CloudFront 也會快取內容至少達到快取政策最短 TTL 中指定的持續時間。

如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

最長 TTL

指定希望物件在 CloudFront 快取中停留的最長時間 (以秒為單位)，然後在 CloudFront 查詢您的原始伺服器原始伺服器，以查看該物件是否已更新。您為 Maximum TTL (最長 TTL) 指定的值僅適用於您的原始伺服器將 HTTP 標題 (例如 Cache-Control max-age、Cache-Control s-maxage 或 Expires) 新增至物件時。如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

若要指定 Maximum TTL (最長 TTL) 的值，您必須為 Object Caching (物件快取) 設定選擇 Customize (自訂) 選項。

Maximum TTL (最長 TTL) 的預設值為 31536000 秒 (一年)。如果將 Minimum TTL (最短 TTL) 或 Default TTL (預設 TTL) 的值變更為 31536000 秒以上，則 Maximum TTL (最長 TTL) 的預設值將變更為 Default TTL (預設 TTL) 的值。

預設 TTL

指定您希望物件在 CloudFront 快取中停留的預設時間，然後 CloudFront 會將另一個請求轉送到原始伺服器，以判斷物件是否已更新。您為預設 TTL 指定的值，僅在您的原始伺服器不會將 HTTP 標題 (例如 Cache-Control max-age、Cache-Control s-maxage 或 Expires) 新增至物件時適用。如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

若要指定 Default TTL (預設 TTL) 的值，您必須為 Object Caching (物件快取) 設定選擇 Customize (自訂) 選項。

Default TTL (預設 TTL) 的預設值為 86400 秒 (一天)。如果將 Minimum TTL (最短 TTL) 的值變更為 86400 秒以上，則 Default TTL (預設 TTL) 的預設值將變更為 Minimum TTL (最短 TTL) 的值。

轉送 Cookie

Note

對於 Amazon S3 原始伺服器，此選項僅適用於配置為網站端點的儲存貯體。

指定您是否希望 CloudFront 轉送 Cookie 到原始伺服器，若要轉送，則請指定是哪些 Cookie。如果您選擇僅轉送所選取的 Cookie (Cookie 的允許清單)，請在允許清單 Cookie 欄位中輸入 Cookie 名稱。如果您選擇所有，無論您的應用程式使用多少，CloudFront 都會轉送所有 Cookie。

Amazon S3 並不處理 Cookie，而且將 Cookie 轉送到原始伺服器的動作，會降低快取的能力。對於將請求轉送到 Amazon S3 原始伺服器的快取行為，請為轉送 Cookie 選擇無。

如需有關將 Cookie 轉發到原始伺服器的詳細資訊，請前往[根據 Cookie 快取內容](#)。

允許清單 Cookie

Note

對於 Amazon S3 原始伺服器，此選項僅適用於配置為網站端點的儲存貯體。

如果您在轉送 Cookies 清單中選擇了允許清單，請在允許清單 Cookies 欄位中輸入您希望 CloudFront 針對此快取行為要轉送至原始伺服器的 Cookie 名稱。在新的列上輸入每個 Cookie 名稱。

您可以指定以下萬用字元，以指定 Cookie 名稱：

- * 符合 Cookie 名稱中的 0 個或多個字元
- ? 僅符合 Cookie 名稱中的一個字元

例如，假設檢視器為包含 Cookie 的物件作請求，其 Cookie 名為：

`userid_member-number`

其中每個使用者都有####的唯一值。您希望 CloudFront 為每個成員快取單獨版本的物件。您可以透過將所有 Cookie 轉送到原始伺服器來達成此目標，但檢視器請求包含一些您不希望 CloudFront 快取的 Cookie。或者，您可以指定以下值做為 Cookie 名稱，這會導致 CloudFront 將所有以 `userid_` 開頭的 Cookie 轉送至原始伺服器：

`userid_*`

如需您可為每個快取行為列入允許清單之 Cookie 名稱的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱[Cookie 的配額 \(舊版快取設定\)](#)。

查詢字串轉送和快取

CloudFront 可以根據查詢字串參數的值快取不同版本的內容。請選擇下列其中一個選項：

無 (提升快取)

如果您的原始來源傳回物件的相同版本，而不考慮查詢字串參數的值，請選擇此選項。這會增加 CloudFront 的可能性，可以提供來自快取的請求，這可以提高效能，並降低您的原始伺服器原始伺服器的負載。

全部轉送，依據允許清單進行快取

如果您的原始伺服器根據一或多個查詢字串參數傳回物件的不同版本，請選擇此選項。接著，在 [查詢字串允許清單](#) 欄位中，指定您希望 CloudFront 使用做為快取基礎的參數。

轉發所有，根據所有快取

如果您的原始伺服器針對所有查詢字串參數傳回物件的不同版本，請選擇此選項。

如需有關根據查詢字串參數進行快取的詳細資訊，包括如何提升效能，請參閱[根據查詢字串參數快取內容](#)。

查詢字串允許清單

此設定僅在您為 [查詢字串轉送和快取](#) 選擇轉送全部，並根據允許清單進行快取時才會套用。您可以指定做為您希望 CloudFront 用於快取基礎的查詢字串參數。

Smooth Streaming

如果您想要以 Microsoft Smooth Streaming 格式分配媒體檔案，而且您沒有 IIS 伺服器，請選擇 Yes (是)。

如果您有 Microsoft IIS 伺服器，而且想要將它當做以 Microsoft Smooth Streaming 格式分配媒體檔案的原始伺服器使用，或者您並非分配 Smooth Streaming 媒體檔案，請選擇 No (否)。

Note

如果您指定 Yes (是)，而內容與 Path Pattern (路徑模式) 的值符合，您仍然可以使用此快取行為分配其他內容。

如需詳細資訊，請參閱[為 Microsoft Smooth Streaming 設定隨需視訊](#)。

限制檢視器存取 (使用已簽章的 URL 或已簽章的 Cookie)

如果您希望與此快取行為之 PathPattern 相符的物件請求使用公有 URL，請選擇 No (否)。

如果您希望與此快取行為之 PathPattern 相符的物件請求使用已簽章的 URL，請選擇 Yes (是)。然後指定您要用來建立簽章 URLs AWS 的帳戶；這些帳戶稱為信任簽署者。

如需有關可信任簽署者的詳細資訊，請參閱[指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

可信簽署者

僅當您在限制檢視器存取權限 (使用已簽署的 URL 或已簽署的 Cookie) 選擇是時才適用此設定。

選擇您要使用此快取行為做為信任簽署者的 AWS 帳戶：

- 自我：使用您目前登入的帳戶 AWS 管理主控台 做為信任的簽署者。如果您目前以 IAM 使用者身分登入，則相關聯的 AWS 帳戶會新增為信任的簽署者。
- Specify Accounts (指定帳戶)：在 AWS Account Numbers (帳號) 欄位中輸入可信任簽署者的帳號。

若要建立簽章URLs，AWS 帳戶必須至少有一個作用中的 CloudFront 金鑰對。

Important

如果您正在更新已用於分配內容的分佈，則只有在準備好開始為您的物件產生已簽章 URL 時，才會新增可信任簽署者。將可信任簽署者新增到分佈後，使用者必須使用已簽名的 URL 來存取與此 PathPattern 快取行為符合的物件。

AWS 帳戶 數字

僅當您在可信簽署者選擇指定帳戶時才適用此設定。

如果您想要使用 AWS 帳戶 除了目前帳戶以外的 或 建立已簽章URLs，請在此欄位的每行輸入一個 AWS 帳戶 數字。注意下列事項：

- 您指定的帳戶必須至少有一個作用中的 CloudFront 金鑰對。如需詳細資訊，請參閱[為您的簽署者建立金鑰對](#)。
- 您無法為 IAM 使用者建立 CloudFront 金鑰對，因此您不能將 IAM 使用者用作可信任簽署者。
- 如需如何取得帳戶 AWS 帳戶 號碼的資訊，請參閱《AWS 帳戶 管理參考指南》中的[檢視 AWS 帳戶 識別符](#)。
- 如果您輸入目前帳戶的帳號，CloudFront 將自動勾選自我核取方塊並從 AWS 帳號清單中移除該帳號。

自動壓縮物件

如果您想要 CloudFront 在檢視器支援壓縮內容時自動壓縮特定類型的檔案，請選擇是。當 CloudFront 壓縮您的內容時，下載的速度會更快，因為檔案較小，而且您的網頁呈現給使用者的速度更快。如需詳細資訊，請參閱[提供壓縮檔案](#)。

CloudFront 事件

此設定適用於 Lambda 函數關聯。

當發生以下一或多個 CloudFront 事件時，您可以選擇執行 Lambda 函數：

- 當 CloudFront 接收到來自檢視器的請求 (檢視器請求) 時
- 在 CloudFront 轉寄請求至來源 (原始請求) 之前
- 當 CloudFront 接收到來自來源的回應 (原始回應) 時
- 在 CloudFront 傳回回應給檢視器 (檢視器回應) 之前

如需詳細資訊，請參閱[選擇要觸發函數的事件](#)。

Lambda 函數 ARN

此設定適用於 Lambda 函數關聯。

指定要為其新增觸發的 Lambda 函數的 Amazon 資源名稱 (ARN)。若要了解如何取得函數的 ARN，請參閱[使用 CloudFront 主控台新增觸發條件](#)的步驟 1。

包含內文

此設定適用於 Lambda 函數關聯。

如需詳細資訊，請參閱[包含本文](#)。

分佈設定

以下值適用於整個分佈。

主題

- [價格分級](#)

- [AWS WAF Web ACL](#)
- [備用網域名稱 \(CNAME\)](#)
- [SSL 憑證](#)
- [自訂 SSL 用戶端支援](#)
- [安全政策 \(最低 SSL/TLS 版本\)](#)
- [支援的 HTTP 版本](#)
- [預設根物件](#)
- [標準記錄](#)
- [連線日誌](#)
- [日誌字首](#)
- [Cookie 記錄](#)
- [啟用 IPv6 \(檢視器請求\)](#)
- [交互身分驗證](#)
- [為自訂原始伺服器啟用 IPv6 \(原始伺服器請求\)](#)
- [Comment](#)
- [分佈狀態](#)

價格分級

選擇對應您希望支付之 CloudFront 服務最高價的價格分級。根據預設，CloudFront 會從所有 CloudFront 區域的節點提供您的物件。

如需價格分級，以及您的價格分級選擇如何影響分佈的 CloudFront 效能詳細資訊，請參閱 [CloudFront 定價](#)。

AWS WAF Web ACL

您可以使用 Web 應用程式防火牆 [AWS WAF](#) 來保護 CloudFront 分發，該防火牆可讓您保護 Web 應用程式和 API，以便在請求到達伺服器之前封鎖請求。在建立或編輯 CloudFront 分發時，您可以 [為分佈啟用 AWS WAF](#)。

或者，您可以稍後在 AWS WAF 主控台 <https://console.aws.amazon.com/wafv2/> 中為應用程式特有的其他威脅設定額外的安全保護。

如需的詳細資訊 AWS WAF，請參閱 [AWS WAF 開發人員指南](#)。

備用網域名稱 (CNAME)

選用。指定一或多個您要用於物件 URL 的網域名稱，而非您建立分佈時 CloudFront 指派的網域名稱。您必須擁有網域名稱，或是擁有使用它的授權；您可以透過新增 SSL/TLS 憑證來進行驗證。

例如，若您希望物件的 URL：

`/images/image.jpg`

看起來像此 URL：

`https://www.example.com/images/image.jpg`

而不是此 URL：

`https://d111111abcdef8.cloudfront.net/images/image.jpg`

請新增 `www.example.com` 的 CNAME。

Important

若您將 `www.example.com` 的 CNAME 新增到您的分佈，您也必須執行以下作業：

- 使用您的 DNS 服務建立 (或更新) CNAME 記錄，以將 `www.example.com` 的查詢路由傳送到 `d111111abcdef8.cloudfront.net`。
- 將來自可信任憑證授權機構 (CA)，涵蓋您新增至分佈網域名稱 (CNAME) 的憑證新增到 CloudFront，以驗證您使用該網域名稱的授權。

您必須擁有使用網域 DNS 服務提供者建立 CNAME 記錄的許可。通常，這表示您擁有網域，或者您正在為網域擁有者開發應用程式。

如需您可為分發新增之備用網域名稱數量的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱 [分佈的一般配額](#)。

如需備用網域名稱的詳細資訊，請參閱 [透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#)。如需 CloudFront URL 的詳細資訊，請參閱 [自訂 CloudFront 中檔案的 URL 格式](#)。

SSL 憑證

若您指定要搭配分佈使用的備用網域名稱，請選擇 Custom SSL Certificate (自訂 SSL 憑證)，然後，為了驗證您的憑證使用其他網域名稱，請選擇涵蓋該名稱的憑證。如果您希望檢視器使用 HTTPS 存取您的物件，請選擇支援該功能的設定。

- 預設 CloudFront 憑證 (*.cloudfront.net) – 若您要在物件的 URL 中使用 CloudFront 網域名稱 (例如 `https://d1111111abcdef8.cloudfront.net/image1.jpg`)，請選擇此選項。
- 自訂 SSL 憑證 – 若您希望在物件的 URL 中使用您自己的網域名稱，做為備用網域名稱 (例如 `https://example.com/image1.jpg`)，請選擇此選項。然後要使用的憑證，其中涵蓋了備用網域名稱。憑證的清單可包括以下任何項目：
 - 提供的憑證 AWS Certificate Manager
 - 您從第三方憑證授權機構購買並上傳到 ACM 的憑證
 - 您從第三方憑證授權機構購買，並上傳到 IAM 憑證存放區的憑證

若您選擇此設定，我們建議您僅使用您物件 URL 中的備用網域名稱 (`https://example.com/logo.jpg`)。若您使用 CloudFront 分佈網域名稱 (`https://d1111111abcdef8.cloudfront.net/logo.jpg`)，而用戶端因使用較舊的檢視器而不支援 SNI，檢視器回應的方式會取決於您為支援用戶端所選取的值：

- 所有用戶端：檢視器會因為 CloudFront 網域名稱與您 SSL/TLS 憑證中的網域名稱不同而顯示警告。
- 僅限支援伺服器名稱指示 (SNI) 的用戶端：CloudFront 失去與檢視器的連線而不傳回物件。

自訂 SSL 用戶端支援

僅當您在 SSL 憑證選擇自訂 SSL 憑證 (example.com) 時才適用。如果您為分佈指定了一或多個備用網域名稱和自訂 SSL 憑證，請選擇您要 CloudFront 提供 HTTPS 請求的方式：

- 支援伺服器名稱指示 (SNI) 的用戶端 - (建議) — 使用此設定，幾乎所有現代網頁瀏覽器和用戶端都可以連線到分佈，因為它們都支援 SNI。不過，有些檢視器可能會使用較舊的網頁瀏覽器或不支援 SNI 的用戶端，這表示他們無法連線至分佈。

若要使用 CloudFront API 套用此設定，請在 `SSLSupportMethod` 欄位中指定 `sni-only`。在 CloudFormation 中，欄位會命名為 `SslSupportMethod` (請注意大小寫)。

- 舊式用戶端支援 — 使用此設定，不支援 SNI 的舊式網頁瀏覽器和用戶端就可以連線到分佈。不過，此設定每月會產生額外費用。如需確切的價格，請移至 [Amazon CloudFront 定價](#) 頁面，然後搜尋專用 IP 自訂 SSL 的頁面。

若要使用 CloudFront API 套用此設定，請在 `SSLSupportMethod` 欄位中指定 `vip`。在 CloudFormation，欄位命名為 `SslSupportMethod`（請注意不同的大寫）。

如需詳細資訊，請參閱[選擇 CloudFront 提供 HTTPS 請求的方式](#)。

安全政策 (最低 SSL/TLS 版本)

指定您希望 CloudFront 用於與檢視器 (用戶端) 之 HTTPS 連線的安全政策。安全政策判斷兩個設定：

- CloudFront 用來與檢視器通訊的最低 SSL/TLS 通訊協定。
- 加密，供 CloudFront 用來加密傳回給檢視器的內容。

如需安全原則 (包括每個原則所包含的通訊協定和密碼) 的詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

可用的安全政策取決於您為 SSL Certificate (SSL 憑證) 和 Custom SSL Client Support (自訂 SSL 用戶端支援) (在 CloudFront API 中稱為 `CloudFrontDefaultCertificate` 和 `SSLSupportMethod`) 指定的值：

- 當 SSL 憑證為預設 CloudFront 憑證 (*.cloudfront.net) 時 (在 API 中則是 `CloudFrontDefaultCertificate` 為 `true`)，CloudFront 會自動將安全政策設定為 `TLSv1`。
- 當 SSL 憑證為自訂 SSL 憑證 (example.com)，且自訂 SSL 用戶端支援為支援伺服器名稱指示 (SNI) 的用戶端 - (建議使用) 時 (在 API 中則是 `CloudFrontDefaultCertificate` 為 `false` 且 `SSLSupportMethod` 為 `sni-only`)，您可以從下列安全政策中進行選擇：
 - `TLSv1.3_2025`
 - `TLSv1.2_2025`
 - `TLSv1.2_2021`
 - `TLSv1.2_2019`
 - `TLSv1.2_2018`
 - `TLSv1.1_2016`
 - `TLSv1_2016`
 - `TLSv1`
- 當 SSL 憑證為自訂 SSL 憑證 (example.com)，且自訂 SSL 用戶端支援為舊式用戶端支援時 (在 API 中則是 `CloudFrontDefaultCertificate` 為 `false` 且 `SSLSupportMethod` 為 `vip`)，您可以從下列安全政策中進行選擇：

- TLSv1
- SSLv3

在此組態中，無法在 CloudFront 主控台或 API 中使用

TLSv1.3_2025、TLSv1.2_2025、TLSv1.2_2021、TLSv1.2_2019、TLSv1.2_2018、TLSv1.1_2016 和 TLSv1_2016 安全政策。如果您想要使用這些安全政策之一，您可以選擇下列選項：

- 評估您的分佈是否需要具有專用 IP 位址的舊式用戶端支援。如果您的檢視器支援[伺服器名稱指示 \(SNI\)](#)，我們建議您將分佈的自訂 SSL 用戶端支援設定更新為支援伺服器名稱指示 (SNI) 的用戶端 (在 API 中則是將 `SSLSupportMethod` 設為 `sni-only`)。這可讓您使用任何可用的 TLS 安全政策，也可以降低 CloudFront 費用。
- 如果您必須保留具有專用 IP 位址的舊式用戶端支援，則可以在 [AWS 支援中心](#) 建立案例，請求其他 TLS 安全原則之一 (TLSv1.3_2025、TLSv1.2_2025、TLSv1.2_2021、TLSv1.2_2019、TLSv1.2_2018、TLSv1.1_2016 或 TLSv1_2016)。

Note

在您聯絡 AWS Support 請求此變更之前，請考慮下列事項：

- 當您將其中一個安全政策 (TLSv1.3_2025, TLSv1.2_2025, TLSv1.2_2021, TLSv1.2_2019, TLSv1.2_2018, TLSv1.1_2016, 或 TLSv1_2016) 新增至舊版用戶端支援分發時，安全政策會套用至您 AWS 帳戶中所有舊版用戶端支援分發的所有非 SNI 檢視器請求。不過，當檢視器將 SNI 請求傳送至具有舊式用戶端支援的分佈時，會套用該分佈的安全政策。若要確保將所需的安全政策套用至傳送至您 AWS 帳戶中所有 Legacy Client Support 分佈的所有瀏覽者請求，請將所需的安全政策個別新增至每個分佈。
- 根據定義，新的安全政策不支援舊版的相同密碼和通訊協定。例如，如果您選擇將分佈的安全政策從 TLSv1 升級到 TLSv1.1_2016，則該分佈將不再支援 DES-CBC3-SHA 加密。如需每個安全政策所支援之密碼和通訊協定的詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

支援的 HTTP 版本

選擇您要在檢視器與 CloudFront 通訊時，您的分佈支援的 HTTP 版本。

針對使用 HTTP/2 的檢視器和 CloudFront，檢視器必須支援 TLSv1.2 或更新版本，和伺服器名稱指示 (SNI)。

針對使用 HTTP/3 的檢視器和 CloudFront，檢視器必須支援 TLSv1.3 和伺服器名稱指示 (SNI)。CloudFront 支援 HTTP/3 連線遷移功能，可讓檢視器在不遺失連線的情況下切換網路。如需連線遷移的詳細資訊，請參閱在 RFC 9000 的[連線遷移](#)。

 Note

如需支援 TLSv1.3 密碼的詳細資訊，請參閱 [檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

 Note

如果您使用 Amazon Route 53，則可以使用 HTTPS 記錄，在用戶端支援通訊協定時允許通訊協定交涉做為 DNS 查詢的一部分。如需詳細資訊，請參閱[Create alias resource record set](#)。

預設根物件

選用。當檢視器請求分佈的根 URL 時 (index.html)，您希望 CloudFront 向原始伺服器請求的物件 (例如 <https://www.example.com/>)，而不是分佈中的物件 (<https://www.example.com/product-description.html>)。指定預設根物件可避免暴露分佈的內容。

該名稱的長度上限為 255 個字元。該名稱可以包含以下任何字元：

- A-Z、a-z
- 0-9
- _ - . * \$ / ~ ' "
- & 通過並以 & 傳回

當您指定預設根物件時，只需輸入物件名稱，例如，index.html。不要在物件名稱前新增 /。

如需詳細資訊，請參閱[指定預設根物件](#)。

標準記錄

指定是否希望 CloudFront 記錄有關物件的每個請求的資訊並儲存日誌檔案。您可以隨時啟用或停用記錄。如果您啟用記錄功能，將不會收取額外費用，但您可以累計費用來存放及存取檔案。您隨時都可刪除日誌。

CloudFront 支援下列標準記錄選項：

- [標準記錄 \(v2\)](#) – 您可以將日誌傳送至交付目的地，包括 Amazon CloudWatch Logs、Amazon Data Firehose 和 Amazon Simple Storage Service (Amazon S3)。
- [標準記錄 \(舊版\)](#) – 您只能將日誌傳送至 Amazon S3 儲存貯體。

連線日誌

當您開啟分佈的[交互身分驗證](#)時，CloudFront 會提供連線日誌，以擷取傳送至分佈之請求的屬性。連線日誌包含的資訊包括用戶端 IP 地址和連接埠、用戶端憑證資訊、連線結果和正在使用的 TLS 密碼。然後，這些連線日誌可用於檢閱請求模式和其他趨勢。

若要進一步了解連線日誌，請參閱[使用連線日誌的可觀測性](#)。

日誌字首

(選擇性) 若啟用標準記錄 (舊版)，請指定您希望 CloudFront 加在此分佈之存取日誌檔名稱之前的字串 (如果需要)，例如：exampleprefix/。結尾的斜線 (/) 是可選的，但建議簡化瀏覽日誌檔案。如需詳細資訊，請參閱[設定標準記錄 \(舊式\)](#)。

Cookie 記錄

如果您希望 CloudFront 在存取日誌中包含 Cookie，請選擇開啟。如果您選擇將 Cookie 包含在日誌中，那麼不管您如何為此分佈配置快取行為，CloudFront 都會記錄所有 Cookie：轉送所有 Cookie，不轉送 Cookie 或將指定的 Cookie 清單轉送至原始伺服器原始伺服器。

Amazon S3 不處理 Cookie，因此除非您的分佈還包含 Amazon EC2 或其他自訂原始伺服器原始伺服器，我們建議您選擇關閉做為 Cookie 記錄的值。

如需 Cookie 的詳細資訊，請參閱[根據 Cookie 快取內容](#)。

啟用 IPv6 (檢視器請求)

如果您希望 CloudFront 回應來自 IPv4 和 IPv6 IP 位址的檢視器請求，請選取啟用 IPv6。如需詳細資訊，請參閱[為 CloudFront 分佈啟用 IPv6](#)。

交互身分驗證

選用。您可以選擇為 CloudFront 分佈開啟交互身分驗證。如需詳細資訊，請參閱[相互 TLS \(mTLS\) 檢視器](#)。

為自訂原始伺服器啟用 IPv6 (原始伺服器請求)

當您使用自訂原始伺服器 (不含 Amazon S3 和 VPC 原始伺服器) 時，您可以自訂分佈的原始伺服器設定，以選擇 CloudFront 如何使用 IPv4 或 IPv6 位址連接到原始伺服器。如需詳細資訊，請參閱[為 CloudFront 分佈啟用 IPv6](#)。

Comment

選用。當您建立分佈，您可以包含高達 128 個字元的評論。您隨時都可以更新評論。

分佈狀態

表示是否要在部署後啟用或停用該分佈：

- 啟用表示一旦分佈完全部署後，您就可以部署使用分佈的網域名稱的連結，使用者可以擷取內容。當分佈被啟用時，CloudFront 接受並處理使用與該分佈關聯的網域名稱的內容的任何最終使用者請求。

當建立、修改或刪除 CloudFront 分佈時，仍需要時間將您的變更傳播到 CloudFront 資料庫。對分佈資訊的立即請求可能不會顯示其變更。傳輸通常可在幾分鐘內完成，但此時可能會增加高系統負載或網路分區。

- 停用表示即使分佈可能已部署並可供使用，但使用者無法使用它。每當分佈停用時，CloudFront 不接受使用與該分佈關聯的網域名稱的任何最終使用者請求。在將分佈從停用狀態切換到啟用狀態之前 (透過更新分佈的組態)，沒有人可以使用它。

您可以根據需要隨時在停用和啟用之間切換分佈。依照更新分佈組態處理。如需詳細資訊，請參閱[更新分佈](#)。

自訂錯誤頁面和錯誤快取

當您的 Amazon S3 或自訂原始伺服器原始伺服器向 CloudFront 傳回 HTTP 4xx 和 5xx 狀態程式碼時，您可以讓 CloudFront 將物件傳回到檢視器 (例如 HTML 檔案)。關於來自原始伺服器原始伺服器或自訂錯誤頁面的錯誤回應，您還可以指定該錯誤回應在 CloudFront 邊緣快取中的時間。如需詳細資訊，請參閱[針對特定的 HTTP 狀態碼建立自訂錯誤頁面](#)。

Note

在建立分佈精靈中不包含以下值，因此只在您更新分佈時才能設定自訂錯誤頁面。

主題

- [HTTP 錯誤代碼](#)
- [回應頁面路徑](#)
- [HTTP 回應代碼](#)
- [錯誤快取最短 TTL \(秒\)](#)

HTTP 錯誤代碼

您希望 CloudFront 傳回自訂錯誤頁面的 HTTP 狀態碼。您可以將 CloudFront 配置為針對 CloudFront 快取的 HTTP 狀態碼 (包括無、一些或全部) 傳回自訂錯誤頁面。

回應頁面路徑

當您的原始伺服器傳回您為錯誤代碼指定的 HTTP 狀態碼 (例如 403) 時，您希望 CloudFront 返回到檢視器的自訂錯誤頁面的路徑 (例如 /4xx-errors/403-forbidden.html)。如果您要在不同位置存放物件和自訂錯誤頁面，則您的分佈必須包含下列為屬實的快取行為：

- Path Pattern (路徑模式) 的值與自訂錯誤訊息的路徑相符。例如，假設您已在 Amazon S3 儲存貯體名為 /4xx-errors 的目錄中儲存了 4xx 錯誤的自訂錯誤頁面。您的分佈必須包含快取行為，其路徑模式會將自訂錯誤頁面的請求路由至該位置，例如 /4xx-errors/* (/4xx-errors/*)。
- Origin (原始伺服器) 的數值將 Origin ID (原始伺服器 ID) 的數值指定給包含自訂錯誤頁面的原始伺服器。

HTTP 回應代碼

您希望 CloudFront 傳回給檢視器的 HTTP 狀態碼以及自訂錯誤頁面。

錯誤快取最短 TTL (秒)

您希望 CloudFront 從原始伺服器快取錯誤回應的最短時間。

地理限制

若您要防止選取的國家/地區的使用者存取您的內容，則可以使用允許清單或封鎖清單配置您的 CloudFront 分佈。設定地理限制無需額外收費。如需詳細資訊，請參閱[限制您內容的地理分佈](#)。

測試分佈

建立分佈後，CloudFront 知道原始伺服器的位置，並且知道與分佈關聯的網域名稱。若要測試您的分佈，請執行下列動作：

1. 等待您的分佈部署完成。
 - 在主控台中檢視您的分佈詳細資訊。您的分佈完成部署時，上次修改欄位會從正在部署變更為日期和時間。
2. 使用下列程序，透過 CloudFront 網域名稱建立物件的連結。
3. 測試連結。CloudFront 會將物件提供給您的網頁或應用程式。

建立物件的連結

使用下列程序為 CloudFront Web 分佈中的物件建立測試連結。

在 Web 分佈中建立連結到物件

1. 將以下 HTML 程式碼複製到新的檔案中，以您的分佈的網域名稱取代 *domain-name*，然後以您的物件的名稱取代 *object-name*。

```
<html>
<head>
  <title>My CloudFront Test</title>
</head>
<body>
  <p>My text content goes here.</p>
  <p></p>
</body>
</html>
```

例如，如果您的網域名稱是 d111111abcdef8.cloudfront.net，物件是 image.jpg，則連結的 URL 將為：

`https://d111111abcdef8.cloudfront.net/image.jpg.`

如果物件位於原始伺服器的資料夾中，則該資料夾必須包含在 URL 中。例如，如果 image.jpg 位於原始伺服器上的影像資料夾中，則 URL 將為：

`https://d111111abcdef8.cloudfront.net/images/image.jpg`

2. 將 HTML 程式碼儲存在副檔名為 .html 的檔案中。
3. 在瀏覽器中開啟您的網頁，以確保您可以查看物件。

瀏覽器傳回具有從 CloudFront 判斷適用來提供物件的節點所提供的內嵌圖片檔案的網頁。

更新分佈

在 CloudFront 主控台中，您可看到與 AWS 帳戶相關的 CloudFront 分佈、檢視分佈設定和更新大多數的設定。請注意，在分佈傳播到 AWS 節點之前，您所做的更新不會生效。

在主控台中更新分佈

下列程序說明如何在主控台中更新 CloudFront 分佈。

Multi-tenant

更新多租用戶分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 搜尋並選擇多租用戶分佈的 ID。
3. 選擇您要更新之設定的索引標籤。
4. 進行更新，接著若要儲存變更，請選擇儲存變更。如需詳細了解您可以更新的設定，請參閱 [預先設定的分佈設定參考](#)。

您也可以使用 CloudFront API 更新分佈：

- 若要更新分佈，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

Important

當您更新您的分佈時，請注意有多個額外的必要欄位，這些欄位在初次建立分佈時是非必要的。為協助確保在您使用 CloudFront API 更新分佈時已包含所有必要欄位，請依照 Amazon CloudFront API 參考中的 [UpdateDistribution](#) 所述步驟執行。

若要變更分佈租用戶的多租用戶分佈，請更新分佈租用戶。您也可以更新分佈租用戶，以更新其網域、憑證、自訂或參數值。如需更新分佈租用戶憑證的詳細資訊，請參閱 [新增網域和憑證 \(分佈租用戶\)](#)。

更新分佈租用戶

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 SaaS 下，選擇分佈租用戶。
3. 搜尋分佈租用戶。使用搜尋列中的下拉式功能表，並且依網域、名稱、分佈 ID、憑證 ID、連線群組 ID 或 Web ACL ID 進行篩選。
4. 選擇分佈租用戶的名稱。
5. 若要更新一般詳細資訊，請選擇編輯、進行更新，然後選擇更新分佈租用戶。
6. 選擇相應的索引標籤以更新其他設定，完成更新後儲存變更。如需您可自訂之分布租用戶設定的詳細資訊，請參閱 [分佈租用戶自訂](#)。

Standard

更新標準分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選取分佈的 ID。此清單包括與您用於登入 CloudFront 主控台之 AWS 帳戶相關聯的所有分發。
3. 若要更新一般設定，請選擇 Edit (編輯)。否則，請選擇您要更新之設定的索引標籤。
4. 進行更新，然後選擇儲存變更。如需欄位的詳細資訊，請參閱下列主題：
 - 一般設定：[分佈設定](#)
 - 原始伺服器設定：[原始設定](#)
 - 快取行為設定：[快取行為設定](#)
5. 如果您想要在分佈中刪除原始伺服器，請執行下列動作：
 - a. 選擇 Behaviors (行為)，然後確認您已將任何與原始伺服器相關的預設快取行為移動至另一個原始伺服器。
 - b. 選擇 Origins (原始伺服器)，然後選取一個原始伺服器。
 - c. 選擇 Delete (刪除)。

您也可以使用 CloudFront API 更新分佈：

- 若要更新分佈，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

Important

當您更新您的分佈時，請注意有多個額外的必要欄位，這些欄位在建立分佈時是非必要的。為協助確保在您使用 CloudFront API 更新分佈時已包含所有必要欄位，請依照 Amazon CloudFront API 參考中的 [UpdateDistribution](#) 所述步驟執行。

當您儲存變更至分佈組態時，CloudFront 開始將變更傳播到所有節點。連續的組態變更依照其各自的順序傳播。直到更新您節點中的組態，CloudFront 才根據之前的組態繼續從該位置提供內容。更新您節點中的組態後，CloudFront 便立即根據新組態從該位置提供內容。

您的變更不會同時傳播到每個邊緣節點。儘管 CloudFront 會傳播您的變更，我們仍無法根據先前的組態或新組態來判斷指定邊緣節點是否提供內容。

Note

在極少數情況下，當主機或網路連結中斷時，某些分佈租用戶流量可能會使用較舊的組態短暫提供，直到您的變更通過網路為止。

若要查看您的變更何時傳播，請在主控台中檢視您的分佈詳細資訊。上次修改欄位會從部署中變更為部署完成時的日期和時間。

標記分佈

標籤是您可以用來識別和組織 AWS 資源的單字或片語。您可以為每個資源新增多個標籤，每個標籤都包含您定義的索引鍵和值。例如，金鑰可能是「網域」，而值可能是「example.com」。您可以根據您新增的標籤來搜尋和篩選資源。

您可以搭配 CloudFront 使用標籤，例如下列範例：

- 在 CloudFront 分佈上強制使用以標籤為基礎的許可。如需詳細資訊，請參閱 [ABAC 搭配 CloudFront](#)。

- 追蹤不同類別中的帳單資訊。當您將標籤套用至 CloudFront 分佈或其他 AWS 資源（例如 Amazon EC2 執行個體或 Amazon S3 儲存貯體）並啟用標籤時，會以逗號分隔值 (CSV 檔案) 的形式 AWS 產生成本分配報告，其中包含作用中標籤彙總的用量和成本。

您可以套用代表業務類別 (例如成本中心、應用程式名稱或擁有者) 的標籤，來整理多個服務中的成本。如需有關使用成本分配標籤的詳細資訊，請參閱 AWS Billing 使用者指南中的[使用成本分配標籤](#)。

備註

- 您可以標記分佈，但無法標記原始存取身分或失效。
- [Tag Editor](#) 和 [Resource groups](#) 目前不支援 CloudFront。
- 如需了解目前可新增到分發的標籤數量上限的詳細資訊，請參閱[一般配額](#)。

內容

- [標籤限制](#)
- [新增、編輯和刪除分佈的標籤](#)
- [程式設計標記](#)

標籤限制

以下基本限制適用於 標籤：

- 如需每個分佈的標籤數量上限，請參閱 [一般配額](#)。
- 金鑰長度上限 - 128 個 Unicode 字元
- 數值長度上限 - 256 個 Unicode 字元
- 金鑰與值的有效值 – a-z、A-Z、0-9、空格和下列字元：_ . : / = + - 及 @
- 標記金鑰與值皆區分大小寫
- 請不要使用 aws：做為索引鍵的字首。此字首已保留供 AWS 使用。

新增、編輯和刪除分佈的標籤

您可以使用 CloudFront 主控台來管理分佈的標籤。

若要為分佈新增標籤、編輯或刪除標籤

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇您希望更新的分佈 ID。
3. 選擇 Tags (標籤) 索引標籤。
4. 選擇管理標籤。
5. 在 Manage tags (管理標籤) 頁面上，可以執行下列操作：
 - 若要新增標籤，請鍵入一個金鑰，並輸入標籤的值 (選擇性)。選擇新增新標籤以新增更多標籤。
 - 若要編輯標籤，請變更標籤的金鑰或其值，或兩者均變更。可以刪除標籤的值，但仍須金鑰。
 - 若要刪除標籤，請選擇 Remove (移除)。
6. 選擇儲存變更。

程式設計標記

您也可以使用 CloudFront API、AWS Command Line Interface (AWS CLI)、AWS SDKs 和 AWS Tools for Windows PowerShell 來套用標籤。如需詳細資訊，請參閱下列主題：

- CloudFront API 操作：
 - [ListTagsForResource](#)
 - [TagResource](#)
 - [UntagResource](#)
- AWS CLI – 請參閱《AWS CLI 命令參考》中的 [cloudfront](#)
- AWS SDKs – 請參閱 文件頁面上適用的開發套件 [AWS 文件](#)
- Tools for Windows PowerShell – 請參閱 [AWS Tools for PowerShell Cmdlet 參考](#) 中的 [Amazon CloudFront](#)

刪除 分發

下列程序會使用 CloudFront 主控台刪除分佈。如需使用 CloudFront API 刪除的資訊，請參閱《Amazon CloudFront API 參考》中的 [DeleteDistribution](#)。

如果您需要刪除連接到 S3 儲存貯體的 OAC 分佈，請參閱 [刪除含附加至 S3 儲存貯體之 OAC 的分佈](#) 以取得重要詳細資訊。

Warning

- 您必須先停用分佈，然後才能刪除它，而這需要更新分佈的許可。分佈一旦刪除，即無法復原。
- 若您停用具有與其建立關聯備用網域名稱的分佈，CloudFront 會停止接受該網域名稱 (例如 `www.example.com`) 的流量，即使另一個分佈具有包含萬用字元 (*) 的其他網域名稱，符合相同的網域 (例如 `*.example.com`) 也一樣。

Multi-tenant

您必須先刪除所有關聯的分佈租用戶，才能刪除多租用戶分佈。

刪除多租用戶分佈

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 CloudFront 主控台的右側窗格中，選取您要刪除的多租用戶分佈名稱。
3. 針對租用戶，選取並刪除所有相關聯的分佈租用戶。
4. 選擇 停用以停用分佈，然後選擇停用分佈以確認。
5. 等到新的時間戳記出現在上次修改的資料欄下方。
 - CloudFront 可能需要幾分鐘的時間將您的變更傳播到所有邊緣節點。
6. 選擇刪除、刪除分佈。

刪除分佈租用戶

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 SaaS 下，選擇分佈租用戶。
3. 搜尋分佈租用戶。使用搜尋列中的下拉式功能表，並且依網域、名稱、分佈 ID、憑證 ID、連線群組 ID 或 Web ACL ID 進行篩選。
4. 選取要刪除的分佈租用戶。
5. 選擇刪除租用戶、刪除分佈租用戶。

Standard

刪除標準分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在適當的 CloudFront 主控台窗格中，請找到您要刪除的分佈。
 - 如果狀態欄顯示分佈已停用，請跳至步驟 6。
 - 如果狀態顯示已啟用，但分佈仍然在上次修改欄中顯示部署中，請等待部署完成，然後再繼續步驟 3。
3. 在 CloudFront 主控台的右側窗格中，選取您要刪除的分佈核取方塊。
4. 選擇 Disable (停用) 來停用分佈，然後選擇 Yes, Disable (是，停用) 以進行確認。然後，選擇 Close (關閉)。
 - 狀態欄的值會立即變更為已停用。
5. 等到新的時間戳記出現在上次修改的資料欄下方。
 - CloudFront 可能需要幾分鐘的時間將您的變更傳播到所有邊緣節點。
6. 選擇您要刪除的分佈的核取方塊。
7. 選擇刪除，刪除。
 - 如果無法使用刪除選項，表示 CloudFront 仍在將您的變更傳播到邊緣節點。等到新的時間戳記出現在上次修改的資料欄下，然後重複步驟 6-7。

將各種原始伺服器與 CloudFront 分佈搭配使用

建立分佈後，您必須指定 CloudFront 為檔案傳送請求的原始伺服器。CloudFront 可以與多種不同類型的原始伺服器搭配使用，例如，您可以使用 Amazon S3 儲存貯體、MediaStore 容器、MediaPackage 頻道、Application Load Balancer 或 AWS Lambda 函數 URL。當您建立 CloudFront 分佈時，CloudFront 會根據您的內容原始伺服器類型設定大部分的分佈設定。如需詳細資訊，請參閱[預先設定的分佈設定參考](#)。

如果您在私有子網路中有 Application Load Balancer、Network Load Balancer 或 EC2 執行個體，則可以將其用作 VPC 原始伺服器。使用 VPC 原始伺服器時，您的應用程式只能在具有 CloudFront 分佈的私有子網路中存取，以防止您的應用程式在公有網際網路上存取。如需詳細資訊，請參閱[the section called “使用 VPC 原始伺服器限制存取”](#)。

 Note

您可以使用邊緣函數，為每個請求動態選取適當的原始伺服器。透過使用 CloudFront Functions 或 Lambda@Edge，您可以根據檢視器的地理位置、請求標頭或查詢字串參數等因素，將請求路由到不同的原始伺服器。如需詳細資訊，請參閱[使用函數在邊緣自訂](#)。

主題

- [使用 Amazon S3 儲存貯體](#)
- [使用 MediaStore 容器或 MediaPackage 頻道](#)
- [使用 Application Load Balancer](#)
- [使用 Network Load Balancer](#)
- [使用 Lambda 函數 URL](#)
- [使用 Amazon EC2 \(或其他自訂原始伺服器\)](#)
- [使用 CloudFront 原始伺服器群組](#)
- [使用 Amazon API Gateway](#)

使用 Amazon S3 儲存貯體

下列主題介紹了多種方法，可協助您將 Amazon S3 儲存貯體用做 CloudFront 分佈的原始伺服器。

主題

- [使用標準 Amazon S3 儲存貯體](#)
- [使用 Amazon S3 Object Lambda](#)
- [使用 Amazon S3 Access Point](#)
- [使用設定為網站端點的 Amazon S3 儲存貯體](#)
- [將 CloudFront 新增至現有 Amazon S3 儲存貯體](#)
- [將 Amazon S3 儲存貯體移至不同的 AWS 區域](#)

使用標準 Amazon S3 儲存貯體

使用 Amazon S3 做為分佈的原始伺服器時，可以將想讓 CloudFront 傳送的物件放置在 Amazon S3 儲存貯體中。您可以使用 Amazon S3 支援的任何方法將物件放入 Amazon S3，例如 Amazon S3 主控台

或 API，也可使用第三方工具。如同使用其他標準 Amazon S3 儲存貯體一樣，您可以在儲存貯體中建立階層結構以存放物件。

使用現有 Amazon S3 儲存貯體做為您的 CloudFront 原始伺服器，不會以任何方式變更儲存貯體；您可以繼續像平常一樣按標準 Amazon S3 價格使用它來存放和存取 Amazon S3 物件。您需要定期支付 Amazon S3 費用才能將物件存放在儲存貯體中。如需 CloudFront 服務使用費的詳細資訊，請參閱 [Amazon CloudFront 定價](#)。如需進一步瞭解如何將 CloudFront 與現有 S3 儲存貯體搭配使用，請參閱 [the section called “將 CloudFront 新增至現有 Amazon S3 儲存貯體”](#)。

Important

為了讓您的儲存貯體使用 CloudFront，該名稱必須符合 DNS 命名請求。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的 [儲存貯體命名規則](#)。

將 Amazon S3 儲存貯體指定為 CloudFront 的原始伺服器時，建議使用以下格式：

bucket-name.s3.*region*.amazonaws.com

當您以此格式指定儲存貯體名稱，可以使用以下 CloudFront 功能：

- 將 CloudFront 設定為使用 SSL/TLS 與 Amazon S3 儲存貯體進行通訊。如需詳細資訊，請參閱 [the section called “透過 CloudFront 使用 HTTPS”](#)。
- 使用原始存取控制來請求檢視器使用 CloudFront URL 存取您的內容，而不是使用 Amazon S3 URL。如需詳細資訊，請參閱 [the section called “限制對 Amazon S3 原始伺服器的存取”](#)。
- 透過向 CloudFront 提交 POST 和 PUT 請求，更新您的儲存貯體的內容。如需詳細資訊，請參閱 [the section called “CloudFront 如何處理和轉送請求至您的 Amazon S3 原始伺服器”](#) 主題中的 [the section called “HTTP 方法”](#)。

請勿使用以下格式指定儲存貯體：

- Amazon S3 路徑樣式：s3.amazonaws.com/*bucket-name*
- Amazon S3 CNAME

Note

CloudFront 支援使用任何儲存類別的 S3 原始伺服器，包括 S3 Intelligent-Tiering。當 CloudFront 從 S3 原始伺服器請求物件時，無論物件目前位於哪個儲存層，都會進行物件擷

取。搭配 S3 Intelligent-Tiering 使用 CloudFront 不會影響分佈的效能或功能。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的使用 [Amazon S3 Intelligent-Tiering 管理儲存成本](#)。

使用 Amazon S3 Object Lambda

在您[建立 Object Lambda 存取點](#)時，Amazon S3 會自動為您的 Object Lambda 存取點產生唯一的別名。您可以[使用此別名](#)，而非 Amazon S3 儲存貯體名稱作為 CloudFront 發佈的原始伺服器。

在您將 Object Lambda 存取點別名用作 CloudFront 的原始伺服器時，建議使用以下格式：

alias.s3.*region*.amazonaws.com

如需有關尋找 *alias* 的詳細資訊，請參閱《Amazon S3 使用者指南》中的[如何為您的 S3 儲存貯體 Object Lambda 存取點使用儲存貯體式別名](#)。

Important

在您使用 Object Lambda 存取點做為 CloudFront 的原始伺服器時，您必須使用[原始伺服器存取控制](#)。

如需範例使用案例，請參閱[將 Amazon S3 Object Lambda 與 Amazon CloudFront 搭配使用，以為最終使用者量身打造內容](#)。

CloudFront 會將 Object Lambda 存取點原始伺服器視為與[標準 Amazon S3 儲存貯體原始伺服器](#)相同。

如您要使用 Amazon S3 Object Lambda 作為分佈的原始伺服器，則必須設定以下四個許可。

Object Lambda Access Point

新增 Object Lambda 存取點的許可

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/>：// 開啟 Amazon S3 主控台。
2. 在導覽窗格中，選擇 物件 Lambda 存取點。
3. 選擇您要使用的 Object Lambda 存取點。
4. 選擇許可索引標籤。

5. 在 物件 Lambda 存取點政策 部分中選擇 編輯。
6. 將以下政策貼入 政策 欄位。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "s3-object-lambda:Get*",
      "Resource": "arn:aws:s3-object-lambda:us-east-1:123456789012:accesspoint/Object-Lambda-Access-Point-name",
      "Condition": {
        "StringEquals": {
          "aws:SourceArn":
            "arn:aws:cloudfront::123456789012:distribution/CloudFront-distribution-ID"
        }
      }
    }
  ]
}
```

7. 選擇儲存變更。

Amazon S3 Access Point

新增 Amazon S3 Access Point 的許可

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/>：// 開啟 Amazon S3 主控台。
2. 在導覽窗格中，選擇 存取點。
3. 選擇您要使用的 Amazon S3 存取點。
4. 選擇許可索引標籤。
5. 在 存取點政策 部分中選擇 編輯。

6. 將以下政策貼入 政策 欄位。

JSON

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "s3objlambda",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-Point-name",
        "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-Point-name/object/*"
      ],
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "s3-object-lambda.amazonaws.com"
        }
      }
    }
  ]
}
```

7. 選擇儲存。

Amazon S3 bucket

新增許可至 Amazon S3 儲存貯體

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/>：// 開啟 Amazon S3 主控台。
2. 在導覽窗格中，選擇 儲存貯體。
3. 選擇您要使用的 Amazon S3 儲存貯體。

4. 選擇許可索引標籤。
5. 在 儲存貯體政策 區段中，選擇 編輯。
6. 將以下政策貼入 政策 欄位。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": [
        "arn:aws:s3:::bucket-name",
        "arn:aws:s3:::bucket-name/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:DataAccessPointAccount": "AWS-account-ID"
        }
      }
    }
  ]
}
```

7. 選擇儲存變更。

AWS Lambda function

新增許可至 Lambda 函數

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/lambda/> 開啟 AWS Lambda 主控台。
2. 在導覽視窗中，選擇函數。
3. 選擇您要使用的 AWS Lambda 函數。
4. 依序選擇 組態 索引標籤和 許可。

5. 在 基於資源的政策聲明 區段中選擇 新增許可。
6. 選擇 AWS 帳戶。
7. 輸入 聲明 ID 的名稱。
8. 在主體 輸入 `cloudfront.amazonaws.com`。
9. 從 動作 下拉式選單中選擇 `lambda:InvokeFunction`。
10. 選擇儲存。

使用 Amazon S3 Access Point

當您[使用 S3 Access Point](#) 時，Amazon S3 會自動為您產生唯一的別名。您可以使用此別名，而非 Amazon S3 儲存貯體名稱作為 CloudFront 發佈的原始伺服器。

在您將 Amazon S3 Access Point 別名用作 CloudFront 的原始伺服器時，建議使用以下格式：

alias.s3.*region*.amazonaws.com

如需有關尋找 *alias* 的詳細資訊，請參閱《Amazon S3 使用者指南》中的[為您的 S3 儲存貯體存取點使用儲存貯體式別名](#)。

Important

在您使用 Amazon S3 Access Point 做為 CloudFront 的原始伺服器時，您必須使用[原始存取控制](#)。

CloudFront 會將 Amazon S3 Access Point 原始伺服器視為與[標準 Amazon S3 儲存貯體原始伺服器](#)相同。

如您要使用 Amazon S3 Object Lambda 作為分佈的原始伺服器，則必須設定以下二個許可。

Amazon S3 Access Point

新增 Amazon S3 Access Point 的許可

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/> 開啟 Amazon S3 主控台。
2. 在導覽窗格中，選擇 存取點。
3. 選擇您要使用的 Amazon S3 存取點。

4. 選擇許可索引標籤。
5. 在 存取點政策 部分中選擇 編輯。
6. 將以下政策貼入 政策 欄位。

JSON

```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "s3objlambda",
      "Effect": "Allow",
      "Principal": {"Service": "cloudfront.amazonaws.com"},
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-Point-name",
        "arn:aws:s3:us-east-1:123456789012:accesspoint/Access-Point-name/object/*"
      ],
      "Condition": {
        "StringEquals": {"aws:SourceArn": "arn:aws:cloudfront::123456789012:distribution/CloudFront-distribution-ID"}
      }
    }
  ]
}
```

7. 選擇儲存。

Amazon S3 bucket

新增許可至 Amazon S3 儲存貯體

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/s3/> 開啟 Amazon S3 主控台。
2. 在導覽窗格中，選擇 儲存貯體。
3. 選擇您要使用的 Amazon S3 儲存貯體。

4. 選擇許可索引標籤。
5. 在 儲存貯體政策 區段中，選擇 編輯。
6. 將以下政策貼入 政策 欄位。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": [
        "arn:aws:s3::bucket-name",
        "arn:aws:s3:::bucket-name/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:DataAccessPointAccount": "AWS-account-ID"
        }
      }
    }
  ]
}
```

7. 選擇儲存變更。

使用設定為網站端點的 Amazon S3 儲存貯體

您可以將設定為網站端點的 Amazon S3 儲存貯體當做自訂原始伺服器，與 CloudFront 搭配使用。當您配置 CloudFront 分佈時，在原始伺服器中為您的儲存貯體輸入 Amazon S3 靜態網站託管端點。這個值會顯示在 [Amazon S3 主控台](#) Properties (屬性) 索引標籤中的 Static website hosting (靜態網站託管) 窗格裡。例如：

`http://bucket-name.s3-website-region.amazonaws.com`

如需指定 Amazon S3 靜態網站端點的詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的[網站端點](#)。

當您以此格式指定儲存貯體名稱做為您的原始伺服器時，可以使用 Amazon S3 重新引導和 Amazon S3 自訂錯誤文件。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的[設定自訂錯誤文件](#)和[設定重新引導](#)部分。(CloudFront 還提供自訂錯誤頁面。如需詳細資訊，請參閱[the section called “針對特定的 HTTP 狀態碼建立自訂錯誤頁面”](#)。)

使用 Amazon S3 儲存貯體做為 CloudFront 的原始伺服器，不會導致儲存貯體出現任何變動。您可以如一般情況般繼續使用，且需支付一般 Amazon S3 費用。如需 CloudFront 服務使用費的詳細資訊，請參閱 [Amazon CloudFront 定價](#)。

Note

如果您使用 CloudFront API 來透過已設為網站端點的 Amazon S3 儲存貯體建立分發，您必須使用 CustomOriginConfig 來配置，雖然網站託管於 Amazon S3 儲存貯體。如需使用 CloudFront API 建立分佈的詳細資訊，請參閱 Amazon CloudFront API 參考中的 [CreateDistribution](#)。

將 CloudFront 新增至現有 Amazon S3 儲存貯體

如果您將物件存放於 Amazon S3 儲存貯體，您可以讓使用者直接從 S3 取得您的物件，或配置 CloudFront 從 S3 取得物件，然後將它們分配給使用者。如果您的使用者經常存取您的物件，則使用 CloudFront 更具成本效益，因為在較高用量，CloudFront 資料傳輸的價格低於 Amazon S3 資料傳輸的價格。此外，使用 CloudFront 下載的速度比單獨使用 Amazon S3 要快，因為物件存放在較靠近使用者的位置。

Note

如果您希望 CloudFront 尊重 Amazon S3 跨原始伺服器資源共享配置，請配置 CloudFront 將 Origin 標頭轉送到 Amazon S3。如需詳細資訊，請參閱[the section called “根據請求標頭快速存取內容”](#)。

如果您目前使用自己的網域名稱 (例如 example.com) 直接從 Amazon S3 儲存貯體分配內容，而非從 Amazon S3 儲存貯體的網域名稱 (例如 amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com)，您可以使用下列程序不受干擾的新增 CloudFront。

在您已經從 Amazon S3 分佈內容時新增 CloudFront

1. 建立 CloudFront 分佈。如需詳細資訊，請參閱[the section called “建立分發”](#)。

當您建立分佈時，請指定 Amazon S3 儲存貯體的名稱做為原始伺服器。

 Important

為了讓您的儲存貯體使用 CloudFront，該名稱必須符合 DNS 命名請求。如需詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的[儲存貯體命名規則](#)。

如果您將 Amazon S3 與 CNAME 搭配使用，也請為您的分佈指定 CNAME。

2. 建立一個在 Amazon S3 儲存貯體中包含公開可讀取物件的連結之測試 Web 頁面，並測試該連結。針對此初始測試，請使用物件 URL 中分佈的 CloudFront 網域名稱，例如 `https://d111111abcdef8.cloudfront.net/images/image.jpg`。

如需 CloudFront URL 格式的詳細資訊，請參閱 [the section called “自訂檔案 URL”](#)。

3. 如果您目前使用 Amazon S3 CNAME，您的應用程式會使用網域名稱 (例如 `example.com`) 參照 Amazon S3 儲存貯體中的物件，而不是使用儲存貯體的名稱 (例如 `amzn-s3-demo-bucket.s3.amazonaws.com`)。若要繼續使用您的網域名稱來參考物件，而不是使用分佈的 CloudFront 網域名稱 (例如 `d111111abcdef8.cloudfront.net`)，您需要透過 DNS 服務供應商來更新設定。

針對使用 Amazon S3 CNAME，您的 DNS 服務供應商必須有網域的 CNAME 資源紀錄集，該記錄目前將網域查詢路由到您的 Amazon S3 儲存貯體。例如，如果使用者請求該物件：

`https://example.com/images/image.jpg`

該請求會自動重新路由，且使用者看到此物件：

`https://amzn-s3-demo-bucket.s3.amazonaws.com/images/image.jpg`

若要路由查詢到 CloudFront 分佈，而非 Amazon S3 儲存貯體，您需要使用 DNS 服務供應商提供的方法來更新網域的 CNAME 資源紀錄集。這項經過更新的 CNAME 記錄會將 DNS 查詢從您的網域重新引導到您的分佈的 CloudFront 網域名稱。如需詳細資訊，請參閱 DNS 服務供應商提供的說明文件。

 Note

如果您使用 Route 53 做為 DNS 服務，您可以使用 CNAME 資源紀錄集或別名資源紀錄集。如需編輯資源紀錄集的詳細資訊，請參閱[編輯記錄](#)。如需別名資源紀錄集的詳細資訊，請參閱[選擇別名或非別名記錄](#)。這兩個主題都在 Amazon Route 53 開發人員指南中。

如需搭配 CloudFront 使用 CNAME 的詳細資訊，請參閱 [the section called “使用自訂 URL”](#)。

在您更新 CNAME 資源紀錄集後，傳播變更到整個 DNS 系統需要 72 小時，但通常更快。這段期間內，會持續將一些內容的請求路由到 Amazon S3 儲存貯體，並將其他的內容路由到 CloudFront。

將 Amazon S3 儲存貯體移至不同的 AWS 區域

如果您使用 Amazon S3 做為 CloudFront 分佈的原始伺服器，並將儲存貯體移至不同的 AWS 區域，則當下列兩者都成立時，CloudFront 最多可能需要一小時的時間來更新其記錄以使用新區域：

- 您使用 CloudFront 原始存取身分 (OAI) 限制對儲存貯體的存取。
- 您將儲存貯體移至需要簽章版本 4 進行身分驗證的 Amazon S3 區域。

當您使用 OAI 時，CloudFront 使用區域 (其他值中)，以計算用來從儲存貯體請求物件的簽章。如需 OAI 的詳細資訊，請參閱[the section called “使用原始存取身分 \(舊版，不建議使用\)”](#)。如需支援 Signature 第 2 版 AWS 區域的清單，請參閱中的 [Signature 第 2 版簽署程序](#) Amazon Web Services 一般參考。

若要強制執行更快速的 CloudFront 記錄更新，您可以更新 CloudFront 分佈，例如更新 CloudFront 主控台中 General (一般) 索引標籤裡的 Description (描述) 欄位；您更新分佈後，CloudFront 便會立即檢查儲存貯體所在的區域。將變更傳播到所有節點只需幾分鐘即可完成。

使用 MediaStore 容器或 MediaPackage 頻道

若要使用 CloudFront 串流視訊，您可以設定已配置為 MediaStore 容器的 Amazon S3 儲存貯體，或使用 MediaPackage 建立頻道和端點。然後，您可在 CloudFront 中建立及配置分佈來串流視訊。

如需詳細資訊和逐步說明，請參閱以下主題：

- [the section called “使用 AWS Elemental MediaStore 做為原始伺服器來提供影片”](#)
- [the section called “提供以 AWS Elemental MediaPackage 格式化的即時視訊”](#)

使用 Application Load Balancer

您可以使用 CloudFront 將流量路由到內部和面向網際網路的 Application Load Balancer。

如果您的原始伺服器是託管在一或多個 Amazon EC2 執行個體上的一或多個 HTTP 伺服器 (Web 伺服器)，您可以使用面向網際網路的 Application Load Balancer 將流量分配到執行個體。面對網際網路的負載平衡器具有可公開解析的 DNS 名稱，並會透過網際網路將用戶端的請求路由至目標。

如想進一步了解如何使用面向網際網路的 Application Load Balancer 做為 CloudFront 的原始伺服器，包括如何確保檢視器只能透過 CloudFront (無法直接透過負載平衡器) 存取您的 web 伺服器，請參閱 [the section called “限制對 Application Load Balancers 的存取”](#)。

或者，您可以使用 VPC 原始伺服器，從虛擬私有雲端 (VPC) 私有子網路中以內部 Application Load Balancer 託管的應用程式交付內容。VPC 原始伺服器可防止您的應用程式在公有網際網路上被存取。如需詳細資訊，請參閱 [使用 VPC 原始伺服器限制存取](#)。

使用 Network Load Balancer

您可以將內部和面向網際網路的 Network Load Balancer 與 Amazon CloudFront 搭配使用。您可以使用 VPC 原始伺服器，將私有子網路內的內部 Network Load Balancer 與 CloudFront 搭配使用。CloudFront VPC 來源可讓您從私有 VPC 子網路中託管的應用程式提供內容，而無需將其暴露至公共網際網路。如需詳細資訊，請參閱 [使用 VPC 原始伺服器限制存取](#)。

或者，您也可以使用 CloudFront 從面向網際網路的 Network Load Balancer 交付流量。面向網際網路的負載平衡器具有可公開解析的 DNS 名稱，並且可以在網際網路和 CloudFront 分佈上接收來自用戶端的請求。

使用 Lambda 函數 URL

[Lambda 函數 URL](#) 是 Lambda 函數專用的 HTTPS 端點。使用 Lambda 函數 URL，您便可以在 Lambda 內完成所有無伺服器 Web 應用程式建置作業。您可以直接透過這個函數 URL 叫用 Lambda Web 應用程式，完全無需與 API Gateway 或 Application Load Balancer 進行整合。

使用含有函數 URL 的 Lambda 函數建置無伺服器 Web 應用程式，可讓 CloudFront 享有一系列優勢。

- 在更接近檢視者的位置快取內容，加快應用程式的速度

- 讓您的 Web 應用程式使用自訂網域
- 使用 CloudFront 快取行為將不同的 URL 路徑轉送至不同的 Lambda 函數
- 使用 CloudFront 地理限制和/或 AWS WAF 封鎖特定請求
- AWS WAF 搭配 CloudFront 使用 可協助保護您的應用程式免受惡意機器人攻擊、協助防止常見的應用程式入侵，並增強對 DDoS 攻擊的保護

若要使用 Lambda 函數 URL 做為 CloudFront 分佈的原始伺服器，請將 Lambda 函數 URL 的完整網域名稱指定為原始伺服器網域。Lambda 函數 URL 網域名稱必須使用以下格式：

function-URL-ID.lambda-url.AWS-Region.on.aws

使用 Lambda 函數 URL 做為 CloudFront 分佈的原始伺服器時，必須確保函數 URL 可供公開存取。若要執行此操作，請使用下列其中一個選項：

- 如果您使用原始存取控制 (OAC)，Lambda 函數 URL 的 AuthType 參數必須使用 AWS_IAM 值，並在資源型政策中允許 `lambda:InvokeFunctionUrl` 和 `lambda:InvokeFunction` 許可。如需有關 Lambda 函數 URL 用於 OAC 的詳細資訊，請參閱 [限制對 AWS Lambda 函數 URL 原始伺服器的存取](#)。
- 若您不使用 OAC，可將函數 URL 的 AuthType 參數設定為 NONE，並在資源型政策中啟用 `lambda:InvokeFunctionUrl` 許可。

您也可以將 CloudFront 傳送至原始伺服器的請求中 [新增自訂原始伺服器標頭](#)，並編寫函數程式碼，讓系統在收到不含該標頭的請求時傳回錯誤回應，以確保使用者只能透過 CloudFront (無法直接使用 Lambda 函數 URL) 存取您的 Web 應用程式。

如需 Lambda 函數 URL 的詳細資訊，請參閱 AWS Lambda 開發人員指南：

- [Lambda 函數 URL](#) – Lambda 函數 URL 功能的一般概觀
- [叫用 Lambda 函數 URL](#) – 針對編寫無伺服器 Web 應用程式程式碼時須用到的請求和回應承載，提供詳細資訊
- [Lambda 函數 URL 的安全與驗證模型](#) – 包含 Lambda 驗證類型的詳細資訊

使用 Amazon EC2 (或其他自訂原始伺服器)

您可以將內部和面向網際網路的 EC2 執行個體與 Amazon CloudFront 搭配使用。您可以使用 VPC 原始伺服器，將私有子網路內的內部 EC2 執行個體與 CloudFront 搭配使用。CloudFront VPC 來源可讓

您從私有 VPC 子網路中託管的應用程式提供內容，而無需將其暴露至公共網際網路。如需詳細資訊，請參閱[使用 VPC 原始伺服器限制存取](#)。

自訂原始伺服器是 HTTP(S) Web 伺服器，具有可公開解析的 DNS 名稱，可透過網際網路將請求從用戶端路由到目標。HTTP(S) 伺服器可以託管在 AWS Amazon EC2 執行個體上，也可以託管在其他地方。設為網站端點的 Amazon S3 原始伺服器也會被視為自訂原始伺服器。如需詳細資訊，請參閱[the section called “使用設定為網站端點的 Amazon S3 儲存貯體”](#)。

使用自有的 HTTP 伺服器做為自訂原始伺服器時，您可以指定伺服器的 DNS 名稱，以及 CloudFront 從您的原始伺服器擷取物件時使用的 HTTP 與 HTTPS 連接埠和通訊協定。

使用自訂原始伺服器時，大多數 CloudFront 功能都受支援，但私有內容除外：雖然您可以使用已簽署的 URL 來分配自訂原始伺服器的內容，但是為了讓 CloudFront 存取自訂原始伺服器，該伺服器必須維持在可公開存取的狀態。如需詳細資訊，請參閱[the section called “限制已簽署 URL 和已簽署 Cookie 的內容”](#)。

請依照這些準則，以便使用 Amazon EC2 執行個體和在 CloudFront 下的其他自訂原始伺服器原始伺服器。

- 在所有伺服器上託管與服務相同內容，這些伺服器為相同的 CloudFront 原始伺服器提供的內容。如需詳細資訊，請參閱 [the section called “所有分佈設定”](#) 主題中的 [the section called “原始設定”](#)。
- 如果您需要 支援 或 CloudFront 使用此值進行偵錯，請在所有伺服器上記錄 X-Amz-Cf-Id 標頭項目。
- 限制對您自訂原始伺服器所監聽的 HTTP 與 HTTPS 連接埠提出請求。
- 同步實作中所有伺服器的時鐘。請注意，CloudFront 對已簽署 URL 和已簽署 Cookie、日誌及報告皆使用國際標準時間 (UTC)。此外，如果您使用 CloudWatch 指標來監控 CloudFront 活動，請注意，CloudWatch 也會使用 UTC。
- 使用冗餘伺服器來處理故障。
- 如需有關使用自訂原始伺服器來提供私有內容的詳細資訊，請參閱[the section called “在自訂原始伺服器上限制存取檔案”](#)。
- 如需有關請求和回應行為，以及有關支援的 HTTP 狀態代碼的詳細資訊，請參閱[請求和回應行為](#)。

如果您使用 Amazon EC2 做為自訂原始伺服器，建議您執行以下操作：

- 使用 Amazon Machine Image，會自動為 Web 伺服器安裝軟體。如需詳細資訊，請參閱 [Amazon EC2 說明文件](#)。

- 使用 ELB 負載平衡器來處理多個 Amazon EC2 執行個體的流量，並隔離您的應用程式與 Amazon EC2 執行個體的變更。例如，如果您使用負載平衡器，則可以新增和刪除 Amazon EC2 執行個體，無需變更應用程式。如需詳細資訊，請參閱 [ELB 文件](#)。
- 當您建立 CloudFront 分佈時，請為您的原始伺服器的網域名稱指定負載平衡器的 URL。如需詳細資訊，請參閱 [the section called “建立分發”](#)。

使用 CloudFront 原始伺服器群組

例如，當您需要高可用性時，若想要針對案例配置原始伺服器容錯移轉，則您可以為 CloudFront 原始伺服器指定原始伺服器群組。使用原始伺服器容錯移轉來指定 CloudFront 的主要原始伺服器，另外再建立第二部原始伺服器，當主要原始伺服器傳回特定 HTTP 狀態碼故障回應時，CloudFront 會自動切換到第二部伺服器。

如需詳細資訊，包括原始伺服器群組的設定步驟，請參閱 [the section called “使用原始伺服器容錯移轉增加可用性”](#)。

使用 Amazon API Gateway

您可以使用 API Gateway 做為 CloudFront 分佈的自訂原始伺服器。如需詳細資訊，請參閱下列主題：

- [使用 Amazon CloudFront 部落格文章使用安全密碼來保護 Amazon API Gateway Amazon CloudFront AWS](#)
- [如何使用自己的 CloudFront 分佈來設定 API Gateway？](#) AWS re:Post

為 CloudFront 分佈啟用 IPv6

Amazon CloudFront 支援從用戶端到 AWS 節點的 IPv4 和 IPv6。CloudFront 亦支援 IPv6，以及對原始伺服器的雙重堆疊 (IPv4 與 IPv6) 連線。這可協助您實現端對端 IPv6 交付。

IPv6 是新一代網際網路通訊協定，旨在取代 IPv4。雖然 IPv4 使用 32 位元位址 (例如 192.0.2.44)，但 IPv6 使用 128 位元位址 (例如 2001:0db8:85a3::8a2e:0370:7334)。IPv6 提供擴充的位址空間，以容納更多的網際網路連線裝置。

主題

- [IPv6 檢視器請求](#)
- [IPv6 原始伺服器請求](#)

IPv6 檢視器請求

一般而言，如果在 IPv6 網路上有需要存取您的內容的使用者，則應該啟用 IPv6。不過，如果您使用簽章的 URL 或簽章的 Cookie 來限制對內容的存取，並且如果您使用自訂政策，包含 `IpAddress` 參數以限制存取您的內容的 IP 位址，請不要啟用 IPv6。如果您想要透過 IP 位址限制存取一些內容，並且不限制對其他內容的存取 (或限制存取但不透過 IP 位址)，則可以建立兩個分佈。如需有關使用自訂政策建立簽章的 URL 的詳細資訊，請參閱[使用自訂政策建立已簽署 URL](#)。如需有關使用自訂政策建立簽章的 Cookie 的詳細資訊，請參閱[使用自訂政策設定已簽署 Cookie](#)。

如果您使用 Route 53 別名資源紀錄集將流量路由至 CloudFront 分佈，則在滿足以下兩個條件時，您必須建立第二個別名資源紀錄集：

- 您為分佈啟用 IPv6
- 您在物件的 URL 中使用備用網域名稱

如需詳細資訊，請參閱 Amazon Route 53 開發人員指南中的[使用網域名稱將流量路由傳送到 Amazon CloudFront 分佈](#)。

如果您透過 Route 53 或其他 DNS 服務建立了 CNAME 資源紀錄集，則不需要進行任何變更。無論檢視器請求的 IP 位址格式如何，CNAME 記錄都會把流量路由到您的分佈。

如果您啟用 IPv6 和 CloudFront 存取日誌，`c-ip` 欄會包含 IPv4 和 IPv6 格式的值。如需詳細資訊，請參閱[日誌檔案欄位](#)。

Note

為了保持較高的客戶可用性，如果我們的資料建議 IPv4 將提供更好的使用者體驗，CloudFront 會透過使用 IPv4 來回應檢視器請求。要了解 CloudFront 透過 IPv6 提供請求的百分比，請為您的分佈啟用 CloudFront 記錄，並剖析 `c-ip` 列，其中包含發出請求的檢視器的 IP 位址。這個百分比應該隨著時間的推移而增長，但它仍然是少數的流量，因為 IPv6 還沒有得到全球所有檢視器網路的支援。有些檢視器網路擁有優異的 IPv6 支援，但其他檢視器網路則完全不支援 IPv6。(檢視器網路類似於家用網路或無線電信業者)。

如需我們對 IPv6 的支援的詳細資訊，請參閱[CloudFront 常見問答集](#)。如需有關啟用存取日誌的詳細資訊，請參閱[標準記錄](#)、和 [日誌字首](#) 欄位。

IPv6 原始伺服器請求

當您使用自訂原始伺服器 (不含 Amazon S3 和 VPC 原始伺服器) 時，您可以自訂分佈的原始伺服器設定，以選擇 CloudFront 如何使用 IPv4 或 IPv6 位址連接到原始伺服器。對於自訂原始伺服器 (不含 Amazon S3 和 VPC 原始伺服器)，您有下列連線選項：

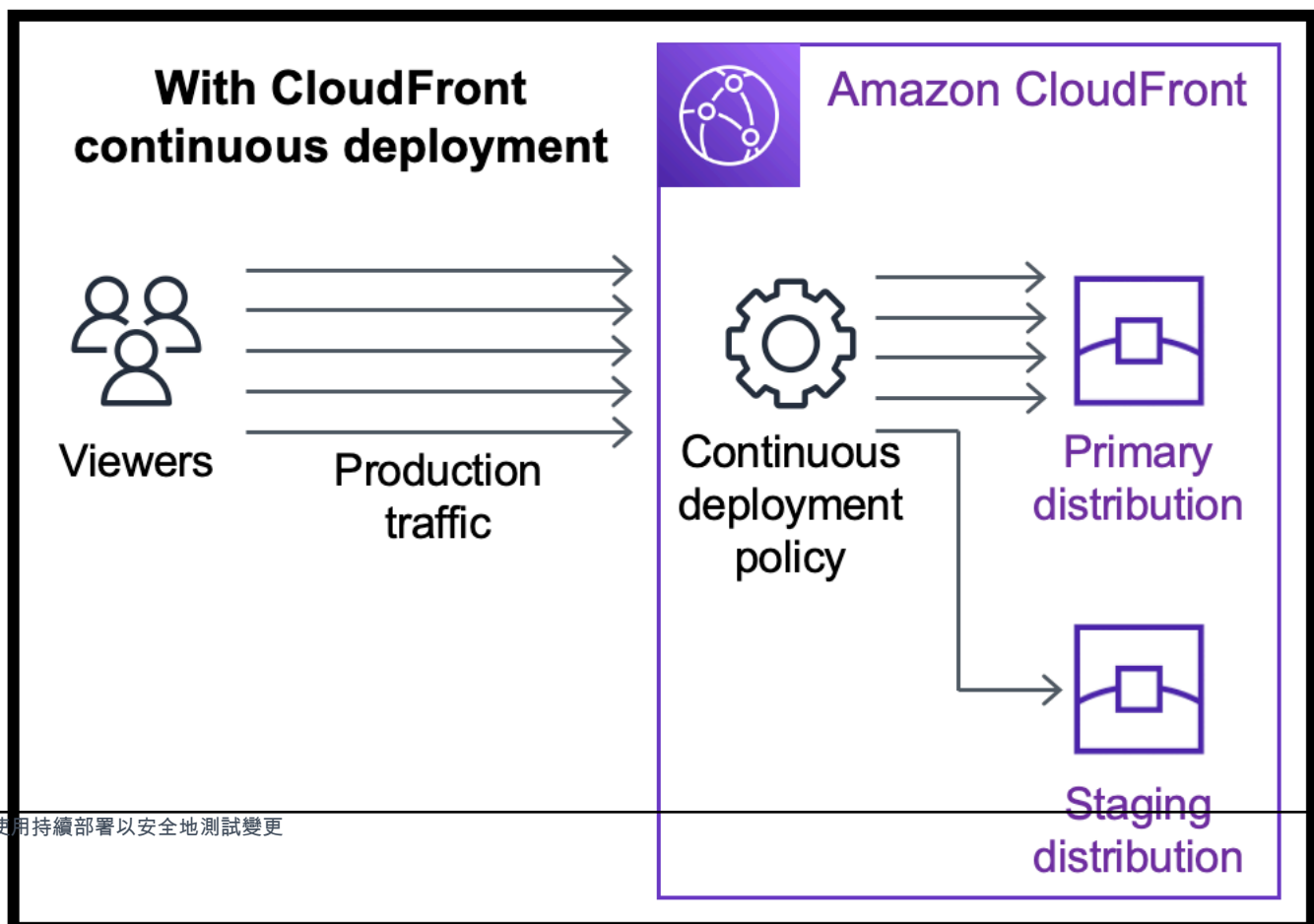
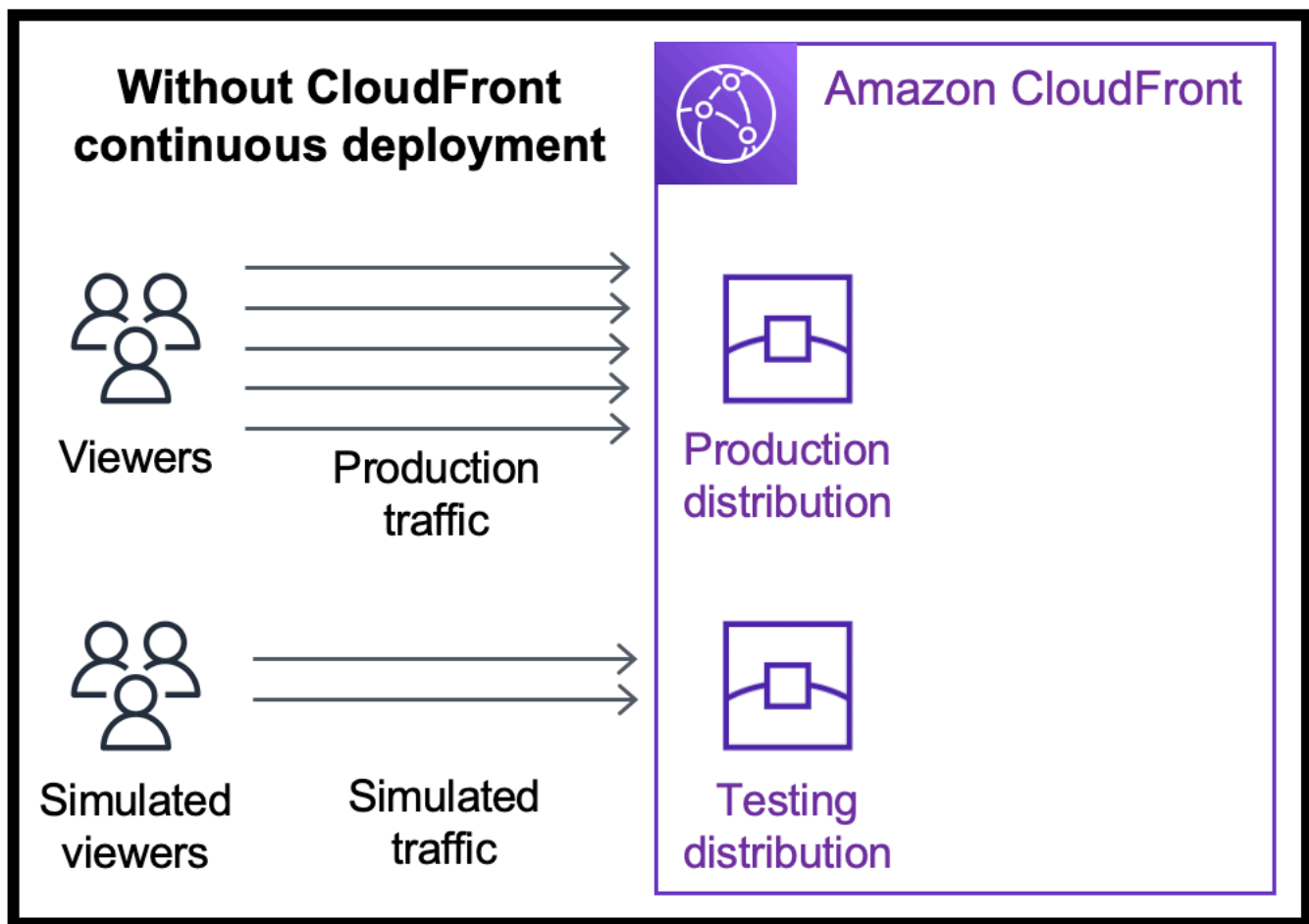
- 僅限 IPv4 (預設) – 這是 CloudFront 用來透過 IPv4 連線至原始伺服器的預設組態。
- 僅限 IPv6 – 需要您的原始伺服器網域解析為 IPv6 位址。CloudFront 將專門使用 IPv6 位址進行原始伺服器連線。
- 雙堆疊 – 啟用透過 IPv4 和 IPv6 的連線。CloudFront 會自動選擇 IPv4 或 IPv6 原始伺服器連線，以排定效能和可用性的優先順序，讓您可以使用 CloudFront 做為 Web 應用程式的 IPv6 和 IPv4 雙堆疊網際網路閘道。

選擇符合您原始伺服器網路組態和連線需求的選項。如需詳細資訊，請參閱[IPv6 設計 DNS](#) 和 [IPv6 安全性和監控考量](#)。

使用 CloudFront 持續部署，安全地測試 CDN 組態變更

透過 Amazon CloudFront 持續部署，您可以先使用生產流量子集進行測試，安全地將變更部署到 CDN 組態。您可以使用臨時分佈和持續部署政策，將真實 (生產) 檢視者的部份流量傳送至新的 CDN 組態，並驗證其是否如預期般運作。您可以即時監控新組態的效能，並在準備就緒時提升新組態，以便透過主要分佈為所有流量提供服務。

下列圖表顯示 CloudFront 持續部署的優勢。若沒有此部署，您就必須使用模擬流量測試 CDN 組態變更。透過持續部署，您可以使用生產流量子集來測試變更，然後在準備就緒時將變更提升至主要分佈。



進一步了解如何在下列主題中使用持續部署。

主題

- [CloudFront 持續部署工作流程](#)
- [使用臨時分佈和持續部署政策](#)
- [監控臨時分佈](#)
- [了解持續部署的運作方式](#)
- [持續部署的配額和其他考量](#)

CloudFront 持續部署工作流程

下列高階工作流程說明如何透過 CloudFront 持續部署，安全地測試和部署組態變更。

1. 選擇您要做為主要分佈的分佈。主要分佈是目前為生產流量提供服務的分佈。
2. 從主要分佈中，建立臨時分佈。臨時分佈一開始會是主要分佈的副本。
3. 在持續部署政策內建立流量組態，並將其連接至主要分佈。這會決定 CloudFront 如何將流量路由至臨時分佈。如需詳細了解臨時分佈的路由請求，請參閱 [the section called “將請求路由至臨時分佈”](#)。
4. 更新臨時分佈的組態。如需詳細了解您可以更新的設定，請參閱 [the section called “更新主要分佈或臨時分佈”](#)。
5. 監控臨時分佈，以判斷組態變更是否如預期般運作。如需監控臨時分佈的詳細資訊，請參閱 [the section called “監控臨時分佈”](#)。

當您監控臨時分佈時，您可以：

- 再次更新臨時分佈的組態，以持續測試組態變更。
 - 更新持續部署政策 (流量組態)，以傳送更多或更少流量至臨時分佈。
6. 臨時分佈的效能符合您的需求時，請將臨時分佈的組態提升為主要分佈，如此會將臨時分佈的組態複製到主要分佈。這項操作也會停用持續部署政策，這表示 CloudFront 會將所有流量路由至主要分佈。

您可以建置自動化來監控臨時分佈的效能 (步驟 5)，並在符合特定條件時自動提升組態 (步驟 6)。

提升組態之後，您可以在下次測試組態變更時，重複使用相同的臨時分佈。

如需在 CloudFront 主控台、AWS CLI、或 CloudFront API 中使用預備分佈和持續部署政策的詳細資訊，請參閱下一節。

使用臨時分佈和持續部署政策

您可以使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API，在 CloudFront 主控台中建立、更新和修改預備分佈和持續部署政策。

使用持續部署政策建立臨時分佈

以下程序介紹如何使用持續部署政策來建立臨時分佈。

Console

您可以使用 AWS 管理主控台建立具有連續部署政策的臨時分佈。

建立臨時分佈和持續部署政策 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇您要做為主要分佈的分佈。主要分佈目前為生產流量提供服務，也就是您從中建立臨時分佈的地方。
4. 在 Continuous deployment (持續部署) 區段中，選擇 Create staging distribution (建立臨時分佈)。這項操作會開啟 Create staging distribution (建立臨時分佈) 精靈。
5. 在 Create staging distribution (建立臨時分佈) 精靈中，執行下列動作：
 - a. (選用) 輸入臨時分佈的描述。
 - b. 選擇下一步。
 - c. 修改臨時分佈的組態。如需詳細了解您可以更新的設定，請參閱 [the section called “更新主要分佈或臨時分佈”](#)。

修改完臨時分佈的組態後，請選擇 Next (下一步)。

- d. 使用控制台指定 Traffic configuration (流量組態)。這會決定 CloudFront 如何將流量路由至臨時分佈。(CloudFront 會將流量組態儲存在持續部署政策中。)

如需 Traffic configuration (流量組態) 選項的詳細資訊，請參閱 [the section called “將請求路由至臨時分佈”](#)。

完成 Traffic configuration (流量組態) 時，請選擇 Next (下一步)。

- e. 檢閱包含流量組態的臨時分佈組態，然後選擇 Create staging distribution (建立臨時分佈)。

在 CloudFront 主控台完成 Create staging distribution (建立臨時分佈) 精靈後，CloudFront 會執行下列動作：

- 使用您在步驟 5c 中指定的設定，建立臨時分佈
- 使用您在步驟 5d 中指定的流量組態，建立持續部署政策
- 將持續部署政策連接至您從中建立臨時分佈的主要分佈

當主要分佈的組態使用連接的持續部署政策部署到邊緣節點時，CloudFront 會開始根據流量組態，將指定的部分流量傳送至臨時分佈。

CLI

若要使用 建立預備分佈和持續部署政策 AWS CLI，請使用下列程序。

建立臨時分佈 (CLI)

1. 同時使用 `aws cloudfront get-distribution` 和 `grep` 命令，針對您要做為主要分佈的分佈，取得其 ETag 值。主要分佈目前為生產流量提供服務，也就是您從中建立臨時分佈的地方。

以下為命令的範例。在下列範例中，請將 *primary_distribution_ID* 取代為主要分佈的 ID。

```
aws cloudfront get-distribution --id primary_distribution_ID | grep 'ETag'
```

請複製 ETag 值，因為您會在以下步驟使用該值。

2. 使用 `aws cloudfront copy-distribution` 命令建立臨時分佈。下列範例命令使用逸出字元 (\) 和換行符號以提高可讀性，但您應在命令中省略這些字元。以下是範例命令：
 - 將 *primary_distribution_ID* 取代為主要分佈的 ID。
 - 將 *primary_distribution_ETag* 取代為主要分佈的 ETag 值，也就是您在上一個步驟中取得的值。
 - (可選) 將 *CLI_example* 取代為所需的呼叫者參考 ID。


```
aws cloudfront copy-distribution --primary-distribution-id primary_distribution_ID \
                                --if-match primary_distribution_ETag \
                                --staging \
                                --caller-reference 'CLI_example'
```

命令的輸出會顯示有關臨時分佈及其組態的資訊。請複製臨時分佈的 CloudFront 網域名稱，因為您需在接下來的步驟使用。

建立持續部署政策 (包含輸入檔案的 CLI)

1. 使用下列命令建立一個名為 `continuous-deployment-policy.yaml` 的檔案，其中包含 `create-continuous-deployment-policy` 命令的所有輸入參數。下列命令使用逸出字元 (\) 和換行符號以提高可讀性，但您應在命令中省略這些字元。

```
aws cloudfront create-continuous-deployment-policy --generate-cli-skeleton yaml-input \
                                                    > continuous-deployment-policy.yaml
```

2. 開啟您剛才建立且命名為 `continuous-deployment-policy.yaml` 的檔案。編輯檔案以指定您想要的持續部署政策設定，然後儲存檔案。當您編輯檔案時：
 - `StagingDistributionDnsNames` 區段：
 - 將 `Quantity` 的值變更為 1。
 - 對於 `Items`，貼上於上一個步驟儲存的臨時分佈的 CloudFront 網域名稱。
 - `TrafficConfig` 區段：
 - 選擇 `Type`，可為 `SingleWeight` 或 `SingleHeader`。
 - 移除其他類型的設定。例如，若您要以權重為基礎的流量組態，請將 `Type` 設定為 `SingleWeight`，然後移除 `SingleHeaderConfig` 設定。
 - 若要使用以權重為基礎的流量組態，請將 `Weight` 值設定為介於 .01 (百分之一) 到 .15 (百分之十五) 間的十進位數字。

如需 TrafficConfig 選項的詳細資訊，請參閱 [the section called “將請求路由至臨時分佈”](#) 和 [the section called “以權重為基礎之組態的工作階段黏性”](#)。

3. 使用下列命令，使用 continuous-deployment-policy.yaml 檔案中的輸入參數建立持續部署政策。

```
aws cloudfront create-continuous-deployment-policy --cli-input-yaml file://continuous-deployment-policy.yaml
```

複製命令輸出中的 Id 值。這是持續部署政策 ID，您會在接下來的步驟使用。

將持續部署政策連接至主要分佈 (包含輸入檔案的 CLI)

1. 使用下列命令，將主要分佈的組態儲存至名為 primary-distribution.yaml 的檔案。將 *primary_distribution_ID* 取代為主要分佈的 ID。

```
aws cloudfront get-distribution-config --id primary_distribution_ID --output yaml > primary-distribution.yaml
```

2. 開啟您剛才建立且命名為 primary-distribution.yaml 的檔案。編輯檔案，進行下列變更：
 - 將您從上一個步驟複製的持續部署政策 ID 貼到 ContinuousDeploymentPolicyId 欄位。
 - 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新主要分佈，以使用持續部署政策。將 *primary_distribution_ID* 取代為主要分佈的 ID。

```
aws cloudfront update-distribution --id primary_distribution_ID --cli-input-yaml file://primary-distribution.yaml
```

當主要分佈的組態使用連接的持續部署政策部署到邊緣節點時，CloudFront 會開始根據流量組態，將指定的部分流量傳送至臨時分佈。

API

若要使用 CloudFront API 建立臨時分佈和持續部署政策，請使用下列 API 操作：

- [CopyDistribution](#)
- [CreateContinuousDeploymentPolicy](#)

如需詳細了解您在這些 API 呼叫中指定的欄位，請參閱以下內容：

- [the section called “將請求路由至臨時分佈”](#)
- [the section called “以權重為基礎之組態的工作階段黏性”](#)
- SDK AWS 或其他 API 用戶端的 API 參考文件

建立臨時分佈和持續部署政策之後，請使用 [UpdateDistribution](#) (在主要分佈上)，將持續部署政策連接至主要分佈。

更新臨時分佈

以下程序介紹如何使用持續部署政策來更新臨時分佈。

Console

您可以更新主要和臨時分佈的特定組態。如需詳細資訊，請參閱[更新主要分佈或臨時分佈](#)。

更新臨時分佈 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇主要分佈。這是目前為生產流量提供服務的分佈，也就是您從中建立臨時分佈的地方。
4. 選擇 View staging distribution (檢視臨時分佈)。
5. 使用主控台修改臨時分佈的組態。如需詳細了解您可以更新的設定，請參閱 [the section called “更新主要分佈或臨時分佈”](#)。

一旦臨時分佈的組態部署到邊緣節點，即對路由至臨時分佈的傳入流量產生效果。

CLI

更新臨時分佈 (包含輸入檔案的 CLI)

1. 使用下列命令，將臨時分佈的組態儲存至名為 `staging-distribution.yaml` 的檔案。將 `staging_distribution_ID` 取代為暫存分佈的 ID。

```
aws cloudfront get-distribution-config --id staging_distribution_ID --output  
yaml > staging-distribution.yaml
```

2. 開啟您剛才建立且命名為 `staging-distribution.yaml` 的檔案。編輯檔案，進行下列變更：
 - 修改臨時分佈的組態。如需詳細了解您可以更新的設定，請參閱 [the section called “更新主要分佈或臨時分佈”](#)。
 - 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新臨時分佈的組態。將 `staging_distribution_ID` 取代為暫存分佈的 ID。

```
aws cloudfront update-distribution --id staging_distribution_ID --cli-input-yaml  
file://staging-distribution.yaml
```

一旦臨時分佈的組態部署到邊緣節點，即對路由至臨時分佈的傳入流量產生效果。

API

若要更新臨時分佈的組態，請使用 [UpdateDistribution](#) (在臨時分佈上)，修改臨時分佈的組態。如需詳細了解您可以更新的設定，請參閱 [the section called “更新主要分佈或臨時分佈”](#)。

更新持續部署政策

下列程序說明如何更新持續部署政策。

Console

您可以透過更新持續部署政策來更新分佈的流量組態。

更新持續部署政策 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇主要分佈。這是目前為生產流量提供服務的分佈，也就是您從中建立臨時分佈的地方。
4. 在 Continuous deployment (持續部署) 區段中，選擇 Edit policy (編輯政策)。
5. 修改持續部署政策中的流量組態。完成時，請選擇 Save changes (儲存變更)。

當主要分佈的組態使用更新的持續部署政策部署到邊緣節點時，CloudFront 會開始根據更新的流量組態，將流量傳送至臨時分佈。

CLI

更新持續部署政策 (包含輸入檔案的 CLI)

1. 使用下列命令，將持續部署政策的組態儲存至名為 continuous-deployment-policy.yaml 的檔案。將 *continuous_deployment_policy_ID* 取代為持續部署政策的 ID。下列命令使用逸出字元 (\) 和換行符號以提高可讀性，但您應在命令中省略這些字元。

```
aws cloudfront get-continuous-deployment-policy-config --  
id continuous_deployment_policy_ID \  
                                     --output yaml >  
continuous-deployment-policy.yaml
```

2. 開啟您剛才建立且命名為 continuous-deployment-policy.yaml 的檔案。編輯檔案，進行下列變更：
 - 視需要修改持續部署政策的組態。例如，您可以從以標頭為基礎的流量組態，改為使用以權重為基礎的流量組態，或變更以權重為基礎的組態的流量百分比 (權重)。如需詳細資訊，請參閱 [the section called “將請求路由至臨時分佈”](#) 及 [the section called “以權重為基礎之組態的工作階段黏性”](#)。
 - 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新持續部署政策。將 `continuous_deployment_policy_ID` 取代為持續部署政策的 ID。下列命令使用逸出字元 (\) 和換行符號以提高可讀性，但您應在命令中省略這些字元。

```
aws cloudfront update-continuous-deployment-policy --  
id continuous_deployment_policy_ID \  
continuous-deployment-policy.yaml --cli-input-yaml file://
```

當主要分佈的組態使用更新的持續部署政策部署到邊緣節點時，CloudFront 會開始根據更新的流量組態，將流量傳送至臨時分佈。

API

若要更新持續部署政策，請使用 [UpdateContinuousDeploymentPolicy](#)。

提升臨時分佈組態

下列程序說明如何提升臨時分佈組態。

Console

提升臨時分佈時，CloudFront 會將組態從臨時分佈複製到主要分佈。CloudFront 也會停用持續部署政策，並將所有流量路由至主要分佈。

提升組態之後，您可以在下次測試組態變更時，重複使用相同的臨時分佈。

提升臨時分佈組態 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇主要分佈。這是目前為生產流量提供服務的分佈，也就是您從中建立臨時分佈的地方。
4. 在 Continuous deployment (持續部署) 區段中，選擇 Promote (提升)。
5. 輸入 **confirm**，然後選擇 Promote (提升)。

CLI

提升臨時分佈時，CloudFront 會將組態從臨時分佈複製到主要分佈。CloudFront 也會停用持續部署政策，並將所有流量路由至主要分佈。

提升組態之後，您可以在下次測試組態變更時，重複使用相同的臨時分佈。

提升臨時分佈組態 (CLI)

- 使用 `aws cloudfront update-distribution-with-staging-config` 命令，將臨時分佈的組態提升為主要分佈。下列範例命令使用逸出字元 (\) 和換行符號以提高可讀性，但您應在命令中省略這些字元。以下是範例命令：
 - 將 `primary_distribution_ID` 取代為主要分佈的 ID。
 - 將 `staging_distribution_ID` 取代為臨時分佈的 ID。
 - 將 `primary_distribution_ETag` 和 `staging_distribution_ETag` 取代為主要分佈和臨時分佈的 ETag 值。請確認主要分佈的值是第一個，如範例所示。

```
aws cloudfront update-distribution-with-staging-config --
id primary_distribution_ID \
                                     --staging-distribution-
id staging_distribution_ID \
                                     --if-match
'primary_distribution_ETag,staging_distribution_ETag'
```

API

若要將臨時分佈的組態提升為主要分佈，請使用 [UpdateDistributionWithStagingConfig](#)。

監控臨時分佈

若要監控臨時分佈的效能，您可以使用 CloudFront 提供給所有分佈的相同[指標、日誌和報告](#)。例如：

- 您可以在 CloudFront 主控台檢視[預設的 CloudFront 分佈指標](#) (例如總請求和錯誤率)，也可以額外付費[開啟其他指標](#) (例如按狀態碼分類的快取命中率和錯誤率)。您也可以根據這些指標建立警示。

- 您可以檢視[標準日誌](#)和[即時存取日誌](#)，以取得臨時分佈所接收請求的詳細資訊。標準日誌包含下列兩個欄位，可協助您識別 CloudFront 將請求路由至臨時分佈之前，最初傳送到的主要分佈：`primary-distribution-id` 和 `primary-distribution-dns-name`。
- 您可以在 CloudFront 主控台檢視和下載[報告](#)，例如快取統計資料報告。

了解持續部署的運作方式

下列主題說明 CloudFront 持續部署的運作方式。

主題

- [將請求路由至臨時分佈](#)
- [以權重為基礎之組態的工作階段黏性](#)
- [更新主要分佈或臨時分佈](#)
- [主要分佈和臨時分佈不會共用快取](#)

將請求路由至臨時分佈

使用 CloudFront 持續部署時，您不必變更檢視者請求的任何相關資訊。檢視者無法使用 DNS 名稱、IP 位址或 CNAME，將請求直接傳送至臨時分佈。反之，檢視者會將請求傳送至主要 (生產) 分佈，而 CloudFront 會根據持續部署政策的流量組態設定，將部份請求路由至臨時分佈。有兩種類型的流量組態：

以權重為基礎的

以權重為基礎的組態會將指定百分比的檢視者請求路由至臨時分佈。當您使用以權重為基礎的組態時，您也可以啟用工作階段黏性，有助於確保 CloudFront 會將來自相同檢視者的請求視為單一工作階段的一部份。如需詳細資訊，請參閱[the section called “以權重為基礎之組態的工作階段黏性”](#)。

以標頭為基礎的

當檢視者請求包含特定 HTTP 標頭 (您可以指定該標頭和值)，以標頭為基礎的組態會將請求路由至臨時分佈。不包含指定標頭和值的請求則會路由至主要分佈。此組態對於本地測試或您可以控制檢視者請求時非常有用。

Note

路由至臨時分佈的標頭必須包含字首 `aws-cf-cd-`。

以權重為基礎之組態的工作階段黏性

當您使用以權重為基礎的組態，將流量路由至臨時分佈時，您也可以啟用工作階段黏性，有助於確保 CloudFront 會將來自相同檢視者的請求視為單一工作階段。當您啟用工作階段黏性，CloudFront 會設定 Cookie，如此一來，單一工作階段中來自相同檢視者的所有請求都會由一個分佈 (主要或臨時) 提供服務。

當您啟用工作階段黏性，您也可以指定閒置時間。如果檢視者在這段時間內處於閒置狀態 (不傳送任何請求)，工作階段會過期，CloudFront 之後會將來自此檢視者的請求視為新工作階段。您能以秒為單位，將閒置時間設定為 300 (5 分鐘) 到 3600 (一小時)。

在下列情況中，CloudFront 會重設所有工作階段 (甚至是作用中的工作階段)，並將所有請求視為新工作階段：

- 您停用或啟用持續部署政策
- 您停用或啟用工作階段黏性設定

更新主要分佈或臨時分佈

當主要分佈具有連接的持續部署政策時，主要分佈和臨時分佈皆可使用下列組態變更：

- 所有快取行為設定，包括預設快取行為
- 所有原始伺服器設定 (原始伺服器和原始伺服器群組)
- 自訂錯誤回應 (錯誤頁面)
- 地理限制
- 預設根物件
- 日誌設定
- 描述 (評論)

您也可以更新分佈組態中參考的外部資源，例如快取政策、回應標頭政策、CloudFront 函數或 Lambda @Edge 函數。

主要分佈和臨時分佈不會共用快取

主要分佈和臨時分佈不會共用快取。CloudFront 將第一個請求傳送至臨時分佈時，其快取會是空的。當請求抵達臨時分佈時，便會開始快取回應 (如果設定為如此)。

持續部署的配額和其他考量

CloudFront 持續部署受下列配額和其他考量約束。

配額

- 每個的預備分佈數目上限 AWS 帳戶：20
- 每個的持續部署政策數目上限 AWS 帳戶：20
- 在以權重為基礎的組態中，您可以傳送至臨時分佈的流量百分比上限：15%
- 工作階段黏性閒置時間的上下限：300–3600 秒

如需詳細資訊，請參閱[配額](#)。

Note

當您使用持續部署，且主要分佈已設定為使用 OAC 存取 S3 儲存貯體時，請更新您的 S3 儲存貯體政策，以允許臨時分佈的存取。如需 S3 儲存貯體政策的範例，請參閱 [the section called “授予 CloudFront 存取 S3 儲存貯體的許可”](#)。

AWS WAF Web ACLs

如果您為分佈啟用持續部署，則下列考量適用於 AWS WAF：

- 如果 AWS WAF Web 存取控制清單 (ACL) 是 ACL 第一次與分佈建立關聯，則無法將其與分佈建立關聯。
- 您無法取消 AWS WAF Web ACL 與分佈的關聯。

您必須先刪除生產分佈的持續部署政策，才能執行上述任務。這也會刪除臨時分佈。如需詳細資訊，請參閱[使用 AWS WAF 保護](#)。

CloudFront 將所有請求傳送至主要分佈的案例

在部份情況下 (例如資源使用率高的期間)，不論持續部署政策指定的內容為何，CloudFront 可能將所有請求傳送至主要分佈。

在流量尖峰時段，不論持續部署政策指定的內容為何，CloudFront 會將所有請求傳送至主要分佈。峰值流量是指 CloudFront 服務上的流量，而不是分佈上的流量。

HTTP/3

您無法搭配使用支援 HTTP/3 的分佈與持續部署。

透過新增備用網域名稱 (CNAME) 使用自訂 URL

建立分佈後，CloudFront 會為該分佈提供網域名稱，例如 `d1111111abcdef8.cloudfront.net`。您可以使用替代網域名稱 (也稱為 CNAME)，而不是使用此提供的網域名稱。

若要了解如何使用自己的網域名稱 (例如 `www.example.com`)，請參閱下列主題：

主題

- [使用備用網域名稱的需求](#)
- [使用備用網域名稱的限制](#)
- [新增替代網域名稱](#)
- [移動替代網域名稱](#)
- [移除替代網域名稱](#)
- [在備用網域名稱中使用萬用字元](#)

使用備用網域名稱的需求

當您將備用網域名稱 (例如 `www.example.com`) 新增至 CloudFront 分佈時，下列為必要項目：

備用網域名稱必須是小寫

所有替代網域名稱 (CNAME) 都必須是小寫。

備用網域名稱必須由有效的 TLS 憑證涵蓋

若要將替代網域名稱 (CNAME) 新增到 CloudFront 分佈，必須將信任、有效且涵蓋替代網域名稱的 TLS 憑證連接到您的分佈。這可確保只有具備存取您網域憑證權限的人員能將與您網域相關的 CNAME 與 CloudFront 建立關聯。

信任憑證是由 AWS Certificate Manager (ACM) 或其他有效憑證授權機構 (CA) 發行的憑證。您可以使用自我簽署憑證來驗證現有的 CNAME，但不能用於新的 CNAME。CloudFront 支援與 Mozilla 相同的憑證授權機構。如需目前的清單，請參閱 [Mozilla 內建 CA 憑證清單](#)。如需有關在使用第三方 CA 時的中繼憑證資訊，請參閱 [中繼憑證](#)。

為了使用您連線的憑證驗證備用網域名稱 (包括包含萬用字元的備用網域名稱)，CloudFront 會檢查憑證上的主體別名 (SAN)。您新增的備用網域名稱必須由 SAN 涵蓋。

 Note

一次只能有一個憑證能連線到 CloudFront 分佈。

您可以執行以下作業，證明您已獲得授權，將特定的備用網域名稱新增到您的分佈：

- 附加包含替代網域名稱的憑證，例如 `product-name.example.com`。
- 連接網域名稱開頭包含 * 萬用字元的憑證，以使用單一憑證涵蓋多個子網域。當您指定萬用字元時，您可以在 CloudFront 中新增多個子網域做為備用網域名稱。

以下範例會示範如何在憑證作業中，於網域名稱內使用萬用字元來授權您在 CloudFront 中新增特定備用網域名稱。

- 您想要新增 `marketing.example.com` 做為替代網域名稱。在您的憑證中列出以下網域名稱：
`*.example.com`。將此憑證連接到 CloudFront 後，您可以為您的分佈新增任何替代網域名稱，取代該層級的萬用字元，包括 `marketing.example.com`。您也可以新增下列備用網域名稱 (範例)：
 - `product.example.com`
 - `api.example.com`

但是，您無法新增比萬用字元層級高或低的備用網域名稱。例如，您無法新增替代網域名稱 `example.com` 或 `marketing.product.example.com`。

- 您想要新增 `mexample.com` 做為替代網域名稱。若要執行此作業，您必須在您連接到分佈的憑證上列出網域名稱 `example.com` 本身。
- 您想要新增 `marketing.product.example.com` 做為替代網域名稱。若要執行此作業，您可以在憑證上列出 `*.example.com`，或是在憑證上列出 `marketing.product.example.com`。

變更 DNS 組態的許可

新增備用網域名稱時，必須建立 CNAME 紀錄，將替代網域名稱的 DNS 查詢路由到您的 CloudFront 分佈。若要執行此作業，您必須擁有為您正在使用備用網域名稱，使用 DNS 服務提供者建立 CNAME 紀錄的許可。一般而言，這表示您擁有此網域，但您也可能是在為網域擁有者開發應用程式。

備用網域名稱和 HTTPS

若您希望檢視器搭配備用網域名稱使用 HTTPS，您必須完成額外設定。如需詳細資訊，請參閱[使用備用網域名稱和 HTTPS](#)。

使用備用網域名稱的限制

請在使用備用網域名稱時，注意以下限制：

備用網域名稱的數量上限

如需您可為分發新增之備用網域名稱數量的目前上限，或是有關請求更高配額 (先前稱為限制) 的詳細資訊，請參閱[分佈的一般配額](#)。

重複和重疊的備用網域名稱

如果另一個 CloudFront 分佈中已存在相同的替代網域名稱，即使您 AWS 帳戶 擁有另一個分佈，也無法將替代網域名稱新增至 CloudFront 分佈。

不過，您可以新增萬用字元替代網域名稱，例如 *.example.com，其中包含 (重疊) 非萬用字元的替代網域名稱，例如 www.example.com。如果您在兩個分佈中有重疊的備用網域名稱，無論 DNS 記錄指向哪個分佈，CloudFront 都會將請求傳送至名稱相符程度更具體的分佈。例如，marketing.domain.com 比 *.domain.com 更為具體。

如果您有指向 CloudFront 分佈的現有萬用字元 DNS 項目，且在嘗試新增具有更特定名稱的新 CNAME 時，收到設定不正確的 DNS 錯誤，請參閱[當我嘗試新增 CNAME 時，CloudFront 會傳回 DNS 記錄設定不正確的錯誤](#)。

網域 Fronting

CloudFront 在不同 AWS 帳戶中發生的網域 Fronting 擁有保護。當非標準用戶端建立一個與網域名稱的 TLS/SSL 連線 AWS 帳戶，然後在另一個 中對不相關的網域名稱提出 HTTPS 請求時，就會發生這種情況 AWS 帳戶。

例如，TLS 連線可能會連線到 www.example.com，接著又發出 www.example.org 請求。

為了判斷請求是否為網域前端，CloudFront 會執行下列檢查：

- SNI 延伸項目等於 HTTP 請求 Host 標頭
- 憑證屬於 AWS 帳戶 與請求分佈相同的。
- HTTP 請求 Host 被涵蓋在 TLS 交握期間提供的憑證中

如果這些條件均不符合，則 CloudFront 會判斷請求是網域前端。CloudFront 會使用 421 HTTP 錯誤回應拒絕請求。

Note

如果用戶端未提供 SNI 延伸模組，而是取得預設的 *.cloudfront.net 憑證，則 CloudFront 會接受傳入的請求。

CloudFront 如何識別請求的分佈

CloudFront 會根據 Host 標頭識別 HTTP 請求的分佈。CloudFront 不依賴您連線的 CloudFront IP 位址，或在 TLS 交握期間提供的 SNI 交握。

當 CloudFront 收到請求時，將使用 Host 標頭的值，將請求與特定分佈進行比對。

例如，假設您有兩個分佈，而且已更新 DNS 組態，因此備用網域名稱會路由至下列端點：

- primary.example.com 指向 d111111primary.cloudfront.net
- secondary.example.com 指向 d222222secondary.cloudfront.net

如果您向 https://primary.example.com 提出請求，但將 Host 標頭指定為 secondary.example.com，例如 `curl https://primary.example.com -H "Host: secondary.example.com"`，則請求將改為路由至次要分佈。

在網域的頂端節點 (Zone Apex) 新增備用網域名稱

如果您將備用網域名稱新增到分佈，您通常會在 DNS 組態中建立 CNAME 記錄以將網域名稱的 DNS 查詢路由到 CloudFront 分佈。不過，您不能為 DNS 命名空間，也稱為 Zone Apex 的頂端節點建立 CNAME 記錄；DNS 通訊協定不允許此操作。例如，如果您註冊 DNS 名稱 example.com，Zone Apex 就是 example.com。您無法為 example.com 建立 CNAME 紀錄，但可以為 www.example.com、newproduct.example.com 等建立 CNAME 紀錄。

如果您使用 Route 53 做為您的 DNS 服務，則可以建立別名資源紀錄集，其與 CNAME 記錄相比有以下優勢：

- 您可以在頂端節點 (example.com) 建立網域名稱的別名資源紀錄集。
- 如果用戶端支援，您可以為替代網域名稱建立 HTTPS 記錄，以允許通訊協定交涉做為 DNS 查詢的一部分。如需詳細資訊，請參閱 [Create alias resource record set](#)。
- 在使用別名資源紀錄集時，您不需要支付 Route 53 查詢。

Note

如果您啟用 IPv6，必須建立兩個別名資源紀錄集：一個給路由 IPv4 流量 (A 紀錄)，一個給路由 IPv6 流量 (AAAA 紀錄)。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [啟用 IPv6 \(檢視器請求\)](#)。

如需詳細資訊，請參閱 Amazon Route 53 開發人員指南中的 [使用網域名稱將流量路由傳送到 Amazon CloudFront Web 分佈](#)。

如果您不是針對 DNS 使用 Route 53，您可以請求 Anycast 靜態 IP 位址，將 example.com 等 Apex 網域路由至 CloudFront。如需詳細資訊，請參閱 [請求用於列入允許清單的 Anycast 靜態 IP](#)。

新增替代網域名稱

以下任務清單會說明如何使用 CloudFront 主控台，將備用網域名稱新增到您的分佈，讓您可以在連結中使用您自己的網域名稱，而非 CloudFront 網域名稱。如需使用 CloudFront API 更新分發的詳細資訊，請參閱 [設定分佈](#)。

Note

如果您要檢視器使用具有備用網域名稱的 HTTPS，請參閱 [使用備用網域名稱和 HTTPS](#)。

在您開始進行前：請先確認您已執行以下作業，再更新您的分佈以新增備用網域名稱：

- 向 Route 53 或其他網域註冊商註冊網域名稱。
- 從涵蓋網域名稱的授權憑證認證機構 (CA) 取得 TLS 憑證。將憑證新增至您的分佈，來驗證您是否已獲得使用網域的授權。如需詳細資訊，請參閱 [使用備用網域名稱的需求](#)。

新增替代網域名稱

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇您希望更新的分佈 ID。
3. 在一般索引標籤上，選擇新增網域。

4. 輸入最多五個要服務的網域。
5. 選擇下一步。
6. 對於 TLS 憑證，如果 CloudFront 在 AWS 帳戶 的 中找不到網域的現有 AWS Certificate Manager (ACM) 憑證 us-east-1 AWS 區域，您可以選擇自動建立憑證或在 ACM 中手動建立憑證。
7. 佈建憑證時，您必須使用 DNS 提供者更新您的 DNS 記錄，以證明網域擁有權。您需要對 DNS 記錄進行的項目會在 CloudFront 主控台中提供給您。
8. 更新 DNS 記錄後，請選擇驗證憑證。
9. 驗證憑證時，請選擇下一步。
10. 檢閱您的變更，然後選擇新增網域。
11. 在分佈的 General (一般) 索引標籤上，請確認 Distribution Status (分佈狀態) 是否已變更為 Deployed (已部署)。如果您在部署分佈的更新之前，嘗試使用備用網域名稱，則您在下列步驟中建立的連結可能無法運作。
12. 設定替代網域名稱 (例如 www.example.com) 的 DNS 服務，以便將流量路由至分佈的 CloudFront 網域名稱 (例如 d111111abcdef8.cloudfront.net)。您使用的方法取決於您是使用 Route 53 做為網域的 DNS 服務提供者，還是其他提供者。如需詳細資訊，請參閱[將網域新增至您的 CloudFront 標準分佈](#)。

Route 53

請建立別名資源紀錄集。使用別名資源紀錄集時，您不需要支付 Route 53 查詢。您也可以建立根網域名稱 (example.com) 的別名資源紀錄集，該 DNS 不允許 CNAME。如需建立別名資源紀錄集的詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的[使用網域名稱將流量路由傳送到 Amazon CloudFront Web 分佈](#)。

或者，如果用戶端支援，您可以為替代網域名稱建立 HTTPS 記錄，以允許通訊協定交涉做為 DNS 查詢的一部分。

使用 HTTPS 記錄建立別名資源紀錄集 (選用)

1. 在 CloudFront 分佈設定中啟用 HTTP/2 或 HTTP/3。如需詳細資訊，請參閱[支援的 HTTP 版本及更新分佈](#)。
2. 在 Route 53 主控台中，建立別名資源紀錄集。依照[使用網域名稱將流量路由到 Amazon CloudFront Web 分佈](#)的程序執行。
3. 當您建立別名資源紀錄集時，請使用記錄類型 HTTPS 建立別名紀錄。

其他 DNS 服務供應商

使用 DNS 服務供應商提供的方法，為您的網域新增 CNAME 記錄。這個新的 CNAME 紀錄會將 DNS 查詢從您的替代網域名稱 (例如 `www.example.com`) 重新引導到分佈的 CloudFront 網域名稱 (例如 `d111111abcdef8.cloudfront.net`)。如需詳細資訊，請參閱 DNS 服務供應商提供的說明文件。

Important

如果您的替代網域名稱已有現有的 CNAME 紀錄，請更新該紀錄或將其替換為指向分佈 CloudFront 網域名稱的新紀錄。

13. 使用 `dig` 或類似 DNS 工具，請確認您在先前步驟建立的 DNS 組態，是否指向分佈的網域名稱。

以下範例顯示了一個在 `www.example.com` 網域的 `dig` 請求和回應的相關部分。

```
PROMPT> dig www.example.com

; <<> DiG 9.3.3rc2 <<> www.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;www.example.com.      IN      A

;; ANSWER SECTION:
www.example.com. 10800 IN CNAME d111111abcdef8.cloudfront.net.
...
```

答案區段會顯示一個 CNAME 記錄，將 `www.example.com` 查詢路由傳送至 CloudFront 分佈網域名稱 `d111111abcdef8.cloudfront.net`。如果 CNAME 右側的名稱為 CloudFront 分佈的網域名稱，則 CNAME (正式名稱記錄) 記錄配置正確。如果它是任何其他值，例如，Amazon S3 儲存貯體的網域名稱，則 CNAME 紀錄配置錯誤。在這種情況下，請回到步驟 7 並更正 CNAME 紀錄以指向分佈的網域名稱。

14. 使用您的網域名稱造訪 URL，而非您分佈的 CloudFront 網域名稱，以測試備用網域名稱。

15. 在您的應用程式中，請為您的物件變更 URL 以使用備用網域名稱，而非 CloudFront 分佈的網域名稱。

移動替代網域名稱

如果您嘗試將替代網域名稱新增至標準分佈或分佈租用戶，且替代網域名稱已與不同的資源建立關聯，您會收到錯誤訊息。

例如，當您嘗試將 `www.example.com` 新增至標準分佈或分佈租用戶時，若該替代網域名稱已與其他資源建立關聯，您會收到 `CNAMEAlreadyExists` 錯誤訊息 (您所提供的一個或多個 CNAME 已與不同的資源關聯)。

在這種情況下，您可能想要將現有的替代網域名稱從一個資源移至另一個資源。這是來源分佈和目標分佈。您可以在標準分佈和/或分佈租用戶之間移動替代網域名稱。

若要移動替代網域名稱，請參閱下列主題：

主題

- [設定目標標準分佈或分佈租用戶](#)
- [尋找來源標準分佈或分佈租用戶](#)
- [移動替代網域名稱](#)

設定目標標準分佈或分佈租用戶

您必須先設定目標資源，才能移動替代網域名稱。這是您要移動替代網域名稱的目標標準分佈或分佈租用戶。

Standard distribution

設定目標標準分佈

1. 請求 TLS 憑證。此憑證包含替代網域名稱做為主體或主體備用網域 (SAN)，或涵蓋您正在移動之替代網域名稱的萬用字元 (*)。如果沒有憑證，可以在 AWS Certificate Manager (ACM) 中申請一個憑證，或是從另一個憑證認證機構 (CA) 取得憑證，然後將其匯入 ACM。

Note

您必須在美國東部 (維吉尼亞北部) (us-east-1) 區域請求或匯入憑證。

如需詳細資訊，請參閱《AWS Certificate Manager 使用者指南》中的[使用主控台申請公有憑證與在 AWS Certificate Manager 中匯入憑證](#)章節。

2. 如果尚未建立目標標準分佈，請立即建立。在建立標準分佈的過程中，請將憑證與此標準分佈建立關聯。如需詳細資訊，請參閱[建立分發](#)。

如果您已經有目標標準分佈，請將憑證與標準分佈建立關聯。如需詳細資訊，請參閱[更新分佈](#)。

3. 如果您要在相同 內移動替代網域名稱 AWS 帳戶，請略過此步驟。

若要將備用網域名稱從一個移至 AWS 帳戶 另一個，您必須在 DNS 組態中建立 TXT 記錄。此驗證步驟有助於防止未經授權的網域傳輸。CloudFront 會使用此 TXT 記錄來驗證您對替代網域名稱的擁有權。

在您的 DNS 設定中，建立一筆 DNS TXT 紀錄，將替代網域名稱與目標標準分佈建立關聯。TXT 記錄格式可能會因網域類型而有所不同。

- 對於子網域，請在替代網域名稱前面指定底線 (_)。以下顯示範例 TXT 紀錄。

```
_www.example.com TXT d111111abcdef8.cloudfront.net
```

- 對於 Apex (或根網域)，請在網域名稱前面指定底線和句點 (._.)。以下顯示範例 TXT 紀錄。

```
_.example.com TXT d111111abcdef8.cloudfront.net
```

Distribution tenant

設定目標分佈租用戶

1. 請求 TLS 憑證。此憑證包含替代網域名稱做為主體或主體備用網域 (SAN)，或涵蓋您正在移動之替代網域名稱的萬用字元 (*)。如果沒有憑證，可以在 AWS Certificate Manager (ACM) 中申請一個憑證，或是從另一個憑證認證機構 (CA) 取得憑證，然後將其匯入 ACM。

Note

您必須在美國東部 (維吉尼亞北部) (us-east-1) 區域請求或匯入憑證。

如需詳細資訊，請參閱《AWS Certificate Manager 使用者指南》中的[使用主控台申請公有憑證與在 AWS Certificate Manager 中匯入憑證](#)章節。

2. 如果尚未建立目標分佈租用戶，請立即建立。做為建立分佈租用戶的一部分，請將憑證與分佈租用戶建立關聯。如需詳細資訊，請參閱[建立分發](#)。

如果您已經有目標分佈租用戶，請將憑證與分佈租用戶建立關聯。如需詳細資訊，請參閱[新增網域和憑證 \(分佈租用戶\)](#)。

3. 如果您要在相同 內移動替代網域名稱 AWS 帳戶，請略過此步驟。

若要將備用網域名稱從一個移至 AWS 帳戶 另一個，您必須在 DNS 組態中建立 TXT 記錄。此驗證步驟有助於防止未經授權的網域傳輸，CloudFront 會使用此 TXT 記錄來驗證替代網域名稱的擁有權。

在您的 DNS 設定中，建立一筆 DNS TXT 紀錄，將替代網域名稱與目標分佈租用戶建立關聯。TXT 記錄格式可能會因網域類型而有所不同。

- 對於子網域，請在替代網域名稱前面指定底線 (_)。以下顯示範例 TXT 紀錄。

```
_www.example.com TXT d111111abcdef8.cloudfront.net
```

- 對於 Apex (或根網域)，請在網域名稱前面指定底線和句點 (._)。以下顯示範例 TXT 紀錄。

```
_.example.com TXT d111111abcdef8.cloudfront.net
```

接下來，請參閱下列主題，尋找已與替代網域名稱相關聯的來源標準分佈或分佈租用戶。

尋找來源標準分佈或分佈租用戶

在您可以將替代網域名稱從一個分佈 (標準或租用戶) 移至另一個分佈之前，請先尋找來源分佈。這是替代網域名稱已關聯的資源。當您知道來源和目標分佈資源的 AWS 帳戶 ID 時，您可以決定如何移動替代網域名稱。

備註

- 我們建議您使用 [ListDomainConflicts](#) API 操作，因為它同時支援標準分佈和分佈租用戶。
- [ListConflictingAliases](#) API 操作僅支援標準分佈。

請依照這些範例來尋找來源分佈 (標準或租用戶)。

list-domain-conflicts

Tip

- 對於標準分佈，您必須擁有 `cloudfront:GetDistribution` 和 `cloudfront:ListDomainConflicts` 許可。
- 對於分佈租用戶，您必須擁有 `cloudfront:GetDistributionTenant` 和 `cloudfront:ListDomainConflicts` 許可。

使用 **list-domain-conflicts** 尋找來源標準分佈或分佈租用戶

1. 請使用 `list-domain-conflicts` 指令，如下列範例所示。
 - a. 以網域名稱取代 `www.example.com`。
 - b. 針對 `domain-control-validation-resource`，指定[您先前設定的](#)目標標準分佈或分佈租用戶 ID。您必須擁有與涵蓋指定網域的憑證相關聯的標準分佈或分佈租用戶。
 - c. 使用 AWS 帳戶 與目標標準分佈或分佈租用戶位於相同 的登入資料來執行此命令。

請求

此範例會指定分佈租用戶。

```
aws cloudfront list-domain-conflicts \  
--domain www.example.com \  
--domain-control-validation-resource  
"DistributionTenantId=dt_2x9GhoK0TZRsohWzv1b9It8JABC"
```

回應

對於命令輸出中的每個網域名稱，您可以看到以下內容：

- 與網域相關聯的資源類型
- 資源 ID
- 擁有資源的 AWS 帳戶 ID

資源 ID 和帳戶 ID 會部分隱藏。這可讓您識別屬於您帳戶的標準分佈或分佈租用戶，並協助保護非您擁有的分佈或分佈租用戶的資訊。

```
{
  "DomainConflicts": [
    {
      "Domain": "www.example.com",
      "ResourceType": "distribution-tenant",
      "ResourceId": "*****ohWzv1b9It8JABC",
      "AccountId": "*****112233"
    }
  ]
}
```

回應會列出與您指定的網域名稱衝突或重疊的所有網域名稱。

範例

- 如果您指定 *tenant1.example.com*，回應會包含 tenant1.example.com 和重疊的萬用字元備用網域名稱 (*.example.com，如果有的話)。
 - 如果您指定 **.tenant1.example.com*，回應會包含 *.tenant1.example.com 和該萬用字元涵蓋的任何備用網域名稱 (例如 test.tenant1.example.com、dev.tenant1.example.com 等)。
- 在回應中，尋找您要移動之替代網域名稱的來源標準分佈或分佈租用戶，並記下 AWS 帳戶 ID。
 - 將來源標準分佈或分佈租用戶的帳戶 ID，與您在上一個步驟中建立目標標準分佈或分佈租用戶的帳戶 ID 進行比較。然後，您可以判斷來源和目標是否位於相同的 AWS 帳戶中。這可協助您判斷如何移動替代網域名稱。

如需詳細資訊，請參閱《AWS Command Line Interface 參考》中的 [list-domain-conflicts](#) 命令。

list-conflicting-aliases (standard distributions only)

 Tip

您必須擁有目標標準分佈的 `cloudfront:GetDistribution` 和 `cloudfront:ListConflictingAliases` 許可。

使用 **list-conflicting-aliases** 尋找來源標準分佈

1. 請使用 `list-conflicting-aliases` 指令，如下列範例所示。
 - a. 以替代網域名稱取代 `www.example.com`，用[您先前設定之](#)目標標準分佈的 ID 取代 `EDFDVBD6EXAMPLE`。
 - b. 使用與目標標準分佈相同 AWS 帳戶 中的憑證來執行此命令。

請求

此範例會指定標準分佈。

```
aws cloudfront list-conflicting-aliases \  
--alias www.example.com \  
--distribution-id EDFDVBD6EXAMPLE
```

回應

針對命令輸出中的每個替代網域名稱，您可以看到與替代網域名稱關聯之標準分佈的 ID，以及擁有該標準分佈的 AWS 帳戶 ID。標準分佈和帳戶 ID 會部分隱藏，這樣既可以讓您識別出您擁有的標準分佈和帳戶，同時又可以避免洩漏非您擁有之標準分佈和帳戶的資訊。

```
{  
  "ConflictingAliasesList": {  
    "MaxItems": 100,  
    "Quantity": 1,  
    "Items": [  
      {  
        "Alias": "www.example.com",  
        "DistributionId": "*****EXAMPLE",  
        "AccountId": "*****112233"  
      }  
    ]  
  }  
}
```

```
    ]  
  }  
}
```

回應會列出與您指定的網域名稱衝突或重疊的備用網域名稱。

範例

- 如果您指定 *www.example.com*，則回應會包含 `www.example.com` 以及 (如果存在) 重疊的萬用字元替代網域名稱 (`*.example.com`)。
 - 如果您指定 **.example.com*，則回應會包含 `*.example.com` 以及該萬用字元所涵蓋的任何替代網域名稱 (例如，`www.example.com`、`test.example.com`、`dev.example.com` 等)。
2. 尋找您要移動之備用網域名稱的標準分佈，並記下 AWS 帳戶 ID。將此帳戶 ID 與您在[上一個步驟](#)中建立目標標準分佈的帳戶 ID 進行比較。然後，您可以判斷這兩個標準分佈是否位於相同，AWS 帳戶 以及如何移動替代網域名稱。

如需詳細資訊，請參閱《AWS Command Line Interface 參考》中的 [list-conflicting-aliases](#) 命令。

然後，請參閱下列主題以移動替代網域名稱。

移動替代網域名稱

根據您的狀況，從以下內容選擇移動替代網域名稱的方式：

來源和目標分佈 (標準或租用戶) 位於相同的 AWS 帳戶

使用 AWS Command Line Interface (AWS CLI) 中的 `update-domain-association` 命令來移動替代網域名稱。

此命令適用於所有相同帳戶的移動，包括當替代網域名稱是 Apex 網域時 (也稱為根網域，例如 `example.com`)。

來源和目標分佈 (標準或租用戶) 位於不同的 AWS 帳戶

如果您具有來源標準分佈或分佈租用戶的存取權，替代網域名稱不是 Apex 網域，而且您尚未使用與該替代網域名稱重疊的萬用字元，請使用萬用字元來移動替代網域名稱。如需詳細資訊，請參閱[the section called “使用萬用字元來移動替代網域名稱”](#)。

如果您無法存取 AWS 帳戶 具有來源標準分佈或分佈租用戶的，您可以嘗試使用 `update-domain-association` 命令來移動替代網域名稱。您必須先停用來源標準分佈或分佈租用戶，才能移動替代網域名稱。如需其他說明，請參閱 [the section called “聯絡 AWS 支援 以移動替代網域名稱”](#)。

Note

您可以使用 `associate-alias` 命令，但此命令僅支援標準分佈。請參閱《Amazon CloudFront API 參考》中的 [AssociateAlias](#)。

`update-domain-association` (standard distributions and distribution tenants)

如要使用 **`update-domain-association`** 移動替代網域名稱

1. 使用 `update-domain-association` 命令，如下列範例所示。
 - a. 以替代網域名稱取代 *example.com*，並指定目標標準分佈或分佈租用戶的 ID。
 - b. 使用與目標標準分佈或分佈租用戶相同 AWS 帳戶 中的憑證來執行此命令。

請注意以下限制

- 除了 `cloudfront:UpdateDomainAssociation` 許可之外，您還必須擁有更新標準分佈的 `cloudfront:UpdateDistribution` 許可。若要更新分佈租用戶，您必須擁有 `cloudfront:UpdateDistributionTenant` 許可。
- 如果來源和目標分佈（標準或租用戶）位於不同位置 AWS 帳戶，則必須先停用來源，才能移動網域。
- 目標分佈必須依照 [the section called “設定目標標準分佈或分佈租用戶”](#) 所述的方式設定。

請求

```
aws cloudfront update-domain-association \  
  --domain "www.example.com" \  
  --target-resource DistributionTenantId=dt_9Fd3xTZq7Hl2KABC \  
  --if-match E3UN6WX5ABC123
```


回應

```
{
  "ETag": "E7Xp1Y3N9DABC",
  "Domain": "www.example.com",
  "ResourceId": "dt_9Fd3xTZq7H12KABC"
}
```

此命令會從來源標準分佈或分佈租用戶中移除替代網域名稱，並將其新增至目標標準分佈或分佈租用戶。

2. 完全部署目標分佈之後，請更新您的 DNS 組態，將您的網域名稱指向 CloudFront 路由端點。例如，您的 DNS 記錄會將您的備用網域名稱 (www.example.com) 指向 CloudFront 提供的網域名稱 d111111abcdef8.cloudfront.net。如果目標是分佈租用戶，請指定連線群組端點。如需詳細資訊，請參閱[將網域指向 CloudFront](#)。

associate-alias (standard distributions only)

如要使用 **associate-alias** 移動替代網域名稱

1. 使用 `associate-alias` 命令，如下列範例所示。
 - a. 以替代網域名稱取代 *www.example.com*，以具有目標標準分佈 ID 取代 *EDFDVBDGEXAMPLE*。
 - b. 使用與目標標準分佈相同 AWS 帳戶 中的憑證來執行此命令。

請注意以下限制

- 您必須擁有目標標準分佈的 `cloudfront:AssociateAlias` 和 `cloudfront:UpdateDistribution` 許可。
- 如果來源和目標標準分佈是相同的 AWS 帳戶，您必須擁有來源標準分佈的 `cloudfront:UpdateDistribution` 許可。
- 如果來源標準分佈和目標標準分佈不同 AWS 帳戶，您必須先停用來源標準分佈。
- 目標標準分佈必須依照 [the section called “設定目標標準分佈或分佈租用戶”](#) 所述的方式設定。

請求

```
aws cloudfront associate-alias \  
--alias www.example.com \  
--target-distribution-id EDFDVBD6EXAMPLE
```

此命令會從來源標準分佈中移除該替代網域名稱，並將其移動至目標標準分佈。

- 目標標準分佈完全部署之後，請更新 DNS 組態，將替代網域名稱的 DNS 紀錄指向目標標準分佈的分佈網域名稱。例如，您的 DNS 記錄會將您的備用網域名稱 (*www.example.com*) 指向 CloudFront 提供的網域名稱 *d111111abcdef8.cloudfront.net*。

如需詳細資訊，請參閱《AWS CLI 命令參考》中的 [associate-alias](#) 命令。

使用萬用字源來移動替代網域名稱

如果來源分佈與目標分佈 AWS 帳戶 不同，且來源分佈已啟用，您可以使用萬用字元來移動替代網域名稱。

Note

無法使用萬用字元來移動 Apex 網域 (例如 *example.com*)。若要在來源分佈和目標分佈位於不同的 AWS 帳戶時移動 Apex 網域，請聯絡 支援。如需詳細資訊，請參閱 [the section called “聯絡 AWS 支援 以移動替代網域名稱”](#)。

使用萬用字源來移動替代網域名稱

Note

此程序涉及對分佈的多次更新。等待每個分佈完整部署最新的變更後，再進行下一步。

- 更新目標分佈，以新增涵蓋您要移動之替代網域名稱的萬用字元替代網域名稱。例如，如果您要移動的替代網域名稱為 *www.example.com*，請將該替代網域名稱 **.example.com* 新增到目標分佈。若要執行這項操作，目標分佈上的 SSL/TLS 憑證必須包含萬用字元網域名稱。如需詳細資訊，請參閱 [the section called “更新分佈”](#)。

2. 更新替代網域名稱的 DNS 設定，以指向目標分佈的網域名稱。例如，如果您要移動的替代網域名稱是 `www.example.com`，請更新 `www.example.com` 的 DNS 紀錄，以將流量路由至目標分佈的網域名稱 (例如，`d1111111abcdef8.cloudfront.net`)。

 Note

在您更新 DNS 設定後，替代網域名稱仍會由來源分佈提供，因為這是替代網域名稱目前設定的所在位置。

3. 更新來源分佈來移除替代網域名稱。如需詳細資訊，請參閱 [更新分佈](#)。
4. 更新目標分佈來新增替代網域名稱。如需詳細資訊，請參閱 [更新分佈](#)。
5. 使用 `dig` (或類似的 DNS 查詢工具)，以驗證替代網域名稱的 DNS 紀錄是否解析為目標分佈的網域名稱。
6. (選用) 更新目標分佈來移除萬用字源替代網域名稱。

聯絡 AWS 支援 以移動替代網域名稱

如果來源和目標分佈位於不同的 AWS 帳戶，而且您無法存取來源分佈的 AWS 帳戶 或 無法停用來源分佈，您可以聯絡 支援 來移動替代網域名稱。

聯絡 支援 以移動替代網域名稱

1. 設定目標分佈，包括指向目標分佈的 DNS TXT 紀錄。如需詳細資訊，請參閱 [設定目標標準分佈或分佈租用戶](#)。
2. [請聯絡 支援](#) 來請求他們驗證您是否擁有該網域，並為您將網域新增到新的 CloudFront 分佈。
3. 完全目標分佈部署之後，請更新 DNS 組態，將替代網域名稱的 DNS 紀錄指向目標分佈的分佈網域名稱。

移除替代網域名稱

如果您想要停止路由網域或子網域的流量到 CloudFront 分佈，請依照此節中的下列步驟來更新 DNS 組態和 CloudFront 分佈。

您務必要從分佈移除備用網域名稱，並更新您的 DNS 組態。若您想要將網域名稱與另一個 CloudFront 分佈建立關聯，將有助於避免之後發生問題。如果備用網域名稱已與一個分佈建立關聯，即無法以另一個分佈設定。

 Note

如果您想要從這個分發移除備用網域名稱來將該地址新增到另一個分發，請按照 [移動替代網域名稱](#) 中的步驟。如果您依據此處的步驟 (來移除網域)，然後新增網域到另一個分佈，將有一段時間內網域不會連結到新的分佈，因為 CloudFront 正在傳輸更新到節點。

從分佈移動備用網域名稱

1. 若要開始，請將網域的網際網路流量路由到另一個非 CloudFront 分佈的資源，例如 ELB 負載平衡器。或者，您可以刪除不將流量路由傳送到 CloudFront 的 DNS 記錄。

執行以下其中一項，取決於您網域的 DNS 服務：

- 如果您使用的是 Route 53，請更新或刪除別名記錄或 CNAME 記錄。如需詳細資訊，請參閱[編輯記錄](#)或[刪除記錄](#)。
 - 如果您使用的是另一個 DNS 服務供應商，請使用由 DNS 服務供應商提供的方法更新或刪除將流量導向 CloudFront 的 CNAME 記錄。如需詳細資訊，請參閱 DNS 服務供應商提供的說明文件。
2. 在您更新網域的 DNS 紀錄，請等待變更傳播且 DNS 解析程式會路由流量到新的資源。您可以建立一些使用 URL 中網域的測試連結來確認此步驟是否完成。
 3. 登入 AWS 管理主控台 並開啟位於的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>，然後執行下列動作，更新您的 CloudFront 分佈以移除網域名稱：
 - a. 選擇您希望更新的分佈 ID。
 - b. 在 General (一般) 索引標籤上，選擇 Edit (編輯)。
 - c. 在 Alternate Domain Names (CNAMEs) (正式名稱記錄 (CNAME)) 中，請移除您已不想再用於分佈的備用網域名稱 (或網域名稱)。
 - d. 請選擇 Yes, Edit (是，編輯)。

在備用網域名稱中使用萬用字元

當您新增備用網域名稱時，可以改為在網域名稱的開頭使用 * 萬用字元，而非新增個別子網域。例如，使用替代網域名稱 *.example.com，您可以在 URL 中使用任何結尾為 example.com 的網域名稱，例如 www.example.com、product-name.example.com、marketing.product-name.example.com 等。不管網域名稱為何，物件的路徑都是一樣的，例如：

- `www.example.com/images/image.jpg`
- `product-name.example.com/images/image.jpg`
- `marketing.product-name.example.com/images/image.jpg`

按照這些包括萬用字元的備用網域名稱要求：

- 替代網域名稱必須以星號和點號 (*) 開頭。
- 您不能使用萬用字元來取代子網域名稱的部分，例如：`*domain.example.com`。
- 您不得取代網域名稱中的子網域，例如：`subdomain.*.example.com`。
- 所有備用網域名稱 (包括使用萬用字元的備用網域名稱) 都必須涵蓋在憑證的主體別名 (SAN) 下。

萬用字元替代網域名稱 (例如 `*.example.com`) 可以包含其他正在使用中的替代網域名稱，例如 `example.com`。

使用 WebSockets 搭配 CloudFront 分佈

Amazon CloudFront 支援使用 WebSocket，這是一種 TCP 型通訊協定，當您在用戶端與伺服器之間需要長時間的雙向連線時，此通訊協定很有用。持久性連線通常是即時應用程式的一項要求。您可能在其中使用 WebSockets 的案例包括社交聊天平台、線上協作工作區、多玩家遊戲，以及可提供即時資料饋送的服務，像是金融交易平台。若為全雙工通訊，透過 WebSocket 連線的資料可以雙向流通。

WebSocket 功能會自動啟用，以使用任何分佈。若要使用 WebSocket，請在連接至分佈的快取行為中設定下列其中一項：

- 轉送所有檢視器請求標頭到原始伺服器。您可以使用 [AllViewer 受管原始伺服器請求政策](#)。
- 具體轉送您原始伺服器請求政策中的 `Sec-WebSocket-Key` 和 `Sec-WebSocket-Version` 請求標頭。

WebSocket 通訊協定的運作方式

WebSocket 通訊協定是一種獨立的 TCP 型通訊協定，可讓您避免一些負荷，但可能增加 HTTP 的延遲。

若要建立 WebSocket 連線，用戶端會傳送一般 HTTP 請求，其會使用 HTTP 的升級語意來變更通訊協定。然後，伺服器便可以完成交握。WebSocket 連線會保持開啟狀態，而且用戶端或伺服器可以彼此傳送資料框架，而無需每次建立新連線。

根據預設，WebSocket 通訊協定會使用連接埠 80，進行一般 WebSocket 連線，以及使用連接埠 443，進行透過 TLS 的 WebSocket 連線。您為 CloudFront [檢視器通訊協定政策](#) 和 [通訊協定 \(僅限自訂原始伺服器\)](#) 選擇的選項適用於 WebSocket 連線，也適用於 HTTP 流量。

WebSocket 要求

WebSocket 請求必須採取以下標準格式，以符合 [RFC 6455](#)。

Example 範例用戶端請求

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: https://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
```

Example 範例伺服器回應

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+x0o=
Sec-WebSocket-Protocol: chat
```

如果 WebSocket 連線是由用戶端或伺服器中斷，或由於網路干擾而中斷，則預期用戶端應用程式會重新啟動與伺服器的連線。

建議的 WebSocket 標頭

為了避免在使用 WebSockets 時發生非預期的壓縮相關問題，建議您在[原始伺服器請求政策](#)中包含下列標頭：

- Sec-WebSocket-Key
- Sec-WebSocket-Version
- Sec-WebSocket-Protocol
- Sec-WebSocket-Accept
- Sec-WebSocket-Extensions

 Note

目前，CloudFront 僅支援透過 HTTP/1.1 通訊協定的 WebSocket 連線。

請求用於列入允許清單的 Anycast 靜態 IP

您可以從 CloudFront 請求 Anycast 靜態 IP 以搭配您的分佈使用。任何傳送靜態 IP 清單只能包含 IPv4 IP 地址或 IPv4 和 IPv6 IP 地址。這些 IP 地址專用於您的 AWS 帳戶，並分散到各個地理區域。

您可以申請 21 個 Anycast 靜態 IP 位址，並將它們加入網路提供者的允許清單，以便讓存取您應用程式的使用者免除資料傳輸費用。或者，您可以在輸出安全防火牆內使用這些靜態 IP，以控制與已核准應用程式的流量交換。Anycast 靜態 IP 清單可與一或多個分佈搭配使用。

如果您想要直接為 CloudFront 分佈啟用 Apex 網域路由 (例如 example.com)，您可以為此使用案例請求 3 個 Anycast 靜態 IP 位址。然後，在您的 DNS 中新增 A 記錄，將 Apex 網域指向 CloudFront。

Anycast 靜態 IP 可與[伺服器名稱指示 \(SNI\)](#) 搭配使用。如需詳細資訊，請參閱[使用 SNI 提供 HTTPS 請求 \(適用於大部分用戶端\)](#)。

先決條件

若要搭配 CloudFront 分佈使用 Anycast 靜態 IP 清單，您必須為分佈的價格分級選取使用所有節點。如需定價的詳細資訊，請參閱[CloudFront 定價](#)。

請求 Anycast 靜態 IP 清單

請求 Anycast 靜態 IP 清單以搭配 CloudFront 分佈使用。

請求 Anycast 靜態 IP 清單

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在左側導覽窗格中，選擇靜態 IP。
3. 針對請求，請選擇連結以聯絡 CloudFront 支援工程。
4. 提供工作負載資訊 (每秒請求位元組數和每秒請求數)。
5. CloudFront 支援工程審查您的請求。審核程序最多可能需要兩天的時間。

請求獲准後，您可以建立 Anycast 靜態 IP 清單，並將其與一或多個分佈建立關聯。

建立 Anycast 靜態 IP 清單

在開始之前，請先請求 Anycast 靜態 IP 清單，如上一節所述。

建立 Anycast 靜態 IP 清單

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在左側導覽窗格中，選擇靜態 IP。
3. 選擇建立 Anycast IP 清單。
4. 針對名稱，輸入名稱。
5. 針對靜態 IP 使用案例，選取適當的使用案例。
6. 針對 IP 地址類型，指定下列其中一個選項：
 - IPv4 – 配置僅 IPv4 地址的清單
 - Dualstack – 配置 IPv4 和 IPv6 地址的清單
7. 檢閱服務條款和定價，然後選擇提交。

建立靜態 IP 清單後，您可以在靜態 IP 清單詳細資訊頁面上檢視配置的 IP 位址。您也可以將分佈與靜態 IP 清單建立關聯。

將 Anycast 靜態 IP 清單與現有分佈建立關聯

在開始之前，請求並建立 Anycast 靜態 IP 清單，如前幾節所述。

確認下列分佈設定與您的 Anycast 靜態 IP 清單相容：

- [價格分級](#) 具有使用所有節點（最佳效能）設定。
- 如果啟用 [IPv6](#)，您可以建立雙堆疊 Anycast 靜態 IP 清單的關聯。僅具有 IPv4 地址的 Anycast 靜態 IP 清單無法與已啟用 IPv6 的分佈建立關聯。

將 Anycast 靜態 IP 清單與現有分佈建立關聯

- 執行以下任意一項：

- 從靜態 IP 清單詳細資訊頁面關聯靜態 IP 清單：
 1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
 2. 在左側導覽窗格中，選擇 固定 IP。
 3. 選擇固定 IP 清單的名稱。
 4. 選擇關聯分佈。
 5. 選取一或多個分佈，然後選擇關聯分佈。
- 從分佈詳細資訊頁面關聯靜態 IP 清單：
 1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
 2. 在左側導覽窗格中，選擇分佈。
 3. 選擇您分佈的名稱。
 4. 在設定窗格下的一般標籤中，選擇編輯。
 5. 針對 Anycast IP 清單，選取要與此分佈搭配使用的 Anycast 靜態 IP 清單。
 6. 選擇儲存變更。

將 Anycast 靜態 IP 清單與新分佈建立關聯

在開始之前，請求並建立 Anycast 靜態 IP 清單，如前幾節所述。

將 Anycast 靜態 IP 清單與新分佈建立關聯

- 建立新的 分佈。如需詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。針對設定，您必須進行下列選擇，才能使用 Anycast 靜態 IP 清單：
 - 針對 Anycast IP 清單，從下拉式清單中選取您的 Anycast 靜態 IP 清單。
 - 針對價格方案，選擇使用所有邊緣節點 (最佳效能)。
 - 注意：如果您的 Anycast 靜態 IP 僅使用 IPv4 而非雙堆疊，請在 IPv6 中選取關閉。

完成建立分佈。您可以根據您的需求，選擇 Anycast 靜態 IP 清單中不需要的任何其他設定和組態。

如需 Anycast 靜態 IP 清單相關配額的詳細資訊，請參閱 AWS 一般參考 中的 [Amazon CloudFront 端點和配額](#)。

將 Anycast 靜態 IP 清單與連線群組建立關聯

開始之前，請請求並建立 Anycast 靜態 IP 清單，如前幾節所述。

將 Anycast 靜態 IP 清單與新的連線群組建立關聯

1. 請確定您已在設定下啟用連線群組。
2. 建立連線群組。如需詳細資訊，請參閱[建立自訂連線群組](#)。
3. 針對設定，您必須進行下列選擇，才能使用 Anycast 靜態 IP 清單。
 - 針對 Anycast IP 清單，從下拉式清單中選取您的 Anycast 靜態 IP 清單。
4. 完成建立連線群組。

Note

如果您的 Anycast 靜態 IP 僅使用 IPv4 而非雙堆疊，請針對 IPv6 選取關閉。

如需 Anycast 靜態 IP 清單相關配額的詳細資訊，請參閱 Amazon Web Services 一般參考 中的 [Amazon CloudFront 端點和配額](#)。

更新 Anycast 靜態 IP 清單

建立 Anycast 靜態 IP 地址並將其關聯至分佈後，您可以變更 Anycast 靜態 IP 清單的 IP 地址類型。

更新 Anycast 靜態 IP 清單

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在左側導覽窗格中，選擇靜態 IP。
3. 選擇固定 IP 清單的名稱。
4. 選擇編輯。
5. 針對 IP 地址類型，指定下列其中一個選項：
 - IPv4 – 配置僅 IPv4 地址的清單
 - Dualstack – 配置 IPv4 和 IPv6 地址的清單

Note

如果您的關聯分佈已啟用 IPv6，則無法選擇 IPv4。IPv6 若要這麼做，請先停用 IPv6，然後才能更新 Anycast 靜態 IP 的 IP 地址類型。如需詳細資訊，請參閱[為 CloudFront 分佈啟用 IPv6](#)。

6. 選擇提交以儲存變更並更新 Anycast 靜態 IP 清單。

使用 IPAM 在 CloudFront 使用自有 IP

本教學課程說明如何使用 IPAM 來管理 CloudFront Anycast 靜態 IP 清單的 BYOIP CIDRs。

主題

- [什麼是 Anycast 靜態 IPs 的 BYOIP？](#)
- [為什麼要使用此功能？](#)
- [先決條件](#)
- [步驟 1：請求 Anycast 靜態 IP 清單](#)
- [步驟 2：建立 Anycast 靜態 IP 清單](#)
- [步驟 3：建立 CloudFront 分佈](#)
- [步驟 4：與 CloudFront 資源建立關聯](#)
- [步驟 5：準備遷移](#)
- [步驟 6：全域公告 CIDR](#)

什麼是 Anycast 靜態 IPs 的 BYOIP？

CloudFront 支援透過 IPAM 全球服務的 BYOIP 攜帶您自己的 IPv4 地址。透過 IPAM 的統一界面，客戶可以使用自己的 IP 地址 (BYOIP) 建立專用 IP 地址集區，並將其指派給 CloudFront 分發，同時利用 AWS 全球內容交付網路來交付其應用程式和內容。您的 IP 地址會使用任何廣播路由，從多個 CloudFront 節點同時公告。

為什麼要使用此功能？

在允許清單中控制網路存取，以：

- 允許列出網路供應商的 IP 地址，以免除已核准瀏覽者的資料費用

- 設定傳出安全防火牆，將流量限制為僅核准的應用程式

簡化操作和遷移

- 透過新增指向靜態 IPs 的 A 記錄，將 apex 網域 (example.com : //) 直接路由至 CloudFront
- 從其他 CDNs 遷移而不更新 IP 基礎設施或防火牆組態
- 與合作夥伴和用戶端維護現有的 IP 允許清單
- 在多個 CloudFront 分佈之間共用單一 Anycast 靜態 IP 清單

一致的品牌

- 移至 時，請保留現有的 IP 地址空間以保持一致的品牌 AWS

先決條件

若要搭配 CloudFront 分佈使用 Anycast 靜態 IP 清單，您必須為分佈的價格分級選取使用所有節點。如需定價的詳細資訊，請參閱 [CloudFront 定價](#)。對於自攜 IP (BYOIP)，您也需要停用分發或連線群組的 IPv6。

請先完成下列步驟，再開始：

- IPAM 設定：請參閱[整合 IPAM 與帳戶](#)和[建立 IPAM](#)。
- 網域驗證：[驗證網域控制](#)。
- 建立頂層集區：請遵循將[您自己的 IPv4 CIDR 帶到 IPAM](#) 中的步驟 1 到 2。
- 建立具有全域地區設定的 IPAM 集區，以搭配 CloudFront 使用。如需詳細資訊，請參閱[使用 IPAM 將您自己的 IP 帶到 CloudFront](#)。

Note

需要三個 /24 IPv4 CIDR 區塊。

步驟 1：請求 Anycast 靜態 IP 清單

請求 Anycast 靜態 IP 清單以搭配 CloudFront 分佈使用。

請求 Anycast 靜態 IP 清單

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在左側導覽窗格中，選擇靜態 IP。
3. 針對請求，請選擇連結以聯絡 CloudFront 支援工程。
4. 提供工作負載資訊 (每秒請求位元組數和每秒請求數)。
5. CloudFront 支援工程審查您的請求。審核程序最多可能需要兩天的時間。
6. 請求獲准後，您可以建立 Anycast 靜態 IP 清單，並將其與一或多個分佈建立關聯。

步驟 2：建立 Anycast 靜態 IP 清單

在開始之前，請先請求 Anycast 靜態 IP 清單，如上一節所述。

建立 Anycast 靜態 IP 清單

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在左側導覽窗格中，選擇靜態 IP。
3. 選擇建立 Anycast IP 清單。
4. 針對名稱，輸入名稱。
5. 針對靜態 IP 使用案例，選取 BYOIP 做為您的使用案例。

下列步驟與標準區域 BYOIP 程序不同，並建立全域服務的模式：

AWS CLI

安裝或更新至最新版本的 AWS CLI。如需詳細資訊，請參閱「[AWS Command Line Interface 使用者指南](#)」。

1. 擷取佈建 CIDR 區塊之 IPAM 集區的 IpamPoolArn。如需詳細資訊，請參閱[僅使用 CLI 將您自己的公有 IPv4 CIDR AWS 帶入 IPAM](#)。
2. 使用 CIDR 區塊和 IPAM 組態建立 Anycast IP 清單：

```
aws cloudfront create-anycast-ip-list \  
    --name byoip-aip-1 \  
    --ipam-pool-arn IPAM_POOL_ARN
```

```
--ip-count 3 \  
--region us-east-1 \  
--ip-address-type ipv4 \  
--ipam-cidr-configs  
'[{"Cidr": "1.1.1.0/24", "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-pool-005d58a8aa8147abc"},  
{"Cidr": "2.2.2.0/24", "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-pool-005d58a8aa8147abc"},  
{"Cidr": "3.3.3.0/24", "IpamPoolArn": "arn:aws:ec2::123456789012:ipam-pool/ipam-pool-005d58a8aa8147abc"}]'
```

Note

您無法從集區選取特定 IP 地址。CloudFront 會自動執行此操作。

步驟 3：建立 CloudFront 分佈

對於 CloudFront，您可以遵循指示[來建立標準分佈](#)或使用[多租用戶分佈](#)。

步驟 4：與 CloudFront 資源建立關聯

- [將 Anycast 靜態 IP 清單與現有分佈建立關聯](#)
- [將 Anycast 靜態 IP 清單與新分佈建立關聯](#)
- [將 Anycast 靜態 IP 清單與連線群組建立關聯](#)

步驟 5：準備遷移

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的[步驟 4：準備遷移](#)。

步驟 6：全域公告 CIDR

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的[步驟 5：全域公告 CIDR](#)。

使用 gRPC 搭配 CloudFront 分佈

Amazon CloudFront 支援 gRPC，這是一種以 HTTP/2 為基礎建置的開放原始碼遠端程序呼叫 (RPC) 架構。gRPC 提供雙向串流和二進位通訊協定緩衝承載，因此非常適合需要低延遲通訊的應用程式。

CloudFront 會接收您的 gRPC 請求，並將其直接代理到您的原始伺服器。您可以使用 CloudFront 來代理四種類型的 gRPC 服務：

- 一元 RPC
- 伺服器串流 RPC
- 用戶端串流 RPC
- 雙向串流 RPC

gRPC 如何在 CloudFront 中運作

若要在 CloudFront 中設定 gRPC，請將提供 gRPC 服務的原始伺服器設定為分佈的原始伺服器。您可以使用同時提供非 gRPC 和 gRPC 服務的原始伺服器。CloudFront 會根據 Content-Type 標頭判斷傳入請求是 gRPC 請求還是 HTTP/HTTPS 請求。如果請求的 Content-Type 標頭值為 application/grpc，則該請求會被視為 gRPC 請求，且 CloudFront 會將請求代理到您的原始伺服器。

Note

如要啟用發佈以處理 gRPC 請求，請將 HTTP/2 納入為支援的 HTTP 版本之一，並允許 HTTP 方法 (包括 POST)。您的 gRPC 原始伺服器端點必須設定為支援 HTTPS，因為 CloudFront 端僅支援安全 (以 HTTP 為基礎的) gRPC 連線。gRPC 僅支援端對端 HTTPS。如果您使用的是自訂原始伺服器，請確認您的[通訊協定](#)設定支援 HTTPS。

若要為您的分佈啟用 gRPC 支援，請完成下列步驟：

1. 更新分佈的快取行為以允許 HTTP 方法，包括 POST 方法。
2. 在您選取 POST 方法後，選取出現的 gRPC 核取方塊。
3. 將 HTTP/2 指定為其中一個支援的 HTTP 版本。

如需詳細資訊，請參閱下列主題：

- [允許透過 HTTP/2 進行 gRPC 請求](#)
- 《Amazon CloudFront API 參考》中的 [GrpcConfig](#)

由於 gRPC 僅適用於不可快取的 API 流量，因此您的快取設定不會影響 gRPC 請求。您可以使用原始伺服器請求政策將自訂標頭新增至傳送至 gRPC 原始伺服器的 gRPC 請求。您可以搭配 CloudFront 使用 AWS WAF 來管理對 gRPC 分佈的存取、控制機器人，以及保護您的 gRPC 應用程式免受網路入侵攻擊。CloudFront gRPC 支援 [CloudFront Functions](#)。

除了 HTTPS 狀態之外，您還會收到 gRPC 回應的 grpc-status。如需 grpc-status 的可能值清單，請參閱[狀態碼及其在 gRPC 中的使用狀況](#)。

備註

gRPC 不支援以下 CloudFront 功能：

- [自訂錯誤回應](#)
- gRPC 不支援[原始伺服器容錯移轉](#)，因為 gRPC 使用 POST 方法。只有當檢視器請求的 HTTP 方法為 GET、HEAD 或 OPTIONS 時，CloudFront 才會容錯移轉至次要原始伺服器。
- CloudFront 會將 gRPC 請求直接傳送至原始伺服器，並略過區域邊緣快取 (REC)。由於 gRPC 會略過 REC，所以 gRPC 不支援 [Lambda@Edge](#) 或 [Origin Shield](#)。
- gRPC 不支援 AWS WAF 請求內文檢查規則。如果您在 Web ACL 上為分佈啟用這些規則，任何使用 gRPC 的請求都會忽略請求本文檢查規則。所有其他 AWS WAF 規則仍將適用。如需詳細資訊，請參閱[為分佈啟用 AWS WAF](#)。

在 CloudFront 中使用共用資源

Amazon CloudFront 與 AWS Resource Access Manager (AWS RAM) 整合以啟用資源共用。AWS RAM 可讓您與其他 AWS 帳戶 或透過 共用一些 CloudFront 資源 AWS Organizations。您可以透過 AWS RAM 建立資源共享，以共用您擁有的資源。資源共享指定要共用的資源，以及共用它們的消費者。消費者可包括：

- 中的特定組織 AWS 帳戶 內部或外部 AWS Organizations
- 中的組織單位 AWS Organizations
- 其在 中的整個組織 AWS Organizations

如需 的詳細資訊 AWS RAM，請參閱 [AWS RAM 使用者指南](#)。

本主題說明如何共用您擁有的參數，以及如何使用與您共用的參數。

目錄

- [共用資源的先決條件](#)
- [共用 VPC 原始伺服器](#)
- [使用共用 VPC 原始伺服器](#)
- [識別共用 VPC 原始伺服器](#)
- [取消共用共用的 VPC 原始伺服器](#)
- [共用 VPC 原始伺服器的責任和許可](#)
- [計費和計量](#)
- [共用資源配額](#)

共用資源的先決條件

- 您必須擁有 `AWSRAMDefaultPermissionCloudFront` 受管政策，才能授予資源共享的唯讀存取權。如需詳細資訊，請參閱 [AWSRAMDefaultPermissionCloudFront](#)。
- 若要共用 VPC 原始伺服器，您必須在 中擁有該原始伺服器 AWS 帳戶。這表示必須在您的帳戶中配置或佈建資源。您無法共用已與您共用的資源。
- 若要與組織或 中的組織單位共用資源 AWS Organizations，您必須啟用與 共用 AWS Organizations。如需詳細資訊，請參閱《AWS RAM 使用者指南》中的 [透過 AWS Organizations 啟用共用](#)。

共用 VPC 原始伺服器

Note

目前，CloudFront 支援共用 VPC 原始伺服器。如果您尚未建立，請參閱 [使用 VPC 原始伺服器限制存取](#)。

當您與其他人共用您擁有的 VPC 原始伺服器時 AWS 帳戶，您可以讓他們使用該資源作為其 CloudFront 分佈的原始伺服器。

若要共用 VPC 原始伺服器，您必須將其新增至資源共用。資源共用是可讓您在 AWS 帳戶之間共用資源的一種 AWS RAM 資源。

資源共享會指定下列項目：

- 您要共用的資源
- 與其共用的消費者
- 決定資源許可的服務受管政策

當您使用 CloudFront 主控台共用 VPC 原始伺服器時，您可以將其新增至現有的資源共用。如果您還沒有資源共用，則可以在從 CloudFront 主控台共用 VPC 原始伺服器時建立資源共用。您也可以使用 [AWS RAM 主控台](#) 或單獨 AWS CLI 建立。

您可以與其他 AWS 帳戶 和 共用 VPC 原始伺服器 AWS Organizations。

- 如果您要與 AWS 組織共用資源，則該特定組織中的所有取用者都可以存取 VPC 原始伺服器。
- 如果您要與 AWS 帳戶 或不屬於您的組織共用，消費者會收到接受資源共用的邀請。一旦接受，他們可以使用 VPC 原始伺服器。

您可以使用 CloudFront 主控台、AWS RAM 主控台或 共用您擁有的 VPC 原始伺服器 AWS CLI。

使用 CloudFront 主控台建立資源共享

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 VPC 原始伺服器。
3. 選取一或多個資源，然後選擇共用 VPC 原始伺服器。

4. 選擇 Create resource share (建立資源共用)。
5. 在名稱中，輸入資源共享的名稱。
6. 針對委託人類型，選取下列其中一個選項：
 - AWS 帳戶 – 授予特定的存取權 AWS 帳戶。
 - 組織單位 – 授予特定組織單位 (OU) 的存取權。
 - 組織 – 授予整個組織的存取權，包括其子 OUs 和 AWS 帳戶。
- a. 如果您選擇 AWS 帳戶，請輸入帳戶 ID 號碼。您可以選擇新增帳戶以新增最多 5 個帳戶 AWS 帳戶。
- b. 如果您選擇組織單位，請輸入 OU 單位 ARN。您只能輸入 1 個 OU。
- c. 如果您選擇組織，請輸入組織 ARN。您只能輸入 1 個組織。
7. 選擇共用資源。

根據預設，CloudFront 會在資源共用上套用 [AWSRAMDefaultPermissionCloudFront](#) AWS 受管政策。此政策允許資源共用的唯讀動作，因此取用帳戶無法更新或刪除共用資源。您無法從資源共用編輯或移除此政策。

 Tip

建立資源共享之後，您可以從 AWS RAM 主控台新增其他 AWS 帳戶。如需詳細資訊，請參閱 AWS RAM 《使用者指南》中的 [更新 AWS RAM 中的資源共享](#)。

使用 CloudFront 主控台共用您擁有的 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 VPC 原始伺服器。
3. 選取資源，然後選擇共用 VPC 原始伺服器。
4. 在共用 VPC 原始伺服器頁面上，您可以選取要將此 VPC 原始伺服器新增至其中的現有資源共用。
5. 選擇共用資源。

在資源詳細資訊頁面的共用下，您可以看到您的 VPC 原始伺服器與下列詳細資訊共用：

- 資源共用名稱

- 共用狀態
- 上次修改日期

在您建立資源共享並與取用帳戶共用後，他們有 12 小時的時間接受邀請。如需詳細資訊，請參閱AWS RAM 《使用者指南》中的[接受和拒絕資源共用邀請](#)。

Important

若要讓取用帳戶將 VPC 原始伺服器用於其 CloudFront 分佈，您還必須提供它們 VPC 原始伺服器的 ELB 或 Amazon EC2 端點。

使用 AWS RAM 主控台共用您擁有的 VPC 原始伺服器

建立資源共用，然後選擇您要新增至其中的 CloudFront 資源。如需詳細資訊，請參閱AWS RAM 《使用者指南》中的[建立資源共享](#)。

使用 共享您擁有的 VPC 原始伺服器 AWS CLI

使用 [create-resource-share](#) 命令。

使用共用 VPC 原始伺服器

若要使用共用 VPC 原始伺服器，接收邀請的帳戶必須接受資源共用。您可以透過導覽至美國東部（維吉尼亞北部）區域的 AWS Resource Access Manager 主控台，並在待定索引標籤中接受任何待定請求來執行此操作。如需詳細資訊，請參閱AWS RAM 《使用者指南》中的[接受共用資源](#)。

接受資源共享之後，您就可以使用 VPC 原始伺服器做為 CloudFront 分佈的原始伺服器。

使用共用 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，針對分佈執行下列其中一項操作：
 - 對於新分佈，請選擇建立分佈。
 - 針對現有的分佈，選擇分佈 ID。
3. 針對原始伺服器類型，選擇 VPC 原始伺服器，然後指定與您共用的 VPC 原始伺服器。

4. 針對 VPC 原始伺服器端點，輸入 Amazon EC2 執行個體或 ELB 負載平衡器的私有 DNS 名稱，或原始網域。如果您還沒有此值，則必須從 AWS 帳戶擁有 VPC 原始伺服器的取得。如果您還沒有此端點，您可以從 AWS 帳戶擁有 VPC 原始伺服器的取得。
5. 請依照主控台的其餘步驟來建立或更新分佈。

識別共用 VPC 原始伺服器

擁有者和消費者可以使用 CloudFront 主控台和 來識別共用 VPC 原始伺服器 AWS CLI。

使用 CloudFront 主控台識別共用 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 VPC 原始伺服器。您可以使用擁有者 ID 欄來識別 AWS 帳戶 資源所屬的。
3. 選取資源。
4. 在資源詳細資訊頁面的共用下，您可以看到您的 VPC 原始伺服器與下列詳細資訊共用：
 - 資源共用名稱
 - 共用狀態
 - 上次修改日期

取消共用共用的 VPC 原始伺服器

當您取消共用資源時，AWS 帳戶（耗用帳戶）無法再將該資源用於新的分佈或更新現有的分佈。

Note

如果您取消共用資源，仍然使用該資源的現有分佈會保持作用中狀態，並繼續為流量提供服務。不過，在移除未共用資源做為原始伺服器之前，無法編輯這些分佈。我們建議您確保在取消共用之前，任何取用帳戶都會停止使用未共用的資源。

若要取消共用您擁有的共用 VPC 原始伺服器，您必須將其從資源共用中移除。您可以使用 CloudFront 主控台、AWS RAM 主控台或 來執行此操作 AWS CLI。

使用 CloudFront 主控台取消共用您擁有的共用 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。

2. 在導覽窗格中，選擇 VPC 原始伺服器。
3. 選取資源，然後選擇取消共用。
4. 檢閱取消共用資源對話方塊中的詳細資訊，然後選擇取消共用。列出的委託人將無法再存取您的共用資源。

使用 AWS RAM 主控台取消共用您擁有的共用 VPC 原始伺服器

請參閱《AWS RAM 使用者指南》中的[更新資源共享](#)。

使用 取消共用您擁有的共用 VPC 原始伺服器 AWS CLI

使用 [disassociate-resource-share](#) 命令。

共用 VPC 原始伺服器的責任和許可

擁有者的許可

作為資源擁有帳戶，請確保任何取用帳戶在取消共用或刪除資源之前停止使用資源。

消費者的許可

使用帳戶可以使用共用資源做為其 CloudFront 分佈的原始伺服器，但無法編輯或刪除資源。根據預設，[AWSRAMDefaultPermissionCloudFront](#) AWS 受管政策會套用至共用帳戶中的資源共用（擁有資源的帳戶）。

AWSRAMDefaultPermissionCloudFront

當您在 CloudFront 中建立資源共享時，CloudFront 會使用 AWSRAMDefaultPermissionCloudFront AWS 受管政策，並將其套用至您的資源共享。此政策會將唯讀許可授予 CloudFront 資源，這些資源可從資源擁有者共用給取用帳戶。

如需在 中管理許可的詳細資訊 AWS RAM，請參閱AWS Resource Access Manager 《使用者指南》中的[在 中管理許可 AWS RAM](#)。

計費和計量

與其他 共用 VPC 原始伺服器無需額外費用 AWS 帳戶。使用共用 VPC 原始伺服器之分佈的流量使用成本，將移至擁有該分佈的耗用帳戶。

共用資源配額

CloudFront 使用與 指定的相同資源共用配額 AWS RAM。從 CloudFront 主控台，您最多可以新增 5 AWS 帳戶個、1 個 OU 或 1 個組織。若要新增更多，請使用 AWS RAM 主控台或 AWS RAM API。

如需詳細資訊，請參閱《AWS RAM 使用者指南》中的 [AWS RAM 的 Service Quotas](#)。

快取和可用性

您可以使用 CloudFront 以減少原始伺服器必須直接回應的請求數。透過 CloudFront 快取，可從更接近您使用者的 CloudFront 節點提供更多物件。這樣可以降低原始伺服器上的負載並降低延遲。

CloudFront 可以從邊緣快取提取的請求越多，CloudFront 必須轉送至原始伺服器才能取得物件的最新版本或唯一版本的檢視者請求就越少。若要最佳化 CloudFront 以盡量減少對原始伺服器的請求，請考慮使用 CloudFront Origin Shield。如需更多詳細資訊，請參閱 [使用 Amazon CloudFront Origin Shield](#)。

直接從 CloudFront 快取提供的請求與所有請求的比例，稱為快取命中率。可以檢視檢視器在 CloudFront 主控台的命中數、未命中數和錯誤請求的百分比。如需更多詳細資訊，請參閱 [檢視 CloudFront 快取統計資料報告](#)。

諸多因素皆會影響快取命中率。您可以依照 [增加從 CloudFront 快取直接提供的請求比例 \(快取命中率\)](#) 所述指引來調整 CloudFront 分佈組態，藉此提高快取命中率。

若要了解如何新增及移除您希望 CloudFront 提供的內容，請參閱 [新增、移除或取代 CloudFront 分配的內容](#)。

主題

- [增加從 CloudFront 快取直接提供的請求比例 \(快取命中率\)](#)
- [使用 Amazon CloudFront Origin Shield](#)
- [透過 CloudFront 原始伺服器容錯移轉最佳化高可用性](#)
- [管理內容保持在快取中達多久時間 \(過期\)](#)
- [根據查詢字串參數快取內容](#)
- [根據 Cookie 快取內容](#)
- [根據請求標頭快取內容](#)

增加從 CloudFront 快取直接提供的請求比例 (快取命中率)

你可以藉由增加從 CloudFront 快取直接提供的瀏覽者請求比例 (而不是轉送到原始伺服器以取得內容) 的方式提升效能。這就是所謂的改善快取命中率。

下列各節說明如何提高您的快取命中率。

主題

- [指定 CloudFront 快取物件的時間長短](#)
- [使用 Origin Shield](#)
- [根據查詢字串參數快取](#)
- [根據 Cookie 值快取](#)
- [根據請求標頭快取](#)
- [不需要壓縮時，移除 Accept-Encoding 標頭](#)
- [透過 HTTP 提供媒體內容](#)

指定 CloudFront 快取物件的時間長短

若要提高快取命中率，您可以設定原始伺服器，將 [Cache-Control max-age](#) 指令新增至物件，並為 max-age 指定最長的實際值。快取持續時間越短，CloudFront 將請求傳送到原始伺服器的頻率越高，該請求旨在確定物件是否更改並獲取最新版本。您可以以 stale-while-revalidate 和 stale-if-error 指令補充 max-age，以在特定情況下進一步改善快取命中率。如需更多詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

使用 Origin Shield

CloudFront Origin Shield 可協助提高您 CloudFront 分佈的快取命中率，因為它會在您的原始伺服器前面提供額外的一層快取。當您使用 Origin Shield 時，從所有 CloudFront 快取層向原始伺服器提出的所有請求都來自單一位置。CloudFront 可以使用來自 Origin Shield 的單一原始伺服器請求擷取每個物件，而 CloudFront 快取的所有其他層 (節點和 [區域節點快取](#)) 都可以從 Origin Shield 擷取物件。

如需更多詳細資訊，請參閱 [使用 Amazon CloudFront Origin Shield](#)。

根據查詢字串參數快取

如果設定 CloudFront 來根據查詢字串參數快取，如果您執行下列操作便可以提升快取：

- 設定 CloudFront 僅轉送查詢字串參數，原始伺服器將回傳唯一物件給該參數。
- 對所有相同參數的執行個體使用相同的大小寫 (大寫或小寫)。例如，如果一個請求包含 parameter1=A 和另一個包含 parameter1=a，則 CloudFront 轉送個別的請求到原始伺服器，當請求包含 parameter1=A 以及當請求包含 parameter1=a 時。然後，CloudFront 分別快取由原始伺服器回傳的對應物件，即使物件相同。如果只是使用 A 或者 a，則 CloudFront 轉送較少的請求到原始伺服器。

- 以相同順序列出參數。就像案例中的差異，如果一個包含查詢字串的物件請求 `parameter1=a&parameter2=b` 和其他包含 `parameter2=b&parameter1=a` 的相同物件請求，則 CloudFront 同時轉送請求到原始伺服器，並分別快取對應的物件，即使他們是相同的。如果您一律使用相同的參數順序，則 CloudFront 轉送較少的請求到您的原始伺服器。

如需更多詳細資訊，請參閱 [根據查詢字串參數快取內容](#)。如果您要檢視 CloudFront 轉送至原始伺服器的查詢字串，請參閱 CloudFront 日誌檔案的 `cs-uri-query` 欄中的值。如需更多詳細資訊，請參閱 [存取日誌（標準日誌）](#)。

根據 Cookie 值快取

如果設定 CloudFront 來根據 Cookie 值快取，如果您執行下列操作便可以提升快取：

- 設定 CloudFront 只轉送指定的 Cookie，而不是轉送所有 Cookie。對於您設定 CloudFront 轉送到原始伺服器的 Cookie，CloudFront 會轉送 Cookie 名稱和值的每個組合。然後它分別快取原始伺服器傳回的物件，即使它們都是相同的。

例如，假設每個檢視器請求包含兩個 Cookie，每個 Cookie 有三種可能的值，且所有 Cookie 值的組合都有可能。CloudFront 轉送最多九個不同的請求到每一個物件的原始伺服器。如果原始伺服器僅根據其中一個 Cookie 傳回物件的不同版本，則 CloudFront 會轉送多過於需求的請求到原始伺服器，且不必要快取多個相同的物件版本。

- 為靜態和動態內容建立個別的快取行為，並設定 CloudFront 只需為動態內容轉送 Cookie 到原始伺服器。

例如，假設您的分佈只有一個快取行為，且您正使用這個適用於動態內容的分佈，例如 `.js` 檔案，以及很少變更的 `.css` 檔案。CloudFront 根據 Cookie 值快取個別 `.css` 檔案的版本，因此 CloudFront 節點為每個新的 Cookie 值或 Cookie 值的組合轉送請求到原始伺服器。

如果您建立快取行為的路徑模式是 `*.css` 以及 CloudFront 不根據 Cookie 值快取，則 CloudFront 會轉送 `.css` 檔案的請求到原始伺服器時，僅針對節點收到指定 `.css` 檔案的第一個請求以及在 `.css` 檔案過期後的第一個請求。

- 可能的話，每個使用者的 Cookie 值是唯一的 (例如使用者 ID) 時，為動態內容建立個別的快取行為，且動態內容根據較少數量的唯一值而有所不同。

如需更多詳細資訊，請參閱 [根據 Cookie 快取內容](#)。如果您要檢視 CloudFront 轉送至原始伺服器的 Cookie，請參閱 CloudFront 日誌檔案的 `cs(Cookie)` 欄中的值。如需更多詳細資訊，請參閱 [存取日誌（標準日誌）](#)。

根據請求標頭快取

如果設定 CloudFront 來根據請求標頭快取，如果您執行下列操作便可以提升快取：

- 設定 CloudFront 根據指定的標頭轉送和快取，而不是根據所有標頭轉送和快取。對於您指定的標頭，CloudFront 會轉送標頭名稱和值的每個組合。然後它分別快取原始伺服器傳回的物件，即使它們都是相同的。

Note

CloudFront 一律轉送到原始伺服器，在下列主題指定的標頭：

- CloudFront 如何處理和轉送請求至您的 Amazon S3 原始伺服器 > [CloudFront 移除或更新的 HTTP 請求標頭](#)
- CloudFront 如何處理和轉送請求至您的自訂原始伺服器 > [HTTP 請求標頭和 CloudFront 行為 \(自訂和 Amazon S3 原始伺服器\)](#)

當您設定 CloudFront 根據請求標頭快取時，不要變更 CloudFront 轉送的標頭，只有 CloudFront 是否根據標頭值來快取物件。

- 請嘗試避免根據有大量唯一值的請求標頭來快取。

例如，如果您想要根據使用者裝置提供不同大小的影像，則不要設定 CloudFront 根據有大量可能值的 User-Agent 標頭來快取。反之，設定 CloudFront 根據 CloudFront 裝置型標頭 CloudFront-Is-Desktop-Viewer、CloudFront-Is-Mobile-Viewer、CloudFront-Is-SmartTV-Viewer 及 CloudFront-Is-Tablet-Viewer 來進行快取。此外，如果您是傳回適用於平板電腦和桌面的影像相同版本，則僅轉送 CloudFront-Is-Tablet-Viewer 標頭，而非 CloudFront-Is-Desktop-Viewer 標頭。

如需更多詳細資訊，請參閱 [根據請求標頭快取內容](#)。

不需要壓縮時，移除 **Accept-Encoding** 標頭

如果因為原始伺服器不支援、CloudFront 不支援或內容無法壓縮而未啟用壓縮，您可以將分佈中的快取行為與設定 Custom Origin Header 如下的原始檔關聯，以增加快取命中率：

- Header name (標頭名稱)：Accept-Encoding
- Header value (標頭值)：(保留空白)

當您使用此組態時，CloudFront 會從快取金鑰移除 Accept-Encoding 標頭，並且不會在原始伺服器請求中包含標頭。此組態會套用到 CloudFront 提供的所有內容以及來自該原始伺服器的分佈。

透過 HTTP 提供媒體內容

如需如何最佳化隨需視訊 (VOD) 及串流視訊內容的相關資訊，請參閱 [使用 CloudFront 提供隨需視訊和即時串流視訊](#)。

使用 Amazon CloudFront Origin Shield

CloudFront Origin Shield 是 CloudFront 快取基礎設施中的額外層，可協助您將原始伺服器的負載降至最低、改善其可用性，並降低營運成本。CloudFront Origin Shield 為您提供下列優勢：

緩衝區快取命中率

Origin Shield 在您的原始伺服器前端多加了一層快取，所以可協助提高您 CloudFront 發佈的快取命中率。當您使用 Origin Shield 時，從所有 CloudFront 快取層向原始伺服器提出的所有請求都會經過 Origin Shield，從而增加快取命中的可能性。CloudFront 可以從 Origin Shield 到您的原始伺服器，擷取每個只有單一原始伺服器請求的物件，而 CloudFront 快取的所有其他層 (節點和 [區域節點快取](#)) 都可以從 Origin Shield 中擷取物件。

減少原始伺服器負載

Origin Shield 可以進一步減少針對相同物件傳送到原始伺服器的 [同時請求](#) 數量。對於不在 Origin Shield 快取中的內容請求會與相同物件的其他請求整合，因此只需要一個請求，即可傳送到您的原始伺服器。在您的原始伺服器處理較少的請求，可以在尖峰負載或意外流量尖峰期間保留原始伺服器的可用性，並且可以降低即時封裝、映像轉換和資料傳出 (DTO) 等項目的成本。

更好的網路效能

當您在 [對原始伺服器延遲最低](#) 的 AWS 區域中啟用 Origin Shield 時，可以獲得更好的網路效能。至於 AWS 區域中的原始伺服器，CloudFront 網路流量會留在高輸送量的 CloudFront 網路上，一路到您的原始伺服器。對於外部的原始伺服器 AWS，CloudFront 網路流量會一直保留在 CloudFront 網路上，直到 Origin Shield 與您的原始伺服器具有低延遲連線。

使用 Origin Shield 會產生額外費用。如需詳細資訊，請參閱 [CloudFront 定價](#)。

 Note

gRPC 請求不支援 Origin Shield。如果支援 gRPC 的分佈已啟用 Origin Shield，gRPC 請求將繼續運作。不過請求將直接代理到 gRPC 原始伺服器，不會經過 Origin Shield。如需詳細資訊，請參閱[使用 gRPC 搭配 CloudFront 分佈](#)。

主題

- [Origin Shield 的使用案例](#)
- [選擇 Origin Shield AWS 的區域](#)
- [啟用 Origin Shield](#)
- [預估 Origin Shield 成本](#)
- [Origin Shield 高可用性](#)
- [Origin Shield 與其他 CloudFront 功能如何互動](#)

Origin Shield 的使用案例

CloudFront Origin Shield 對許多使用案例都很有幫助，包括：

- 分佈在不同地理區域的檢視器
- 為即時串流或即時映像處理提供即時封裝的原始伺服器
- 具有容量或頻寬限制的內部部署原始伺服器
- 使用多個內容交付網路 (CDN) 的工作負載

Origin Shield 在其他情況下可能不太適合，例如代理至原始伺服器的動態內容、快取性低的內容，或很少請求的內容。

以下各節說明 Origin Shield 對於下列使用案例的好處。

使用案例

- [不同地理區域的檢視器](#)
- [多個 CDN](#)

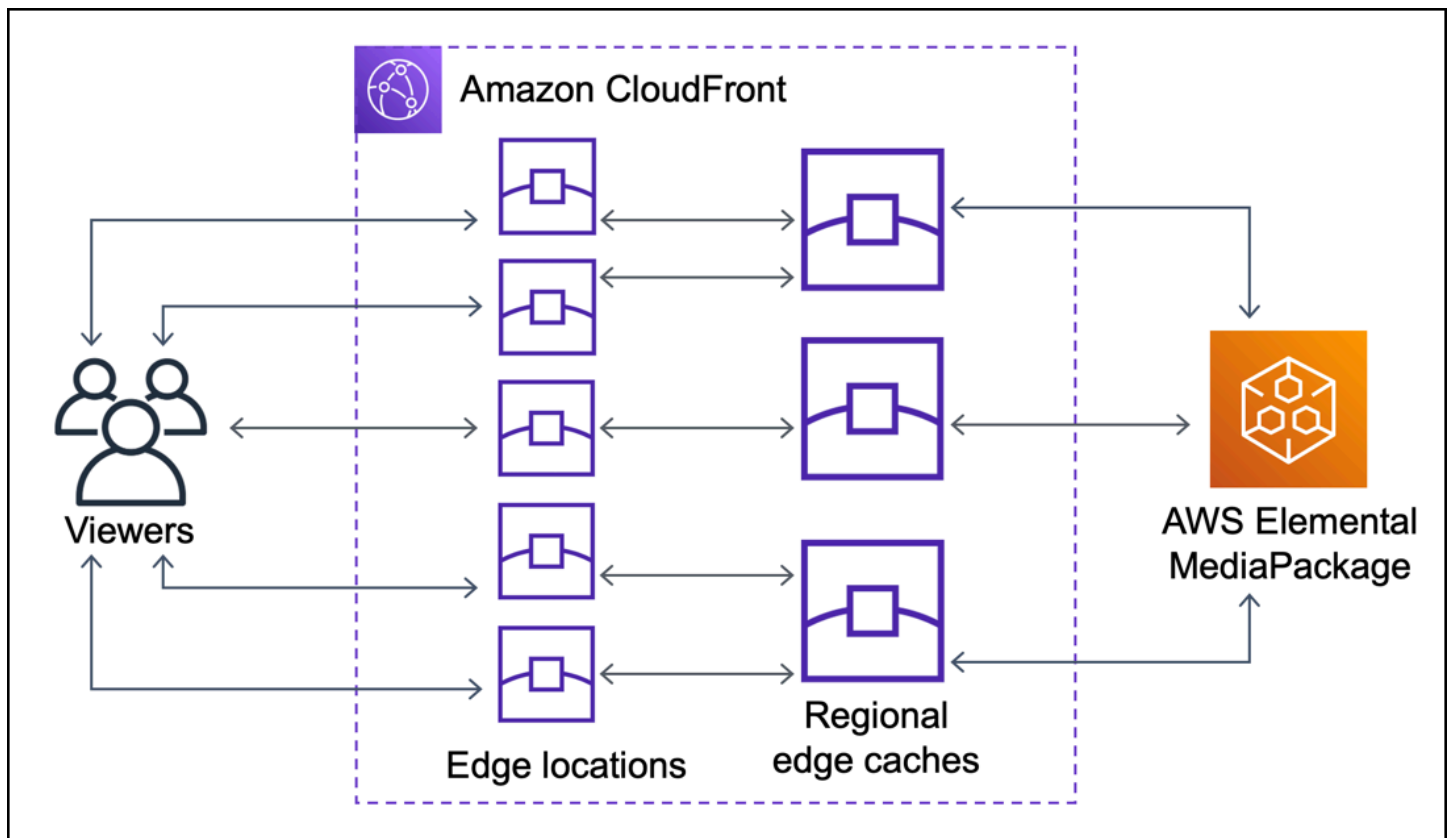
不同地理區域的檢視器

使用 Amazon CloudFront，本質上會減少原始伺服器的負載，因為 CloudFront 可從快取提供的請求，不會前往原始伺服器。除了 CloudFront 的[節點全域網路](#)以外，[區域節點快取](#)也可做為提供快取命中的中層快取層，並整合附近地理區域之檢視器的原始伺服器請求。檢視器請求會先路由到附近的 CloudFront 節點，如果未在該位置快取物件，則將請求傳送至區域節點快取。

當檢視器位於不同的地理區域時，可以透過不同的區域邊緣快取路由請求，每個請求都可以將相同內容的請求傳送至您的原始伺服器。但是，有了 Origin Shield，您可以在區域邊緣快取和您的原始伺服器之間獲得額外的一層快取。來自所有區域邊緣快取的所有請求都會經過 Origin Shield，進一步減少原始伺服器的負載。下圖說明此概念。在下圖中，原始伺服器是 AWS Elemental MediaPackage。

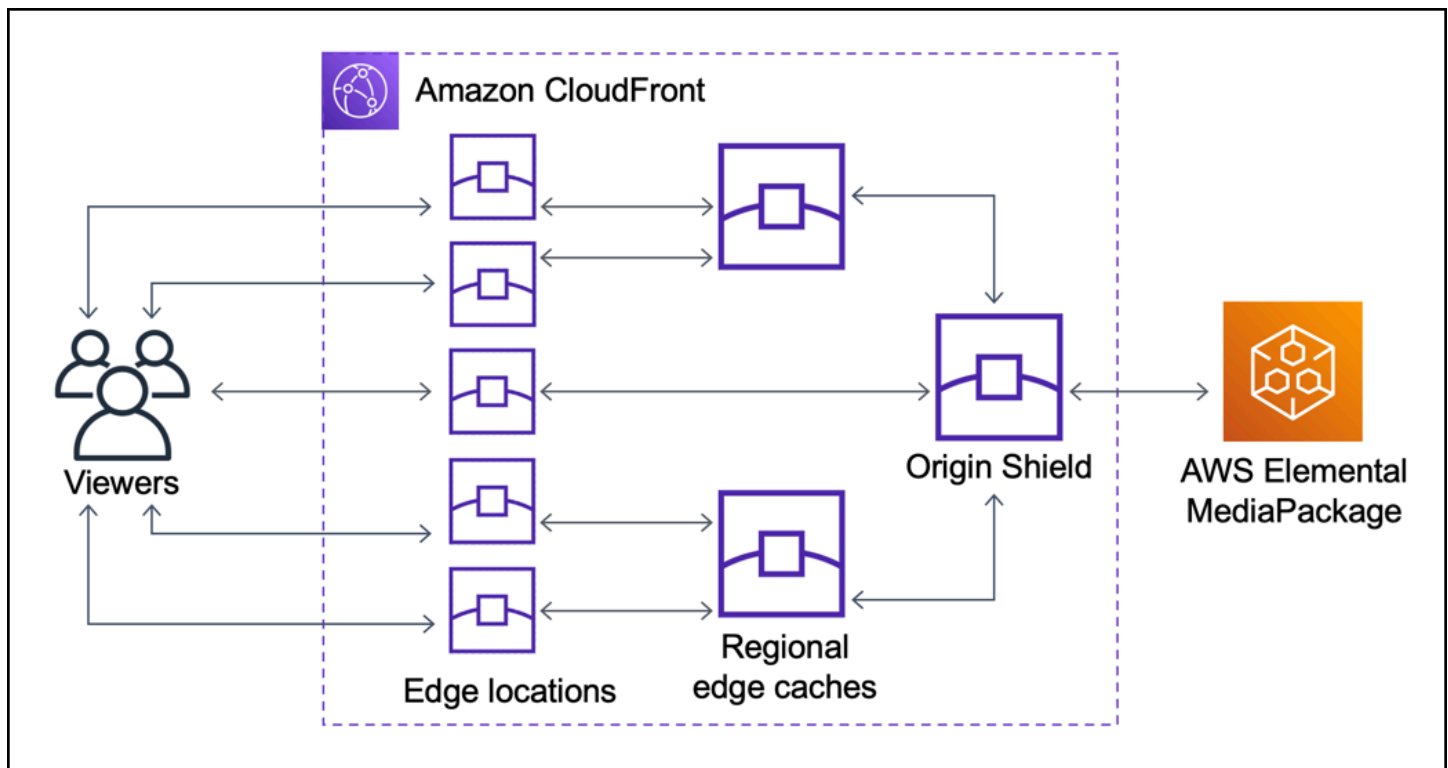
沒有 Origin Shield

如果沒有 Origin Shield，您的原始伺服器可能會收到相同內容的重複請求，如下圖所示。



具備 Origin Shield

使用 Origin Shield 有助於減少原始伺服器的負載，如下圖所示。



多個 CDN

若要提供即時視訊活動或熱門隨需內容，您可以使用多個內容交付網路 (CDN)。使用多個 CDN 可以提供某些好處，但這也表示您的原始伺服器可能會收到許多相同內容的重複請求，每個請求都來自不同的 CDN 或相同 CDN 中的不同位置。這些冗餘請求可能會對原始伺服器的可用性造成不利影響，或對於即時封裝或資料傳出 (DTO) 到網際網路等程序，造成額外的營運成本。

使用 Origin Shield 並將 CloudFront 發佈做為其他 CDN 的原始伺服器使用，您可以獲得下列好處：

- 減少原始伺服器收到的冗餘請求，這有助於降低使用多個 CDN 的負面影響。
- 跨 CDN 的常見[快取金鑰](#)，以及集中式管理原始伺服器面相的功能。
- 改善的網路效能。來自其他 CDN 的網路流量會終止於附近的 CloudFront 節點，這可能會提供來自本機快取的命中。如果請求的物件不在節點快取中，原始伺服器的請求會維持在 CloudFront 網路上一路到 Origin Shield，這樣可為原始伺服器提供高輸送量和低延遲。如果請求的物件位於 Origin Shield 的快取中，可完全避免原始伺服器的請求。

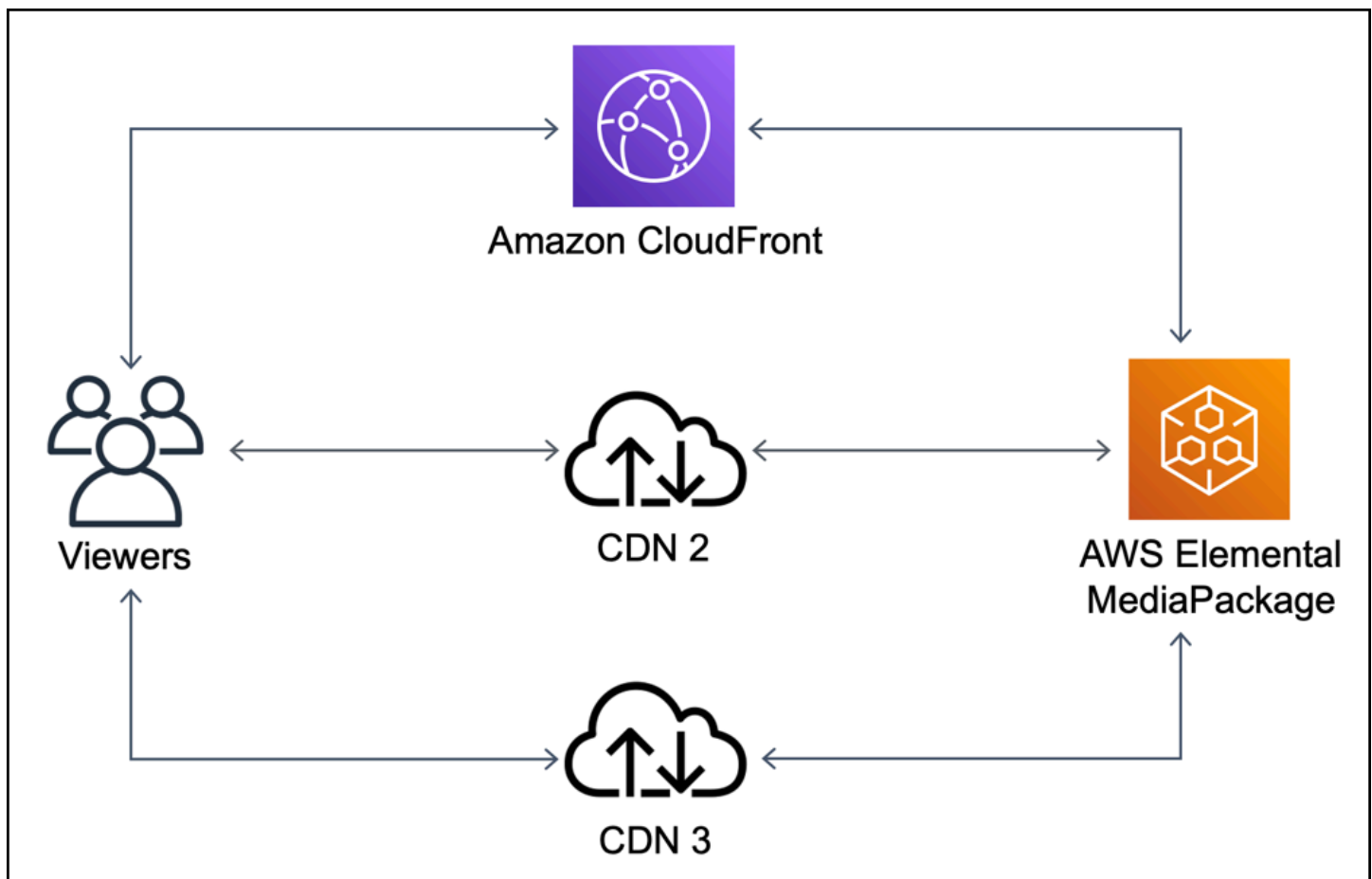
⚠ Important

如果您有興趣在多 CDN 架構中使用 Origin Shield，並享有折扣定價，[請聯絡我們](#)或您的 AWS 銷售代表以取得詳細資訊。可能會收取額外費用。

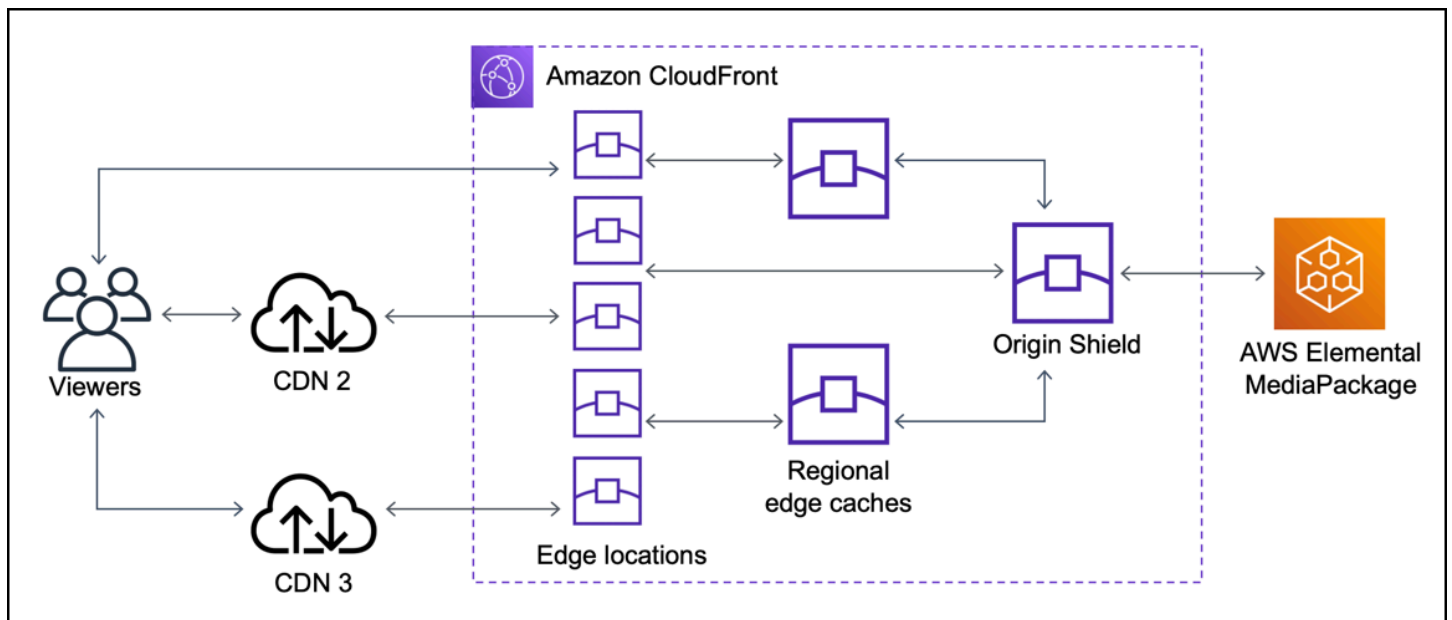
下圖顯示當您使用多個 CDN 提供熱門即時視訊活動時，此組態如何幫助將原始伺服器的負載降至最低。在下圖中，原始伺服器為 AWS Elemental MediaPackage。

沒有 Origin Shield (多個 CDN)

沒有 Origin Shield，您的原始伺服器可能會收到相同內容的重複請求，每個請求都來自不同的 CDN，如下圖所示。

**具備 Origin Shield (多個 CDN)**

使用 Origin Shield 並將 CloudFront 做為其他 CDN 的原始伺服器使用，有助於減少原始伺服器的負載，如下圖所示。



選擇 Origin Shield AWS 的區域

Amazon CloudFront 在 CloudFront 具有 [區域節點快取](#) 的 AWS 區域中提供 Origin Shield。當您啟用 Origin Shield 時，您可以選擇 Origin Shield AWS 的區域。您應該為原始伺服器選擇最低延遲的 AWS 區域。您可以將 Origin Shield 與 AWS 區域中的原始伺服器以及不在其中的原始伺服器搭配使用 AWS。

對於 AWS 區域中的原始伺服器

如果您的原始伺服器位於 AWS 區域，請先判斷您的原始伺服器是否位於 CloudFront 提供 Origin Shield 的區域。CloudFront 在以下 AWS 區域中提供 Origin Shield。

- 美國東部 (俄亥俄) – us-east-2
- 美國東部 (維吉尼亞北部) – us-east-1
- 美國西部 (奧勒岡) – us-west-2
- 亞太地區 (孟買) – ap-south-1
- 亞太地區 (首爾) – ap-northeast-2
- 亞太地區 (新加坡) – ap-southeast-1
- 亞太地區 (雪梨) – ap-southeast-2
- 亞太地區 (東京) – ap-northeast-1
- 歐洲 (法蘭克福) – eu-central-1

- 歐洲 (愛爾蘭) – eu-west-1
- 歐洲 (倫敦) – eu-west-2
- 南美洲 (聖保羅) – sa-east-1
- 中東 (阿拉伯聯合大公國) – me-central-1

如果您的原始伺服器位於 CloudFront 提供 Origin Shield AWS 的區域

如果您的原始伺服器位於 CloudFront 提供 Origin Shield AWS 的區域（請參閱上述清單），請在與您的原始伺服器相同的區域中啟用 Origin Shield。

如果您的原始伺服器不在 CloudFront 提供 Origin Shield AWS 的區域

如果您的原始伺服器不在 CloudFront 提供 Origin Shield 的 AWS 區域中，請參閱下表以決定要在哪個區域中啟用 Origin Shield。

如果您的原始伺服器在...	Origin Shield 啟用於...
美國西部 (加利佛尼亞北部) – us-west-1	美國西部 (奧勒岡) – us-west-2
非洲 (開普敦) – af-south-1	歐洲 (愛爾蘭) – eu-west-1
亞太區域 (香港) – ap-east-1	亞太區域 (新加坡) – ap-southeast-1
加拿大 (中部) – ca-central-1	美國東部 (維吉尼亞北部) – us-east-1
歐洲 (米蘭) – eu-south-1	歐洲 (法蘭克福) – eu-central-1
歐洲 (巴黎) – eu-west-3	歐洲 (倫敦) – eu-west-2
歐洲 (斯德哥爾摩) – eu-north-1	歐洲 (倫敦) – eu-west-2
中東 (巴林) – me-south-1	亞太區域 (孟買) – ap-south-1

對於 外部的原始伺服器 AWS

您可以將 Origin Shield 與內部部署的原始伺服器或不在 AWS 區域中的原始伺服器搭配使用。在此情況下，請在對您的原始伺服器延遲最低的 AWS 區域中啟用 Origin Shield。如果您不確定哪個 AWS 區域對您的原始伺服器延遲最低，您可以使用下列建議來協助您做出判斷。

- 您可以參閱上表，根據原始伺服器的地理區域，了解哪個 AWS 區域對您的原始伺服器有最低延遲。
- 您可以在地理位置接近原始伺服器的幾個不同 AWS 區域中啟動 Amazon EC2 執行個體，並使用執行一些測試ping，以測量這些區域與原始伺服器之間的典型網路延遲。

啟用 Origin Shield

您可以啟用 Origin Shield 來改善快取命中率、降低原始伺服器的負載，並協助改善效能。若要啟用 Origin Shield，請變更 CloudFront 發佈中的原始伺服器設定。Origin Shield 是原始伺服器的屬性。對於 CloudFront 發佈中的每部原始伺服器，您可以在為該原始伺服器提供最佳效能的任何 AWS 區域中分別啟用 Origin Shield。

您可以在 CloudFront 主控台中，使用 CloudFormation 或 CloudFront API 啟用 Origin Shield。

Console

若要為現有的原始伺服器啟用 Origin Shield (主控台)

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇包含您要更新之原始伺服器的分佈。
3. 選擇 Origins (原始伺服器) 索引標籤。
4. 選擇要更新的原始伺服器，然後選擇 Edit (編輯)。
5. 對於 Enable Origin Shield (啟用 Origin Shield)，選擇 Yes (是)。
6. 對於 Origin Shield Region (原始伺服器 Shield Region)，選擇您要從中啟用 Shield Region 的 AWS 區域。如需選擇區域的說明，請參閱[選擇 Origin Shield AWS 的區域](#)。
7. 選擇儲存變更。

當您的分佈狀態為 Deployed (已部署)，Origin Shield 便已就緒。這需要幾分鐘的時間。

為新的原始伺服器啟用 Origin Shield (主控台)

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 若要在現有分佈中建立新的原始伺服器，請執行下列動作：
 1. 選擇您要在其中建立原始伺服器的分佈。
 2. 選擇 Create Origin (建立原始伺服器)，然後繼續步驟 3。

若要在新標準分佈中建立新的原始伺服器，請執行下列動作：

1. 依照步驟在主控台中建立標準分佈。如需詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。
2. 在設定區段中選取自訂設定。繼續步驟 3。
3. 對於 Enable Origin Shield (啟用 Origin Shield)，選擇 Yes (是)。
4. 對於 Origin Shield Region (原始伺服器 Shield Region)，選擇您要從中啟用 Shield Region 的 AWS 區域。如需選擇區域的說明，請參閱[選擇 Origin Shield AWS 的區域](#)。
5. 請依照主控台其中的步驟完成建立原始伺服器或分佈。

當您的分佈狀態為 Deployed (已部署)，Origin Shield 便已就緒。這需要幾分鐘的時間。

CloudFormation

若要使用 啟用 Origin Shield CloudFormation，請在 `AWS::CloudFront::Distribution` 資源的 `OriginShield` 屬性類型中使用 `Origin` 屬性。您可以將 `OriginShield` 屬性新增至現有的 `Origin`，或在建立新的 `Origin` 時，加入該屬性。

下列範例顯示 YAML 格式的語法，適用於在美國西部 (奧勒岡) 區域 (OriginShield) 中啟用 `us-west-2`。如需選擇區域的說明，請參閱[the section called “選擇 Origin Shield AWS 的區域”](#)。此範例僅顯示 `Origin` 屬性類型，而不是整個 `AWS::CloudFront::Distribution` 資源。

```
Origins:
- DomainName: 3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com
  Id: Example-EMP-3ae97e9482b0d011
  OriginShield:
    Enabled: true
    OriginShieldRegion: us-west-2
  CustomOriginConfig:
    OriginProtocolPolicy: match-viewer
    OriginSSLProtocols: TLSv1
```

如需詳細資訊，請參閱 AWS CloudFormation 使用者指南中「資源和屬性參考」一節的[AWS::CloudFront::Distribution Origin](#)。

API

若要使用 AWS SDKs 或 AWS Command Line Interface () 搭配 CloudFront API 啟用 Origin Shield AWS CLI，請使用 `OriginShield` 類型。在 `OriginShield` 中，指定 `Origin` 中的

DistributionConfig,。如需 OriginShield 類型的相關資訊,請參閱「Amazon CloudFront API 參考」中的下列資訊。

- [OriginShield](#) (類型)
- [Origin](#) (類型)
- [DistributionConfig](#) (類型)
- [UpdateDistribution](#) (操作)
- [CreateDistribution](#) (操作)

使用這些類型和操作的特定語法會根據軟體開發套件、CLI 或 API 用戶端而有所不同。如需詳細資訊,請參閱軟體開發套件、CLI 或用戶端的參考文件。

預估 Origin Shield 成本

您可以根據進入 Origin Shield 做為增量層的請求數量來累算 Origin Shield 的費用。

對於代理至原始伺服器的動態 (非可快取) 請求,Origin Shield 一律為增量層。動態請求使用 PUT、POST、PATCH 和 DELETE 等 HTTP 方法。

若 GET 和 HEAD 請求的存留時間 (TTL) 設定少於 3600 秒,則視為動態請求。此外已停用快取的 GET 和 HEAD 請求也會視為動態請求。

若要針對動態請求預估 Origin Shield 的費用,請使用下列公式:

動態請求總數 x 每 10,000 個請求的 Origin Shield 費用 / 10,000

對於使用 HTTP 方法 GET、HEAD 和 OPTIONS 的非動態請求,Origin Shield 有時是增量層。啟用 Origin Shield 時,您可以選擇 Origin Shield AWS 區域的。若請求是自然移至與 Origin Shield 相同區域中的[區域邊緣快取](#),則 Origin Shield 不是增量層。您不會對這些請求累算 Origin Shield 費用。若請求移至與 Origin Shield 不同區域的區域邊緣快取,然後移至 Origin Shield,則 Origin Shield 是增量層。您會對這些請求累算 Origin Shield 費用。

若要針對可快取請求預估 Origin Shield 的費用,請使用下列公式:

可快取請求的總數 x (1 – 快取命中率) x 不同區域中從區域節點快取移至 Origin Shield 的請求百分比 x 每 10,000 個請求的 Origin Shield 費用 / 10,000

如需每 10,000 個請求的詳細 Origin Shield 費用,請參閱[CloudFront 定價](#)。

Origin Shield 高可用性

Origin Shield 利用 CloudFront 的[區域邊緣快取](#)功能。每個邊緣快取都是使用至少三個[可用區域](#)與自動擴展 Amazon EC2 執行個體機群建置在 AWS 區域中。CloudFront 位置與 Origin Shield 的連線也會針對每項請求使用有效的錯誤追蹤，以便在主要的 Origin Shield 位置無法使用時，自動將請求路由到輔助的 Origin Shield 位置。

Origin Shield 與其他 CloudFront 功能如何互動

下列各節會說明 Origin Shield 與其他 CloudFront 功能如何互動。

Origin Shield 與 CloudFront 記錄日誌

若要查看 Origin Shield 何時處理請求，您必須啟用下列其中一項：

- [CloudFront 標準日誌 \(存取日誌\)](#)。標準日誌是免費提供的。
- [CloudFront 即時存取日誌](#)。使用即時存取日誌會產生額外費用。請參閱 [Amazon CloudFront 定價](#)。

來自 Origin Shield 的快取命中在 CloudFront 日誌的 `OriginShieldHit` 欄位中顯示為 `x-edge-detailed-result-type`。Origin Shield 利用 Amazon CloudFront 的[區域節點快取](#)。如果請求是從 CloudFront 節點路由到做為 Origin Shield 的區域節點快取，日誌會將其回報為 `Hit`，而不是回報為 `OriginShieldHit`。

Origin Shield 和原始伺服器群組

Origin Shield 與 [CloudFront 原始伺服器群組](#)相容。由於 Origin Shield 是原始伺服器的屬性，因此，即使原始伺服器是原始伺服器群組的一部分，請求一律會針對每個原始伺服器透過 Origin Shield 來傳輸。對於指定的請求，CloudFront 會透過主要原始伺服器的 Origin Shield，將請求路由至原始伺服器群組中的主要原始伺服器。如果該請求失敗 (根據原始伺服器群組容錯移轉條件)，CloudFront 會透過輔助原始伺服器的 Origin Shield，將請求路由至輔助原始伺服器。

Origin Shield 和 Lambda@Edge

Origin Shield 不會影響 [Lambda@Edge](#) 函數的功能，但它會影響這些函數執行所在的 AWS 區域。

當您搭配 Lambda@Edge 使用 Origin Shield 時，[原始伺服器面向觸發程序](#) (原始伺服器請求和原始伺服器回應) 會在啟用 Origin Shield 的 AWS 區域中執行。如果主要 Origin Shield 位置無法使用，且 CloudFront 將請求路由到次要 Origin Shield 位置，則 Lambda@Edge 面向原始伺服器的觸發條件也會切換為使用次要 Origin Shield 位置。

檢視器面向觸發程序不會受到影響。

透過 CloudFront 原始伺服器容錯移轉最佳化高可用性

對於需要高可用性的案例，您可以將 CloudFront 設定為具有原始伺服器容錯移轉。如要開始使用，您可以使用兩個原始伺服器建立原始伺服器群組：主要及次要。如果主要原始伺服器無法使用，或是傳回指出故障的特定 HTTP 回應狀態碼，CloudFront 會自動切換至次要原始伺服器。

若要設定原始伺服器容錯移轉，您必須至少分佈兩部原始伺服器。接著，您可以為您的分佈建立原始伺服器群組，其中包含兩個原始伺服器，並將其中一個設為主要原始伺服器。最後，您可以建立或更新快取行為，以使用原始伺服器群組。

如要查看設定原始伺服器群組和設定特定原始伺服器容錯移轉選項的步驟，請參閱 [建立原始伺服器群組](#)。

在設定快取行為的原始伺服器容錯移轉之後，CloudFront 會針對檢視者請求執行下列操作：

- 發生快取命中時，CloudFront 會傳回請求的物件。
- 快取未命中時，CloudFront 會將請求路由至原始伺服器群組中的主要原始伺服器。
- 當主要原始伺服器傳回不是為容錯移轉設定的狀態碼 (例如 HTTP 2xx 或 3xx 狀態碼) 時，CloudFront 便會將請求的物件提供給檢視者。
- 發生下列任何一種情況時：
 - 主要原始伺服器會傳回您為容錯移轉設定的 HTTP 狀態碼
 - CloudFront 無法連線至主要原始伺服器 (當 503 設定為容錯移轉代碼時)
 - 來自主要原始伺服器的回應需要太長的時間 (逾時) (當 504 設定為容錯移轉代碼時)

接著，CloudFront 會將請求路由至原始伺服器群組中的輔助原始伺服器。

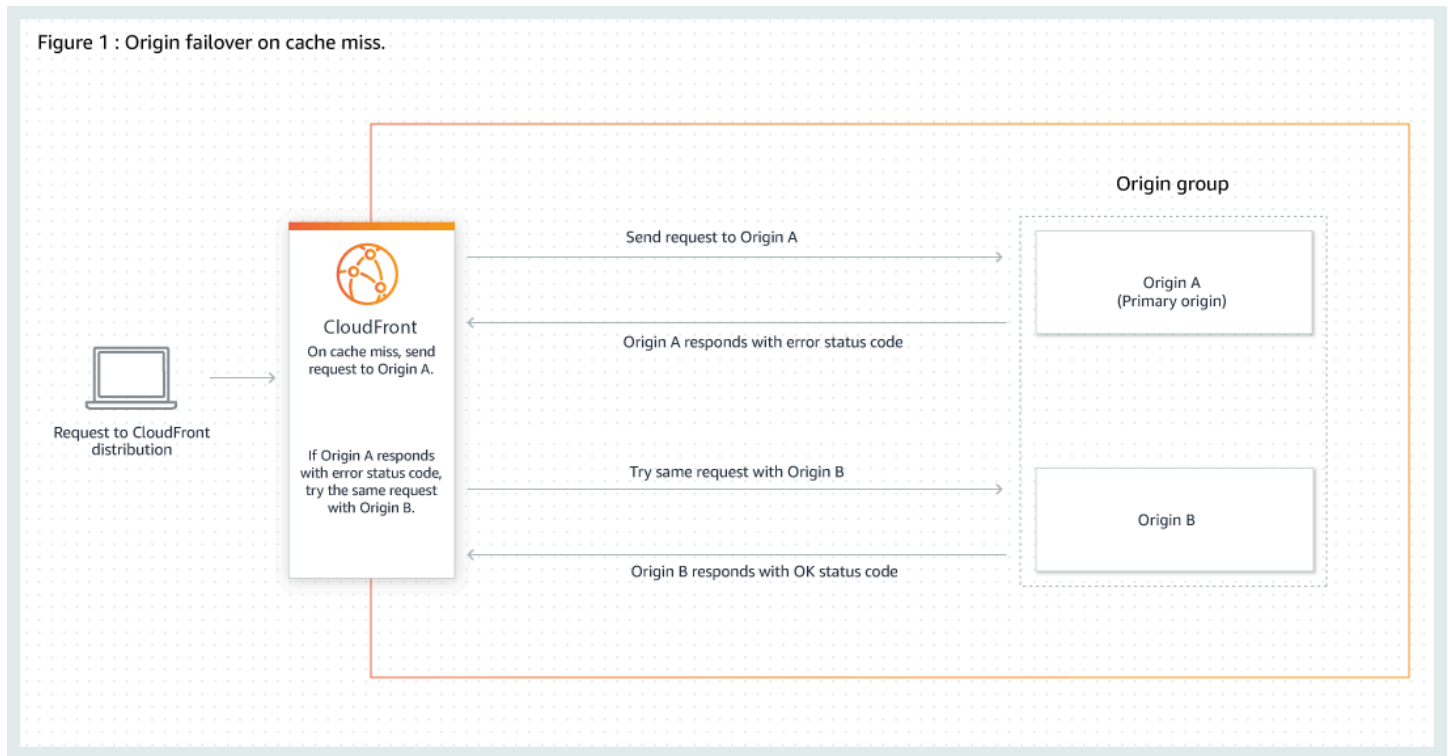
Note

對於某些使用案例 (例如串流視訊內容)，您可能希望 CloudFront 快速容錯移轉至次要原始伺服器。若要調整 CloudFront 容錯移轉至次要原始伺服器的速度，請參閱[控制原始伺服器逾時和嘗試次數](#)。

CloudFront 會將所有傳入要求路由到主要原始伺服器，即使先前的要求容錯移轉到次要原始伺服器也是如此。CloudFront 只會在向主要原始伺服器傳送請求失敗後，才會向次要原始伺服器傳送請求。

只有當檢視器請求的 HTTP 方法為 GET、HEAD 或 OPTIONS 時，CloudFront 才會容錯移轉至次要原始伺服器。當檢視器傳送不同的 HTTP 方法 (例如，POST、PUT 等等) 時，CloudFront 不會容錯移轉。

下圖說明原始伺服器容錯移轉的運作方式。



主題

- [建立原始伺服器群組](#)
- [控制原始伺服器逾時和嘗試次數](#)
- [使用原始伺服器容錯移轉與 Lambda@Edge 函數搭配](#)
- [搭配原始伺服器容錯移轉使用自訂錯誤頁面](#)

建立原始伺服器群組

建立來源群組

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 選擇您要為原始伺服器群組建立的分佈。
3. 選擇 Origins (原始伺服器) 索引標籤。
4. 請確定分佈有多個原始伺服器。如果沒有多個原始伺服器，請新增第二個原始伺服器。

5. 在 Origins (原始伺服器) 索引標籤上的 Origin groups (原始伺服器群組) 窗格中，選擇 Create Origin Group (建立原始伺服器群組)。
6. 選擇原始伺服器群組的原始伺服器。新增原始伺服器後，使用箭號來設定優先順序，即哪一個原始伺服器為主要原始伺服器，哪一個為輔助。
7. 輸入原始伺服器群組的名稱。
8. 選擇要用來做為容錯移轉條件的 HTTP 代碼。您可以選擇以下狀態碼的任意組合：400、403、404、416、500、502、503 或 504。當 CloudFront 收到的請求中包含其中一個您指定的狀態碼時，便會容錯移轉至輔助原始伺服器。

 Note

只有當檢視器請求的 HTTP 方法為 GET、HEAD 或 OPTIONS 時，CloudFront 才會容錯移轉至次要原始伺服器。當檢視器傳送不同的 HTTP 方法 (例如，POST、PUT 等等) 時，CloudFront 不會容錯移轉。

9. 在原始伺服器選取條件下，指定當您的分佈路由檢視器請求時，如何選取原始伺服器。您可以選擇下列選項。

預設

CloudFront 將使用您在設定頁面上指定的預設原始伺服器優先順序。

媒體品質分數

CloudFront 會追蹤並使用此分數來決定轉送請求的第一個原始伺服器。這也會授權 CloudFront 向原始伺服器群組中的替代原始伺服器提出非同步 HEAD 請求，以判斷其媒體品質分數。您只能為 AWS Elemental MediaPackage v2 原始伺服器選擇此選項。如需更多詳細資訊，請參閱 [媒體品質感知彈性](#)。

10. 選擇 Create origin group (建立原始伺服器群組)。

請務必將原始伺服器群組指派為分佈的快取行為的原始伺服器。如需更多詳細資訊，請參閱 [Name](#)。

控制原始伺服器逾時和嘗試次數

依預設，CloudFront 會嘗試連接至原始群組中的主要原始伺服器，長達 30 秒 (嘗試 3 次連線，每次 10 秒)，然後再容錯移轉至次要原始伺服器。對於某些使用案例 (例如串流視訊內容)，您可能希望 CloudFront 更快速容錯移轉至次要原始伺服器。您可以調整以下設定，影響 CloudFront 容錯移轉至次

要原始伺服器的速度。如果原始伺服器是次要原始伺服器或不屬於原始群組的原始伺服器，這些設定會影響 CloudFront 回應將 HTTP 504 傳回瀏覽者的速度。

若要更快地容錯移轉，請指定較短的連線逾時、較少的連線嘗試次數或兩者。對於自訂原始伺服器 (包括使用靜態網站託管所設定的 Amazon S3 儲存貯體)，您也可以調整原始伺服器回應逾時。

原始伺服器連線逾時

原始伺服器連線逾時設定會影響當嘗試建立與原始伺服器的連線時 CloudFront 的等待時間長短。依預設，CloudFront 會等待 10 秒來建立連線，但您可以指定 1-10 秒 (含)。如需更多詳細資訊，請參閱 [連線逾時](#)。

原始伺服器連線嘗試次數

原始伺服器連線嘗試次數設定會影響 CloudFront 嘗試連線至原始伺服器的次數。依預設，CloudFront 會嘗試連線 3 次，但您可以指定 1-3 (含)。如需更多詳細資訊，請參閱 [連線嘗試](#)。

對於自訂原始伺服器 (包括使用靜態網站託管所設定的 Amazon S3 儲存貯體)，此設定也會影響在原始伺服器回應逾時時，CloudFront 嘗試從原始伺服器取得回應的次數。

原始伺服器回應逾時

原始伺服器回應逾時 (也稱原始伺服器讀取逾時) 會影響 CloudFront 從原始伺服器接收回應 (或接收完整回應) 的等待時間長短。依預設，CloudFront 會等待 30 秒，但您可以指定 1-120 秒 (含)。如需更多詳細資訊，請參閱 [回應逾時](#)。

如何變更這些設定

在 [CloudFront 主控台](#) 中變更這些設定

- 對於新原始伺服器或新的分佈，您可以在建立資源時指定這些數值。
- 對於現有分佈中的現有原始伺服器，您可以在編輯原始伺服器時指定這些數值。

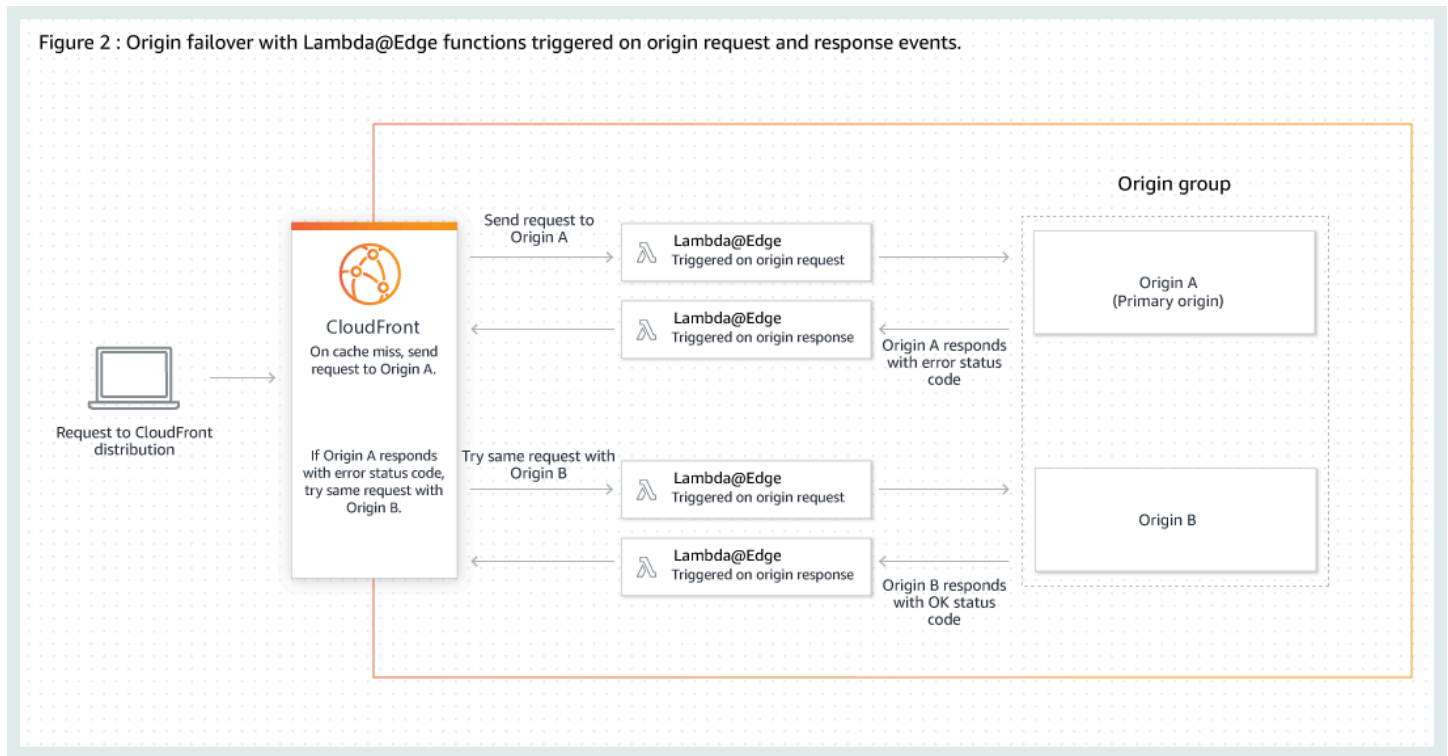
如需更多詳細資訊，請參閱 [所有分佈設定參考](#)。

使用原始伺服器容錯移轉與 Lambda@Edge 函數搭配

您可以使用 Lambda@Edge 函數，與您已利用原始伺服器群組設定的 CloudFront 分佈搭配。如要使用 Lambda 函數，請在建立快取行為時，於原始伺服器群組的 [原始伺服器請求或原始伺服器回應觸發條件](#) 中進行指定。當您搭配原始伺服器群組使用 Lambda@Edge 函數時，可能會針對單一瀏覽者請求觸發函數兩次。例如，考量以下情境：

1. 您可以使用原始伺服器請求觸發條件建立 Lambda@Edge 函數。
2. Lambda 函數會在 CloudFront 傳送請求至主要原始伺服器 (快取未中) 時觸發一次。
3. 主要原始伺服器會回應為容錯移轉設定的 HTTP 狀態碼。
4. 當 CloudFront 向次要原始伺服器傳送相同的請求時，會再次觸發 Lambda 函數。

下圖說明當您在原始伺服器請求或回應觸發中包括 Lambda@Edge 函數時，原始伺服器容錯移轉的運作方式。



如需使用 Lambda@Edge 觸發的詳細資訊，請參閱[the section called “為 Lambda@Edge 函數新增觸發條件”](#)。

如需有關管理 DNS 備援的詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的[設定 DNS 備援](#)。

搭配原始伺服器容錯移轉使用自訂錯誤頁面

您可以使用自訂錯誤頁面與原始伺服器群組搭配，其方式與您將它們與未針對原始伺服器容錯移轉設定之原始伺服器搭配使用的方式類似。

當您使用原始伺服器容錯移轉時，您可以設定 CloudFront 針對主要或輔助原始伺服器 (或兩者) 傳回自訂錯誤頁面：

- 針對主要原始伺服器傳回自訂錯誤頁面 – 如果主要原始伺服器傳回未針對容錯移轉設定的 HTTP 狀態碼，CloudFront 便會將自訂錯誤頁面傳回瀏覽者。
- 針對輔助原始伺服器傳回自訂錯誤頁面 – 如果 CloudFront 從輔助原始伺服器收到故障狀態碼，CloudFront 便會傳回自訂錯誤頁面。

如需使用自訂錯誤頁面與 CloudFront 搭配的詳細資訊，請參閱 [產生自訂錯誤回應](#)。

管理內容保持在快取中達多久時間 (過期)

您可以控制在 CloudFront 將另一個請求轉送至原始伺服器之前，檔案在 CloudFront 快取中停留的時間。降低持續時間允許您提供動態內容。增加持續時間表示您的使用者取得更好的效能，因為檔案更有可能是直接透過節點快取提供。較長的持續時間也能減少的原始伺服器的負載。

一般而言，CloudFront 從節點提供檔案，直到您指定的快取持續時間已經達到，亦即直到檔案過期。過期之後，下次邊緣節點取得檔案的使用者請求時，CloudFront 會將請求轉送到原始伺服器來驗證快取是否包含檔案的最新版本。來自原始伺服器的回應取決於檔案是否已變更：

- 如果 CloudFront 快取已經有最新版本，則原始伺服器傳回狀態碼 304 Not Modified。
- 如果 CloudFront 快取沒有最新的版本，則原始伺服器會傳回一個狀態碼 200 OK 和檔案的最新版本。

如果節點中的物件未被頻繁請求，CloudFront 可能會移出該檔案 (在檔案過期日期前移除檔案)，以將足夠的空間提供給最近被請求的檔案。

我們建議您透過更新分佈的快取政策來管理快取持續時間。如果您選擇不使用快取政策，預設 TTL (生存時間) 為 24 小時，但您可以更新下列設定來覆寫預設值：

- 若要為符合相同路徑模式的所有檔案變更快取持續期間，您可以變更快取行為的 Minimum TTL (最小 TTL)、Maximum TTL (最大 TTL)，以及 Default TTL (預設 TTL) CloudFront 設定。如需個別設定的詳細資訊，請參閱 [最短 TTL](#)、[最長 TTL](#) 和 [預設 TTL](#)。
- 若要變更個別檔案的快取持續期間，您可以設定原始伺服器來使用 max-age 或 s-maxage 指示詞將 Cache-Control 標頭，或是 Expires 標頭欄位新增到檔案。如需更多詳細資訊，請參閱 [使用標頭來控制個別物件的快取持續時間](#)。

如需有關 Minimum TTL (最短 TTL)、Default TTL (預設 TTL) 及 Maximum TTL (最長 TTL) 與 max-age 和 s-maxage 命令及 Expires 標頭欄位互動方式的詳細資訊，請參閱 [the section called “指定 CloudFront 快取物件的時間”](#)。

您也可以控制錯誤 (例如 404 Not Found) 在 CloudFront 快取中保留多久時間，然後 CloudFront 將再試一次藉由轉送另一個請求至原始伺服器以取得請求的物件。如需更多詳細資訊，請參閱 [the section called “CloudFront 如何處理來自原始伺服器的 HTTP 4xx 和 5xx 狀態碼”](#)。

主題

- [使用標頭來控制個別物件的快取持續時間](#)
- [提供過時 \(過期\) 內容](#)
- [指定 CloudFront 快取物件的時間](#)
- [使用 Amazon S3 主控台新增標頭到物件](#)

使用標頭來控制個別物件的快取持續時間

您可以使用 Cache-Control 和 Expires 標頭來控制物件在快取中保持多久的時間。Minimum TTL (最短 TTL)、Default TTL (預設 TTL) 及 Maximum TTL (最長 TTL) 的設定也會影響快取持續時間，但以下是標頭會如何影響快取持續時間的概觀：

- Cache-Control max-age 指令可讓您指定您想要物件在快取中維持多久的時間 (以秒為單位)，然後 CloudFront 從原始伺服器再次取得物件。CloudFront 支援的最短到期時間是 0 秒。最長值為 100 年。請在下列格式中指定值：

Cache-Control: max-age=#

例如，下列指令告訴 CloudFront 在快取中維持關聯的物件 3600 秒 (1 小時)：

Cache-Control: max-age=3600

如果您想要物件在 CloudFront 節點快取中保持的持續時間不同於在瀏覽器快取中保持的，您可以 Cache-Control max-age 和 Cache-Control s-maxage 指令併用。如需更多詳細資訊，請參閱 [指定 CloudFront 快取物件的時間](#)。

- Expires 標頭欄位可讓您使用 [RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1 Section 3.3.1, Full Date](#) 中指定的格式來指定過期日期和時間，例如：

Sat, 27 Jun 2015 23:59:59 GMT

我們建議您使用 Cache-Control max-age 指令，而不是 Expires 標頭欄位來控制物件快取。如果您同時為 Cache-Control max-age 與 Expires 指定值，則 CloudFront 僅使用 Cache-Control max-age 的值。

如需更多詳細資訊，請參閱 [指定 CloudFront 快取物件的時間](#)。

您不能使用 HTTP Cache-Control 或來自檢視者的 Pragma 請求的 GET 標頭欄位，來強制 CloudFront 返回物件的原始伺服器。CloudFront 忽略檢視者請求中的這些標頭欄位。

如需有關 Cache-Control 與 Expires 標頭欄位的詳細資訊，請參閱 RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1 中的以下各節：

- [Section 14.9 Cache Control \(14.9 節快取控制\)](#)
- [Section 14.21 Expires \(14.21 節過期\)](#)

提供過時 (過期) 內容

CloudFront 支援 Stale-While-Revalidate 和 Stale-If-Error 快取控制指令。您可以使用這些指令來指定檢視器可使用過時內容的時間長度。

主題

- [Stale-While-Revalidate](#)
- [Stale-If-Error](#)
- [使用兩種指令](#)

Stale-While-Revalidate

此指令允許 CloudFront 從快取中提供過時的內容，同時 CloudFront 會以非同步方式從原始伺服器擷取新版本。當檢視器立即從邊緣節點接收回應時，這可改善延遲，而不必等待背景擷取。新內容會在背景載入以供未來請求使用。

Example 範例：Stale-While-Revalidate

當您將 Cache-Control 標頭設定為使用這些指令時，CloudFront 會執行下列動作。

```
Cache-Control: max-age=3600, stale-while-revalidate=600
```

1. CloudFront 會將回應快取一小時 (max-age=3600)。
2. 如果請求在此持續時間之後發出，CloudFront 會提供過時內容，同時向原始伺服器傳送請求，以重新驗證和重新整理快取的內容。
3. 內容被重新驗證的同時，CloudFront 可為過時內容最多提供 10 分鐘 (stale-while-revalidate=600)。

 Note

CloudFront 會將過時的內容提供至 `stale-while-revalidate` 指令的值或 CloudFront [最大 TTL](#) 的值，以較小者為準。在最長 TTL 持續時間之後，無論 `stale-while-revalidate` 值為何，都不會從邊緣節點快取提供過時的物件。

Stale-If-Error

在原始伺服器無法存取或傳回 500 至 600 的錯誤代碼時，此指令可讓 CloudFront 從快取中提供過時內容。這可確保檢視者即使在原始伺服器中斷期間也能存取內容。

Example 範例：Stale-If-Error

當您將 `Cache-Control` 標頭設定為使用這些指令時，CloudFront 會執行下列動作。

```
Cache-Control: max-age=3600, stale-if-error=86400
```

1. CloudFront 會快取回應一小時 (`max-age=3600`)。
2. 如果原始伺服器關閉或在此持續時間之後傳回錯誤，CloudFront 會持續提供過時內容最多 24 小時 (`stale-if-error=86400`)
3. 如您已設定自訂錯誤回應，如果在指定的 `stale-if-error` 持續時間內發生錯誤，CloudFront 會先嘗試提供過時內容。如果過時內容無法使用，則 CloudFront 會依對應錯誤狀態碼的設定提供自訂錯誤回應。如需更多詳細資訊，請參閱 [產生自訂錯誤回應](#)。

 備註

- CloudFront 會將過時的內容提供至 `stale-if-error` 指令的值或 CloudFront [最大 TTL](#) 的值，以較小者為準。在最長 TTL 持續時間之後，無論 `stale-if-error` 值為何，都不會從邊緣節點快取提供過時的物件。
- 如果您未設定 `stale-if-error` 或自訂錯誤回應，CloudFront 會根據請求的物件是否在邊緣節點快取中，傳回過時物件或將錯誤回應轉送回檢視器。如需更多詳細資訊，請參閱 [您尚未設定自訂錯誤頁面時 CloudFront 如何處理錯誤](#)。

使用兩種指令

`stale-while-revalidate` 與 `stale-if-error` 是獨立的快取控制指令，您可以一起使用以減少延遲，並為原始伺服器新增緩衝區以便回應或復原。

Example 範例：使用這兩個指令

當您將 `Cache-Control` 標頭設定為使用下列指令時，CloudFront 會執行下列動作。

```
Cache-Control: max-age=3600, stale-while-revalidate=600, stale-if-error=86400
```

1. CloudFront 會快取回應一小時 (`max-age=3600`)。
2. 如果請求在此持續時間之後提出，CloudFront 會在重新驗證內容期間最多提供 10 分鐘 (`stale-while-revalidate=600`) 的時間。
3. 如果原始伺服器在 CloudFront 嘗試重新驗證內容時傳回錯誤，CloudFront 將持續提供過時內容長達 24 小時 (`stale-if-error=86400`)。

快取是效能和更新狀態之間的平衡。使用類似 `stale-while-revalidate` 與 `stale-if-error` 指令可增強效能和使用者的體驗，但請確保組態與您希望內容的更新狀態保持一致。過時內容指令最適合需要重新整理內容但不需要最新版本的使用案例。此外，如果您的內容沒有變更或很少變更，`stale-while-revalidate` 可能會新增不必要的網路要求。反之，請考慮設定較長的快取持續時間。

指定 CloudFront 快取物件的時間

若要控制 CloudFront 在傳送另一個請求至原始伺服器前將物件保留在快取中的時間，您可以：

- 設定 CloudFront 分散式快取行為中的最短、最長和預設 TTL 值。您可以在連接至快取行為的[快取政策](#) (建議使用) 或舊版快取設定中設定這些值。
- 在來自原始伺服器的回應中包含 `Cache-Control` 或 `Expires` 標頭。這些標頭還有助於判定在傳送另一個請求到 CloudFront 之前，瀏覽器將物件保留在瀏覽器快取中的時間長度。

下表解釋了從原始伺服器傳送的 `Cache-Control` 和 `Expires` 標頭如何與快取行為中的 TTL 設定一起運作，以影響快取。

原始標頭	最短 TTL = 0	最短 TTL > 0
	CloudFront 快取	CloudFront 快取

原始標頭	最短 TTL = 0	最短 TTL > 0
原始伺服器將 Cache-Control: max-age 指示詞新增至物件	CloudFront 會針對 Cache-Control: max-age 指令值或 CloudFront 最大 TTL 值兩者之中較小的值快取物件。 瀏覽器快取 瀏覽器會快取 Cache-Control: max-age 指令值的物件。	CloudFront 快取取決於 CloudFront 最小 TTL 與最大 TTL 及 Cache-Control max-age 指示詞的值： - 如果最小 TTL < max-age < 最大 TTL，則 CloudFront 會快取 Cache-Control: max-age 指令值的物件。 - 如果 max-age < 最小 TTL，則 CloudFront 會快取 CloudFront 最小 TTL 值的物件。 - 如果 max-age > 最大 TTL，則 CloudFront 會快取 CloudFront 最大 TTL 值的物件。 瀏覽器快取 瀏覽器會快取 Cache-Control: max-age 指令值的物件。

原始標頭	最短 TTL = 0	最短 TTL > 0
原始伺服器不會將 Cache-Control: max-age 指示詞新增至物件	CloudFront 快取 CloudFront 會快取 CloudFront 預設 TTL 值的物件。 瀏覽器快取 取決於瀏覽器。	CloudFront 快取 CloudFront 會針對 CloudFront 最小 TTL 值或預設 TTL 值兩者之中較大的值快取物件。 瀏覽器快取 取決於瀏覽器。

原始標頭	最短 TTL = 0	最短 TTL > 0
原始伺服器將 Cache-Control: max-age 與 Cache-Control: s-maxage 指示詞新增至物件	CloudFront 快取 CloudFront 會針對 Cache-Control: s-maxage 指令值或 CloudFront 最大 TTL 值兩者之中較小的值快取物件。 瀏覽器快取 瀏覽器會快取 Cache-Control: max-age 指令值的物件。	CloudFront 快取 CloudFront 快取取決於 CloudFront 最小 TTL 與最大 TTL 及 Cache-Control: s-maxage 指示詞的值： • 如果最小 TTL < s-maxage < 最大 TTL，則 CloudFront 會快取 Cache-Control: s-maxage 指令值的物件。 • 如果 s-maxage < 最小 TTL，則 CloudFront 會快取 CloudFront 最小 TTL 值的物件。 • 如果 s-maxage > 最大 TTL，則 CloudFront 會快取 CloudFront 最大 TTL 值的物件。 瀏覽器快取 瀏覽器會快取 Cache-Control: max-age 指令值的物件。

原始標頭	最短 TTL = 0	最短 TTL > 0
原始伺服器將 Expires 標頭新增至物件	CloudFront 快取 CloudFront 會快取物件直到 Expires 標頭中的日期或 CloudFront 最大 TTL 值，以先到者為準。 瀏覽器快取 瀏覽器會快取物件直到 Expires 標頭中的日期。	CloudFront 快取 CloudFront 快取取決於 CloudFront 最短 TTL 與最長 TTL 及 Expires 標頭的值： • 如果最小 TTL Expires < 最大 TTL，則 CloudFront 會快取物件，直到 Expires 標頭中的日期和時間為止。 • 如果 Expires < 最小 TTL，則 CloudFront 會快取 CloudFront 最小 TTL 值的物件。 • 如果 Expires > 最大 TTL，則 CloudFront 會快取 CloudFront 最大 TTL 值的物件。 瀏覽器快取 瀏覽器會快取物件直到 Expires 標頭中的日期與時間。

原始標頭	最短 TTL = 0	最短 TTL > 0
原始伺服器將 Cache-Control: no-cache 、 no-store 和/或 private 指示詞新增至物件	CloudFront 和瀏覽器遵守標頭。	CloudFront 快取 CloudFront 會快取 CloudFront 最小 TTL 值的物件。 請參閱此資料表下方的警告。 瀏覽器快取 瀏覽器遵守標頭。

Warning

- 如果您的最小 TTL 大於 0，則即使原始標頭中存在 Cache-Control: no-cache、no-store 和/或 private 指令，CloudFront 都會使用快取政策的最小 TTL。
- 如果原始伺服器可以存取，CloudFront 會從原始伺服器取得物件並將其傳回給檢視器。
- 如果原始伺服器無法連線，且最小或最大 TTL 大於 0，則 CloudFront 會處理之前從原始伺服器取得的物件。

為了避免這種行為，請將 Cache-Control: stale-if-error=0 指示詞與從原始伺服器傳回的物件一同包含。如果原始伺服器無法連線，這會導致 CloudFront 傳回錯誤以回應未來的請求，而不是傳回其之前從原始伺服器取得的物件。

- 當原始伺服器標頭包含 Cache-Control: no-cache、no-store 和/或 private 指令時，CloudFront 不會從 S3 原始伺服器快取 HTTP 501 狀態碼 (未實作)。這是 S3 原始伺服器的預設行為，即使您的[最小 TTL](#) 設定大於 0 也一樣。

如需有關如何使用 CloudFront 主控台變更分佈設定的詳細資訊，請參閱[更新分佈](#)。如需有關如何使用 CloudFront API 為分佈變更設定的詳細資訊，請參閱[UpdateDistribution](#)。

使用 Amazon S3 主控台新增標頭到物件

您可以將 Cache-Control 或 Expires 標頭欄位新增至 Amazon S3 物件。若要執行此操作，請修改物件的中繼資料欄位。

將 **Cache-Control** 或 **Expires** 標頭欄位新增至 Amazon S3 物件

1. 請遵循取代系統定義的中繼資料一節中的程序。該程序位於 Amazon S3 使用者指南的[在 Amazon S3 主控台中編輯物件中繼資料](#)主題中。
2. 對於 Key (金鑰)，選擇您要新增的標頭名稱 (Cache-Control (快取控制) 或 Expires (過期))。
3. 對於 Value (值)，輸入標頭值。例如，對於 Cache-Control 標題，您可以輸入 max-age=86400。對於 Expires，您可以輸入過期日期和時間，例如 Wed, 30 Jun 2021 09:28:00 GMT。
4. 請遵循其餘程序以儲存中繼資料變更。

根據查詢字串參數快取內容

有些 Web 應用程式使用查詢字串來傳送資訊到原始伺服器。查詢字串是 Web 請求的一部分，其顯示在 ? 字元之後；此字串可包含以 & 字元分隔的一或多個參數。在下列範例中，查詢字串包含兩個參數，*color=red* 及 *size=large*：

`https://d111111abcdef8.cloudfront.net/images/image.jpg?color=red&size=large`

針對分佈，您可以選擇是否要讓 CloudFront 將查詢字串轉送到原始伺服器，以及是否根據所有參數或選取的參數快取內容。為什麼說這可能很有用？請考量下列範例。

假設您的網站提供五種語言。網站所有五種版本的目錄結構和檔案名稱都相同。當使用者檢視網站時，轉送給 CloudFront 的請求包含一個根據使用者所選語言的語言查詢字串參數。可以設定 CloudFront 轉送查詢字串到原始伺服器，並根據語言參數快取。如果您設定 Web 伺服器來傳回與所選語言對應的指定頁面版本，則 CloudFront 會根據語言查詢字串參數的值分別快取每個語言版本。

在這個範例中，如果網站的主要頁面為 main.html，以下五個請求會造成 CloudFront 快取 main.html 五次，對各語言查詢字串參數的值進行一次快取：

- `https://d111111abcdef8.cloudfront.net/main.html?language=de`
- `https://d111111abcdef8.cloudfront.net/main.html?language=en`
- `https://d111111abcdef8.cloudfront.net/main.html?language=es`
- `https://d111111abcdef8.cloudfront.net/main.html?language=fr`
- `https://d111111abcdef8.cloudfront.net/main.html?language=jp`

注意下列事項：

- 有些 HTTP 伺服器不處理查詢字串參數，因此不會傳回根據參數值的物件不同版本。針對這些原始伺服器，如果您設定 CloudFront 轉送查詢字串參數至原始伺服器，即使原始伺服器為每個參數值傳回相同物件的版本給 CloudFront，CloudFront 仍會根據參數值執行快取。
- 為使查詢字串參數能如以上範例所述搭配多種語言運作，各查詢字串參數間必須使用 & 字元做為分隔符號。如果您使用不同的分隔符號，即有可能得到非預期的結果，具體取決於您所指定供 CloudFront 做為快取基礎的參數以及這些參數在查詢字串中顯示的順序。

以下範例顯示使用不同的分隔符號並將 CloudFront 設定為僅根據 color 參數進行快取時將會發生的情況：

- 在下列請求中，CloudFront 根據 color 參數值快取內容，但 CloudFront 將該值解譯為 *red;size=large*：

```
https://d111111abcdef8.cloudfront.net/images/  
image.jpg?color=red;size=large
```

- 在下列請求中，CloudFront 快取內容但不根據查詢字串參數快取。這是因為根據 color 參數設定 CloudFront 來快取，但 CloudFront 解譯以下字串為只包含一個 size 參數，該參數的值為 *large;color=red*：

```
https://d111111abcdef8.cloudfront.net/images/  
image.jpg?size=large;color=red
```

您可設定 CloudFront 執行以下其中一項：

- 完全不要轉送查詢字串到原始伺服器。如果您不轉送查詢字串，CloudFront 不會根據查詢字串參數快取。
- 轉送查詢字串到原始伺服器，以及在查詢字串中根據所有參數快取。
- 轉送查詢字串到原始伺服器，以及在查詢字串中根據指定的參數快取。

如需更多詳細資訊，請參閱 [the section called “最佳化快取”](#)。

主題

- [查詢字串轉送和快取的主控台和 API 設定](#)
- [最佳化快取](#)
- [查詢字串參數和 CloudFront 標準記錄檔 \(存取日誌\)](#)

查詢字串轉送和快取的主控制台和 API 設定

當您在 CloudFront 主控台中建立分佈時，CloudFront 會根據原始伺服器類型為您設定查詢字串的轉送和快取。或者，您可以手動編輯這些設定。如需詳細資訊，請參閱 [the section called “所有分佈設定”](#) 中的下列主題：

- [the section called “查詢字串轉送和快取”](#)
- [the section called “查詢字串允許清單”](#)

如要使用 CloudFront API 設定查詢字串轉送和快取，請參閱《Amazon CloudFront API 參考》中的 [CachePolicy](#) 和 [OriginRequestPolicy](#)。

最佳化快取

根據查詢字串參數將 CloudFront 設為快取時，可以採取下列步驟來減少 CloudFront 轉送至原始伺服器的請求數。CloudFront 節點存取物件時，您可以減少原始伺服器的負載並降低延遲，因為物件由離您使用者較近的位置提供。

快取僅根據原始伺服器傳回物件不同版本的參數

針對每個 Web 應用程式轉送到 CloudFront 的查詢字串參數，CloudFront 為每個參數值轉送請求到原始伺服器，且為每個參數值快取個別的物件版本。即使原始伺服器一律傳回相同物件，無論參數值如何，情況都是如此。對於多個參數，請求的數量和物件的數量相乘。

我們建議您設定 CloudFront 快取僅根據原始伺服器傳回物件不同版本的查詢字串參數，並且根據每個參數仔細考慮快取的優點。例如，假設您有一個零售網站。您有六種不同顏色的夾克圖片，以及夾克有 10 種不同的大小。您已有的圖片顯示夾克的不同顏色，但不顯示不同大小。為了最佳化快取，應設定 CloudFront 僅根據顏色參數，而非根據大小參數來快取。這會增加 CloudFront 的可能性，可以提供來自快取的請求，這可以提高效能，並降低您的原始伺服器原始伺服器的負載。

一律以相同順序列出參數

在查詢字串中參數的順序很重要。在下列範例中，查詢字串完全相同，除了參數的順序不同。這會造成 CloudFront 轉送兩個個別的 image.jpg 請求到原始伺服器並快取兩個個別的物件版本：

- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=red&size=large`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large&color=red`

我們建議您一律以相同順序列出參數名稱，例如字母順序。

參數名稱和值請一律使用相同的大小寫

根據查詢字串參數快取時，CloudFront 考慮參數名稱和值的大小寫。在下列範例中，查詢字串完全相同，除了參數名稱和值的大小寫以外。這會造成 CloudFront 轉送四個個別的 image.jpg 請求到原始伺服器並快取四個個別的物件版本：

- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?color=Red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?Color=red`
- `https://d111111abcdef8.cloudfront.net/images/image.jpg?Color=Red`

我們建議您為參數名稱和值使用一致的大小寫，例如全部小寫。

不使用與簽署 URL 衝突的參數名稱

如果您使用簽署 URL 來限制存取內容 (如果您新增可信任簽署者到分佈)，在轉送 URL 的其他部分到原始伺服器之前，則 CloudFront 移除以下查詢字串參數：

- Expires
- Key-Pair-Id
- Policy
- Signature

如果您使用簽署 URL，且想要設定 CloudFront 將查詢字串轉送到原始伺服器，則無法命名自己的查詢字串參數為 Expires、Key-Pair-Id、Policy 或 Signature。

查詢字串參數和 CloudFront 標準記錄檔 (存取日誌)

如果您啟用記錄，則 CloudFront 會記錄完整的 URL，包括查詢字串參數。無論您是否已設定 CloudFront 將查詢字串轉送給原始伺服器，這都是如此。如需 CloudFront 記錄的詳細資訊，請參閱 [the section called “存取日誌 \(標準日誌\)”](#)。

根據 Cookie 快取內容

根據預設，CloudFront 在處理請求和回應時，以及在節點中快取物件時，不會考慮 Cookie。如果 CloudFront 接收到除了 Cookie 標頭中的內容外的兩個相同請求，則根據預設，CloudFront 會將請求視為相同，並為兩個請求傳回相同物件。

您可以設定 CloudFront 將檢視者請求中的部分或所有 Cookie 轉送到原始伺服器，並根據轉送的中 Cookie 值，快取個別物件版本。執行這項操作時，CloudFront 會在檢視者請求中使用部分或全部 Cookie (無論設定了哪個轉送)，以唯一識別快取中的物件。

例如，假設 `locations.html` 的請求包含具有 `country` 或 `uk` 值的 `fr` Cookie。當您設定 CloudFront 根據 `country` Cookie 值快取物件時，CloudFront 會將 `locations.html` 的請求轉送至原始伺服器，並包含 `country` Cookie 和其值。原始伺服器傳回 `locations.html`，CloudFront 會為 `country` Cookie 為 `uk` 的請求快取其物件一次，以及為該值為 `fr` 的請求快取其物件一次。

Important

Amazon S3 和一些 HTTP 伺服器不處理 Cookie。不要設定 CloudFront，將 Cookie 轉送到不處理 Cookie 或不根據 Cookie 改變其回應的原始伺服器。這會導致 CloudFront 轉送更多的請求到相同物件的原始伺服器，這會降低效能且增加原始伺服器的負載。如果考慮到前面的範例，原始伺服器不處理 `country` Cookie 或始終將相同版本的 `locations.html` 傳回 CloudFront，無論 `country` Cookie 的值如何，請不要設定 CloudFront 來轉送該 Cookie。反之，如果自定義原始伺服器取決於特定 cookie，或根據 Cookie 傳送不同回應，請務必設定 CloudFront，將該 cookie 轉送到原始伺服器。否則，CloudFront 會在轉送請求至您的原始伺服器之前就移除該 Cookie。

若要設定 Cookie 轉送，請更新分佈的快取行為。如需有關快取行為的詳細資訊，請參閱 [快取行為設定](#)，尤其是 [轉送 Cookie](#) 和 [允許清單 Cookie](#) 小節。

您可以設定每個快取行為，執行下列其中一項動作：

- 將所有 Cookie 轉送到原始伺服器 - CloudFront 包括瀏覽器將請求轉送到原始伺服器時，檢視者發送的所有 Cookie。當原始伺服器傳回回應時，CloudFront 會使用檢視者請求中的 Cookie 名稱和值來快取回應。如果原始回應包含 `Set-Cookie` 標頭，CloudFront 會將它們傳回給具有請求物件的檢視者。CloudFront 也會使用從原始伺服器傳回的物件快取 `Set-Cookie` 標頭，並在所有快取點擊時將這些 `Set-Cookie` 標頭傳送至所有快取命中的檢視者。
- 轉送您指定的一組 Cookie - CloudFront 會先移除檢視者傳送的任何不在允許清單上的 Cookie，然後再將請求轉寄給原始伺服器。CloudFront 會使用檢視者請求中列入清單的 Cookie 名稱和值來快取回應。如果原始回應包含 `Set-Cookie` 標頭，CloudFront 會將它們傳回給具有請求物件的檢視者。CloudFront 也會使用從原始伺服器傳回的物件快取 `Set-Cookie` 標頭，並在所有快取點擊時將這些 `Set-Cookie` 標頭傳送至所有快取命中的檢視者。

如需有關在 Cookie 名稱中指定萬用字元的詳細資訊，請參閱 [允許清單 Cookie](#)。

如需有關針對每個快取行為轉送的 Cookie 名稱數量的目前配額，或是有關請求更高配額的詳細資訊，請參閱 [查詢字串的配額 \(舊版快取設定\)](#)。

- 不轉送 Cookie 至原始伺服器 - CloudFront 不會根據檢視器傳送的 Cookie 來快取物件。此外，CloudFront 會先移除 Cookie，再將請求轉送至原始伺服器，然後在回應傳回檢視者之前，移除回應中的 Set-Cookie 標頭。由於這不是使用原始伺服器資源的最佳方式，因此當您選取此快取行為時，您應該確保原始伺服器預設不會在原始伺服器回應中包含 Cookie。

請注意以下有關指定您要轉送的 Cookie：

存取日誌

如果您設定 CloudFront 以記錄請求及記錄 Cookie，CloudFront 將記錄所有 Cookie 和所有 Cookie 屬性，即使您設定 CloudFront 不轉送 Cookie 至原始伺服器，或設定 CloudFront 僅轉送指定 Cookie。如需 CloudFront 記錄的詳細資訊，請參閱 [存取日誌 \(標準日誌\)](#)。

區分大小寫

Cookie 名稱和值都是區分大小寫。例如，如果 CloudFront 設為轉送所有 Cookie，且同一個物件的兩個瀏覽者請求具有相同的 Cookie，則 CloudFront 會快取物件兩次。

CloudFront 排序 Cookie

若 CloudFront 設為轉送 Cookie (所有或子集)，則 CloudFront 在自然順序中藉由 Cookie 名稱排序 Cookie 然後再轉送請求到原始伺服器。

Note

系統不支援以 \$ 字元開頭的 Cookie 名稱。否則，CloudFront 在轉送請求至您的原始伺服器之前就會移除該 Cookie。您可以在 Cookie 名稱的開頭移除 \$ 字元或指定不同的字元。

If-Modified-Sinceand(If-None-Match)

若 CloudFront 設為轉送 Cookie (所有或子集)，則不支援 If-Modified-Since 和 If-None-Match 條件請求。

需要標準名稱值組格式

當值符合[標準名稱值組格式](#)時，CloudFront 才會轉送 Cookie 標頭，例如："Cookie: cookie1=value1; cookie2=value2"

停用 **Set-Cookie** 標頭的快取功能

如果設定 CloudFront 為將 Cookie 轉送至原始伺服器 (全部或特定 Cookies)，它也會快取在原始伺服器回應接收到的 Set-Cookie 標頭。CloudFront 對原始瀏覽者的回應中包含這些 Set-Cookie 標頭，且亦包含於 CloudFront 快取提供的後續回應中。

如果您想在原始伺服器接收 cookie，但不希望 CloudFront 快取原始伺服器回應中的 Set-Cookie 標頭，請設定原始伺服器以新增 Cache-Control 標頭，該標頭具有指定 no-cache 為欄位名稱的 Set-Cookie 指令。例如：Cache-Control: no-cache="Set-Cookie"。如需詳細資訊，請參閱 Hypertext Transfer Protocol (HTTP/1.1)：快取標準中的[回應快取控制指令](#)。

Cookie 名稱的長度上限

若您設定 CloudFront 轉送指定 Cookie 到原始伺服器，在設定 CloudFront 轉送的所有 Cookie 名稱中所有位元組總數不能超過 512 減去您正轉送的 Cookie 數量。例如，如果設定 CloudFront 轉送 10 個 Cookie 到原始伺服器，則 10 個 Cookie 名稱的總和長度不能超過 502 位元 (512 – 10)。

如果您設定 CloudFront 轉送所有 Cookie 到原始伺服器，則 Cookie 名稱的長度不重要。

如需有關使用 CloudFront 主控台來更新分佈，讓 CloudFront 能轉送 Cookie 到原始伺服器的詳細資訊，請參閱[更新分佈](#)。如需使用 CloudFront API 更新分佈的詳細資訊，請參閱 Amazon CloudFront API 參考中的[UpdateDistribution](#)。

根據請求標頭快取內容

CloudFront 讓您選擇是否要讓 CloudFront 將標頭轉送至原始伺服器，並根據檢視器請求中的標頭值，快取指定物件的個別版本。這可讓您根據使用者使用的裝置、檢視器的位置、檢視器使用的語言，以及各種其他條件，提供內容的不同版本。

主題

- [標頭和分佈 - 概觀](#)
- [選取快取時所依據的標頭](#)
- [設定 CloudFront 以遵循 CORS 設定](#)
- [根據裝置類型設定快取](#)
- [根據檢視器語言設定快取](#)
- [根據檢視器位置設定快取](#)
- [根據請求的通訊協定設定快取](#)

- [設定壓縮檔案的快取](#)
- [快取如何根據標頭影響效能](#)
- [標頭大小寫和標頭值如何影響快取](#)
- [CloudFront 傳回給檢視器的標頭](#)

標頭和分佈 - 概觀

在預設情況下，CloudFront 在節點中快取物件時不會考慮標頭。如果原始伺服器傳回兩個物件，且只有請求標頭中的值不同，則 CloudFront 只快取一個物件版本。

您可以設定 CloudFront 轉送標頭到原始伺服器，這讓 CloudFront 根據一個或多個請求標頭中的值來快取多個物件版本。要設定 CloudFront 根據特定標頭的值來快取物件，您可以為您的分佈指定快取行為設定。如需詳細資訊，請參閱[根據選取的請求標頭執行快取](#)。

例如，假設 logo.jpg 的檢視器請求包含具有 Product 或 Acme 值的客戶 Apex 標頭。當您設定 CloudFront 根據 Product 標頭值快取物件時，CloudFront 會轉送 logo.jpg 的請求到原始伺服器，並包含 Product 標頭和標頭值。CloudFront 為請求中 logo.jpg 標頭值為 Product 與請求中值為 Acme 各快取 Apex 一次。

您可以在分佈中設定每個快取行為，以執行以下其中一項：

- 轉送所有標頭到原始伺服器

Note

對於舊版快取設定 – 如果您設定 CloudFront 將所有標頭轉送到原始伺服器，則 CloudFront 不會快取與此快取行為關聯的物件。反之，它會傳送每個請求到原始伺服器。

- 轉送您指定的標頭清單。CloudFront 根據所有指定標頭中的值來快取物件。CloudFront 也轉送預設情況下轉送的標頭，但只根據您指定的標頭來快取物件。
- 只轉送預設標頭。這個組態中，CloudFront 根據請求標頭中的值不會快取物件。

如需有關針對每個快取行為轉送單的標頭數量的目前配額，或是有關請求更高配額的詳細資訊，請參閱[標頭的配額](#)。

如需有關使用 CloudFront 主控台來更新分佈，讓 CloudFront 能轉送標頭到原始伺服器的詳細資訊，請參閱[更新分佈](#)。如需有關使用 CloudFront API 更新現有分佈的資訊，請參閱 Amazon CloudFront API 參考中的[更新分佈](#)。

選取快取時所依據的標頭

您可以轉送到原始伺服器且 CloudFront 可基礎快取的標頭，取決於您的原始伺服器是 Amazon S3 儲存貯體還是自訂原始伺服器。

- Amazon S3 – 您可以設定 CloudFront 根據多個特定標頭轉送機快取物件 (請參閱下列例外狀況清單)。不過，我們建議您避免轉送具有 Amazon S3 原始伺服器的標頭，除非您需要實作跨來源資源共享 (CORS)，或是想要在原始伺服器面向事件中使用 Lambda@Edge 將內容個人化。
- 若要設定 CORS，您必須轉送標頭，以允許 CloudFront 為已啟用跨來源資源共享 (CORS) 的網站分佈內容。如需更多詳細資訊，請參閱 [設定 CloudFront 以遵循 CORS 設定](#)。
- 若要使用您轉發至 Amazon S3 原始伺服器的標頭將內容個人化，請編寫並新增 Lambda@Edge 函數，並將其與要由原始伺服器面向的事件所觸發的 CloudFront 分佈相關聯。如需使用標頭來個人化內容的詳細資訊，請參閱[根據國家/地區或裝置類型標頭個人化 - 範例](#)。

我們建議您避免轉送沒有要用來個人化內容的標頭，因為轉送額外的標頭可能會降低您的快取命中率。也就是說，CloudFront 無法將那麼多來自節點快取的請求做為所有請求的一部分來處理。

- 自訂原始伺服器 - 您可以設定 CloudFront 根據下列之外的任何請求標頭值來執行快取：
 - Connection
 - Cookie - 如果要根據 Cookie 來轉送及快取，您可以在分佈中使用個別的設定。如需更多詳細資訊，請參閱 [根據 Cookie 快取內容](#)。
 - Host (for Amazon S3 origins)
 - Proxy-Authorization
 - TE
 - Upgrade

您可以設定 CloudFront 根據 Date 與 User-Agent 標頭中的值來快取物件，但我們不建議這麼做。這些標頭有許多可能的值，且根據其值來快取可能導致 CloudFront 轉送更多請求到原始伺服器。

如需 HTTP 請求標頭的完整清單，以及 CloudFront 處理這些標頭的方式，請參閱 [HTTP 請求標頭和 CloudFront 行為 \(自訂和 Amazon S3 原始伺服器\)](#)。

設定 CloudFront 以遵循 CORS 設定

如果您已在 Amazon S3 儲存貯體或自訂原始伺服器上啟用跨來源資源共享 (CORS)，您必須選擇特定的標頭進行轉發，以遵守 CORS 設定。您必須轉發的標頭會因原始伺服器 (Amazon S3 或自訂)，以及您是否要快取 OPTIONS 回應而異。

Amazon S3

- 如果您希望快取 OPTIONS 回應，請執行下列動作：
 - 選擇預設快取行為設定的選項，來啟用 OPTIONS 回應的快取。
 - 設定 CloudFront 轉送以下標頭：Origin、Access-Control-Request-Headers 及 Access-Control-Request-Method。
- 如果您不要快取 OPTIONS 回應，請設定 CloudFront 將 Origin 標頭以及原始伺服器所需的任何其他標頭一起轉送 (例如，Access-Control-Request-Headers、Access-Control-Request-Method 或其他)。

自訂原始伺服器 - 轉送 Origin 標頭與原始伺服器需要的其他任何標頭。

若要設定 CloudFront 以根據 CORS 快取回應，您必須設定 CloudFront 以使用快取政策轉送標頭。如需更多詳細資訊，請參閱 [使用政策控制快取金鑰](#)。

如需 CORS 和 Amazon S3 的詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的[使用跨來源資源分享 \(CORS\)](#)。

根據裝置類型設定快取

如果您要讓 CloudFront 根據使用者使用以檢視內容的裝置來快取不同的物件版本，請設定 CloudFront 來轉送適用標頭到自訂原始伺服器：

- CloudFront-Is-Desktop-Viewer
- CloudFront-Is-Mobile-Viewer
- CloudFront-Is-SmartTV-Viewer
- CloudFront-Is-Tablet-Viewer

根據 User-Agent 標頭值，CloudFront 將這些標頭值設定為 true 或 false，然後將請求轉送至原始伺服器。如果裝置屬於多個類別，一個以上的值可能是 true。例如，針對一些平板電腦裝

置，CloudFront 可能將 CloudFront-Is-Mobile-Viewer 與 CloudFront-Is-Tablet-Viewer 同時設定為 true。

根據檢視器語言設定快取

如果您要讓 CloudFront 根據請求中指定的語言，快取物件的不同版本，請設定 CloudFront 轉送 Accept-Language 標頭至原始伺服器。

根據檢視器位置設定快取

如果您要讓 CloudFront 根據請求來源國快取物件的不同版本，請設定 CloudFront 轉送 CloudFront-Viewer-Country 標頭至原始伺服器。CloudFront 自動轉換來自兩個字母國家/地區代碼請求的 IP 位址。如需可依據代碼與國家/地區名稱排序的國家/地區代碼易用清單，請參閱 Wikipedia 項目 [ISO 3166-1 alpha-2](#)。

根據請求的通訊協定設定快取

如果您要讓 CloudFront 根據請求的通訊協定 HTTP 或 HTTPS 來快取物件的不同版本，請設定 CloudFront 將 CloudFront-Forwarded-Proto 標頭轉送至原始伺服器。

設定壓縮檔案的快取

如果您的原始伺服器支援 Brotli 壓縮，您可以根據 Accept-Encoding 標頭快取。只有在您的原始伺服器根據標頭提供不同內容時，才應根據 Accept-Encoding 設定快取。

快取如何根據標頭影響效能

當您設定 CloudFront 根據一或多個標頭和有一個以上可能值的標頭來快取時，CloudFront 轉送多個請求到相同物件的原始伺服器。這會降低效能和增加原始伺服器的負載。如果原始伺服器不管指定標頭值傳回相同物件，我們建議您不要設定 CloudFront 根據該標頭快取。

如果您設定 CloudFront 轉送多個標頭，則檢視器請求中的標頭順序只要值一樣，便不會影響快取。例如，如果一個請求包含標頭 A:1、B:2 和另一個請求包含 B:2、A:1，則 CloudFront 只快取一個物件副本。

標頭大小寫和標頭值如何影響快取

當 CloudFront 根據標頭值快取時，不會考慮標頭名稱的大小寫，但會考慮標頭值的大小寫：

- 如果檢視器請求包含 `Product:Acme` 和 `product:Acme`，CloudFront 只會快取物件一次。它們之間的唯一差別是不會影響快取的標頭大小寫。
- 如果檢視器請求同時包含 `Product:Acme` 與 `Product:acme`，CloudFront 會快取物件兩次，因為有一些請求中的值是 `Acme`，而其他請求中的值則是 `acme`。

CloudFront 傳回給檢視器的標頭

設定 CloudFront 來轉送和快取標頭不會影響 CloudFront 傳回哪一個標頭給檢視器。CloudFront 傳回它從原始伺服器取得的標頭，有幾個例外狀況。如需詳細資訊，請參閱適用的主題：

- Amazon S3 原始伺服器 - 請參閱[CloudFront 移除或更新的 HTTP 回應標頭](#)。
- 自訂原始伺服器 - 請參閱[CloudFront 移除或取代的 HTTP 回應標頭](#)。

使用政策控制快取金鑰

透過 CloudFront 快取政策，您可以指定要由 CloudFront 納入快取金鑰中的 HTTP 標頭、Cookie，以及查詢字串，作為 CloudFront 邊緣節點快取物件的依據。快取金鑰是快取中每個物件的唯一識別碼，它會決定檢視器請求是否會導致快取命中。

當檢視者請求產生與先前請求相同的快取金鑰，且該快取金鑰的物件位於節點的快取中且有效時，就會發生快取命中。當有快取點擊時，物件會從 CloudFront 邊緣節點提供給檢視者，這具有下列優點：

- 降低原始伺服器的負載
- 減少檢視器的延遲

在快取金鑰中包含較少的值會增加快取命中的可能性。因為這會擁有較高的快取命中率 (較高的瀏覽者請求會導致快取命中)，您便可以從網站或應用程式獲得較佳的效能。如需更多詳細資訊，請參閱 [了解快取金鑰](#)。

若要控制快取金鑰，您可以使用 CloudFront 快取政策。您可以將快取政策附加到 CloudFront 分佈中的一或多個快取行為。

您也可以使用快取政策來指定 CloudFront 快取中物件的存留時間 (TTL) 設定，並啟用 CloudFront 以請求和快取壓縮物件。

Note

快取設定不影響 gRPC，因為 gRPC 流量無法進行快取。如需更多詳細資訊，請參閱 [使用 gRPC 搭配 CloudFront 分佈](#)。

主題

- [了解快取政策](#)
- [建立快取政策](#)
- [使用受管快取政策](#)
- [了解快取金鑰](#)

了解快取政策

您可以使用快取政策藉由控制快取金鑰中包含的值 (URL 查詢字串、HTTP 標頭和 Cookie) 來改善快取命中率。CloudFront 會針對常見使用案例提供一些預先定義的快取政策 (稱為受管政策)。您可以使用這些受管政策，也可以建立專屬於您需求的專屬快取政策。如需有關受管政策的詳細資訊，請參閱 [使用受管快取政策](#)。

快取政策包含下列設定，這些設定分類為政策資訊、存留時間 (TTL) 設定，以及快取金鑰設定。

政策資訊

名稱

用來識別快取政策的唯一名稱。在主控台中，您可以使用名稱將快取政策附加到快取行為。

描述

用來描述快取政策的備註。這是選用的，但它可以協助您識別快取政策的目的。

存活期 (TTL) 設定

存留時間 (TTL) 設定會與 Cache-Control 和 Expires HTTP 標頭 (如果它們在原始回應中) 一起運作，以判斷 CloudFront 快取中物件保持有效的時間長度。

最短 TTL

預設時間的長度下限 (以秒為單位)，此時間是您希望在 CloudFront 檢查來源以查看物件是否已更新前，物件要停留在 CloudFront 快取中的時間。如需更多詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

Warning

如果您的最小 TTL 大於 0，則即使原始伺服器標頭中存在 Cache-Control: no-cache、no-store 或 private 指令，CloudFront 也會快取內容至少達到快取政策最短 TTL 中指定的持續時間。

最長 TTL

時間的長度上限 (以秒為單位)，此時間是 CloudFront 檢查來源以查看物件是否已更新前，物件要停留在 CloudFront 快取中的時間。只有在來源與物件一起傳送 Cache-Control 或 Expires 標

頭時，CloudFront 才會使用此設定。如需詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

預設 TTL

預設時間長度 (以秒為單位)，此時間是您希望在 CloudFront 檢查來源以查看物件是否已更新前，物件要停留在 CloudFront 快取中的時間。只有在原始伺服器物件未與物件一起傳送 Cache-Control 或 Expires 標頭時，CloudFront 才會使用此設定的值做為物件的 TTL。如需更多詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

Note

如果最短 TTL、最長 TTL 和預設 TTL 設定都設為 0，這會停用 CloudFront 快取。

快取金鑰設定

快取金鑰設定會指定檢視器請求中 CloudFront 包含在快取金鑰中的值。這些值可以包括 URL 查詢字串、HTTP 標頭和 Cookie。您包含在快取金鑰中的值會自動包含在 CloudFront 傳送至原始伺服器的請求中，稱為原始伺服器請求。如需在不影響快取金鑰的情況下控制原始伺服器請求的相關資訊，請參閱 [使用政策控制原始伺服器請求](#)。

快取金鑰設定包括：

- [標頭](#)
- [Cookie](#)
- [查詢字串](#)
- [壓縮支援](#)

標頭

在檢視器請求中，CloudFront 會包含在快取金鑰和原始伺服器請求中的 HTTP 標頭。針對標頭，您可以選擇下列設定之一：

- 無 - 檢視器請求中的 HTTP 標頭不會包含在快取金鑰中，也不會自動包含在原始伺服器請求中。
- Include the following headers (包含以下標頭) - 您可以指定檢視器請求中的哪些 HTTP 標頭會包含在快取金鑰中，並自動包含在原始伺服器請求中。

當您使用 Include the following headers (包含以下標頭) 設定時，您可以指定 HTTP 標頭的名稱，而不是它們的值。例如，請參閱下列 HTTP 標頭：

```
Accept-Language: en-US,en;q=0.5
```

在這種情況下，您可以將標頭指定為 Accept-Language，而不是 Accept-Language: en-US,en;q=0.5。但是，在快取金鑰和原始伺服器請求中，CloudFront 會包含完整的標頭，包括其值。

您也可以快取金鑰中，包含由 CloudFront 產生的某些標頭。如需更多詳細資訊，請參閱 [the section called “新增 CloudFront 請求標頭”](#)。

Cookie

在檢視器請求中，CloudFront 會包含在快取金鑰和原始伺服器請求中的 Cookie。針對 Cookie，您可以選擇下列設定之一：

- 無 - 檢視器請求中的 Cookie 不會包含在快取金鑰中，也不會自動包含在原始伺服器請求中。
- 所有 - 檢視器請求中的所有 Cookie 都包含在快取金鑰中，並自動包含在原始伺服器請求中。
- Include specified cookies (包含指定的 Cookie) - 您可以指定檢視器請求中的哪些 Cookie 會包含在快取金鑰中，並自動包含在原始伺服器請求中。
- Include all cookies except (包含所有 Cookie，除了：) - 您可以指定檢視器請求中的哪些 Cookie 不會包含在快取金鑰中，且不會自動包含在原始伺服器請求中。所有其他 Cookie、您預期指定的 Cookie，都會包含在快取金鑰中，並自動包含在原始伺服器請求中。

當您使用 Include specified cookies (包含指定 Cookie) 或 Include all cookies except (包含所有 Cookie，除了：) 設定時，您可以依名稱指定 Cookie，而不是依值來指定 Cookie。例如，請參閱下列 Cookie 標頭：

```
Cookie: session_ID=abcd1234
```

在這種情況下，您可以將 Cookie 指定為 session_ID，而不是 session_ID=abcd1234。但是，CloudFront 會在快取金鑰和原始伺服器請求中包括完整的 Cookie，包括它的值。

查詢字串

在檢視器請求中，CloudFront 會包含在快取金鑰和原始伺服器請求中的 URL 查詢字串。對於查詢字串，您可以選擇下列其中一個設定：

- 無 - 檢視器請求中的查詢字串不會包含在快取金鑰中，也不會自動包含在原始伺服器的請求中。

- 所有 – 檢視器請求中的所有查詢字串都會包含在快取金鑰中，並且也會自動包含在原始伺服器請求中。
- Include specified query strings (包含指定的查詢字串) – 您可以指定檢視器請求中的哪些查詢字串會包含在快取金鑰中，並自動包含在原始伺服器請求中。
- Include all query strings except (包含所有查詢字串，除了：) – 您可以指定檢視器請求中的哪些查詢字串不會包含在快取金鑰中，且不會自動包含在原始伺服器請求中。所有其他查詢字串 (您指定的查詢字串) 都會包含在快取金鑰中，並自動包含在原始伺服器請求中。

當您使用 Include specified query strings (包含指定的查詢字串) 或 Include all query strings except (包含所有查詢字串，除了：) 設定時，您可以依名稱來指定查詢字串，而不是依值來指定查詢字串。例如，請參閱下列 URL 路徑。

```
/content/stories/example-story.html?split-pages=false
```

在這種情況下，您可以將查詢字串指定為 split-pages，而不是 split-pages=false。不過，在快取金鑰和原始伺服器請求中，CloudFront 會包含完整的查詢字串 (包括其值)。

 Note

對於快取金鑰設定，CloudFront 會將標頭、查詢字串和 Cookie 的星號字元 (*) 視為文字字串，而不是萬用字元。

壓縮支援

當檢視器支援 CloudFront 時，這些設定可以請求和快取以 Gzip 或 Brotli 壓縮格式壓縮的物件。這些設定也允許 [CloudFront 壓縮](#) 運作。檢視器透過 Accept-Encoding HTTP 標頭表示他們支援這些壓縮格式。

 Note

只有在使用 HTTPS 發送請求時，Chrome 和 Firefox 網頁瀏覽器才支援 Brotli 壓縮。這些瀏覽器不支援使用 HTTP 請求的 Brotli。

當下列任一條件成立時，請啟用這些設定：

- 當查看者支援它們時，您的原始伺服器返回 Gzip 壓縮物件 (請求包含帶有值 Accept-Encoding 的 gzip HTTP 標頭)。在此情況下，請使用已啟用 Gzip 設定 (在 CloudFront API、AWS SDK、AWS CLI 或 CloudFormation 中將 EnableAcceptEncodingGzip 設定為 true)。
- 當查看者支援它們時，您的原始伺服器返回 Brotli 壓縮物件 (請求包含帶有值 Accept-Encoding 的 br HTTP 標頭)。在此情況下，請使用已啟用 Brotli 設定 (在 CloudFront API、AWS SDK、AWS CLI 或 CloudFormation 中將 EnableAcceptEncodingBrotli 設定為 true)。
- 此快取政策連接到的快取行為會設定 [CloudFront 壓縮](#)。在這種情況下，您可以啟用 Gzip 或 Brotli 的快取，或兩者的快取。啟用 CloudFront 壓縮時，啟用兩種格式的快取可協助降低資料傳輸到網際網路的成本。

 Note

如果您針對其中一種或兩種壓縮格式啟用快取，請勿在與相同快取行為相關聯的[原始伺服器請求政策](#)中包含 Accept-Encoding 標頭。當針對這些格式啟用快取時，CloudFront 一律會在原始伺服器請求中包含此標頭，因此在原始伺服器請求政策中包含 Accept-Encoding 沒有任何作用。

如果您的原始伺服器未傳回 Gzip 或 Brotli 壓縮物件，或是快取行為未設定 CloudFront 壓縮，請勿啟用壓縮物件的快取功能。如果您這樣做，可能會導致[快取命中率](#)降低。

以下說明這些設定如何影響 CloudFront 分佈。下列所有案例假設檢視器請求會包含 Accept-Encoding 標頭。當檢視者請求不包含 Accept-Encoding 標頭時，CloudFront 不會在快取金鑰中包含此標頭，也不會將其包含在對應的原始伺服器請求中。

針對兩種壓縮格式啟用快取壓縮物件時

如果檢視器同時支援 Gzip 和 Brotli，亦即如果 gzip 和 br 值都在檢視器請求的 Accept-Encoding 標頭中，則 CloudFront 執行以下操作：

- 將標頭標準化為 Accept-Encoding: br, gzip，並在快取金鑰中包含標準化標頭。快取金鑰不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。
- 如果節點位置在快取中有符合請求且未過期的 Brotli 或 Gzip 壓縮物件，則節點位置會將物件傳回給檢視器。
- 如果邊緣節點在快取中沒有符合請求且未過期的 Brotli 或 Gzip 壓縮物件，則 CloudFront 會在對應的原始伺服器請求中包含標準化標頭 (Accept-Encoding: br, gzip)。原始伺服器請求不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。

如果檢視器支援一種壓縮格式，但不支援另一種壓縮格式——例如，如果 gzip 是檢視器請求中 Accept-Encoding 標頭中的值，但 br 不是——CloudFront 會有以下操作：

- 將標頭標準化為 Accept-Encoding: gzip，並在快取金鑰中包含標準化標頭。快取金鑰不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。
- 如果節點位置在快取中有符合請求且未過期的 Gzip 壓縮物件，則節點位置會將物件傳回給檢視器。
- 如果邊緣節點在快取中沒有符合請求且未過期的 Gzip 壓縮物件，則 CloudFront 會在對應的原始伺服器請求中包含標準化標頭 (Accept-Encoding: gzip)。原始伺服器請求不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。

若要瞭解如果檢視器支援 Brotli 但不支援 Gzip 時，CloudFront 會如何因應，請將前面例子中的兩種壓縮格式相互替換。

如果檢視器不支援 Brotli 或 Gzip——亦即檢視器請求中的 Accept-Encoding 標頭不包含 br 或 gzip 作為值——則 CloudFront：

- 不包含在快取金鑰中的 Accept-Encoding 標頭。
- 包含 Accept-Encoding: identity 在相應的原始伺服器請求中。原始伺服器請求不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。

針對某種壓縮格式啟用快取壓縮物件時，但不啟用另一種壓縮格式

如果檢視器支持啟用快取的格式——例如，如果為 Gzip 啟用了快取壓縮物件，並且檢視器支援 Gzip (gzip 是檢視器請求中 Accept-Encoding 標頭的值之一)——CloudFront 會執行以下操作：

- 將標頭標準化為 Accept-Encoding: gzip，並在快取金鑰中包含標準化標頭。
- 如果節點位置在快取中有符合請求且未過期的 Gzip 壓縮物件，則節點位置會將物件傳回給檢視器。
- 如果邊緣節點在快取中沒有符合請求且未過期的 Gzip 壓縮物件，則 CloudFront 會在對應的原始伺服器請求中包含標準化標頭 (Accept-Encoding: gzip)。原始伺服器請求不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。

當檢視器同時支援 Gzip 和 Brotli (檢視器請求中的 Accept-Encoding 標頭同時包含 gzip 和 br 值) 時，這種行為是相同的，因為在這種情況下，不會啟用快取 Brotli 的壓縮物件。

若要瞭解 CloudFront 如果已針對 Brotli 啟用快取壓縮物件，但未啟用 Gzip，請在前面的範例中將兩種壓縮格式取代。

如果檢視器不支援啟用快取的壓縮格式 (檢視器請求中的 Accept-Encoding 標頭不包含該格式的值)，則 CloudFront：

- 不包含在快取金鑰中的 Accept-Encoding 標頭。
- 包含 Accept-Encoding: identity 在相應的原始伺服器請求中。原始伺服器請求不包含檢視者傳送的 Accept-Encoding 標頭中的其他值。

針對兩種壓縮格式停用快取壓縮物件時

針對這兩種壓縮格式停用快取壓縮物件時，CloudFront 會將 Accept-Encoding 標頭視為檢視器請求中任何其他 HTTP 標頭相同。預設情況下，系統不會將其不會包含在快取金鑰中，也不包含在原始伺服器請求中。您可以將其包含在快取政策中的標頭清單或原始伺服器請求政策中，如同任何其他 HTTP 標頭。

建立快取政策

您可以使用快取政策藉由控制快取金鑰中包含的值 (URL 查詢字串、HTTP 標頭和 Cookie) 來改善快取命中率。您可以使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API 在 CloudFront 主控台中建立快取政策。

建立快取政策之後，您可以將它連接到 CloudFront 分佈中的一或多個快取行為。

Console

建立快取政策 (主控台)

1. 登入 AWS 管理主控台，然後在位於 <https://console.aws.amazon.com/cloudfront/v4/home?#/policies> 的 CloudFront 主控台中開啟 Policies (政策) 頁面。
2. 選擇建立快取政策。
3. 選擇此快取政策所需的設定。如需詳細資訊，請參閱 [了解快取政策](#)。
4. 完成時，請選擇 Create (建立)。

建立快取政策之後，您可以將其附加到快取行為。

若要將快取政策附加至現有分佈 (主控台)

1. 在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/distributions> 的 CloudFront 主控台中開啟 Distributions (分佈) 頁面。
2. 選擇要更新的分佈，然後選擇行為索引標籤。

3. 選擇要更新的快取行為，然後選擇編輯。

或者，若要建立新的快取行為，請選擇 Create behavior (建立行為)。

4. 在 Cache key and origin requests (快取金鑰和原始伺服器請求) 一節中，請確定已選擇 Cache policy and origin request policy (快取政策和原始伺服器請求政策)。
5. 在 Cache policy (快取政策) 中，選擇要連接到此快取行為的快取政策。
6. 請在頁面底部選擇 Save changes (儲存變更)。

若要將快取政策附加至新分佈 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 開啟 CloudFront 主控台。
2. 選擇 Create Distribution (建立分佈)。
3. 在 Cache key and origin requests (快取金鑰和原始伺服器請求) 一節中，請確定已選擇 Cache policy and origin request policy (快取政策和原始伺服器請求政策)。
4. 在 Cache policy (快取政策) 中，選擇要連接到此分佈預設快取行為的快取政策。
5. 為原始伺服器、預設快取行為和其他分佈設定選擇所需的設定。如需詳細資訊，請參閱 [所有分佈設定參考](#)。
6. 完成後，請選擇 Create distribution (建立分佈)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立快取政策，請使用 `aws cloudfront create-cache-policy` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不是將每個個別參數指定為命令列輸入。

建立快取政策 (包含輸入檔案的 CLI)

1. 使用下列命令建立一個命名為 `cache-policy.yaml` 的檔案，其中包含 `create-cache-policy` 命令的所有輸入參數。

```
aws cloudfront create-cache-policy --generate-cli-skeleton yaml-input > cache-policy.yaml
```

2. 開啟您剛才建立且命名為 `cache-policy.yaml` 的檔案。編輯檔案以指定您想要的快取政策設定，然後儲存檔案。您可以從檔案中移除選用欄位，但不要移除必要欄位。

如需有關快取政策設定的詳細資訊，請參閱 [了解快取政策](#)。

3. 使用下列命令，使用 `cache-policy.yaml` 檔案中的輸入參數建立快取政策。

```
aws cloudfront create-cache-policy --cli-input-yaml file://cache-policy.yaml
```

記下命令輸出中的 `Id` 值。這是快取政策 ID，而且您需要它將快取政策連接到 CloudFront 分佈的快取行為。

將快取政策附加至現有分佈 (包含輸入檔案的 CLI)

1. 使用下列命令來儲存您想要更新之 CloudFront 分發的分發組態。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml > dist-config.yaml
```

2. 開啟您剛才建立且命名為 `dist-config.yaml` 的檔案。編輯檔案，對您要更新為使用快取政策的每個快取行為進行下列變更。
 - 在快取行為中，新增名為 `CachePolicyId` 的欄位。對於欄位值，請使用您在建立政策後記下的快取政策 ID。
 - 從快取行為中移除 `MinTTL`、`MaxTTL`、`DefaultTTL` 和 `ForwardedValues` 欄位。這些設定是在快取政策中指定的，因此您無法在相同的快取行為中包含這些欄位和快取政策。
 - 將 `ETag` 欄位重新命名為 `IfMatch`，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈以使用快取政策。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://dist-config.yaml
```

若要將快取政策附加至新分佈 (包含輸入檔案的 CLI)

1. 使用下列命令建立一個命名為 `distribution.yaml` 的檔案，其中包含 `create-distribution` 命令的所有輸入參數。

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input >
distribution.yaml
```

2. 開啟您剛才建立且命名為 `distribution.yaml` 的檔案。在預設快取行為的 `CachePolicyId` 欄位中，輸入您在建立政策後記下的快取政策 ID。繼續編輯檔案以指定所需的分佈設定，然後在完成後儲存檔案。

如需有關分佈設定的詳細資訊，請參閱 [所有分佈設定參考](#)。

3. 使用下列命令，使用 `distribution.yaml` 檔案中的輸入參數建立分佈。

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

若要使用 CloudFront API 建立快取政策，請使用 [CreateCachePolicy](#)。如需有關您在此 API 呼叫中指定欄位的詳細資訊，請參閱 [了解快取政策](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考文件。

建立快取政策之後，您可以使用下列其中一個 API 呼叫，將其附加至快取行為：

- 若要將它附加到現有分佈中的快取行為，請使用 [UpdateDistribution](#)。
- 若要將它附加到新分佈中的快取行為，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請在快取行為中的 `CachePolicyId` 欄位中提供快取請求政策的 ID。如需您在這些 API 呼叫中指定其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考說明文件。

使用受管快取政策

CloudFront 提供一組受管快取政策，您可以將這些政策附加至分佈的任何快取行為。使用受管快取政策，您不需要撰寫或維護自己的快取政策。受管政策會使用針對特定使用案例最佳化的設定。

若要使用受管快取政策，請將其附加至分佈中的快取行為。此程序與您建立快取政策時的程序相同，但您只要附加其中一個受管快取政策，而不是建立新政策。您可以依名稱 (使用主控台) 或 ID (使用 AWS CLI 或軟體開發套件) 附加政策。名稱和 ID 會列在下一節中。

如需更多詳細資訊，請參閱 [建立快取政策](#)。

下列主題說明您可以使用的受管快取政策。

主題

- [Amplify](#)
- [CachingDisabled](#)
- [CachingOptimized](#)
- [CachingOptimizedForUncompressedObjects](#)
- [Elemental-MediaPackage](#)
- [UseOriginCacheControlHeaders](#)
- [UseOriginCacheControlHeaders-QueryString](#)

Amplify

[在 CloudFront 主控台中檢視此政策](#)

此政策是專為與 [AWS Amplify](#) Web 應用程式的原始伺服器搭配使用而設計。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

2e54312d-136d-493c-8eb9-b001f22f67d2

此政策包括以下設定：

- 最小 TTL：2 秒
- 最長 TTL：600 秒 (10 分鐘)
- 預設 TTL：2 秒
- 包含在快取金鑰中的標頭：
 - Authorization
 - CloudFront-Viewer-Country
 - Host

由於已啟用快取壓縮物件設定，因此也會包含標準化的 Accept-Encoding 標頭。如需詳細資訊，請參閱[壓縮支援](#)。

- 快取金鑰中包含的 Cookie：所有 Cookie 都包含在內。
- 快取金鑰中包含的查詢字串：包含所有查詢字串。
- 快取壓縮物件設定：已啟用。如需詳細資訊，請參閱[壓縮支援](#)。

Warning

由於此政策擁有大於 0 的最小 TTL，CloudFront 將快取內容至少達快取政策最短 TTL 中指定的持續時間，即使原始伺服器標頭中存在 Cache-Control: no-cache、no-store 或 private 指令。

AWS Amplify 託管快取政策

Amplify 使用以下受管快取政策來最佳化客戶應用程式的預設快取組態：

- [Amplify-Default](#)
- [Amplify-DefaultNoCookies](#)
- [Amplify-ImageOptimization](#)
- [Amplify-StaticContent](#)

Note

這些政策僅供 Amplify 使用。我們不建議您將這些政策用於分佈。

如需管理 Amplify 託管應用程式快取組態的詳細資訊，請參閱《Amplify 託管使用者指南》中的[管理快取組態](#)。

CachingDisabled

[在 CloudFront 主控台中檢視此政策](#)

此政策會停用快取。此政策對於動態內容和無法快取的請求非常有用。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

4135ea2d-6df8-44a3-9df3-4b5a84be39ad

此政策包括以下設定：

- 最小 TTL：0 秒
- 最長 TTL：0 秒
- 預設 TTL：0 秒
- 快取金鑰中包含的標頭：無
- 快取金鑰中包含的 Cookie：無
- 快取金鑰中包含的查詢字串：無
- 快取壓縮物件設定：停用

CachingOptimized

[在 CloudFront 主控台中檢視此政策](#)

此政策的設計目的是將 CloudFront 包含在快取金鑰中的值降至最低，以最佳化快取效率。CloudFront 不會在快取金鑰中包含任何查詢字串或 Cookie，而且只會包含標準化的 Accept-Encoding 標頭。這可讓 CloudFront 在原點傳回物件或啟用 [CloudFront 邊緣壓縮](#)時，分別以 Gzip 和 Brotli 壓縮格式快取物件。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

658327ea-f89d-4fab-a63d-7e88639e58f6

此政策包括以下設定：

- 最小 TTL：1 秒。
- 最長 TTL：31,536,000 秒 (365 天)。
- 預設 TTL：86,400 秒 (24 小時)。
- 快取金鑰中包含的標頭：沒有明確包含任何標頭。包含標準化的 Accept-Encoding 標頭，因為已啟用快取壓縮物件設定。如需詳細資訊，請參閱[壓縮支援](#)。
- 快取金鑰中包含的 Cookie：無。
- 快取金鑰中包含的查詢字串：無。
- 快取壓縮物件設定：已啟用。如需詳細資訊，請參閱[壓縮支援](#)。

 Warning

由於此政策擁有大於 0 的最小 TTL，CloudFront 將快取內容至少達快取政策最短 TTL 中指定的持續時間，即使原始伺服器標頭中存在 Cache-Control: no-cache、no-store 或 private 指令。

CachingOptimizedForUncompressedObjects

[在 CloudFront 主控台中檢視此政策](#)

此政策的設計目的是將快取金鑰中包含的值降至最低，以最佳化快取效率。不包括查詢字串、標頭或 Cookie。此政策與前一個政策相同，但會停用快取壓縮物件設定。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

b2884449-e4de-46a7-ac36-70bc7f1ddd6d

此政策包括以下設定：

- 最小 TTL：1 秒
- 最長 TTL：31,536,000 秒 (365 天)
- 預設 TTL：86,400 秒 (24 小時)
- 快取金鑰中包含的標頭：無
- 快取金鑰中包含的 Cookie：無
- 快取金鑰中包含的查詢字串：無
- 快取壓縮物件設定：停用

 Warning

由於此政策擁有大於 0 的最小 TTL，CloudFront 將快取內容至少達快取政策最短 TTL 中指定的持續時間，即使原始伺服器標頭中存在 Cache-Control: no-cache、no-store 或 private 指令。

Elemental-MediaPackage

[在 CloudFront 主控台中檢視此政策](#)

此政策是專為與 AWS Elemental MediaPackage 端點的原始伺服器搭配使用而設計。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

08627262-05a9-4f76-9ded-b50ca2e3a84f

此政策包括以下設定：

- 最小 TTL：0 秒
- 最長 TTL：31,536,000 秒 (365 天)
- 預設 TTL：86,400 秒 (24 小時)
- 快取金鑰中包含的標頭：
 - Origin

由於已為 Gzip 啟用快取壓縮物件設定，所以也會包含標準化的 Accept-Encoding 標頭。如需詳細資訊，請參閱[壓縮支援](#)。

- 快取金鑰中包含的 Cookie：無
- 快取金鑰中包含的查詢字串：
 - aws.manifestfilter
 - start
 - end
 - m
- 快取壓縮物件設定：已為 Gzip 啟用。如需詳細資訊，請參閱[壓縮支援](#)。

UseOriginCacheControlHeaders

[在 CloudFront 主控台中檢視此政策](#)

此政策旨在與傳回 Cache-Control HTTP 回應標頭的原始伺服器搭配使用，且不會根據查詢字串中存在的值提供不同的內容。如果您的原始伺服器根據查詢字串中的值提供不同的內容，請考慮使用[UseOriginCacheControlHeaders-QueryString](#)。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

83da9c7e-98b4-4e11-a168-04f0df8e2c65

此政策包括以下設定：

- 最小 TTL : 0 秒
- 最長 TTL : 31,536,000 秒 (365 天)
- 預設 TTL : 0 秒
- 快取金鑰中包含的標頭 :
 - Host
 - Origin
 - X-HTTP-Method-Override
 - X-HTTP-Method
 - X-Method-Override

由於已啟用快取壓縮物件設定，因此也會包含標準化的 Accept-Encoding 標頭。如需詳細資訊，請參閱[壓縮支援](#)。

- 快取金鑰中包含的 Cookie : 所有 Cookie 都包含在內。
- 快取金鑰中包含的查詢字串 : 無。
- 快取壓縮物件設定 : 已啟用。如需詳細資訊，請參閱[壓縮支援](#)。

UseOriginCacheControlHeaders-QueryStrings

[在 CloudFront 主控台中檢視此政策](#)

此政策旨在與傳回 Cache-Control HTTP 回應標頭的原始伺服器搭配使用，並根據查詢字串中存在的值提供不同的內容。如果您的原始伺服器未根據查詢字串中的值提供不同的內容，請考慮使用[UseOriginCacheControlHeaders](#)。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

4cc15a8a-d715-48a4-82b8-cc0b614638fe

此政策包括以下設定：

- 最小 TTL : 0 秒
- 最長 TTL : 31,536,000 秒 (365 天)
- 預設 TTL : 0 秒
- 快取金鑰中包含的標頭 :
 - Host

- Origin
- X-HTTP-Method-Override
- X-HTTP-Method
- X-Method-Override

由於已啟用快取壓縮物件設定，因此也會包含標準化的 Accept-Encoding 標頭。如需詳細資訊，請參閱[壓縮支援](#)。

- 快取金鑰中包含的 Cookie：所有 Cookie 都包含在內。
- 快取金鑰中包含的查詢字串：包含所有查詢字串。
- 快取壓縮物件設定：已啟用。如需詳細資訊，請參閱[壓縮支援](#)。

了解快取金鑰

快取金鑰會決定對 CloudFront 節點的檢視器請求是否會導致快取命中。快取金鑰是快取中物件的唯一識別碼。快取中的每個物件都有唯一的快取金鑰。

當檢視器請求產生與先前請求相同的快取金鑰，且該快取金鑰的物件位於節點的快取中且有效時，就會發生快取命中。當有快取命中時，請求的物件會從 CloudFront 節點提供給檢視器，這具有下列優點：

- 降低原始伺服器的負載
- 減少檢視器的延遲

當您的快取命中率 較高 (造成快取命中率較高的檢視器請求比例較高時)，您可以從網站或應用程式獲得較佳的效能。改善快取命中率的一種方法是，只在快取金鑰中包含必要的最小值。如需詳細資訊，請參閱下列區段。

您可以使用[快取政策](#)來修改快取金鑰中的值 (URL 查詢字串、HTTP 標頭和 Cookie)。(您也可以在檢視器請求上使用 [Lambda@Edgefunction](#) 或 [CloudFront Function](#) 來修改快取金鑰。) 在修改快取金鑰之前，請務必瞭解應用程式的設計方式，以及根據檢視器請求的特性提供不同的回應的時間和方式。當查看器請求中的值確定您的來源返回的回應時，您應該在快取金鑰中包含該值。但是，如果您在快取金鑰中包含一個不影響您來源返回的回應值，則最終可能會快取重複的物件。

預設快取金鑰

根據預設，CloudFront 分佈的快取金鑰包含下列資訊：

- CloudFront 分佈的網域名稱 (例如 d111111abcdef8.cloudfront.net)

- 請求物件的 URL 路徑 (例如 `/content/stories/example-story.html`)

 Note

OPTIONS 方法中包含在 OPTIONS 請求的快取金鑰中。這表示，OPTIONS 請求的回應會與 GET 和 HEAD 請求的回應分開快取。

依預設，來自檢視者請求的其他值不會包含在快取金鑰中。考慮從 Web 瀏覽器下面的 HTTP 請求。

```
GET /content/stories/example-story.html?ref=0123abc&split-pages=false
HTTP/1.1
Host: d111111abcdef8.cloudfront.net
User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/68.0
Accept: text/html, */*
Accept-Language: en-US,en
Cookie: session_id=01234abcd
Referer: https://news.example.com/
```

當像此範例的檢視者請求進入 CloudFront 邊緣節點時，CloudFront 會使用快取金鑰來判斷是否存在快取命中。根據預設，只有下列請求元件包含在快取金鑰中：`/content/stories/example-story.html` 和 `d111111abcdef8.cloudfront.net`。如果請求的物件不在快取中 (快取遺漏)，則 CloudFront 會將請求發送到原點以獲取物件。取得物件後，CloudFront 會將其傳回檢視器，並將其儲存在節點的快取中。

當 CloudFront 收到相同物件的另一個請求時，由快取金鑰決定，會立即將快取物件 CloudFront 提供給檢視器，而不會將請求傳送給原始物件。例如，請考慮下列的 HTTP 請求在先前的請求之後進入。

```
GET /content/stories/example-story.html?ref=xyz987&split-pages=true
HTTP/1.1
Host: d111111abcdef8.cloudfront.net
User-Agent: Mozilla/5.0 AppleWebKit/537.36 Chrome/83.0.4103.116
Accept: text/html, */*
Accept-Language: en-US,en
Cookie: session_id=wxyz9876
Referer: https://rss.news.example.net/
```

此請求適用於與先前請求相同的物件，但與先前請求不同。它具有不同的 URL 查詢字串，不同的 User-Agent 和 Referer 標頭，以及不同的 session_id Cookie。但是，根據預設，這些值都不是快取金鑰的一部分，因此第二個請求會導致快取命中。

自訂快取金鑰

在某些情況下，您可能想要在快取金鑰中包含更多資訊，即使這樣做可能會導致較少的快取點擊次數。您可以使用[快取政策](#)指定要包含在快取金鑰中的內容。

例如，如果您的原始伺服器使用檢視器請求中的 Accept-Language HTTP 標頭，根據檢視器的語言傳回不同的內容，您可能需要將此標頭包含在快取金鑰中。當您這樣做時，CloudFront 會使用此標頭來確定快取命中，並在原始伺服器請求中包含標頭 (當有快取遺漏時，CloudFront 發送到原始伺服器請求的請求)。

在快取金鑰中包含其他值的可能一種結果是，最終 CloudFront 可能會因為檢視器請求中可能發生的變化而快取重複的物件。例如，檢視者可能會針對 Accept-Language 標頭傳送下列任何值：

- en-US, en
- en, en-US
- en-US, en
- en-US

所有這些不同的值都表示檢視器的語言是英文，但變化可能會導致 CloudFront 多次快取相同的物件。這可以減少快取命中並增加原始伺服器請求的數目。您可以避免這種重複，方法是不要在快取金鑰中包含 Accept-Language 標頭，而是將網站或應用程式設定為針對不同語言的內容使用不同的 URL (例如 /en-US/content/stories/example-story.html)。

對於您想要包含在快取金鑰中的任何指定值，您應該確定您瞭解該值的多少種不同變化可能會出現在檢視器請求中。對於某些請求值，將它們包含在快取金鑰中很少具有意義。例如，User-Agent 標頭可以有數千個獨特的變化，所以通常不會是要包含在快取金鑰中的候選項。具有使用者特定或工作階段特定值，以及在數千個 (甚至數百萬個) 請求中是唯一的 Cookie，也不適合包含快取金鑰的候選項。如果您在快取金鑰中包含這些值，每個唯一變化都會產生快取中物件的另一個副本。如果物件的這些副本不是唯一的，或者如果您最終得到如此大量稍微不同的物件，每個物件只會得到少量的快取命中，您可能需要考慮不同的方法。您可以從快取金鑰中排除這些高變數值，也可以將物件標示為不可快取。

自訂快取金鑰時請小心。有時候這是可取的，但它可能會產生意想不到的後果，例如快取重複的物件，降低快取命中率以及增加原始伺服器請求的數量。如果您的來源網站或應用程式需要從檢視器請求接收

分析、遙測或其他用途的特定值，但這些值不會變更來源傳回的物件，請使用[原始伺服器請求政策](#)將這些值包含在原始伺服器請求中，但不會將它們包含在快取金鑰中。

使用政策控制原始伺服器請求

當檢視器請求 CloudFront 導致快取遺漏 (請求的物件未在節點位置快取) 時，CloudFront 會將請求傳送至原始伺服器以擷取物件。這就是所謂的原始伺服器請求。原始伺服器請求會永遠包含來自檢視器請求的以下資訊：

- URL 路徑 (僅路徑，不含 URL 查詢字串或網域名稱)
- 請求內文 (如果有)
- CloudFront 自動納入在每個原始伺服器請求 (包括 Host、User-Agent 和 X-Amz-Cf-Id) 中的 HTTP 標頭

依預設，來自檢視器請求的其他資訊，例如 URL 查詢字串、HTTP 標頭和 Cookie，不會包含在原始伺服器請求中。(例外：使用舊版快取設定時，CloudFront 預設會將標頭轉送至您的原始伺服器。) 但是，您可能想要在原始伺服器接收其中一些其他資訊，例如收集資料以供分析或遙測。您可以使用原始伺服器請求政策來控制原始伺服器請求中包含的資訊。

原始伺服器請求政策與控制[快取金鑰的快取政策](#)分開。如此可讓您在原始伺服器接收其他資訊，並保持良好的快取命中率 (造成快取命中的檢視器請求比例)。您可以單獨控制原始伺服器請求中包含哪些資訊 (使用原始伺服器請求政策) 以及包含在快取金鑰中 (使用快取政策) 來執行此操作。

雖然這兩種政策彼此獨立，但實際上彼此相關聯。您包含在快取金鑰 (使用快取政策) 中的所有 URL 查詢字串、HTTP 標頭和 Cookie 都會自動包含在原始伺服器請求中。使用原始伺服器請求政策，指定您要包含在原始伺服器請求中，但不包含在快取金鑰中的資訊。就像快取原則一樣，您可以將原始伺服器請求原則附加到 CloudFront 分發中的一或多個快取行為。

您也可以使用原始伺服器請求政策，將其他 HTTP 標頭新增至未包含在檢視器請求中的原始伺服器請求。這些額外的標頭是由傳送原始伺服器請求 CloudFront 之前新增的，標頭值會根據檢視器請求自動確定。如需更多詳細資訊，請參閱 [the section called “新增 CloudFront 請求標頭”](#)。

主題

- [了解原始伺服器請求政策](#)
- [建立原始伺服器請求政策](#)
- [使用受管原始伺服器請求政策](#)
- [新增 CloudFront 請求標頭](#)
- [了解原始伺服器請求政策和快取政策如何協同運作](#)

了解原始伺服器請求政策

CloudFront 會針對常見使用案例提供一些預先定義的原始伺服器請求原則 (稱為受管原則)。您可以使用這些受管政策，也可以根據您的需求建立自己的原始伺服器請求政策。如需有關受管政策的詳細資訊，請參閱 [使用受管原始伺服器請求政策](#)。

原始伺服器請求政策包含下列設定，這些設定分類為政策資訊和原始伺服器請求設定。

政策資訊

名稱

用來識別原始伺服器請求政策的唯一名稱。在主控台中，您可以使用名稱將原始伺服器請求政策附加到快取行為。

描述

描述原始伺服器請求政策的備註。這是選用的。

原始伺服器請求設定

原始伺服器請求設定會指定檢視器請求中的值，這些值包含在 CloudFront 發送給原始伺服器的請求中 (稱為原始伺服器請求)。這些值可以包括 URL 查詢字串、HTTP 標頭和 Cookie。您指定的值會包含在原始伺服器請求中，但不會包含在快取金鑰中。如需控制快取金鑰的資訊，請參閱 [使用政策控制快取金鑰](#)。

標頭

在檢視器請求中，CloudFront 會包含在原始伺服器請求中的 HTTP 標頭。針對標頭，您可以選擇下列設定之一：

- 無 – 檢視器請求中的 HTTP 標頭不包含在原始伺服器請求中。
- 所有檢視器標頭 – 檢視器請求中的所有 HTTP 標頭都會包含在原始伺服器請求中。
- All viewer headers and the following CloudFront headers (所有檢視器標頭和下列 CloudFront 標頭) – 檢視器請求中的所有 HTTP 標頭都會包含在原始伺服器請求中。此外，您可以指定要新增至原始伺服器請求的 CloudFront 標頭。如需 CloudFront 標頭的詳細資訊，請參閱 [the section called “新增 CloudFront 請求標頭”](#)。
- Include the following headers (包含下列標頭) – 您可以指定原始伺服器請求中要包含哪些 HTTP 標頭。

 Note

請勿指定已包含在 原始伺服器自訂標頭 設定中的標頭。如需更多詳細資訊，請參閱 [設定 CloudFront 以將自訂標頭新增到原始伺服器請求](#)。

- 所有檢視器標頭，除了 – 您指定不包含在原始伺服器請求中的 HTTP 標頭。檢視器要求中的所有其他 HTTP 標頭都會包含在內，但指定的標頭除外。

當您使用 所有檢視器標頭和下列 CloudFront 標頭、包含下列標頭 或 所有檢視器標頭，除了..... 設定時，您將僅依標頭名稱指定 HTTP 標頭。在原始伺服器請求中，CloudFront 會包含完整標頭，包括其值。

 Note

當您使用 所有檢視器標頭，除了..... 設定以移除檢視器的 Host 標頭時，CloudFront 會向原始伺服器請求新增一個具有原始伺服器網域名稱的新 Host 標頭。

Cookie

在原始伺服器請求中，CloudFront 會包含在原始伺服器請求中的 Cookie。針對 Cookie，您可以選擇下列設定之一：

- 無 – 檢視器請求中的 Cookie 不包含在原始伺服器請求中。
- 所有 – 檢視器請求中的所有 Cookie 都會包含在原始伺服器請求中。
- 包含指定的 Cookie – 您可以指定在原始伺服器請求中納入哪些檢視器請求的 Cookie。
- 所有 Cookie，除了 – 在原始伺服器請求中，您指定哪些 Cookie 不包含在檢視器請求中。檢視器請求中的所有其他 Cookie 都會包含在內。

在您使用 包含指定的 Cookie 或 所有 Cookie，除了..... 設定時，您只會依名稱指定 Cookie。在原始伺服器請求中，CloudFront 會包含完整的 Cookie，包括其值。

查詢字串

在檢視器請求中，CloudFront 包含在原始伺服器請求中的 URL 查詢字串。對於查詢字串，您可以選擇下列其中一個設定：

- 無 – 檢視器請求中的查詢字串不包含在原始伺服器請求中。
- 所有 – 檢視器請求中的查詢字串都包含在原始伺服器請求中。

- 包含指定的查詢字串 – 您可以指定原始伺服器請求中要納入檢視器請求中的哪些查詢字串。
- 所有查詢字串，除了 – 在原始伺服器請求中，您指定哪些查詢字符串不包含在檢視器請求中。包括所有其他查詢字串。

在您使用 包含指定的查詢字串 或 所有查詢字串，除了..... 設定時，您只會依名稱來指定查詢字串。在原始伺服器請求中，CloudFront 會包含完整的查詢字串 (包括其值)。

建立原始伺服器請求政策

您可以使用原始伺服器請求原則來控制 CloudFront 傳送至原始伺服器請求中包含的值 (URL 查詢字串、HTTP 標頭和 Cookie)。您可以使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API 在 CloudFront 主控台中建立原始伺服器請求原則。

建立原始伺服器請求原則後，您可以將其附加到 CloudFront 分發中的一或多個快取行為。

不需要原始伺服器請求政策。當快取行為未附加原始伺服器請求政策時，原始伺服器請求會包含[快取政策](#)中指定的所有值，但僅此而已。

Note

若要使用原始伺服器請求政策，快取行為也必須使用[快取政策](#)。在沒有快取政策的情況下，您無法在快取行為中使用原始伺服器請求政策。

Console

建立原始伺服器請求政策 (主控台)

1. 登入 AWS 管理主控台，然後在位於 <https://console.aws.amazon.com/cloudfront/v4/home?#/policies> 的 CloudFront 主控台中開啟 Policies (政策) 頁面。
2. 選擇 Origin request (原始伺服器請求)，然後選擇 Create origin request policy (建立原始伺服器請求政策)。
3. 選擇此原始伺服器請求政策所需的設定。如需詳細資訊，請參閱 [了解原始伺服器請求政策](#)。
4. 完成時，請選擇 Create (建立)。

建立原始伺服器請求政策後，您可以將其附加至快取行為。

將原始伺服器請求政策附加到現有分佈 (主控台)

1. 在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/distributions> 的 CloudFront 主控台中開啟 Distributions (分佈) 頁面。
2. 選擇要更新的分佈，然後選擇行為索引標籤。
3. 選擇要更新的快取行為，然後選擇編輯。

或者，若要建立新的快取行為，請選擇 Create behavior (建立行為)。

4. 在 Cache key and origin requests (快取金鑰和原始伺服器請求) 一節中，請確定已選擇 Cache policy and origin request policy (快取政策和原始伺服器請求政策)。
5. 針對 Origin request policy (原始伺服器請求政策)，請選擇要連接至此快取行為的原始伺服器請求政策。
6. 請在頁面底部選擇 Save changes (儲存變更)。

將原始伺服器請求政策附加到新分佈 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 開啟 CloudFront 主控台。
2. 選擇 Create Distribution (建立分佈)。
3. 在 Cache key and origin requests (快取金鑰和原始伺服器請求) 一節中，請確定已選擇 Cache policy and origin request policy (快取政策和原始伺服器請求政策)。
4. 對於 Origin request policy (原始伺服器請求政策)，請選擇要連接至此分佈預設快取行為的原始伺服器請求政策。
5. 為原始伺服器、預設快取行為和其他分佈設定選擇所需的設定。如需詳細資訊，請參閱 [所有分佈設定參考](#)。
6. 完成後，請選擇 Create distribution (建立分佈)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立原始伺服器請求原則，請使用 `aws cloudfront create-origin-request-policy` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不是將每個個別參數指定為命令列輸入。

建立原始伺服器請求政策 (包含輸入檔案的 CLI)

1. 使用下列命令建立一個命名為 `origin-request-policy.yaml` 的檔案，其中包含 `create-origin-request-policy` 命令的所有輸入參數。

```
aws cloudfront create-origin-request-policy --generate-cli-skeleton yaml-input >
origin-request-policy.yaml
```

2. 開啟您剛才建立且命名為 `origin-request-policy.yaml` 的檔案。編輯檔案以指定您想要的原始伺服器請求政策設定，然後儲存檔案。您可以從檔案中移除選用欄位，但不要移除必要欄位。

如需有關原始伺服器請求原則設定的詳細資訊，請參閱 [了解原始伺服器請求政策](#)。

3. 使用下列命令，使用 `origin-request-policy.yaml` 檔案中的輸入參數建立原始伺服器請求政策。

```
aws cloudfront create-origin-request-policy --cli-input-yaml file://origin-
request-policy.yaml
```

記下命令輸出中的 `Id` 值。這是原始伺服器請求原則 ID，您需要它將原始伺服器請求原則附加到 CloudFront 分佈的快取行為。

若要將原始伺服器請求政策附加至現有分佈 (包含輸入檔案的 CLI)

1. 使用下列命令來儲存您想要更新之 CloudFront 分佈的分佈組態。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >
dist-config.yaml
```

2. 開啟您剛才建立且命名為 `dist-config.yaml` 的檔案。編輯檔案，對您要更新的每個快取行為進行下列變更，以使用原始伺服器請求政策。
 - 在快取行為中，新增名為 `OriginRequestPolicyId` 的欄位。對於欄位值，請使用您在建立政策後記下的原始伺服器請求政策 ID。
 - 將 `ETag` 欄位重新命名為 `IfMatch`，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用原始伺服器請求政策。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://dist-config.yaml
```

若要將原始伺服器請求政策連接至新分佈 (包含輸入檔案的 CLI)

1. 使用下列命令建立一個命名為 `distribution.yaml` 的檔案，其中包含 `create-distribution` 命令的所有輸入參數。

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input > distribution.yaml
```

2. 開啟您剛才建立且命名為 `distribution.yaml` 的檔案。在預設快取行為的 `OriginRequestPolicyId` 欄位中，輸入您在建立政策後記下的原始伺服器請求政策 ID。繼續編輯檔案以指定所需的分佈設定，然後在完成後儲存檔案。

如需有關分佈設定的詳細資訊，請參閱 [所有分佈設定參考](#)。

3. 使用下列命令，使用 `distribution.yaml` 檔案中的輸入參數建立分佈。

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

若要使用 CloudFront API 建立原始伺服器請求原則，請使用 [CreateOriginRequestPolicy](#)。如需有關您在此 API 呼叫中指定欄位的詳細資訊，請參閱 [了解原始伺服器請求政策](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考文件。

建立原始伺服器請求政策後，您可以使用下列其中一個 API 呼叫，將其附加至快取行為：

- 若要將它附加到現有分佈中的快取行為，請使用 [UpdateDistribution](#)。
- 若要將它附加到新分佈中的快取行為，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請在快取行為中的 `OriginRequestPolicyId` 欄位中提供原始伺服器請求政策的 ID。如需您在這些 API 呼叫中指定其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考說明文件。

使用受管原始伺服器請求政策

CloudFront 提供一組受管原始伺服器請求原則，您可以將這些原則附加到分佈的任何快取行為上。使用受管原始伺服器請求政策，您不需要編寫或維護自己的原始伺服器請求政策。受管政策會使用針對特定使用案例最佳化的設定。

若要使用受管原始伺服器請求政策，請將其附加到分佈中的快取行為。此程序與您建立原始伺服器請求政策時的程序相同，但您只需附加其中一個受管原始伺服器請求政策，而不是建立新的原始伺服器請求政策。您可以依名稱 (使用主控台) 或 ID (使用 AWS CLI 或軟體開發套件) 附加政策。名稱和 ID 會列在下一節中。

如需更多詳細資訊，請參閱 [建立原始伺服器請求政策](#)。

下列主題說明您可以使用的受管原始伺服器請求政策。

主題

- [AllViewer](#)
- [AllViewerAndCloudFrontHeaders-2022-06](#)
- [AllViewerExceptHostHeader](#)
- [CORS-CustomOrigin](#)
- [CORS-S3Origin](#)
- [Elemental-MediaTailor-PersonalizedManifests](#)
- [HostHeaderOnly](#)
- [UserAgentRefererHeaders](#)

AllViewer

[在 CloudFront 主控台中檢視此政策](#)

此政策包含檢視器請求中的所有值 (標頭、Cookie 和查詢字串)。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

216adef6-5c7f-47e4-b989-5492eafa07d3

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：檢視器請求中的所有標頭
- 原始伺服器請求中包含的 Cookie：所有
- 原始伺服器請求中包含的查詢字串：全部

AllViewerAndCloudFrontHeaders-2022-06

[在 CloudFront 主控台中檢視此政策](#)

此政策包含檢視者請求的所有值 (標頭、Cookies 和查詢字串)，以及 2022 年 6 月前發行的所有 [CloudFront 標頭](#) (不包含 2022 年 6 月後發行的 CloudFront 標頭)。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

33f36d7e-f396-46d9-90e0-52428a34d9dc

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：檢視者請求中的所有標頭和下列 CloudFront 標頭：
 - CloudFront-Forwarded-Proto
 - CloudFront-Is-Android-Viewer
 - CloudFront-Is-Desktop-Viewer
 - CloudFront-Is-IOS-Viewer
 - CloudFront-Is-Mobile-Viewer
 - CloudFront-Is-SmartTV-Viewer
 - CloudFront-Is-Tablet-Viewer
 - CloudFront-Viewer-Address
 - CloudFront-Viewer-ASN
 - CloudFront-Viewer-City
 - CloudFront-Viewer-Country
 - CloudFront-Viewer-Country-Name
 - CloudFront-Viewer-Country-Region
 - CloudFront-Viewer-Country-Region-Name

- CloudFront-Viewer-Http-Version
- CloudFront-Viewer-Latitude
- CloudFront-Viewer-Longitude
- CloudFront-Viewer-Metro-Code
- CloudFront-Viewer-Postal-Code
- CloudFront-Viewer-Time-Zone
- CloudFront-Viewer-TLS
- 原始伺服器請求中包含的 Cookie：所有
- 原始伺服器請求中包含的查詢字串：全部

AllViewerExceptHostHeader

[在 CloudFront 主控台中檢視此政策](#)

此政策不包含來自檢視者請求的 Host 標頭，但會包含檢視器請求中的所有其他值 (標頭、Cookie 和查詢字串)。

此政策也包含 HTTP 通訊協定、HTTP 版本、TLS 版本，以及所有裝置類型和檢視器位置標頭的其他 [CloudFront 請求標頭](#)。

此政策旨在與 Amazon API Gateway 和 AWS Lambda 函數 URL 原始伺服器搭配使用。這些原始伺服器預期 Host 標頭包含原始伺服器網域名稱，而不是 CloudFront 分發的網域名稱。將檢視者請求中的 Host 標頭轉寄至這些原始伺服器可能會導致它們無法正常運作。

Note

當您使用此受管原始伺服器請求政策移除檢視器的 Host 標頭時，CloudFront 會將包含原始伺服器網域名稱的新 Host 標頭新增至原始伺服器請求。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

b689b0a8-53d0-40ab-baf2-68738e2966ac

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：檢視器請求中的所有標頭 (Host 標頭除外)

- 原始伺服器請求中包含的 Cookie：所有
- 原始伺服器請求中包含的查詢字串：全部

CORS-CustomOrigin

[在 CloudFront 主控台中檢視此政策](#)

此政策包含在原始伺服器為自訂原始伺服器時，啟用跨原始伺服器資源共用 (CORS) 請求的標頭。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

59781a5b-3903-41f3-afcb-af62929ccde1

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：
 - Origin
- 原始伺服器請求中包含的 Cookie：無
- 原始伺服器請求中包含的查詢字串：無

CORS-S3Origin

[在 CloudFront 主控台中檢視此政策](#)

此原則包括當原始伺服器資源 Amazon S3 儲存貯體時，啟用跨原始伺服器資源共用 (CORS) 請求的標頭。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

88a5eaf4-2fd4-4709-b370-b4c650ea3fcf

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：
 - Origin
 - Access-Control-Request-Headers
 - Access-Control-Request-Method
- 原始伺服器請求中包含的 Cookie：無

- 原始伺服器請求中包含的查詢字串：無

Elemental-MediaTailor-PersonalizedManifests

[在 CloudFront 主控台中檢視此政策](#)

此政策專為搭配本身是 AWS Elemental MediaTailor 端點的原始伺服器使用而設計。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

775133bc-15f2-49f9-abea-afb2e0bf67d2

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：
 - Origin
 - Access-Control-Request-Headers
 - Access-Control-Request-Method
 - User-Agent
 - X-Forwarded-For
- 原始伺服器請求中包含的 Cookie：無
- 原始伺服器請求中包含的查詢字串：全部

HostHeaderOnly

[在 CloudFront 主控台中檢視此政策](#)

此政策僅包含來自原始伺服器的 Host 標頭。它不包括任何查詢字串或 Cookie。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

bf0718e1-ba1e-49d1-88b1-f726733018ae

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：Host
- 原始伺服器請求中包含的 Cookie：無

- 原始伺服器請求中包含的查詢字串：無

UserAgentRefererHeaders

[在 CloudFront 主控台中檢視此政策](#)

此政策只包含 User-Agent 和 Referer 標頭。它不包括任何查詢字串或 Cookie。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

acba4595-bd28-49b8-b9fe-13317c0390fa

此政策包括以下設定：

- 原始伺服器請求中包含的標頭：
 - User-Agent
 - Referer
- 原始伺服器請求中包含的 Cookie：無
- 原始伺服器請求中包含的查詢字串：無

新增 CloudFront 請求標頭

您可以設定 CloudFront，將特定的 HTTP 標頭新增到 CloudFront 從檢視者接收到的請求，並轉送至您的原始伺服器或[邊緣函數](#)。這些 HTTP 標頭的值皆基於檢視者請求的特性。標頭提供有關檢視器的裝置類型、IP 位址、地理位置、請求通訊協定 (HTTP 或 HTTPS)、HTTP 版本、TLS 連線詳細內容以及[JA3 指紋](#)與 JA4 指紋。您也可以設定分佈的快取行為來轉送 WebSocket 標頭。如需更多詳細資訊，請參閱[使用 WebSockets 搭配 CloudFront 分佈](#)。

有了這些標題，您的原始伺服器或邊緣函數即可接收有關檢視器的資訊，而不需要您編寫自己的程式碼來判斷此資訊。如果您的原始伺服器根據這些標頭中的資訊傳回不同的回應，則可以將它們包含在快取金鑰中，以便 CloudFront 單獨快取不同的回應。例如，您的原始伺服器可能會根據檢視器所在國家/地區使用特定語言的內容進行回應，或者使用針對特定裝置類型訂製的內容。您的原始伺服器還可能會將這些標頭寫入日誌檔案，您可以使用這些檔案來確定有關檢視器所在位置、檢視器所在的裝置類型等資訊。

如果您想要在快取金鑰中包含標頭，請使用快取政策。如需詳細資訊，請參閱[使用政策控制快取金鑰](#)及[the section called “了解快取金鑰”](#)。

要在原始伺服器中接收這些表頭，但不將它們包含在快取金鑰中，請使用原始伺服器請求政策。如需更多詳細資訊，請參閱 [使用政策控制原始伺服器請求](#)。

主題

- [裝置類型標頭](#)
- [檢視器位置標頭](#)
- [用於判斷檢視器標頭結構的標頭](#)
- [TLS 相關標頭](#)
- [其他 CloudFront 標題](#)

裝置類型標頭

可以新增下列標頭來判斷檢視者的裝置類型。根據 User-Agent 標頭的值，CloudFront 將這些標頭的值設定為 true 或 false。如果裝置屬於多個類別，一個以上的值可以是 true。例如，針對一些平板電腦裝置，CloudFront 將 CloudFront-Is-Mobile-Viewer 與 CloudFront-Is-Tablet-Viewer 同時設定為 true。

- CloudFront-Is-Android-Viewer - 當 CloudFront 判斷檢視器為具有 Android 作業系統的裝置時，設定為 true。
- CloudFront-Is-Desktop-Viewer - 當 CloudFront 判斷檢視器為桌上型裝置時，設定為 true。
- CloudFront-Is-iOS-Viewer - 當 CloudFront 判斷檢視器為具有 Apple 行動作業系統的裝置 (例如 iPhone、iPod Touch 以及一些 iPad 裝置) 時，設定為 true。
- CloudFront-Is-Mobile-Viewer - 當 CloudFront 判斷檢視器為行動裝置時，設定為 true。
- CloudFront-Is-SmartTV-Viewer - 當 CloudFront 判斷檢視器為智慧型電視時，設定為 true。
- CloudFront-Is-Tablet-Viewer - 當 CloudFront 判斷檢視器為平板電腦時，設定為 true。

檢視器位置標頭

可以新增下列標頭來判斷檢視者的位置。CloudFront 會根據檢視器的 IP 位址來決定這些標頭的值。對於這些標頭值中的非 ASCII 字元，CloudFront 會根據 [RFC 3986 的 1.2 節](#) 對字元進行百分比編碼。

- CloudFront-Viewer-Address - 包含檢視者 IP 位址以及請求的來源連接埠，例如標頭值 198.51.100.10:46532 表示瀏覽器的 IP 位址是 198.51.100.10，請求來源連接埠是 46532。
- CloudFront-Viewer-ASN - 包含檢視器的自治系統編號 (ASN)。

 Note

CloudFront-Viewer-Address 和 CloudFront-Viewer-ASN 可以在原始伺服器請求政策中新增，而不是在快取政策中新增。

- CloudFront-Viewer-Country – 包含檢視器國家/地區的兩個字母國家/地區代碼。如需國家/地區代碼的清單，請參閱 [ISO 3166-1 alpha-2](#)。
- CloudFront-Viewer-City - 包含檢視器所在城市的名稱。

當您新增下列標頭時，CloudFront 會將這些標頭套用至所有請求，但不會套用至源自 AWS 網路的請求：

- CloudFront-Viewer-Country-Name – 包含檢視器所在國家/地區的名稱。
- CloudFront-Viewer-Country-Region – 包含代表檢視器區域的代碼 (最多三個字元)。該區域是 [ISO 3166-2](#) 代碼的第一層細分 (即最廣泛或最不具體的一層)。
- CloudFront-Viewer-Country-Region-Name – 包含檢視器區域的名稱。該區域是 [ISO 3166-2](#) 代碼的第一層細分 (即最廣泛或最不具體的一層)。
- CloudFront-Viewer-Latitude – 包含檢視器的約略緯度。
- CloudFront-Viewer-Longitude – 包含檢視器的約略經度。
- CloudFront-Viewer-Metro-Code – 包含檢視器的地鐵代碼。只有當檢視器在美國時，才會出現此問題。
- CloudFront-Viewer-Postal-Code – 包含檢視器的郵遞區號。
- CloudFront-Viewer-Time-Zone 包含檢視器的時區，採用 [IANA 時區資料庫格式](#) (例如，America/Los_Angeles)。

 Note

CloudFront-Viewer-City、CloudFront-Viewer-Metro-Code 和 CloudFront-Viewer-Postal-Code 可能無法用於每個 IP 位址。部分 IP 位址無法以足夠的特異性進行地理位置定位以取得該資訊。

用於判斷檢視器標頭結構的標頭

您可以新增下列標頭，協助根據檢視者傳送的標頭來識別檢視者。例如，不同的瀏覽器可能會以特定順序傳送 HTTP 標頭。若 User-Agent 標頭中指定的瀏覽器與該瀏覽器的預期標頭順序不同，您可以拒絕該請求。此外，若 CloudFront-Viewer-Header-Count 值與 CloudFront-Viewer-Header-Order 中的標頭數量不同，您也可以拒絕該請求。

- CloudFront-Viewer-Header-Order – 按要求的順序包含檢視者的標頭名稱，並以冒號分隔。例如：CloudFront-Viewer-Header-Order: Host:User-Agent:Accept:Accept-Encoding。超出 7,680 字元限制的標頭會被截斷。
- CloudFront-Viewer-Header-Count – 包含檢視者標頭的總數。

TLS 相關標頭

您可以新增下列標頭，以判斷檢視器的 JA3 指紋、JA4 指紋和 TLS 連線詳細資料：

- CloudFront-Viewer-JA3-Fingerprint – 包含檢視者的 [JA3 指紋](#)。JA3 指紋可協助您判斷請求是否來自已知用戶端、惡意軟體或惡意機器人，或預期的 (允許清單中的) 應用程式。
- CloudFront-Viewer-JA4-Fingerprint – 包含檢視者的 JA4 指紋。與 JA3 指紋相似，[JA4 指紋](#)可協助您判斷請求是否來自已知用戶端、惡意軟體或惡意機器人，或預期的 (允許清單中的) 應用程式。您可以使用指紋來建置已知良好和不良行為者的資料庫，以便在檢查 HTTP 請求時套用。然後，您可以在應用程式 Web 伺服器或 [Lambda@Edge](#) 和 [CloudFront Functions](#) 中檢查標頭值，將標頭值與已知惡意軟體指紋清單進行比較，以封鎖惡意用戶端。
- CloudFront-Viewer-TLS – 包含 SSL/TLS 版本、密碼以及有關用於檢視器和 CloudFront 之間連線的 SSL/TLS 交換的詳細資訊。標頭值的格式如下：

```
SSL/TLS_version:cipher:handshake_information
```

對於 *handshake_information*，標頭可包含下列值：

- fullHandshake – 已針對 SSL/TLS 工作階段進行完整交換。
- sessionResumed – 之前的 SSL/TLS 工作階段已恢復。
- connectionReused – 之前的 SSL/TLS 連線已重複使用。

下列是此標頭的一些範例值：

```
TLSv1.3:TLS_AES_128_GCM_SHA256:sessionResumed
```

```
TLSv1.2:ECDHE-ECDSA-AES128-GCM-SHA256:connectionReused
```

```
TLSv1.1:ECDHE-RSA-AES128-SHA256:fullHandshake
```

```
TLSv1:ECDHE-RSA-AES256-SHA:fullHandshake
```

有關此標頭值中可能存在的 SSL/TLS 版本和密碼的完整列表，請參閱 [the section called “檢視器和 CloudFront 之間支援的通訊協定和密碼”](#)。

備註

- JA3 和 JA4 指紋衍生自 SSL/TLS Client Hello 封包。它們僅適用於 HTTPS 請求。
- 對於這些與 TLS 相關的標頭，您可以將它們新增至[原始伺服器請求政策](#)，而不是在[快取政策](#)中。

其他 CloudFront 標題

您可以新增下列標頭，以判斷檢視器的原始請求 URI、原始請求查詢字串參數和值、通訊協定和版本：

- CloudFront-Error-Uri – 包含從檢視器收到的原始請求 URI。
- CloudFront-Error-Args – 包含原始請求查詢字串參數和值。
- CloudFront-Forwarded-Proto – 包含檢視器請求的通訊協定 (HTTP 或 HTTPS)。
- CloudFront-Viewer-Http-Version – 包含檢視器請求的 HTTP 版本。

了解原始伺服器請求政策和快取政策如何協同運作

您可以使用 CloudFront [原始伺服器請求政策](#) 來控制 CloudFront 傳送到原始伺服器的請求，這些請求稱為原始伺服器請求。若要使用原始伺服器請求政策，您必須連接[快取政策](#)至相同的快取行為。在沒有快取政策的情況下，您無法在快取行為中使用原始伺服器請求政策。如需更多詳細資訊，請參閱 [使用政策控制原始伺服器請求](#)。

原始伺服器請求政策和快取政策共同運作，以確定 CloudFront 包含在原始伺服器請求中的值。您包含在快取金鑰 (使用快取政策) 中的所有 URL 查詢字串、HTTP 標頭和 Cookie 都會自動包含在原始伺服

器請求中。您在原始伺服器請求政策中指定的任何其他查詢字串、標頭和 Cookie 都會包含在原始伺服器請求中 (但不會包含在快取金鑰中)。

原始請求政策和快取政策具有可能彼此衝突的設定。例如，一個政策可能允許某些值，而另一個政策會封鎖它們。下表說明當您同時使用原始伺服器請求政策和快取政策的設定時，CloudFront 包含在原始伺服器請求中的哪些值。這些設定通常適用於所有類型的值 (查詢字串、標頭和 Cookie)，但您無法在快取政策中指定所有標頭或使用標頭封鎖清單的例外。

原始伺服器請求政策				
	無	全部。	允許清單	封鎖清單
快取政策				
無	除了每個原始伺服器請求中包含的預設值以外，來源請求中不會包含任何來自檢視器要求的值。如需更多詳細資訊，請參閱 使用政策控制原始伺服器請求 。	來自檢視器請求的所有值都包含在原始伺服器請求中。	只有在原始伺服器請求政策中指定的值才會包含在原始伺服器請求中。	除了原始伺服器請求政策中指定的值以外，所有來自檢視器請求的值都包含在原始伺服器請求中。
全部。 注意：您無法在快取政策中指定所有標頭。	來自檢視器要求的所有查詢字串和 Cookie 都包含在原始伺服器請求中。	來自檢視器請求的所有值都包含在原始伺服器請求中。	來自檢視器要求的所有查詢字串和 Cookie，以及原始伺服器請求政策中指定的任何標頭，都包含在原始伺服器請求中。	來自檢視器要求的所有查詢字串和 Cookie 都包含在原始伺服器請求中，且包含在原始伺服器請求政策封鎖清單中指定的字串和 Cookie。快取政策設定會覆寫原始伺服器請求政策封鎖清單。

	原始伺服器請求政策			
	無	全部。	允許清單	封鎖清單
允許清單	原始伺服器請求中只包含來自檢視器請求的指定值。	來自檢視器請求的所有值都包含在原始伺服器請求中。	快取政策或原始伺服器請求政策中指定的所有值都包含在原始伺服器請求中。	即使原始伺服器請求政策封鎖清單中指定了相同的值，快取政策中指定的值也會包含在原始伺服器請求中。快取政策允許清單會覆寫原始要求政策封鎖清單。
封鎖清單 注意：您無法在快取政策封鎖清單中指定標頭。	來自檢視器要求的所有查詢字串和 Cookie (指定的字串除外) 都包含在原始伺服器請求中。	來自檢視器請求的所有值都包含在原始伺服器請求中。	即使在快取政策封鎖清單中指定了相同的值，原始伺服器請求政策中指定的值也會包含在原始伺服器請求中。原始伺服器請求政策允許清單會覆寫快取政策封鎖清單。	來自檢視器請求的所有值 (快取政策或原始伺服器請求政策中指定的值除外) 都會包含在原始伺服器請求中。

在 CloudFront 回應中使用政策新增或移除 HTTP 標頭

您可以設定 CloudFront，藉此修改傳送給檢視器之回應的 HTTP 標頭 (Web 瀏覽器及其他用戶端)。CloudFront 可以移除從原始伺服器接收的標頭，或在回應中新增標頭，再將回應傳送給檢視者。且不須編寫程式碼或變更來源，即可進行這些變更。

例如，您可以移除標頭 (例如 X-Powered-By 和 Vary)，如此一來，CloudFront 就不會在傳送給檢視者的回應中包含這些標頭。或者，您可以新增如下所示的 HTTP 標頭：

- 新增 Cache-Control 標頭以控制瀏覽器快取。
- 新增 Access-Control-Allow-Origin 標頭以啟用跨原始來源資源分享 (CORS)。您也可以新增其他 CORS 標頭。
- 新增一組常見的安全性標頭，例如 Strict-Transport-Security、Content-Security-Policy、X-Frame-Options 等。
- 新增 Server-Timing 標頭，以查看與透過 CloudFront 傳送的請求和回應相關的效能和路由資訊。

若要指定 CloudFront 移除或新增至 HTTP 回應的標頭，請使用 response headers policy (回應標頭政策)。將回應標頭政策連接到更多快取行為，CloudFront 會修改回應中的標頭，這些回應針對符合快取行為的請求傳送。CloudFront 會修改從快取中提供的回應以及 CloudFront 從原始伺服器轉送的回應之標頭。如果原始回應包含回應標頭政策中新增的一或多個標頭，則政策可以指定 CloudFront 應使用從原始伺服器接收的標頭，還是使用政策中的標頭覆寫該標頭。

Note

如果您將控制瀏覽器快取的標頭新增至回應標頭政策 (例如 Cache-Control)，CloudFront 只會將這些標頭新增至檢視器回應。這些標頭不會影響 CloudFront 快取請求物件的方式。

CloudFront 會針對常見使用案例提供一些預先定義的回應標頭政策 (稱為受管政策)。您可以[使用這些受管政策](#)，也可以建立您專屬的政策。您可以將單個回應標頭政策連接至 AWS 帳戶 中多個分配內的多個快取行為。

如需詳細資訊，請參閱下列內容：

主題

- [了解回應標頭政策](#)
- [建立回應標頭政策](#)

- [使用受管回應標頭政策](#)

了解回應標頭政策

您可以使用回應標頭政策指定 Amazon CloudFront 對傳送給檢視者的回應所移除或新增的 HTTP 標頭。如需回應標頭政策及其使用原因的詳細資訊，請參閱[使用政策新增或移除回應標頭](#)。

下列主題說明回應標頭政策中的設定。這些設定分為多個類別，下列主題中依次介紹這些類別。

主題

- [政策詳細資訊 \(中繼資料\)](#)
- [CORS 標頭](#)
- [安全性標頭](#)
- [自訂標頭](#)
- [移除標頭](#)
- [Server-Timing 標頭](#)

政策詳細資訊 (中繼資料)

政策詳細資訊設定包含回應標頭政策的中繼資料。

- 名稱 - 用來識別回應標頭政策的名稱。在主控台中，您可以使用名稱將政策連接至快取行為。
- 描述(選用) - 用來描述回應標頭政策的註解。這是選用的設定，但它可以協助您識別政策的目的。

CORS 標頭

跨來源資源分享 (CORS) 設定可讓您在回應標頭政策中新增和設定 CORS 標頭。

此清單著重於如何在回應標頭政策中指定設定和有效值。如需進一步瞭解這些標頭以及如何實際將標頭用於 CORS 請求和回應，請參閱 MDN Web Docs 網站上的[跨原始來源資源分享](#)與[CORS 通訊協定規格](#)。

Access-Control-Allow-Credentials

這是布林值設定 (true 或 false)，用於決定 CloudFront 是否要對 CORS 請求的回應新增 Access-Control-Allow-Credentials 標頭。如這項設定為 true，CloudFront 就會對

CORS 請求的回應新增 `Access-Control-Allow-Credentials: true` 標頭，反之則不會將此標頭新增至回應。

Access-Control-Allow-Headers

指定 CloudFront 用作 CORS 預檢請求回應中 `Access-Control-Allow-Headers` 標頭值的標頭名稱。此設定的有效值包括 HTTP 標頭名稱或萬用字元 (*)，用以表示所有標頭皆受允許。

 **Note**

Authorization 標頭無法使用萬用字元，且必須明確列出。

萬用字元的有效使用範例

範例	相符	不相符
x-amz-*	x-amz-test	x-amz
	x-amz-	
x-*-amz	x-test-amz	
	x--amz	
*	除了 Authorization 之外的所有標頭	Authorization

Access-Control-Allow-Methods

指定 CloudFront 用作 CORS 預檢請求回應中 `Access-Control-Allow-Methods` 標頭值的 HTTP 方法。有效值包含 GET、DELETE、HEAD、OPTIONS、PATCH、POST、PUT、ALL。ALL 是包含所有已列出 HTTP 方法的特殊值。

Access-Control-Allow-Origin

指定 CloudFront 可以在 `Access-Control-Allow-Origin` 回應標頭中使用的值。此設定的有效值包括特定原始伺服器 (例如 `http://www.example.com`)，或者代表允許所有原始伺服器的萬用字元 (*)。

備註

- 允許在子網域最左邊 (*.example.org) 使用萬用字元 (*)。
- 下列位置不允許使用萬用字元 (*) :
 - 頂層網域 (example.*)
 - 子網域 (test.*.example.org) 右側或任何子網域 (*test.example.org) 內
 - 條款內部 (exa*mples.org)

如需使用萬用字元的範例，請參閱下表。

範例	相符	不相符
http://*.example.org	http://www.example.org http://test.example.org	https://test.example.org https://test.example.org:123 http://test.example.org:123
*.example.org	test.example.org test.test.example.org .example.org http://test.example.org https://test.example.org	http://test.example.org:123 https://test.example.org:123
example.org	http://example.org https://example.org	

範例	相符	不相符
<code>http://example.org</code>		<code>https://example.org</code> <code>http://example.org:123</code>
<code>http://example.org:*</code>	<code>http://example.org:123</code> <code>http://example.org</code>	
<code>http://example.org:1*3</code>	<code>http://example.org:123</code> <code>http://example.org:1893</code> <code>http://example.org:13</code>	
<code>*.example.org:1*</code>	<code>test.example.org:123</code>	

Access-Control-Expose-Headers

指定 CloudFront 用作 CORS 請求回應中 Access-Control-Expose-Headers 標頭值的標頭名稱。此設定的有效值包括 HTTP 標頭名稱或萬用字元 (*)。

Access-Control-Max-Age

CloudFront 用作 CORS 請求回應中 Access-Control-Max-Age 標頭值的秒數。

覆寫原始伺服器

布林值設定，可決定當來自原始伺服器的回應包含其中一個也在政策中的 CORS 標頭時 CloudFront 的行為。

- 設定為 `true`，且原始伺服器回應包含列於政策中的 CORS 標頭時，CloudFront 會將政策中的 CORS 標頭新增至回應。然後 CloudFront 會將該回應傳送給檢視器。CloudFront 會忽略從原始伺服器收到的標頭。

- 設定為 `false`，且原始伺服器回應包含 CORS 標頭時 (無論 CORS 標頭是否列於政策中)，CloudFront 會將從原始伺服器收到的 CORS 標頭包含在回應中。CloudFront 不會將政策中的任何 CORS 標頭新增至傳送給檢視器的回應。

安全性標頭

安全性標頭設定可讓您在回應標頭政策中新增及設定數個與安全性相關的 HTTP 回應標頭。

這份清單會說明如何在回應標頭政策中指定設定和有效值，如需進一步瞭解各個標頭以及如何實際將標頭用於 HTTP 回應，請參閱 MDN Web Docs 網站上的相關連結。

Content-Security-Policy

指定 CloudFront 用作 Content-Security-Policy 回應標頭值的內容安全性政策指令。

如需此標頭和有效政策指令的詳細資訊，請參閱 MDN Web 文件中的 [Content-Security-Policy](#)。

Note

Content-Security-Policy 標頭值限制為 1783 個字元。

推薦網站政策

指定 CloudFront 用作 Referrer-Policy 回應標頭值的推薦網站政策指令。此設定的有效值為 `no-referrer`、`no-referrer-when-downgrade`、`origin`、`origin-when-cross-origin`、`same-origin`、`strict-origin`、`strict-origin-when-cross-origin`、`unsafe-url`。

如需此標頭和這些指令的詳細資訊，請參閱 MDN Web 文件中的 [Referrer-Policy](#)。

Strict-Transport-Security

指定 CloudFront 用作 Strict-Transport-Security 回應標頭值的指令和設定。對於此設定，您可以分別指定：

- CloudFront 用作此標頭的 `max-age` 指令值的秒數
- `preload` 的布林值設定 (`true` 或 `false`)，用於決定 CloudFront 是否要在此標頭的值中納入 `preload` 指令
- `includeSubDomains` 的布林值設定 (`true` 或 `false`)，用於決定 CloudFront 是否要在此標頭的值中納入 `includeSubDomains` 指令

如需此標頭和這些指令的詳細資訊，請參閱 MDN Web 文件中的 [Strict-Transport-Security](#)。

X-Content-Type-Options

這是布林值設定 (true 或 false)，用於決定 CloudFront 是否要將 X-Content-Type-Options 標頭新增至回應。當此設定為 true 時，CloudFront 會將 X-Content-Type-Options: nosniff 標頭新增至回應。反之則不會新增此標頭。

如需此標頭的詳細資訊，請參閱 MDN Web 文件中的 [X-Content-Type-Options](#)。

X-Frame-Options

指定 CloudFront 用作 X-Frame-Options 回應標頭值的指令。此設定的有效值為 DENY 或 SAMEORIGIN。

如需此標頭和這些指令的詳細資訊，請參閱 MDN Web 文件中的 [X-Frame-Options](#)。

X-XSS-Protection

指定 CloudFront 用作 X-XSS-Protection 回應標頭值的指令和設定。對於此設定，您可以分別指定：

- 0 (禁用 XSS 篩選) 或 1 (啟用 XSS 篩選) 的 X-XSS-Protection 設定
- block 的布林值設定 (true 或 false)，用於決定 CloudFront 是否要在此標頭的值中納入 mode=block 指令
- 報告 URI，用於決定 CloudFront 是否要在此標頭的值中納入 report=*reporting URI* 指令

您可以將 block 指定為 true，也可以指定報告 URI，但不能同時指定兩者。如需此標頭和這些指令的詳細資訊，請參閱 MDN Web 文件中的 [X-XSS-Protection](#)。

覆寫原始伺服器

每個這些安全性標頭設定均包含一個布林值設定 (true 或 false)，可決定當來自原始伺服器的回應包含此標頭時 CloudFront 的行為。

當此設定為 true，且原始伺服器回應包含此標頭時，CloudFront 會將政策中的標頭新增至傳送給檢視者的回應，同時忽略從原始伺服器收到的標頭。

當此設定為 false，且原始伺服器回應包含此標頭時，CloudFront 會在傳送給檢視者的回應中納入從原始伺服器收到的標頭。

如原始伺服器回應不包含此標頭，CloudFront 會將政策中的標頭新增至傳送給檢視者的回應，無論此設定為 true 還是 false 都一樣。

自訂標頭

自訂標頭設定可讓您在回應標頭政策中新增和設定自訂 HTTP 標頭，CloudFront 會將這些標頭新增至它傳回給檢視者的每個回應中。您可以為每個自訂標頭指定值，但不指定也沒關係，因為 CloudFront 可以新增沒有值的回應標頭。

每個自訂標頭也有自己的原始伺服器覆寫設定：

- 當此設定為 `true`，且原始伺服器回應包含列於政策中的自訂標頭時，CloudFront 會將政策中的自訂標頭新增至傳送給檢視者的回應中，同時忽略從原始伺服器收到的標頭。
- 當此設定為 `false`，且原始伺服器回應包含列於政策中的自訂標頭時，CloudFront 會在傳送給檢視者的回應中納入從原始伺服器收到的自訂標頭。
- 如原始伺服器回應不包含列於政策中的自訂標頭，CloudFront 會將政策中的自訂標頭新增至傳送給檢視者的回應，無論此設定為 `true` 還是 `false` 都一樣。

移除標頭

您可以指定要讓 CloudFront 從原始伺服器接收到的回應中移除的標頭，如此一來，CloudFront 傳送給檢視者的回應就不會包含該標頭。CloudFront 會從傳送給檢視者的每個回應中移除標頭，無論物件是從 CloudFront 的快取或從原始伺服器提供。例如，您可以移除不使用瀏覽器的標頭 (例如 `X-Powered-By` 或 `Vary`)，以便讓 CloudFront 從傳送給檢視者的回應中移除這些標頭。

當您使用回應標頭政策指定要移除的標頭時，CloudFront 會先移除標頭，然後新增在回應標頭政策其他區段 (CORS 標頭、安全標頭、自訂標頭等) 中指定的任何標頭。若您指定了要移除的標頭，但同時在政策的另一個區段新增相同的標頭，CloudFront 會在傳送給檢視者的回應中包含該標頭。

Note

您可以使用回應標頭政策移除 CloudFront 從原始伺服器接收的 `Server` 和 `Date` 標頭，如此一來，這些從原始伺服器接收的標頭就不會包含在 CloudFront 傳送給檢視者的回應中。不過，若採取此操作，CloudFront 會在傳送給檢視者的回應中新增自有的標頭版本。對於 CloudFront 新增的 `Server` 標頭，該標頭的值為 `CloudFront`。

您無法移除的標頭

您無法使用回應標頭政策移除下列標頭。如果您在回應標頭政策 (API 中的 `ResponseHeadersPolicyRemoveHeadersConfig`) 的 `Remove headers (移除標頭)` 區段中指定這些標頭，就會收到錯誤訊息。

- `Connection`
- `Content-Encoding`
- `Content-Length`
- `Expect`
- `Host`
- `Keep-Alive`
- `Proxy-Authenticate`
- `Proxy-Authorization`
- `Proxy-Connection`
- `Trailer`
- `Transfer-Encoding`
- `Upgrade`
- `Via`
- `Warning`
- `X-Accel-Buffering`
- `X-Accel-Charset`
- `X-Accel-Limit-Rate`
- `X-Accel-Redirect`
- `X-Amz-Cf-.*`
- `X-Amzn-Auth`
- `X-Amzn-Cf-Billing`
- `X-Amzn-Cf-Id`
- `X-Amzn-Cf-Xff`
- `X-Amzn-ErrorType`
- `X-Amzn-Fle-Profile`

- X-Amzn-Header-Count
- X-Amzn-Header-Order
- X-Amzn-Lambda-Integration-Tag
- X-Amzn-RequestId
- X-Cache
- X-Edge-.*
- X-Forwarded-Proto
- X-Real-Ip

Server-Timing 標頭

使用 Server-Timing 標頭設定啟用 CloudFront 所傳送的 HTTP 回應中的 Server-Timing 標頭。您可以使用這個標頭檢視有助於洞察 CloudFront 和原始伺服器行為和效能的指標。例如，您可以查看哪個快取層出現了快取命中，或是在發生快取遺漏時，查看來自原始伺服器的第一個位元組延遲時間。Server-Timing 標頭中的指標可以幫助您排解問題，或者測試 CloudFront 或原始伺服器組態的效率。

如需進一步瞭解透過 CloudFront 使用 Server-Timing 標頭，請參閱下列主題。

若要啟用 Server-Timing 標頭，請[建立 \(或編輯\) 回應標頭政策](#)。

主題

- [取樣率和 Pragma 請求標頭](#)
- [來自原始伺服器的 Server-Timing 標頭](#)
- [Server-Timing 標頭指標](#)
- [Server-Timing 標頭範例](#)

取樣率和 Pragma 請求標頭

您在回應標頭政策中啟用 Server-Timing 標頭後，還可以指定抽樣率。抽樣率是介於 0 至 100 之間 (包含 0 與 100) 的數值，用於指定 CloudFront 要新增 Server-Timing 標頭的回應數百分比。如將抽樣率設為 100，則任何 HTTP 回應只要符合回應標頭政策所連接的快取行為，CloudFront 一律會新增 Server-Timing 標頭；如將抽樣率設為 50，則 CloudFront 會對符合快取行為的請求中 50% 的回應新增此標頭。您可以將抽樣率設定為 0 至 100 之間的任何數值，最多可精確到小數點後四位數字。

當抽樣率設定為低於 100 的數值時，您無法控管 CloudFront 會對哪些回應新增 Server-Timing 標頭，只能掌握有多少百分比的回應會加上標頭。不過，假如您為 HTTP 請求新增 Pragma 標頭，並將值設定為 server-timing，那麼該請求的回應便會一定收到 Server-Timing 標頭，無論抽樣率設為多少都一樣；即使抽樣率設為零 (0)，只要是包含 Pragma: server-timing 標頭的請求，CloudFront 一律會對其回應加上 Server-Timing 標頭。

來自原始伺服器的 Server-Timing 標頭

發生快取遺漏且 CloudFront 轉送請求到原始伺服器時，原始伺服器可能包含 Server-Timing 標頭在其對 CloudFront 的回應中。在這種情況下，CloudFront 會新增其[指標](#)到從原始伺服器收到的 Server-Timing 標頭。CloudFront 傳送給檢視器的回應包含單一回應 Server-Timing 標頭，其中包含來自原始伺服器的值以及 CloudFront 新增的指標。來自原始伺服器的標頭值可能位於結尾，或介於 CloudFront 新增至標頭的兩組指標之間。

發生快取命中時，CloudFront 傳送給檢視器的回應會包含單一 Server-Timing 標頭，其標頭值中僅包含 CloudFront 指標 (不包含來自原始伺服器的值)。

Server-Timing 標頭指標

如 CloudFront 對 HTTP 回應新增了 Server-Timing 標頭，標頭的值會包含一或多項指標，可協助您獲得 CloudFront 和原始伺服器行為及效能相關的洞察。以下清單包含所有指標及其可能的值；Server-Timing 標頭僅含有其中一些指標，實際情況取決於 CloudFront 收到的請求和提供的回應所具備的性質。

Server-Timing 標頭中的部分指標僅有名稱 (沒有值)，有些指標則有名稱和一個值。如果指標具有值，名稱和值之間會以半形分號分隔 (;)；假如標頭包含多個指標，則各個指標之間會以半形逗號分隔 (,)。

cdn-cache-hit

CloudFront 提供了來自快取的回應，沒有向原始伺服器提出請求。

cdn-cache-refresh

CloudFront 向原始伺服器提出請求，確認快取物件仍然有效後，提供了來自快取的回應。在這種情況下，CloudFront 並未從原始伺服器擷取完整物件。

cdn-cache-miss

CloudFront 沒有提供來自快取的回應。在這種情況下，CloudFront 在傳回回應之前，先向原始伺服器請求過完整物件。

cdn-pop

包含一個值，用於描述哪個 CloudFront 連接點 (POP) 負責處理請求。

cdn-rid

包含一個值，為該請求的 CloudFront 唯一識別符。透過 支援 排解問題時，您可以使用這個請求標識符 (RID)

cdn-hit-layer

當 CloudFront 在沒有向原始伺服器提出請求的情況下提供了來自快取的回應，就會出現這個指標。包含以下其中一個值：

- EDGE – CloudFront 提供了來自 POP 位置的快取回應。
- REC – CloudFront 提供了來自 [區域節點快取](#) (REC) 位置的快取回應。
- Origin Shield – CloudFront 提供了做為 [Origin Shield](#) 的 REC 傳送的快取回應。

cdn-upstream-layer

如 CloudFront 向原始伺服器請求了完整物件，就會出現這個指標；指標會包含下列其中一個值：

- EDGE – POP 位置將請求直接傳送給了原始伺服器。
- REC – REC 位置將請求直接傳送給了原始伺服器。
- Origin Shield – 做為 [Origin Shield](#) 的 REC 將請求直接傳送給了原始伺服器。

cdn-upstream-dns

包含一個值，用於表示從原始伺服器擷取 DNS 記錄所花費的毫秒數。假如該值為零 (0)，代表 CloudFront 使用的是快取 DNS 結果，或者重複使用了現有連線。

cdn-upstream-connect

包含一個值，用於表示完成原始伺服器 DNS 請求以及 TCP (與 TLS，如有) 完成原始伺服器連線之間所花費的毫秒數。假如該值為零 (0)，代表 CloudFront 重複使用了現有連線。

cdn-upstream-fbl

包含一個值，用於表示完成原始伺服器 HTTP 請求以及來自原始伺服器的回應收到第一個位元組之間所花費的毫秒數 (第一個位元組延遲時間)。

cdn-downstream-fbl

包含一個值，用於表示邊緣節點完成收到請求以及將回應的第一個位元組傳送給檢視者之間所花費的毫秒數。

Server-Timing 標頭範例

下列範例顯示 Server-Timing 標頭設定啟用後，檢視者可能從 CloudFront 收到的 Server-Timing 標頭。

Example – 快取遺漏

下列範例顯示當 CloudFront 快取中沒有請求的物件，檢視者可能會收到的 Server-Timing 標頭。

```
Server-Timing: cdn-upstream-layer;desc="EDGE",cdn-upstream-dns;dur=0,cdn-upstream-connect;dur=114,cdn-upstream-fbl;dur=177,cdn-cache-miss,cdn-pop;desc="PHX50-C2",cdn-rid;desc="yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg==",cdn-downstream-fbl;dur=436
```

此 Server-Timing 標頭表示以下內容：

- 原始伺服器請求是從 CloudFront 連接點 (POP) 位置傳送的 (cdn-upstream-layer;desc="EDGE")。
- CloudFront 使用了原始伺服器的快取 DNS 結果 (cdn-upstream-dns;dur=0)。
- CloudFront 花了 114 毫秒才完成與原始伺服器的 TCP (及 TLS，如有) 連線 (cdn-upstream-connect;dur=114)。
- 完成請求後，CloudFront 花了 177 毫秒才從原始伺服器收到回應的第一個位元組 (cdn-upstream-fbl;dur=177)。
- CloudFront 的快取中沒有請求的物件 (cdn-cache-miss)。
- 請求是由 PHX50-C2 代碼識別的節點接收的 (cdn-pop;desc="PHX50-C2")。
- 此請求的 CloudFront 唯一 ID 是 yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg== (cdn-rid;desc="yNPsyYn7skvTzwWkq3Wcc8Nj_foxUjQUe9H1ifslzWhb0w7aLbFvGg==")。
- CloudFront 在收到檢視器要求後，將回應的第一個位元組傳送給檢視器，花了 436 毫秒 (cdn-downstream-fbl;dur=436)。

Example – 快取命中

下列範例顯示當 CloudFront 快取中含有請求的物件，檢視者可能會收到的 Server-Timing 標頭。

```
Server-Timing: cdn-cache-hit,cdn-pop;desc="SEA19-C1",cdn-rid;desc="nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9di0peVc7xsrlKj-g==",cdn-hit-layer;desc="REC",cdn-downstream-fbl;dur=137
```

此 Server-Timing 標頭表示以下內容：

- 快取中含有請求的物件 (cdn-cache-hit)。
- 請求是由 SEA19-C1 代碼識別的節點接收的 (cdn-pop;desc="SEA19-C1")。
- 此請求的 CloudFront 唯一 ID 是 nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9di0peVc7xsrLKj-g== (cdn-rid;desc="nQBz4aJU2kP9iC3KHEq7vFxfMozu-VYBwGzkW9di0peVc7xsrLKj-g==")。
- 請求的物件是在區域節點快取 (REC) 位置進行快取的 (cdn-hit-layer;desc="REC")。
- CloudFront 在收到檢視器要求後，將回應的第一個位元組傳送給檢視器，花了 137 毫秒 (cdn-downstream-fbl;dur=137)。

建立回應標頭政策

您可以使用回應標頭政策來指定 Amazon CloudFront 移除或新增至 HTTP 回應的 HTTP 標頭。如需回應標頭政策及其使用原因的詳細資訊，請參閱[使用政策新增或移除回應標頭](#)。

您可以在 CloudFront 主控台中建立回應標頭政策，也可以使用 AWS CloudFormation、AWS Command Line Interface (AWS CLI) 或 CloudFront API 建立政策。建立回應標頭政策之後，您可以將它連接到 CloudFront 分佈中的一或多個快取行為。

建立自訂回應標頭政策之前，請先確認是否有任何[受管回應標頭政策](#)適合您的使用案例；如果有，您就可以將該政策連接到您的快取行為，而不需自行建立或管理回應標頭政策。

Console

建立回應標頭政策 (主控台)

1. 登入 AWS 管理主控台，然後前往位於 <https://console.aws.amazon.com/cloudfront/v4/home#/policies/responseHeaders> 的 CloudFront 主控台，進入 Policies (政策) 頁面中的 Response headers (回應標頭) 索引標籤。
2. 選擇 Create response headers policy (建立回應標頭政策)。
3. 在 Create response headers policy (建立回應標頭政策) 表單中執行下列動作：
 - a. 在 Details (詳細資訊) 面板中，輸入回應標頭政策的 Name (名稱) 和 (選用) 解釋政策用途的 Description (描述)。
 - b. 在 Cross-origin resource sharing (CORS) (跨來源資源分享 (CORS)) 面板中，選擇 Configure CORS (設定 CORS) 切換開關，並設定您要新增至政策的任何 CORS 標頭。如

果您希望設定的標頭覆寫 CloudFront 從原始伺服器接收的標頭，請選取 Origin override (原始伺服器覆寫) 核取方塊。

如需 CORS 標頭設定的詳細資訊，請參閱 [the section called “CORS 標頭”](#)。

- c. 在 Security headers (安全性標頭) 面板中，選擇切換開關並設定您要新增至政策的每個安全性標頭。

如需安全性標頭設定的詳細資訊，請參閱 [the section called “安全性標頭”](#)。

- d. 在 Custom headers (自訂標頭) 面板中，新增您想要納入政策的任何自訂標頭。

如需自訂標頭設定的詳細資訊，請參閱 [the section called “自訂標頭”](#)。

- e. 在 Remove headers (移除標頭) 面板中，新增您希望 CloudFront 從原始伺服器回應移除，而不包含於 CloudFront 傳送給檢視者的回應中的任何標頭名稱。

如需移除標頭設定的詳細資訊，請參閱 [the section called “移除標頭”](#)。

- f. 在 Server-Timing header (Server-Timing 標頭) 面板中，選擇 Enable (啟用) 切換，然後輸入採樣率 (須為介於 0 到 100 之間的數值，包括 0 與 100)。

如需有關 Server-Timing 標頭的詳細資訊，請參閱 [the section called “Server-Timing 標頭”](#)。

4. 選擇 Create (建立) 以建立政策。

建立回應標頭政策之後，您可以將它連接到 CloudFront 分佈中的快取行為。

若要將回應標頭政策連接至現有分佈 (主控台)

1. 在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/distributions> 的 CloudFront 主控台中開啟 Distributions (分佈) 頁面。
2. 選擇要更新的分佈，然後選擇 Behaviors (行為) 索引標籤。
3. 選擇要更新的快取行為，然後選擇 Edit (編輯)。

或者，若要建立新的快取行為，請選擇 Create behavior (建立行為)。

4. 對於 Response headers policy (回應標頭政策)，選擇要新增至快取行為的政策。
5. 選擇 Save changes (儲存變更) 以更新快取行為。如果要建立新的快取行為，請選擇 Create behavior (建立行為)。

若要將回應標頭政策連接至新分佈 (主控台)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇 Create Distribution (建立分佈)。
3. 對於 Response headers policy (回應標頭政策)，選擇要新增至快取行為的政策。
4. 為您的分佈選擇其他設定。如需更多詳細資訊，請參閱 [the section called “所有分佈設定”](#)。
5. 選擇 Create distribution (建立分佈) 以建立分佈。

CloudFormation

若要以 CloudFormation 建立回應標頭政策，請使用 `AWS::CloudFront::ResponseHeadersPolicy` 資源類型。下列範例顯示用於建立回應標頭政策的 CloudFormation 範本語法 (YAML 格式)。

```
Type: AWS::CloudFront::ResponseHeadersPolicy
Properties:
  ResponseHeadersPolicyConfig:
    Name: EXAMPLE-Response-Headers-Policy
    Comment: Example response headers policy for the documentation
  CorsConfig:
    AccessControlAllowCredentials: false
    AccessControlAllowHeaders:
      Items:
        - '*'
    AccessControlAllowMethods:
      Items:
        - GET
        - OPTIONS
    AccessControlAllowOrigins:
      Items:
        - https://example.com
        - https://docs.example.com
    AccessControlExposeHeaders:
      Items:
        - '*'
    AccessControlMaxAgeSec: 600
    OriginOverride: false
  CustomHeadersConfig:
    Items:
      - Header: Example-Custom-Header-1
        Value: value-1
```

```

        Override: true
      - Header: Example-Custom-Header-2
        Value: value-2
        Override: true
    SecurityHeadersConfig:
      ContentSecurityPolicy:
        ContentSecurityPolicy: default-src 'none'; img-src 'self'; script-src
'self'; style-src 'self'; object-src 'none'; frame-ancestors 'none'
        Override: false
      ContentTypeOptions: # You don't need to specify a value for 'X-Content-Type-
Options'.
                           # Simply including it in the template sets its value to
'nosniff'.
      Override: false
      FrameOptions:
        FrameOption: DENY
        Override: false
      ReferrerPolicy:
        ReferrerPolicy: same-origin
        Override: false
      StrictTransportSecurity:
        AccessControlMaxAgeSec: 63072000
        IncludeSubdomains: true
        Preload: true
        Override: false
      XSSProtection:
        ModeBlock: true # You can set ModeBlock to 'true' OR set a value for
ReportUri, but not both
        Protection: true
        Override: false
      ServerTimingHeadersConfig:
        Enabled: true
        SamplingRate: 50
      RemoveHeadersConfig:
        Items:
          - Header: Vary
          - Header: X-Powered-By

```

如需詳細資訊，請參閱 AWS CloudFormation 使用者指南中的
[AWS::CloudFront::ResponseHeadersPolicy](#)。

CLI

若要以 AWS Command Line Interface (AWS CLI) 建立回應標頭政策，請使用 `aws cloudfront create-response-headers-policy` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不必分別將每個個別參數指定為命令列輸入。

建立回應標頭政策 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `response-headers-policy.yaml` 的檔案。這個檔案中包含 `create-response-headers-policy` 命令的所有輸入參數。

```
aws cloudfront create-response-headers-policy --generate-cli-skeleton yaml-input  
> response-headers-policy.yaml
```

2. 開啟您剛才建立的 `response-headers-policy.yaml` 檔案。編輯檔案以指定政策名稱和所需的回應標頭政策組態，然後儲存檔案。

如需有關回應標頭政策設定的詳細資訊，請參閱 [the section called “了解回應標頭政策”](#)。

3. 使用下列命令建立回應標頭政策。您所建立的政策會使用 `response-headers-policy.yaml` 檔案中的輸入參數。

```
aws cloudfront create-response-headers-policy --cli-input-yaml file://response-headers-policy.yaml
```

記下命令輸出中的 `Id` 值，這是回應標頭政策的 ID，將此政策連接到 CloudFront 分佈的快取行為時需要使用這個值。

將回應標頭政策連接至現有分佈 (包含輸入檔案的 CLI)

1. 使用下列命令來儲存您想要更新之 CloudFront 分佈的分佈組態。將 `distribution_ID` 改為分佈的 ID。

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >  
dist-config.yaml
```

2. 開啟您剛才建立且命名為 `dist-config.yaml` 的檔案。編輯檔案，對快取行為進行下列變更，以便讓該行為使用此回應標頭政策。

- 在快取行為中新增名為 `ResponseHeadersPolicyId` 的欄位。對於欄位值，請使用您在建立政策後記下的回應標頭政策 ID。
- 將 `ETag` 欄位重新命名為 `IfMatch`，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈以使用回應標頭政策。將 *distribution_ID* 改為分佈的 ID。

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://dist-config.yaml
```

若要將回應標頭政策連接至新分佈 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `distribution.yaml` 的檔案。這個檔案中包含 `create-distribution` 命令的所有輸入參數。

```
aws cloudfront create-distribution --generate-cli-skeleton yaml-input > distribution.yaml
```

2. 開啟您剛才建立的 `distribution.yaml` 檔案。在預設快取行為的 `ResponseHeadersPolicyId` 欄位中，輸入您在建立政策後記下的回應標頭政策 ID。繼續編輯檔案以指定所需的分佈設定，然後在完成後儲存檔案。

如需有關分佈設定的詳細資訊，請參閱 [所有分佈設定參考](#)。

3. 使用下列命令，使用 `distribution.yaml` 檔案中的輸入參數建立分佈。

```
aws cloudfront create-distribution --cli-input-yaml file://distribution.yaml
```

API

若要使用 CloudFront API 建立回應標頭政策，請使用 [CreateResponseHeadersPolicy](#)。如需有關您在此 API 呼叫中指定欄位的詳細資訊，請參閱 [the section called “了解回應標頭政策”](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考文件。

建立回應標頭政策之後，您可以使用下列其中一個 API 呼叫，將其連接至快取行為：

- 若要將它附加到現有分佈中的快取行為，請使用 [UpdateDistribution](#)。
- 若要將它附加到新分佈中的快取行為，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請在快取行為中的 ResponseHeadersPolicyId 欄位中提供回應標頭政策的 ID。如需您在這些 API 呼叫中指定其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS 開發套件或其他 API 用戶端的 API 參考說明文件。

使用受管回應標頭政策

藉助 CloudFront 回應標頭政策，您可以指定 Amazon CloudFront 移除或新增至傳送給檢視者之回應的 HTTP 標頭。如需回應標頭政策及其使用原因的詳細資訊，請參閱 [使用政策新增或移除回應標頭](#)。

CloudFront 提供受管回應標頭政策，您可以將這些政策連接至 CloudFront 發佈中的快取行為。使用受管回應標頭政策，您不需要撰寫或維護自己的政策。受管政策包含適用於常見使用案例的 HTTP 回應標頭集。

若要使用受管回應標頭政策，請將其連接至分佈中的快取行為。這項程序與建立自訂回應標頭政策的程序相同，只不過您並不會建立新政策，而是連接其中一項受管政策。您可以依名稱 (使用主控台) 或 ID (使用 CloudFormation、AWS CLI 或 AWS 開發套件) 連接政策。名稱和 ID 會列在下一節中。

如需更多詳細資訊，請參閱 [the section called “建立回應標頭政策”](#)。

下列主題說明您可以使用的受管回應標頭政策。

主題

- [CORS-and-SecurityHeadersPolicy](#)
- [CORS-With-Preflight](#)
- [CORS-with-preflight-and-SecurityHeadersPolicy](#)
- [SecurityHeadersPolicy](#)
- [SimpleCORS](#)

CORS-and-SecurityHeadersPolicy

[在 CloudFront 主控台中檢視此政策](#)

使用此受管政策允許來自任何原始伺服器的簡單 CORS 請求。此政策也會將一組安全性標頭新增至 CloudFront 傳送給檢視者的所有回應。這項政策將 [the section called “SimpleCORS”](#) 和 [the section called “SecurityHeadersPolicy”](#) 政策結合為一個政策。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

e61eb60c-9c35-4d20-a928-2b84e02af89c

政策設定

	標頭名稱	標頭值	是否覆寫原始伺服器？
CORS 標頭：	Access-Control-Allow-Origin	*	否
安全性標頭：	Referrer-Policy	strict-origin-when-cross-origin	否
	Strict-Transport-Security	max-age=31536000	否
	X-Content-Type-Options	nosniff	是
	X-Frame-Options	SAMEORIGIN	否
	X-XSS-Protection	1; mode=block	否

CORS-With-Preflight

[在 CloudFront 主控台中檢視此政策](#)

使用此受管政策可允許來自任何原始伺服器的 CORS 請求，包括預檢請求。對於預檢請求（使用 HTTP OPTIONS 方法），CloudFront 將以下三個標頭新增至回應中。對於簡單 CORS 請求，CloudFront 只新增 Access-Control-Allow-Origin 標頭。

如果 CloudFront 從原始伺服器收到的回應包含任何這些標頭，CloudFront 會在傳送給檢視者的回應中使用接收到的標頭（及其值），而不是此政策中的標頭。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

5cc3b908-e619-4b99-88e5-2cf7f45965bd

政策設定

	標頭名稱	標頭值	是否覆寫原始伺服器？
CORS 標頭：	Access-Control-Allow-Methods	DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT	否
	Access-Control-Allow-Origin	*	
	Access-Control-Expose-Headers	*	

CORS-with-preflight-and-SecurityHeadersPolicy

[在 CloudFront 主控台中檢視此政策](#)

使用此受管政策可允許來自任何原始伺服器的 CORS 請求，包含預檢請求。此政策也會將一組安全性標頭新增至 CloudFront 傳送給檢視者的所有回應。這項政策將 [the section called “CORS-With-Preflight”](#) 和 [the section called “SecurityHeadersPolicy”](#) 政策結合為一個政策。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

eaab4381-ed33-4a86-88ca-d9558dc6cd63

政策設定

	標頭名稱	標頭值	是否覆寫原始伺服器？
CORS 標頭：	Access-Control-Allow-Methods	DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT	否
	Access-Control-Allow-Origin	*	

	標頭名稱	標頭值	是否覆寫原始伺服器？
	Access-Control-Expose-Headers	*	
安全性標頭：	Referrer-Policy	strict-origin-when-cross-origin	否
	Strict-Transport-Security	max-age=31536000	否
	X-Content-Type-Options	nosniff	是
	X-Frame-Options	SAMEORIGIN	否
	X-XSS-Protection	1; mode=block	否

SecurityHeadersPolicy

[在 CloudFront 主控台中檢視此政策](#)

使用此受管政策，將一組安全性標頭新增至 CloudFront 傳送給檢視者的所有回應。如需這些安全性標頭的詳細資訊，請參閱 [Mozilla 的 Web 安全指南](#)。

使用這項回應標頭政策後，CloudFront 會為所有回應新增 X-Content-Type-Options: nosniff，無論 CloudFront 從原始伺服器收到的回應是否包含這個標頭都一樣。但如果 CloudFront 從原始伺服器收到的回應所包含的是此政策中的其他任何標頭，則 CloudFront 會在傳送給檢視者的回應中使用接收到的標頭 (及其值)，而不是此政策中的標頭。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

67f7725c-6f97-4210-82d7-5512b31e9d03

政策設定

	標頭名稱	標頭值	是否覆寫原始伺服器？
安全性標頭：	Referrer-Policy	strict-origin-when-cross-origin	否
	Strict-Transport-Security	max-age=31536000	否
	X-Content-Type-Options	nosniff	是
	X-Frame-Options	SAMEORIGIN	否
	X-XSS-Protection	1; mode=block	否

SimpleCORS

[在 CloudFront 主控台中檢視此政策](#)

使用此受管政策允許來自任何原始伺服器的[簡單 CORS 請求](#)。使用此政策，CloudFront 將標頭 `Access-Control-Allow-Origin: *` 新增至簡單 CORS 請求的所有回應。

如果 CloudFront 從原始伺服器收到的回應包含 `Access-Control-Allow-Origin` 標頭，CloudFront 會在傳送給檢視者的回應中使用該標頭 (及其值)，而不是此政策中的標頭。

使用 CloudFormation、AWS CLI、或 CloudFront API 時，此政策的 ID 為：

60669652-455b-4ae9-85a4-c4c02393f86c

政策設定

	標頭名稱	標頭值	是否覆寫原始伺服器？
CORS 標頭：	Access-Control-Allow-Origin	*	否

請求和回應行為

下列主題說明 CloudFront 如何處理請求和回應。

您可以瞭解 CloudFront 如何與 Amazon S3 或自訂原始伺服器互動、處理各種 HTTP 方法和標頭、處理狀態碼，以及管理快取和錯誤回應。

主題

- [CloudFront 如何處理 HTTP 和 HTTPS 的請求](#)
- [Amazon S3 原始伺服器之請求和回應行為](#)
- [為自訂原始伺服器之請求和回應行為](#)
- [原始伺服器群組的請求和回應行為](#)
- [將自訂標頭新增到原始伺服器請求](#)
- [CloudFront 如何處理物件的部分請求 \(範圍 GET\)](#)
- [CloudFront 如何處理來自原始伺服器的 HTTP 3xx 狀態碼。](#)
- [CloudFront 如何處理來自原始伺服器的 HTTP 4xx 和 5xx 狀態碼](#)
- [產生自訂錯誤回應](#)

CloudFront 如何處理 HTTP 和 HTTPS 的請求

針對 Amazon S3 原始伺服器，預設情況下，在適用於 CloudFront 分佈中物件的 HTTP 和 HTTPS 通訊協定中 CloudFront 接受請求。然後，CloudFront 使用與請求相同的通訊協定轉送請求到 Amazon S3 儲存貯體。

對於自訂原始伺服器，當您建立分佈時，您可以指定 CloudFront 如何存取原始伺服器：僅限 HTTP，或配合檢視器所使用的通訊協定。如需有關 CloudFront 如何為自訂原始伺服器處理 HTTP 和 HTTPS 請求的詳細資訊，請參閱[通訊協定](#)。

如需有關如何限制分佈，讓最終使用者使用 HTTPS 只能存取物件的詳細資訊，請參閱[透過 CloudFront 使用 HTTPS](#)。

Note

HTTPS 請求的費用高過於 HTTP 請求的費用。如需計費費率的詳細資訊，請參閱 [CloudFront 定價](#)。

Amazon S3 原始伺服器之請求和回應行為

若要瞭解您使用 Amazon S3 做為原始伺服器時 CloudFront 如何處理請求和回應，請參閱下列章節：

主題

- [CloudFront 如何處理和轉送請求至您的 Amazon S3 原始伺服器](#)
- [CloudFront 如何處理來自您 Amazon S3 原始伺服器的回應](#)

CloudFront 如何處理和轉送請求至您的 Amazon S3 原始伺服器

瞭解 CloudFront 如何處理檢視器請求和轉送請求至 Amazon S3 原始伺服器。

內容

- [快取持續時間和最短 TTL](#)
- [用戶端 IP 位址](#)
- [條件式 GET 請求](#)
- [Cookie](#)
- [跨來源資源共享 \(CORS\)](#)
- [包括本文的 GET 請求](#)
- [HTTP 方法](#)
- [CloudFront 移除或更新的 HTTP 請求標頭](#)
- [最大請求長度和最大 URL 長度](#)
- [OCSP 裝訂](#)
- [通訊協定](#)
- [查詢字串](#)
- [原始伺服器連線逾時和嘗試次數](#)
- [原始伺服器回應逾時](#)
- [相同物件之同步請求 \(請求折疊\)](#)

快取持續時間和最短 TTL

若要控制在 CloudFront 將另一個請求轉送到原始伺服器之前，您的物件留存於 CloudFront 快取中的時間長度，您可以：

- 設定原始伺服器在每個物件中新增 Cache-Control 或 Expires 標頭欄位。
- 指定在 CloudFront 快取行為中最短 TTL 的值。
- 使用預設值為 24 小時。

如需詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

用戶端 IP 位址

如果檢視器將請求傳送到 CloudFront，而且未包含 X-Forwarded-For 請求標頭，則 CloudFront 會從 TCP 連線取得檢視器的 IP 位址、加入包含該 IP 位址的 X-Forwarded-For 標頭，然後將該請求轉送給原始伺服器。例如，如果 CloudFront 從 TCP 連線取得 IP 位址 192.0.2.2，則會將下列的標頭轉送到原始伺服器：

X-Forwarded-For: 192.0.2.2

如果檢視器將請求傳送到 CloudFront，並且包含 X-Forwarded-For 請求標頭，則 CloudFront 會從 TCP 連線取得檢視器的 IP 位址、將該 IP 位址附加到 X-Forwarded-For 標頭的結尾，然後將該請求轉送給原始伺服器。例如，如果檢視器的請求包含 X-Forwarded-For: 192.0.2.4,192.0.2.3，而且 CloudFront 從 TCP 連線取得 IP 位址 192.0.2.2，則會將下列的標頭轉送到原始伺服器：

X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2

Note

X-Forwarded-For 標頭包含 IPv4 地址 (如 192.0.2.44) 和 IPv6 地址 (如 2001:0db8:85a3::8a2e:0370:7334)。

條件式 GET 請求

如果 CloudFront 收到請求，向邊緣快取要求已過期的物件，會將該請求轉送到 Amazon S3 原始伺服器，以取得物件的最新版本，或是從 Amazon S3 取得 CloudFront 邊緣快取已具有最新版本的確認。當 Amazon S3 最初將物件傳送到 CloudFront 時，在回應中加入了 ETag 值和 LastModified 值。在 CloudFront 轉送到 Amazon S3 的新請求中，CloudFront 會新增下列其中一個或兩個標頭：

- 含有已過期版本物件 If-Match 值的 If-None-Match 或 ETag 標頭。
- 含有已過期版本物件 If-Modified-Since 值的 LastModified 標頭。

Amazon S3 使用此資訊來判斷物件是否已被更新，且因此是否需傳回整個物件至 CloudFront 或只傳回 HTTP 304 狀態碼 (而非修改)。

Cookie

Amazon S3 不處理 cookie。如果您設定快取行為將 Cookie 轉送到 Amazon S3 原始伺服器，則 CloudFront 會轉送 Cookie，但 Amazon S3 會忽略這些 Cookie。所有相同物件的未來請求，無論是否變更 Cookie，透過快取中的現有物件提供。

跨來源資源共享 (CORS)

如果想讓 CloudFront 遵循 Amazon S3 跨來源資源共享設定，請設定 CloudFront 將選取的標頭轉送到 Amazon S3。如需詳細資訊，請參閱 [根據請求標頭快取內容](#)。

包括本文的 GET 請求

如果檢視器的 GET 請求包含本文，CloudFront 會將 HTTP 狀態碼 403 (禁止) 傳回給檢視器。

HTTP 方法

如果設定 CloudFront 處理其支援的所有 HTTP 方法，CloudFront 會接受從檢視器傳來的下列請求，然後將這些請求轉送到您的 Amazon S3 原始伺服器：

- DELETE
- GET
- HEAD
- OPTIONS
- PATCH
- POST
- PUT

CloudFront 一律會快取 GET 與 HEAD 請求的回應。您也可以設定 CloudFront 快取 OPTIONS 請求的回應。CloudFront 不會快取對使用其他方法之請求的回應。

如果您想要使用分段上傳的方式，將物件新增至 Amazon S3 儲存貯體，您必須將 CloudFront 原始存取控制 (OAC) 新增至分佈，並將所需要的許可授予該 OAC。如需詳細資訊，請參閱 [the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

Important

如果設定 CloudFront 接受 CloudFront 支援的所有 HTTP 方法並轉送到 Amazon S3，您必須建立一個 CloudFront OAC，以限制對您 Amazon S3 內容的存取，並將所需要的許可權限授予該 OAC。例如若因為想要使用 PUT 方法，而設定 CloudFront 接受和轉送這些方法，則您必須設定 Amazon S3 儲存貯體政策，以適當地處理 DELETE 請求，讓檢視器無法刪除您不希望其刪除的資源。如需詳細資訊，請參閱[the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

如需關於 Amazon S3 支援操作的詳細資訊，請參閱 [Amazon S3 文件](#)。

CloudFront 移除或更新的 HTTP 請求標頭

在將請求轉送到您的 Amazon S3 原始伺服器之前，CloudFront 會先移除或更新一些標頭。對於大多數標頭，這種行為與自訂原始伺服器相同。如需 HTTP 請求標頭的完整清單，以及 CloudFront 處理這些標頭的方式，請參閱 [HTTP 請求標頭和 CloudFront 行為 \(自訂和 Amazon S3 原始伺服器\)](#)。

最大請求長度和最大 URL 長度

最大請求長度，包括路徑、查詢字串 (如果有) 和標頭是 20,480 位元組。

CloudFront 從請求建構 URL。此 URL 的最大長度為 8192 位元組。

如果請求或 URL 超過最大長度，CloudFront 會傳回 HTTP 狀態碼 413 (請求實體太大) 至檢視器，然後終止 TCP 連線到檢視器。

OCSP 裝訂

檢視器提交對物件的 HTTPS 請求時，CloudFront 或檢視器都必須向憑證認證機構 (CA) 確認尚未撤銷網域的 SSL 憑證。OCSP 裝訂藉由讓 CloudFront 從 CA 驗證憑證和快取回應加速憑證驗證，所以用戶端不需要直接向 CA 驗證憑證。

CloudFront 收到針對同一個網域中物件的大量 HTTPS 請求時，OCSP 裝訂的效能提升會更加明顯。每個在 CloudFront 節點的伺服器必須提交單獨的驗證請求。CloudFront 收到針對同一個網域的大量 HTTPS 請求時，在邊緣節點中的每個伺服器很快就會收到來自 CA 的回應，伺服器可將其裝訂到 SSL 交握中的封包。檢視器確認憑證為有效時，CloudFront 即可提供請求的物件。如果您的分佈無法取得在 CloudFront 節點更多的流量，新的請求更有可能被導向至尚未向 CA 驗證憑證的伺服器。在這種情況下，檢視器會單獨執行驗證步驟，而且 CloudFront 伺服器會提供物件。該 CloudFront 伺服器也會提交驗證請求至 CA，所以下一次收到包含相同網域名稱的請求時，會有來自 CA 的驗證回應。

通訊協定

CloudFront 會根據檢視器請求的通訊協定 (HTTP 或 HTTPS)，轉送 HTTP 或 HTTPS 請求至原始伺服器。

Important

如果設定 Amazon S3 儲存貯體為網站端點，則您無法設定 CloudFront 使用 HTTPS 與原始伺服器進行通訊，因為 Amazon S3 在該組態中不支援 HTTPS 連線。

查詢字串

您可以設定 CloudFront 是否轉送查詢字串參數至您的 Amazon S3 原始伺服器。如需詳細資訊，請參閱 [根據查詢字串參數快取內容](#)。

原始伺服器連線逾時和嘗試次數

「原始伺服器連線逾時」是嘗試建立與原始伺服器連線時 CloudFront 所等待的秒數。

「原始伺服器連線嘗試次數」是 CloudFront 嘗試連線至原始伺服器的次數。

這些設定共同決定了在容錯移轉至次要原始伺服器 (如果是原始伺服器群組) 或將錯誤回應傳回給檢視器之前，CloudFront 嘗試連線到原始伺服器的時間長度。依預設，CloudFront 會等待 30 秒 (嘗試 3 次，每次 10 秒)，然後再嘗試連線至次要原始伺服器或傳回錯誤回應。您可以指定較短的連線逾時、較少的嘗試次數或兩者，以縮短此時間。

如需詳細資訊，請參閱 [控制原始伺服器逾時和嘗試次數](#)。

原始伺服器回應逾時

「原始伺服器回應逾時」，也稱為「原始伺服器讀取逾時」或「原始伺服器請求逾時」，適用於以下兩個數值：

- 在將請求轉送到原始伺服器之後，CloudFront 等待回應的時間 (以秒為單位)。
- CloudFront 在收到來自原始伺服器的回應封包後，並在接收下一個封包前所等待的時間 (以秒為單位)。

CloudFront 行為取決於檢視器請求的 HTTP 方法：

- GET 與 HEAD 請求 – 如果原始伺服器未在 30 秒內回應，或是停止回應 30 秒，CloudFront 會結束連線。如果指定的[原始伺服器連線嘗試次數](#)超過 1，CloudFront 會再次嘗試取得完整的回應。CloudFront 最多可嘗試 3 次，由「原始伺服器連線嘗試次數」設定的值決定。如果原始伺服器在最後一次嘗試時未回應，則 CloudFront 在收到相同原始伺服器上內容的另一個請求之前不會再嘗試。
- DELETE、OPTIONS、PATCH、PUT 與 POST 請求 – 如果原始伺服器未在 30 秒內回應，CloudFront 會結束連線，而且不會再嘗試聯絡原始伺服器。用戶端可以視需要重新提交請求。

您無法變更 Amazon S3 原始伺服器（「非」使用靜態網站託管所設定的 S3 儲存貯體）的回應逾時。

相同物件之同步請求 (請求折疊)

如果 CloudFront 邊緣節點收到對物件的請求，而且該物件目前不在快取中或快取物件已過期，CloudFront 會立即將該請求傳送到原始伺服器。不過，如果有對相同物件的同步請求，意即：如果在 CloudFront 接收到第一個請求之前，(具有相同快取金鑰) 其他相同物件的請求抵達邊緣節點，則在將其他請求轉送到原始伺服器之前，CloudFront 將暫停。此短暫的暫停有助於減輕原始伺服器的負載。暫停期間，CloudFront 會將原始伺服器請求的回應傳送至接收到的所有請求。這就是所謂的請求摺疊。在 CloudFront 日誌中，會將第一個請求識別為 x-edge-result-type 欄位中的 Miss，而崩潰的請求會識別為 Hit。如需有關 CloudFront 日誌的詳細資訊，請參閱 [the section called “CloudFront 和邊緣函數記錄”](#)。

CloudFront 只會摺疊共用[快取金鑰](#)的請求。如果其他的請求沒有共用快取金鑰，(例如，因為您設定 CloudFront 根據請求標頭或查詢字串進行快取)，則 CloudFront 會將所有獨特的請求轉送到原始伺服器。

如果您想要防止所有請求摺疊，可使用受管快取政策 CachingDisabled (也可防止快取)。如需詳細資訊，請參閱[使用受管快取政策](#)。

若您想防止特定物件的請求折疊，您可以將快取行為的最短 TTL 設為 0 並設定原始伺服器傳送 Cache-Control: private、Cache-Control: no-store、Cache-Control: no-cache、Cache-Control: max-age=0 或 Cache-Control: s-maxage=0。這些組態會增加原始伺服器的負載，並於 CloudFront 等待第一個請求的回應時，為暫停的同步請求引入額外的延遲。

Important

目前如果您在[快取政策](#)、[原始伺服器請求政策](#)或舊版快取設定中啟用 Cookie 轉送，CloudFront 不支援請求摺疊。

CloudFront 如何處理來自您 Amazon S3 原始伺服器的回應

瞭解 CloudFront 如何處理來自您 Amazon S3 原始伺服器的回應。

內容

- [已取消請求](#)
- [CloudFront 移除或更新的 HTTP 回應標頭](#)
- [可快取檔案大小上限](#)
- [重新引導](#)

已取消請求

如果物件不在邊緣快取中，而且在 CloudFront 從原始伺服器取得物件之後，但尚未傳送請求的物件之前，檢視器就終止了工作階段 (例如，關閉瀏覽器)，則 CloudFront 不會在節點快取物件。

CloudFront 移除或更新的 HTTP 回應標頭

在將 Amazon S3 原始伺服器傳來的回應轉送給檢視器之前，CloudFront 會移除或更新下列的標頭欄位：

- X-Amz-Id-2
- X-Amz-Request-Id
- Set-Cookie – 如果設定 CloudFront 轉送 Cookie，此服務會將 Set-Cookie 標頭欄位轉送給用戶端。如需詳細資訊，請參閱 [根據 Cookie 快取內容](#)。
- Trailer
- Transfer-Encoding – 如果 Amazon S3 原始伺服器傳回此標頭欄位，CloudFront 會將此值設定為 chunked，再將回應傳回給檢視器。
- Upgrade
- Via – CloudFront 在對檢視器的回應中將值設為以下值：

Via: *http-## ##-##*.cloudfront.net (CloudFront)

值的範例如下所示：

Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)

可快取檔案大小上限

CloudFront 在快取中儲存의 回應內文大小上限為 50 GB。此包含未指定 Content-Length 標頭值的區塊傳輸回應。

使用範圍請求方法來請求每個部分為 50 GB 或更小的多部分物件，即可使用 CloudFront 快取大於此大小的物件。CloudFront 會快取這些部分，因為每個部分皆為 50 GB 或更小。檢視器擷取物件的所有部分之後，就可以重建原始、較大的物件。如需詳細資訊，請參閱 [使用範圍請求快取大物件](#)。

重新引導

您可以設定 Amazon S3 儲存貯體，將所有的請求重新引導到另一個主機名稱，這可以是另一個 Amazon S3 儲存貯體或 HTTP 伺服器。如果設定儲存貯體，將所有的請求重新引導，而且如果儲存貯體是 CloudFront 分佈的來源，則我們建議您設定該儲存貯體，使用分佈的網域名稱 (例如，d1111111abcdef8.cloudfront.net)，或使用與分佈具有關聯的替代網域名稱 (CNAME) (例如，example.com)，來將所有的請求重新引導到 CloudFront 分佈。否則，檢視器會略過 CloudFront 請求，並從新的原始伺服器直接提供物件。

Note

如果您重新引導請求到備用網域名稱，您必須藉由新增 CNAME 記錄，為您的網域更新 DNS 服務。如需詳細資訊，請參閱 [透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#)。

以下是當您設定儲存貯體重新引導所有請求時，發生的情況：

1. 檢視器 (例如，瀏覽器) 自 CloudFront 請求物件。
2. CloudFront 會將請求轉送到做為您分佈來源的 Amazon S3 儲存貯體。
3. Amazon S3 傳回 HTTP 狀態碼 301 (永久移動) 以及新的位置。
4. CloudFront 快取重新引導狀態碼和新的位置，並傳回值給檢視器。CloudFront 不遵循重新引導，以從新位置取得物件。
5. 檢視器傳送物件的另一個請求，但這次檢視器指定從 CloudFront 取得的新位置：
 - 如果 Amazon S3 儲存貯體使用 CloudFront 分佈的網域名稱或替代網域名稱，來將所有的請求重新引導到分佈，則 CloudFront 會向 Amazon S3 儲存貯體或新位置中的 HTTP 伺服器請求物件。當新的位置傳回物件時，CloudFront 會將其傳回至檢視器並在節點快取。

- 如果 Amazon S3 儲存貯體將請求重新引導至另一個位置，則第二個請求會略過 CloudFront。Amazon S3 儲存貯體或新位置中的 HTTP 伺服器，會將物件直接傳回給檢視器，因此一律不會在 CloudFront 邊緣快取中快取物件。

為自訂原始伺服器之請求和回應行為

若要瞭解 CloudFront 如何在您使用自訂原始伺服器時處理請求和回應，請參閱下列章節：

主題

- [CloudFront 如何處理和轉送請求至您的自訂原始伺服器](#)
- [CloudFront 如何處理來自您的自訂原始伺服器的回應](#)

CloudFront 如何處理和轉送請求至您的自訂原始伺服器

瞭解 CloudFront 如何處理檢視器請求和轉送請求至您的自訂原始伺服器。

內容

- [身分驗證](#)
- [快取持續時間和最短 TTL](#)
- [用戶端 IP 位址](#)
- [用戶端 SSL 身分驗證](#)
- [壓縮](#)
- [條件式請求](#)
- [Cookie](#)
- [跨來源資源共享 \(CORS\)](#)
- [加密](#)
- [包括本文的 GET 請求](#)
- [HTTP 方法](#)
- [HTTP 請求標頭和 CloudFront 行為 \(自訂和 Amazon S3 原始伺服器\)](#)
- [HTTP 版本](#)
- [最大請求長度和最大 URL 長度](#)
- [OCSP 裝訂](#)

- [持久性連線](#)
- [通訊協定](#)
- [查詢字串](#)
- [原始伺服器連線逾時和嘗試次數](#)
- [原始伺服器回應逾時](#)
- [相同物件之同步請求 \(請求折疊\)](#)
- [User-Agent 標頭](#)

身分驗證

如果您將 Authorization 標頭轉送至原始伺服器，則可以設定原始伺服器，以針對下列請求類型請求用戶端身分驗證：

- DELETE
- GET
- HEAD
- PATCH
- PUT
- POST

對於 OPTIONS 請求，只有在您使用下列 CloudFront 設定時才能設定用戶端身分驗證：

- CloudFront 設定為將 Authorization 標頭轉送到您的原始伺服器。
- CloudFront 設定為不快取對 OPTIONS 請求的回應。

如需詳細資訊，請參閱 [設定 CloudFront 以轉送 Authorization 標頭](#)。

您可以使用 HTTP 或 HTTPS 將請求轉送至原始伺服器。如需詳細資訊，請參閱 [透過 CloudFront 使用 HTTPS](#)。

快取持續時間和最短 TTL

若要控制在 CloudFront 將另一個請求轉送到原始伺服器之前，您的物件留存於 CloudFront 快取中的時間長度，您可以：

- 設定原始伺服器在每個物件中新增 Cache-Control 或 Expires 標頭欄位。
- 指定在 CloudFront 快取行為中最短 TTL 的值。
- 使用預設值為 24 小時。

如需詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

用戶端 IP 位址

如果檢視器將請求傳送到 CloudFront，而且未包含 X-Forwarded-For 請求標頭，則 CloudFront 會從 TCP 連線取得檢視器的 IP 位址、加入包含該 IP 位址的 X-Forwarded-For 標頭，然後將該請求轉送給原始伺服器。例如，如果 CloudFront 從 TCP 連線取得 IP 位址 192.0.2.2，則會將下列的標頭轉送到原始伺服器：

X-Forwarded-For: 192.0.2.2

如果檢視器將請求傳送到 CloudFront，並且包含 X-Forwarded-For 請求標頭，則 CloudFront 會從 TCP 連線取得檢視器的 IP 位址、將該 IP 位址附加到 X-Forwarded-For 標頭的結尾，然後將該請求轉送給原始伺服器。例如，如果檢視器的請求包含 X-Forwarded-For: 192.0.2.4,192.0.2.3，而且 CloudFront 從 TCP 連線取得 IP 位址 192.0.2.2，則會將下列的標頭轉送到原始伺服器：

X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2

某些應用程式，例如負載平衡器 (包括 Elastic Load Balancing)、Web 應用程式防火牆、反向代理伺服器、入侵防護系統和 API Gateway，會針對轉送請求的 CloudFront 邊緣伺服器，將該伺服器的 IP 位址附加到 X-Forwarded-For 標頭的結尾。例如，如果 CloudFront 在轉送給 ELB 的請求中加入 X-Forwarded-For: 192.0.2.2，而且 CloudFront 邊緣伺服器的 IP 位址為 192.0.2.199，則您的 EC2 執行個體所收到的請求，會包含下列的標頭：

X-Forwarded-For: 192.0.2.2,192.0.2.199

Note

X-Forwarded-For 標頭包含 IPv4 地址 (如 192.0.2.44) 和 IPv6 地址 (如 2001:0db8:85a3::8a2e:0370:7334)。

另請注意，目前伺服器 (CloudFront) 路徑上的每個節點都可以修改 X-Forwarded-For 標頭。如需詳細資訊，請參閱 [RFC 7239](#) 的第 8.1 節。您也可以使用 CloudFront 邊緣運算函數修改標頭。

用戶端 SSL 身分驗證

CloudFront 支援交互 TLS (mTLS) 身分驗證，其中用戶端和伺服器都會使用憑證互相驗證。設定 mTLS 後，CloudFront 可以在 TLS 交握期間驗證用戶端憑證，並選擇性地執行 CloudFront Functions 以實作自訂驗證邏輯。

對於未設定 mTLS 時請求用戶端憑證的原始伺服器，CloudFront 會捨棄請求。

如需設定 mTLS 的詳細資訊，請參閱 [???。](#)

CloudFront 不支援使用用戶端 SSL 憑證之用戶端身分驗證。如果原始伺服器請求用戶端憑證，CloudFront 會放棄請求。

壓縮

如需詳細資訊，請參閱 [提供壓縮檔案。](#)

條件式請求

如果 CloudFront 收到請求，向邊緣快取要求已過期的物件，會將該請求轉送到原始伺服器，以取得物件的最新版本，或是從原始伺服器取得 CloudFront 邊緣快取已具有最新版本的確認。一般而言，當原始伺服器最後將物件傳送到 CloudFront 時，會在回應中加入 ETag 值、LastModified 值，或是這兩個值。在 CloudFront 轉送到原始伺服器的新請求中，CloudFront 會加入下列其中一個或兩個項目：

- 含有已過期版本物件 If-Match 值的 If-None-Match 或 ETag 標頭。
- 含有已過期版本物件 If-Modified-Since 值的 LastModified 標頭。

原始伺服器使用此資訊來判斷物件是否已被更新，且因此是否需傳回整個物件至 CloudFront 或只傳回 HTTP 304 狀態碼 (而非修改)。

Note

若 CloudFront 設為轉送 Cookie (所有或子集)，則不支援 If-Modified-Since 和 If-None-Match 條件請求。

如需詳細資訊，請參閱[根據 Cookie 快取內容。](#)

Cookie

您可以設定 CloudFront 轉送 Cookie 至您的原始伺服器。如需詳細資訊，請參閱 [根據 Cookie 快取內容](#)。

跨來源資源共享 (CORS)

如果想讓 CloudFront 遵循跨來源資源共享設定，請設定 CloudFront 將 Origin 標頭轉送到您的原始伺服器。如需詳細資訊，請參閱 [根據請求標頭快取內容](#)。

加密

您可以要求檢視器使用 HTTPS 來將請求傳送至 CloudFront，並要求 CloudFront 使用檢視器所使用的通訊協定，將請求轉送給您的自訂原始伺服器。如需詳細資訊，請參閱下列分佈設定：

- [檢視器通訊協定政策](#)
- [通訊協定 \(僅限自訂原始伺服器\)](#)

CloudFront 使用 SSLv3、TLSv1.0、TLSv1.1、TLSv1.2 和 TLSv1.3 通訊協定，轉送 HTTPS 請求至原始伺服器。對自訂原始伺服器，您可以選擇您希望 CloudFront 在與原始伺服器通訊時要使用的 SSL 通訊協定：

- 如果使用 CloudFront 主控台，請透過勾選 Origin SSL Protocols (原始伺服器 SSL 通訊協定) 核取方塊，來選擇通訊協定。如需詳細資訊，請參閱 [建立分發](#)。
- 如果使用 CloudFront API，請利用 OriginSslProtocols 元素來指定通訊協定。如需詳細資訊，請參閱《Amazon CloudFront API 參考》中的 [OriginSslProtocols](#) 和 [DistributionConfig](#)。

如果該原始伺服器是 Amazon S3 儲存貯體，則 CloudFront 預設使用 TLSv1.3。

Important

其他 SSL 和 TLS 的版本不支援。

如需有關使用 CloudFront 搭配 HTTPS 的詳細資訊，請參閱[透過 CloudFront 使用 HTTPS](#)。針對檢視器與 CloudFront 之間，以及 CloudFront 與原始伺服器之間的 HTTPS 通訊，如需 CloudFront 支援密碼的清單，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

包括本文的 GET 請求

如果檢視器的 GET 請求包含本文，CloudFront 會將 HTTP 狀態碼 403 (禁止) 傳回給檢視器。

HTTP 方法

如果設定 CloudFront 處理其支援的所有 HTTP 方法，CloudFront 會接受從檢視器傳來的下列請求，然後將這些請求轉送到您的自訂原始伺服器：

- DELETE
- GET
- HEAD
- OPTIONS
- PATCH
- POST
- PUT

CloudFront 一律會快取 GET 與 HEAD 請求的回應。您也可以設定 CloudFront 快取 OPTIONS 請求的回應。CloudFront 不會快取對使用其他方法之請求的回應。

如需有關設定您的自訂原始伺服器是否處理這些方法的詳細資訊，請參閱您的原始伺服器的文件。

Important

如果設定 CloudFront 接受 CloudFront 支援的所有 HTTP 方法，並將這些方法轉送到原始伺服器，請設定您的原始伺服器，來處理所有的方法。例如，如果因為想要使用 POST，而設定 CloudFront 接受和轉送這些方法，則您必須設定原始伺服器，以適當地處理 DELETE 請求，讓檢視器無法刪除您不希望其刪除的資源。如需詳細資訊，請參閱您的 HTTP 伺服器文件。

HTTP 請求標頭和 CloudFront 行為 (自訂和 Amazon S3 原始伺服器)

下表列出可以轉送到自訂和 Amazon S3 原始伺服器的 HTTP 請求標頭 (已指明例外狀況)。對於每個標頭，該表包含下列資訊：

- 未設定 CloudFront 將標頭轉送到原始伺服器時，CloudFront 的行為，此行為會讓 CloudFront 根據標頭值來快取物件。

- 您是否可以設定 CloudFront 根據該標頭的標頭值來快取物件。

您可以設定 CloudFront 根據 Date 與 User-Agent 標頭中的值來快取物件，但我們不建議這麼做。這些標頭有許多可能的值，且根據其值進行快取可能導致 CloudFront 轉送更多請求到原始伺服器。

如需有關根據標頭值快取的詳細資訊，請參閱[根據請求標頭快取內容](#)。

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
其他定義的標頭	舊版快取設定 – CloudFront 轉送標頭至原始伺服器。	是
Accept	CloudFront 移除標頭。	是
Accept-Charset	CloudFront 移除標頭。	是
Accept-Encoding	如果該值包含 gzip 或 br，CloudFront 會將標準化 Accept-Encoding 標頭轉送給您的原始伺服器。 如需詳細資訊，請參閱 壓縮支援 及 提供壓縮檔案 。	是
Accept-Language	CloudFront 移除標頭。	是
Authorization	• GET 與 HEAD 請求 – CloudFront 會在將請求轉送到您的原始伺服器之前，移除 Authorization 標頭欄位。 • OPTIONS 請求 – 如果設定 CloudFront 快取 Authorization 請求的回應，則 CloudFront 會在將請求轉送到原始伺服器之前，移除 OPTIONS 標頭欄位。	是

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
	如果未設定 CloudFront 快取 OPTIONS 請求的回應，CloudFront 會將 Authorization 標頭欄位轉送到您的原始伺服器。 DELETE、PATCH、POST，和 PUT 請求 – CloudFront 在轉送請求至您的原始伺服器之前，不會移除標頭欄位。	
Cache-Control	CloudFront 轉送標頭至原始伺服器。	否
CloudFront-Forwarded-Proto	CloudFront 在轉送請求至您的原始伺服器之前，不會加入標頭。 如需詳細資訊，請參閱 根據請求的通訊協定設定快取。	是
CloudFront-Is-Desktop-Viewer	CloudFront 在轉送請求至您的原始伺服器之前，不會加入標頭。 如需詳細資訊，請參閱 根據裝置類型設定快取。	是
CloudFront-Is-Mobile-Viewer	CloudFront 在轉送請求至您的原始伺服器之前，不會加入標頭。 如需詳細資訊，請參閱 根據裝置類型設定快取。	是
CloudFront-Is-Tablet-Viewer	CloudFront 在轉送請求至您的原始伺服器之前，不會加入標頭。 如需詳細資訊，請參閱 根據裝置類型設定快取。	是

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
CloudFront-Viewer-Country	CloudFront 在轉送請求至您的原始伺服器之前，不會加入標頭。	是
Connection	CloudFront 在轉送請求至您的原始伺服器之前，以 Connection: Keep-Alive 取代此標頭。	否
Content-Length	CloudFront 轉送標頭至原始伺服器。	否
Content-MD5	CloudFront 轉送標頭至原始伺服器。	是
Content-Type	CloudFront 轉送標頭至原始伺服器。	是
Cookie	如果設定 CloudFront 轉送 Cookie，此服務會將 Cookie 標頭欄位轉送給您的原始伺服器。如果未如此設定，CloudFront 會移除 Cookie 標頭欄位。如需詳細資訊，請參閱 根據 Cookie 快取內容 。	否
Date	CloudFront 轉送標頭至原始伺服器。	可以，但不建議
Expect	CloudFront 移除標頭。	是
From	CloudFront 轉送標頭至原始伺服器。	是

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
Host	CloudFront 將值設定為與請求的物件相關聯之原始伺服器的網域名稱。 您無法根據 Amazon S3 或 MediaStore 原始伺服器的主機標頭進行快取。	是 (自訂) 否 (S3 和 MediaStore)
If-Match	CloudFront 轉送標頭至原始伺服器。	是
If-Modified-Since	CloudFront 轉送標頭至原始伺服器。	是
If-None-Match	CloudFront 轉送標頭至原始伺服器。	是
If-Range	CloudFront 轉送標頭至原始伺服器。	是
If-Unmodified-Since	CloudFront 轉送標頭至原始伺服器。	是
Max-Forwards	CloudFront 轉送標頭至原始伺服器。	否
Origin	CloudFront 轉送標頭至原始伺服器。	是
Pragma	CloudFront 轉送標頭至原始伺服器。	否
Proxy-Authenticate	CloudFront 移除標頭。	否
Proxy-Authorization	CloudFront 移除標頭。	否

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
Proxy-Connection	CloudFront 移除標頭。	否
Range	CloudFront 轉送標頭至原始伺服器。如需詳細資訊，請參閱 CloudFront 如何處理物件的部分請求 (範圍 GET) 。	是，根據預設。
Referer	CloudFront 移除標頭。	是
Request-Range	CloudFront 轉送標頭至原始伺服器。	否
TE	CloudFront 移除標頭。	否
Trailer	CloudFront 移除標頭。	否
Transfer-Encoding	CloudFront 轉送標頭至原始伺服器。	否
Upgrade	除非您已建立 WebSocket 連線，否則 CloudFront 會移除標頭。	否 (WebSocket 連線除外)
User-Agent	CloudFront 以 Amazon CloudFront 取代此標頭欄位的值。如果您希望 CloudFront 根據使用者所使用的裝置來快取內容，請參閱 根據裝置類型設定快取 。	可以，但不建議
Via	CloudFront 轉送標頭至原始伺服器。	是
Warning	CloudFront 轉送標頭至原始伺服器。	是

標頭	未設定 CloudFront 根據標頭值進行快取時的行為	支援根據標頭值進行快取
X-Amz-Cf-Id	CloudFront 在轉送請求至您的原始伺服器之前，將標頭加入檢視器。此標頭值包含可唯一識別請求的加密字串。	否
X-Edge-*	CloudFront 會移除所有的 X-Edge-* 標頭。	否
X-Forwarded-For	CloudFront 轉送標頭至原始伺服器。如需詳細資訊，請參閱 用戶端 IP 位址 。	是
X-Forwarded-Proto	CloudFront 移除標頭。	否
X-HTTP-Method-Override	CloudFront 移除標頭。	是
X-Real-IP	CloudFront 移除標頭。	否

HTTP 版本

CloudFront 使用 HTTP/1.1 轉送請求到您的自訂原始伺服器。

最大請求長度和最大 URL 長度

最大請求長度，包括路徑、查詢字串 (如果有) 和標頭是 20,480 位元組。

CloudFront 從請求建構 URL。此 URL 的最大長度為 8192 位元組。

如果請求或 URL 超過這些最大值，CloudFront 會傳回 HTTP 狀態碼 413、請求實體對檢視器太大，然後終止 TCP 連線到檢視器。

OCSP 裝訂

當檢視器提交對物件的 HTTPS 請求時，CloudFront 或檢視器都必須向憑證授權單位 (CA) 確認尚未撤銷網域的 SSL 憑證。OCSP 裝訂藉由讓 CloudFront 從 CA 驗證憑證和快取回應加速憑證驗證，所以用戶端不需要直接向 CA 驗證憑證。

當 CloudFront 收到針對同一個網域中物件的大量 HTTPS 請求時，OCSP 裝訂的效能提升會更加明顯。每個在 CloudFront 節點的伺服器必須提交單獨的驗證請求。當 CloudFront 收到針對同一個網域的大量 HTTPS 請求時，在節點中的每個伺服器很快就會收到來自 CA 的回應，伺服器可將此回應「裝釘」到 SSL 交握中的封包；當檢視器確認憑證為有效時，CloudFront 即可提供請求的物件。如果您的分佈無法取得在 CloudFront 節點更多的流量，新的請求更有可能被導向至尚未向 CA 驗證憑證的伺服器。在這種情況下，檢視器會單獨執行驗證步驟，而且 CloudFront 伺服器會提供物件。該 CloudFront 伺服器也會提交驗證請求至 CA，所以下一次收到包含相同網域名稱的請求時，會有來自 CA 的驗證回應。

持久性連線

當 CloudFront 從您的原始伺服器獲得回應時，會嘗試維持幾秒鐘連線，以確保在該段期間內另一個請求送達。維護持久性連線可節省重新建立 TCP 連線所需的時間，並為後續請求執行另一個 TLS 交握。

如需包括如何設定持續連線時間的詳細資訊，請參閱[保持連線逾時 \(僅限自訂與 VPC 原始伺服器\)](#)一節的[所有分佈設定參考](#)。

通訊協定

CloudFront 根據下列項目轉送 HTTP 或 HTTPS 請求到原始伺服器：

- 檢視器傳送請求到 CloudFront 的通訊協定 (HTTP 或 HTTPS)。
- CloudFront 主控台中 Origin Protocol Policy (原始伺服器通訊協定政策) 欄位的值，或者，如果使用 CloudFront API，則為 `OriginProtocolPolicy` 複雜類型中的 `DistributionConfig` 元素。在 CloudFront 主控台中，選項為 HTTP Only (僅限 HTTP)、HTTPS Only (僅限 HTTPS) 與 Match Viewer (配合檢視器)。

如果指定 HTTP Only (僅限 HTTP) 或 HTTPS Only (僅限 HTTPS)，則無論檢視器請求中的通訊協定為何，CloudFront 都會使用指定的通訊協定，來將請求轉送到原始伺服器。

如果指定 Match Viewer (配合檢視器)，CloudFront 會使用檢視器請求中的通訊協定，來將請求轉送到原始伺服器。請注意 CloudFront 僅會在檢視器使用 HTTP 和 HTTPS 通訊協定發出請求時，快取物件一次。

Important

如果 CloudFront 使用 HTTPS 通訊協定來將請求轉送到原始伺服器，而且原始伺服器傳回無效的憑證或自簽憑證，CloudFront 會結束 TCP 連線。

如需有關如何使用 CloudFront 主控台更新分佈的詳細資訊，請參閱[更新分佈](#)。如需有關如何使用 CloudFront API 更新分佈的詳細資訊，請移至《Amazon CloudFront API 參考》中的[UpdateDistribution](#)。

查詢字串

您可以設定 CloudFront 是否轉送查詢字串參數至您的原始伺服器。如需詳細資訊，請參閱[根據查詢字串參數快取內容](#)。

原始伺服器連線逾時和嘗試次數

「原始伺服器連線逾時」是嘗試建立與原始伺服器連線時 CloudFront 所等待的秒數。

「原始伺服器連線嘗試次數」是 CloudFront 嘗試連線至原始伺服器的次數。

這些設定共同決定了在容錯移轉至次要原始伺服器 (如果是原始伺服器群組) 或將錯誤回應傳回給檢視器之前，CloudFront 嘗試連線到原始伺服器的時間長度。依預設，CloudFront 會等待 30 秒 (嘗試 3 次，每次 10 秒)，然後再嘗試連線至次要原始伺服器或傳回錯誤回應。您可以指定較短的連線逾時、較少的嘗試次數或兩者，以縮短此時間。

如需詳細資訊，請參閱[控制原始伺服器逾時和嘗試次數](#)。

原始伺服器回應逾時

「原始伺服器回應逾時」，也稱為「原始伺服器讀取逾時」或「原始伺服器請求逾時」，適用於以下兩個數值：

- 在將請求轉送到原始伺服器之後，CloudFront 等待回應的時間 (以秒為單位)。
- CloudFront 在收到來自原始伺服器的回應封包後，並在接收下一個封包前所等待的時間 (以秒為單位)。

CloudFront 行為取決於檢視器請求的 HTTP 方法：

- GET 與 HEAD 請求 – 如果原始伺服器在回應逾時期間未回應或是停止回應，CloudFront 會結束連線。如果指定的[原始伺服器連線嘗試次數](#)超過 1，CloudFront 會再次嘗試取得完整的回應。CloudFront 最多可嘗試 3 次，由「原始伺服器連線嘗試次數」設定的值決定。如果原始伺服器在最後一次嘗試時未回應，則 CloudFront 在收到相同原始伺服器上內容的另一個請求之前不會再嘗試。
- DELETE、OPTIONS、PATCH、PUT 及 POST 請求 – 如果在讀取逾時期間原始伺服器未回應，CloudFront 將會結束連線，並且不再嘗試聯絡原始伺服器。用戶端可以視需要重新提交請求。

如需詳細資訊，包括如何設定原始伺服器回應逾時，請參閱[回應逾時](#)。

相同物件之同步請求 (請求折疊)

如果 CloudFront 邊緣節點收到對物件的請求，而且該物件目前不在快取中或快取物件已過期，CloudFront 會立即將該請求傳送到原始伺服器。不過，如果有對相同物件的同步請求，意即：如果在 CloudFront 接收到第一個請求之前，(具有相同快取金鑰) 其他相同物件的請求抵達邊緣節點，則在將其他請求轉送到原始伺服器之前，CloudFront 將暫停。此短暫的暫停有助於減輕原始伺服器的負載。暫停期間，CloudFront 會將原始伺服器請求的回應傳送至接收到的所有請求。這就是所謂的請求摺疊。在 CloudFront 日誌中，會將第一個請求識別為 x-edge-result-type 欄位中的 Miss，而崩潰的請求會識別為 Hit。如需有關 CloudFront 日誌的詳細資訊，請參閱 [the section called “CloudFront 和邊緣函數記錄”](#)。

CloudFront 只會摺疊共用[快取金鑰](#)的請求。如果其他的請求沒有共用快取金鑰，(例如，因為您設定 CloudFront 根據請求標頭或查詢字串進行快取)，則 CloudFront 會將所有獨特的請求轉送到原始伺服器。

如果您想要防止所有請求摺疊，可使用受管快取政策 CachingDisabled (也可防止快取)。如需詳細資訊，請參閱[使用受管快取政策](#)。

若您想防止特定物件的請求折疊，您可以將快取行為的最短 TTL 設為 0 並設定原始伺服器傳送 Cache-Control: private、Cache-Control: no-store、Cache-Control: no-cache、Cache-Control: max-age=0 或 Cache-Control: s-maxage=0。這些組態會增加原始伺服器的負載，並於 CloudFront 等待第一個請求的回應時，為暫停的同步請求引入額外的延遲。

Important

目前如果您在[快取政策](#)、[原始伺服器請求政策](#)或舊版快取設定中啟用 Cookie 轉送，CloudFront 不支援請求摺疊。

User-Agent 標頭

如果您希望 CloudFront 根據使用者使用以檢視內容的裝置來快取不同的物件版本，我們建議您設定 CloudFront，以將下列其中一或多個標頭轉送至自訂原始伺服器：

- CloudFront-Is-Desktop-Viewer
- CloudFront-Is-Mobile-Viewer
- CloudFront-Is-SmartTV-Viewer
- CloudFront-Is-Tablet-Viewer

根據 User-Agent 標頭值，CloudFront 將這些標頭值設定為 true 或 false，然後將請求轉送至原始伺服器。如果裝置屬於多個類別，一個以上的值可能是 true。例如，針對一些平板電腦裝置，CloudFront 可能將 CloudFront-Is-Mobile-Viewer 與 CloudFront-Is-Tablet-Viewer 同時設定為 true。如需有關如何設定 CloudFront 根據請求標頭進行快取的詳細資訊，請參閱[根據請求標頭快取內容](#)。

您可以設定 CloudFront 根據 User-Agent 標頭中的值來快取物件，但我們不建議這麼做。User-Agent 標頭有許多可能的值，且根據這些值進行快取可能導致 CloudFront 轉送更多請求到原始伺服器。

如果未設定 CloudFront 根據 User-Agent 標頭中的值來快取物件，CloudFront 會在轉送請求到您的原始伺服器之前，加入具有下列值的 User-Agent 標頭：

User-Agent = Amazon CloudFront

無論檢視器所傳來的請求是否包含 User-Agent 標頭，CloudFront 都會加入此標頭。如果檢視器所傳來的請求包含 User-Agent 標頭，CloudFront 會移除此標頭。

CloudFront 如何處理來自您的自訂原始伺服器的回應

瞭解 CloudFront 如何處理來自您的自訂原始伺服器的回應。

內容

- [100 Continue 回應](#)
- [快取](#)
- [已取消請求](#)
- [內容議價](#)

- [Cookie](#)
- [捨棄 TCP 連線](#)
- [CloudFront 移除或取代的 HTTP 回應標頭](#)
- [可快取檔案大小上限](#)
- [原始伺服器無法使用](#)
- [重新引導](#)
- [Transfer-Encoding 標頭](#)

100 Continue 回應

您的原始伺服器無法傳送一個以上的 100-Continue 回應給 CloudFront。在第一個 100-Continue 回應之後，CloudFront 會預期 HTTP 200 OK 回應。如果您的原始伺服器在第一個 100-Continue 回應之後傳送另一個回應，CloudFront 將傳回錯誤。

快取

- 確保原始伺服器集為有效且為 Date 和 Last-Modified 標頭欄位準確的值。
- CloudFront 通常會遵循原始伺服器回應中的 Cache-Control: no-cache 標頭。如需例外，請參閱[相同物件之同步請求 \(請求折疊\)](#)。

已取消請求

如果物件不在邊緣快取中，而且在 CloudFront 從原始伺服器取得物件之後，但尚未傳送請求的物件之前，檢視器就終止了工作階段 (例如，關閉瀏覽器)，則 CloudFront 不會在節點快取物件。

內容議價

如果您的原始伺服器在回應中傳回 Vary: *，而且對應快取行為的 Minimum TTL (最短 TTL) 值為 0，則 CloudFront 會快取物件，但仍會將對於該物件的每個後續請求，轉送到原始伺服器，以確認快取包含物件的最新版本。CloudFront 不包含任何條件式標頭，例如 If-None-Match 或 If-Modified-Since。因此，您的原始伺服器傳回物件至 CloudFront 以回應每個請求。

如果您的原始伺服器在回應中傳回 Vary: *，而且對應快取行為的 Minimum TTL (最短 TTL) 值是任何其他值，則 CloudFront 會根據 Vary 中的說明，來處理 [CloudFront 移除或取代的 HTTP 回應標頭](#) 標頭。

Cookie

如果您為快取行為啟用 Cookie，而且如果原始伺服器依物件傳回 Cookie 時，CloudFront 會快取物件和 Cookie。請注意，這可減少物件的快取能力。如需詳細資訊，請參閱 [根據 Cookie 快取內容](#)。

捨棄 TCP 連線

如果在您的原始伺服器將物件傳回 CloudFront 時，CloudFront 與原始伺服器之間的 TCP 連線中斷，則 CloudFront 行為取決於您的原始伺服器是否在回應中包含 Content-Length 標頭：

- Content-Length 標頭 – CloudFront 會在從您的原始伺服器取得物件時，將物件傳回給檢視器。不過，如果 Content-Length 標頭的值不符合物件的大小，CloudFront 不會快取物件。
- Transfer-Encoding: Chunked – CloudFront 會在從您的原始伺服器取得物件時，將物件傳回給檢視器。不過，如果區塊回應不完整，CloudFront 不會快取物件。
- 沒有 Content-Length 標頭 – CloudFront 會將物件傳回給檢視器並進行快取，但物件可能會不完整。如果沒有 Content-Length 標頭，CloudFront 就無法判斷 TCP 連線中斷是意外的或刻意的。

我們建議您設定 HTTP 伺服器加入 Content-Length 標頭，來防止 CloudFront 只快取部分物件。

CloudFront 移除或取代的 HTTP 回應標頭

在將原始伺服器傳來的回應轉送給檢視器之前，CloudFront 會移除或更新下列的標頭欄位：

- Set-Cookie – 如果設定 CloudFront 轉送 Cookie，此服務會將 Set-Cookie 標頭欄位轉送給用戶端。如需詳細資訊，請參閱 [根據 Cookie 快取內容](#)。
- Trailer
- Transfer-Encoding – 如果原始伺服器傳回此標頭欄位，CloudFront 會將此值設定為 chunked，再將回應傳回給檢視器。
- Upgrade
- Vary – 請注意以下各項：
 - 如果設定 CloudFront 將任何裝置特定的標頭，轉送到您的原始伺服器 (CloudFront-Is-Desktop-Viewer、CloudFront-Is-Mobile-Viewer、CloudFront-Is-SmartTV-Viewer、CloudFront-Is-Tablet-Viewer)，而且設定原始伺服器將 Vary:User-Agent 傳回給 CloudFront，則 CloudFront 會將 Vary:User-Agent 傳回給檢視器。如需詳細資訊，請參閱 [根據裝置類型設定快取](#)。
 - 如果設定原始伺服器，在 Accept-Encoding 標頭中加入 Cookie 或 Vary，則 CloudFront 會在給檢視器的回應中加入這些值。

- 如果您設定 CloudFront 將標頭轉送到原始伺服器，並且設定原始伺服器將 Vary 標頭中的標頭名稱傳回給 CloudFront (例如，Vary:Accept-Charset,Accept-Language)，則 CloudFront 會將 Vary 標頭及這些值傳回給瀏覽者。
- 如需有關 CloudFront 如何處理 * 標頭中 Vary 值的詳細資訊，請參閱[內容議價](#)。
- 如果設定原始伺服器在 Vary 標頭中加入其他任何值，CloudFront 會移除這些值，再將回應傳回給檢視器。
- Via – CloudFront 在對檢視器的回應中將值設為以下值：

Via: *http-##-##-##.cloudfront.net* (CloudFront)

值的範例如下所示：

Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)

可快取檔案大小上限

CloudFront 在快取中儲存의 回應內文大小上限為 50 GB。此包含未指定 Content-Length 標頭值的區塊傳輸回應。

使用範圍請求方法來請求每個部分為 50 GB 或更小的多部分物件，即可使用 CloudFront 快取大於此大小的物件。CloudFront 會快取這些部分，因為每個部分皆為 50 GB 或更小。檢視器擷取物件的所有部分之後，就可以重建原始、較大的物件。如需詳細資訊，請參閱[使用範圍請求快取大物件](#)。

原始伺服器無法使用

如果您的原始伺服器無法使用，而且 CloudFront 收到了請求，要求邊緣快取中已過期的物件 (例如，因為 Cache-Control max-age 指令中所指定的期間已過)，則 CloudFront 會提供該物件已過期的版本，或是提供自訂的錯誤頁面。如需有關已設定自訂錯誤頁面時 CloudFront 行為的詳細資訊，請參閱[當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤](#)。

在某些情況下，很少請求的物件會被移出並在節點快取中不可再用。CloudFront 無法提供已移出的物件。

重新引導

如果您在原始伺服器中變更物件的位置，您可以設定您的 Web 伺服器重新引導請求至新的位置。在您設定重新引導之後，檢視器第一次提交對物件的請求時，CloudFront 會傳送請求到原始伺服器，而原始伺服器會以重新引導回應 (例如 302 Moved Temporarily)。CloudFront 會快取此重新引導並傳回給檢視器。CloudFront 不會遵循重新引導。

您可以設定您的 Web 伺服器重新引導請求以下其中一個位置：

- 在原始伺服器中物件的新 URL。當檢視器遵循重新引導到新 URL 時，檢視器會略過 CloudFront 並直接至原始伺服器。因此，我們建議您不要重新引導請求到原始伺服器中物件的新 URL。
- 物件的新 CloudFront URL。如果檢視器提交請求，其中包含新的 CloudFront URL，CloudFront 會從原始伺服器上的新位置取得物件、在節點快取該物件，然後將物件傳回給檢視器。物件的後續請求會被節點提供。這可避免與從原始伺服器檢視器請求的物件有關的延遲和負載。然而，物件的每個新請求將會產生為兩個請求至 CloudFront 的費用。

Transfer-Encoding 標頭

CloudFront 只支援 chunked 標頭的 Transfer-Encoding 值。如果您的原始伺服器傳回 Transfer-Encoding: chunked，CloudFront 會在節點收到物件時，將物件傳回到用戶端，並以區塊格式快取物件，來回應後續的請求。

如果檢視器提出 Range GET 請求，而原始伺服器傳回 Transfer-Encoding: chunked，CloudFront 會將整個物件 (而非請求的範圍) 傳回至檢視器。

如果您無法預定回應內容的長度，我們建議您使用區塊編碼。如需詳細資訊，請參閱 [捨棄 TCP 連線](#)。

原始伺服器群組的請求和回應行為

請求原始群組的運作方式與請求未設為原始群組的原始伺服器的運作方式相同，但存在原始伺服器容錯移轉時例外。與任何其他原始伺服器相同，當 CloudFront 收到請求且內容已在節點遭到快取時，會從快取將內容提供給檢視器。當發生命中遺漏時，會將檢視器請求轉送到原始伺服器群組中的主要原始伺服器。

主要原始伺服器的請求和回應行為是相同的，因為它是針對不在原始伺服器群組中的原始伺服器。如需詳細資訊，請參閱 [Amazon S3 原始伺服器之請求和回應行為](#) 及 [為自訂原始伺服器之請求和回應行為](#)。

以下描述當主要原始伺服器傳回特定 HTTP 狀態碼時，原始伺服器容錯移轉的行為：

- HTTP 2xx 狀態碼 (成功)：CloudFront 快取檔案並傳回給檢視器。
- HTTP 3xx 狀態碼 (重新引導)：CloudFront 將狀態碼傳回給檢視器。
- HTTP 4xx 和 5xx 狀態碼 (用戶端/伺服器錯誤)：如果已針對容錯移轉設定傳回的狀態碼，則 CloudFront 會傳送相同的請求到原始伺服器群組中的次要原始伺服器。

- HTTP 4xx 和 5xx 狀態碼 (用戶端/伺服器錯誤)：如果未針對容錯移轉設定傳回的狀態碼，則 CloudFront 會將錯誤傳回給檢視器。

只有當檢視器請求的 HTTP 方法為 GET、HEAD 或 OPTIONS 時，CloudFront 才會容錯移轉至次要原始伺服器。當檢視器傳送不同的 HTTP 方法 (例如，POST、PUT 等等) 時，CloudFront 不會容錯移轉。

當 CloudFront 將請求傳送到次要原始伺服器時，回應行為與不在原始伺服器群組中的 CloudFront 原始伺服器相同。

如需原始伺服器群組的詳細資訊，請參閱[透過 CloudFront 原始伺服器容錯移轉最佳化高可用性](#)。

將自訂標頭新增到原始伺服器請求

您可以將 CloudFront 配置為將自訂標頭新增到傳送給您原始伺服器的請求。您可以使用自訂標頭傳送並收集來自原始伺服器的資訊，該資訊無法透過一般檢視器請求取得。您甚至可以針對每個原始伺服器自訂標頭。CloudFront 支援自訂原始伺服器和 Amazon S3 原始伺服器的自訂標頭。

內容

- [使用案例](#)
- [設定 CloudFront 以將自訂標頭新增到原始伺服器請求](#)
- [CloudFront 無法新增到原始伺服器請求的自訂標頭](#)
- [設定 CloudFront 以轉送 Authorization 標頭](#)

使用案例

您可以使用自訂標頭，例如下列範例：

識別來自 CloudFront 的請求

您可以識別原始伺服器從 CloudFront 收到的請求。如果您想知道使用者正繞過 CloudFront，還是您正使用多個 CDN，並且想知道來自每個 CDN 的請求的資訊，這會是很實用的方法。

Note

如果您使用的是 Amazon S3 原始伺服器，並且啟用 [Amazon S3 伺服器存取記錄](#)，則日誌不包含標頭資訊。

判斷哪些請求是來自特定分發

如果您將多個 CloudFront 分發配置為使用同一個原始伺服器，您可以在每個分發中新增不同的自訂標頭。然後，您可以使用來自原始伺服器的日誌，判斷哪些請求來自哪個 CloudFront 分發。

啟用跨來源資源分享 (CORS)

如果某些檢視器不支援跨原始伺服器資源分享 (CORS)，您可以將 CloudFront 配置為一律將 Origin 標頭新增到傳送給您原始伺服器的請求。接著，您可以將原始伺服器設定為針對每個請求傳回 Access-Control-Allow-Origin 標頭。您也必須[配置 CloudFront，才能遵循 CORS 設定](#)。

控制內容的存取

您可以使用自訂標頭來控制內容的存取。您可以將原始伺服器配置為只有在包含 CloudFront 所新增的自訂標頭時才回應請求，藉此防止使用者繞過 CloudFront 並直接在原始伺服器上存取您的內容。如需更多詳細資訊，請參閱[在自訂原始伺服器上限制存取檔案](#)。

設定 CloudFront 以將自訂標頭新增到原始伺服器請求

若要將分佈設定為將自訂標頭新增到傳送給您原始伺服器的請求，請使用下列其中一種方法來更新原始伺服器設定：

- CloudFront 主控台 – 當您建立或更新分佈時，請在新增自訂標頭設定中，指定標頭的名稱和值。如需更多詳細資訊，請參閱[新增自訂標頭](#)。
- CloudFront API – 針對您要為其新增自訂標頭的每個原始伺服器，在 Origin 內的 CustomHeaders 欄位中指定標頭名稱和值。如需詳細資訊，請參閱《Amazon CloudFront API 參考》中的[CreateDistribution](#) 或 [UpdateDistribution](#)。

如果您指定的標頭名稱和值尚未出現在檢視器請求中，CloudFront 會將它們新增到原始伺服器請求中。如果存在標頭，則在將請求轉送到原始伺服器之前，CloudFront 將覆寫標頭值。

如需套用至原始伺服器自訂標頭的配額，請參閱[標頭的配額](#)。

CloudFront 無法新增到原始伺服器請求的自訂標頭

您無法將 CloudFront 配置為將任何以下標頭新增到傳送給您原始伺服器的請求：

- Cache-Control
- Connection

- Content-Length
- Cookie
- Host
- If-Match
- If-Modified-Since
- If-None-Match
- If-Range
- If-Unmodified-Since
- Max-Forwards
- Pragma
- Proxy-Authenticate
- Proxy-Authorization
- Proxy-Connection
- Range
- Request-Range
- TE
- Trailer
- Transfer-Encoding
- Upgrade
- Via
- 以 X-Amz- 做為開頭的標頭
- 以 X-Edge- 做為開頭的標頭
- X-Real-Ip

設定 CloudFront 以轉送 **Authorization** 標頭

當 CloudFront 將檢視器請求轉送至您的原始伺服器時，CloudFront 預設會移除某些檢視器標頭，包括 Authorization 標頭。為了確保您的原始伺服器始終收到原始伺服器請求中的 Authorization 標頭，您有以下選項：

- 使用快取政策將 Authorization 標頭新增至快取金鑰。快取金鑰中的所有標頭都會自動包含在原始伺服器請求中。如需更多詳細資訊，請參閱 [使用政策控制快取金鑰](#)。

- 使用將所有檢視器標頭轉寄至原始伺服器的原始伺服器請求政策。您無法在原始伺服器請求政策中個別轉寄 Authorization 標頭，但當您轉寄所有檢視器標頭時，CloudFront 會在檢視器請求中包含 Authorization 標頭。CloudFront 為此使用案例提供受管的原始伺服器請求政策，稱為 Managed-AllViewer。如需更多詳細資訊，請參閱 [使用受管原始伺服器請求政策](#)。

CloudFront 如何處理物件的部分請求 (範圍 GET)

針對大型物件，檢視器 (Web 瀏覽器或用戶端) 可能提出多個 GET 請求，並使用 Range 請求標頭下載分為多個較小部分的物件。這些位元範圍的請求，有時稱為 Range GET 請求，改善部分下載的效率與從部分失敗的傳輸復原。

當 CloudFront 收到 Range GET 請求時，會檢查收到請求之節點中的快取。如果節點中的快取已包含整個物件或物件的請求部分，則 CloudFront 會立即從快取提供請求的範圍。

如果快取不包含請求的範圍，則 CloudFront 會轉送請求到原始伺服器。(若要最佳化效能，CloudFront 請求的範圍可能會比用戶端在 Range GET 中請求的更大。) 接下來，會發生什麼狀況取決於原始伺服器是否支援 Range GET 請求：

- 如果原始伺服器支援 **Range GET** 請求：會傳回請求的範圍。CloudFront 會提供請求的範圍，並同時進行快取來回應未來的請求。(如同許多 HTTP 伺服器一樣，Amazon S3 支援 Range GET 請求。)
- 如果原始伺服器不支援 **Range GET** 請求：會傳回整個物件。CloudFront 會針對目前的請求傳送整個物件，並同時進行快取來回應未來的請求。CloudFront 在邊緣快取中快取整個物件之後，會提供請求的範圍以回應新的 Range GET 請求。

在這兩種情況下，只要第一位元組從原始伺服器送達，CloudFront 就會立即開始提供請求的範圍或物件給最終使用者。

Note

如果檢視器提出 Range GET 請求，而原始伺服器傳回 Transfer-Encoding: chunked，CloudFront 會將整個物件 (而非請求的範圍) 傳回至檢視器。

一般來說，CloudFront 遵循適用於 Range 標頭的 RFC 規格。但是，如果您的 Range 標頭未遵循下列要求，CloudFront 會傳回包含完整物件的 HTTP 狀態碼 200，而非包含指定範圍的狀態碼 206：

- 必須以遞增順序列出的範圍。例如，100-200,300-400 是有效的，300-400,100-200 是無效的。
- 範圍不得重疊。例如，100-200,150-250 是無效的。
- 所有範圍規格必須有效。例如，您無法指定負值為範圍的一部分。

如需 Range 請求標頭的更多資訊，請參閱 RFC 7233 中的[範圍請求](#)，或者參閱 MDN Web 文件中的[範圍](#)。

使用範圍請求快取大物件

啟用快取時，CloudFront 不會擷取或快取大於 50 GB 的物件。當原始伺服器指出物件大於此大小時 (在 Content-Length 回應標頭中)，CloudFront 會關閉與原始伺服器的連線，並傳回錯誤給檢視器。(停用快取時，CloudFront 可以從原始伺服器擷取大於此大小的物件，並將其傳遞給檢視器。但是，CloudFront 不會快取物件。)

然而，藉助範圍請求，您可以使用 CloudFront 來快取大於[可快取檔案大小上限](#)的物件。

Example 範例

1. 考慮具有 100 GB 物件的原始伺服器。啟用快取時，CloudFront 不會擷取或快取如此大的物件。不過，檢視器可以傳送多個範圍請求，以擷取多個部分的物件，其中每個部分都小於 50 GB。
2. 檢視器可以請求包含 20 GB 部分的物件，方法是傳送帶有標頭 Range: bytes=0-21474836480 的請求來擷取第一個部分，然後傳送帶有標頭 Range: bytes=21474836481-42949672960 的另一個請求來擷取下一個部分，依此類推。
3. 檢視器收到所有的部分之後，它可以將這些部分組合起來建構原始的 100 GB 物件。
4. 在此情況下，CloudFront 會快取物件的每個 20 GB 部分，並且可以回應快取中相同部分的後續要求。

對於壓縮物件的範圍請求，位元組範圍請求是根據壓縮大小，而不是物件的原始大小。如需壓縮檔案的相關資訊，請參閱[提供壓縮檔案](#)。

CloudFront 如何處理來自原始伺服器的 HTTP 3xx 狀態碼。

當 CloudFront 向您的 Amazon S3 儲存貯體或自訂原始伺服器請求物件時，原始伺服器有時會傳回 HTTP 3xx 狀態碼。這通常代表下列其中一項：

- 物件的 URL 已變更 (例如，狀態碼 301、302、307 或 308)
- 自上次 CloudFront 請求該物件以來，該物件不曾變更過 (狀態碼 304)

CloudFront 會根據您的 CloudFront 分佈中的設定和回應中的標頭來快取 3xx 回應。只有在您將 Cache-Control 標頭包含在原始伺服器回應中時，CloudFront 才會快取 307 和 308 回應。如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

如果您的原始伺服器傳回重新引導狀態碼 (例如，301 或 307)，CloudFront 就不會遵循重新引導。CloudFront 會沿著 301 或 307 回應傳遞至檢視器，而檢視器可以透過傳送新的請求來遵循重新引導。

CloudFront 如何處理來自原始伺服器的 HTTP 4xx 和 5xx 狀態碼

當 CloudFront 向您的 Amazon S3 儲存貯體或自訂原始伺服器請求物件時，您的原始伺服器有時會傳回 HTTP 4xx 或 5xx 狀態碼，這表示錯誤已發生。CloudFront 行為取決於：

- 您是否已設定自訂錯誤頁面
- 您是否已設定讓 CloudFront 快取原始伺服器錯誤回應的時間長度 (錯誤快取最短 TTL)
- 狀態碼
- 對於 5xx 狀態碼，請求的物件目前是否在 CloudFront 邊緣快取
- 對於某些 4xx 狀態代碼，原始伺服器是否傳回 Cache-Control max-age 或 Cache-Control s-maxage 標頭

CloudFront 一律會快取 GET 與 HEAD 請求的回應。您也可以設定 CloudFront 快取 OPTIONS 請求的回應。CloudFront 不會快取對使用其他方法之請求的回應。

如果原始伺服器未回應，原始伺服器的 CloudFront 請求則會逾時，而此逾時會被視為是原始伺服器的 HTTP 5xx 錯誤，即使原始伺服器未以該錯誤回應。在這種情況下，CloudFront 會繼續提供快取內容。如需詳細資訊，請參閱[原始伺服器無法使用](#)。

如果您已啟用記錄日誌，則不論 HTTP 狀態碼為何，CloudFront 都會將結果寫入日誌。

如需與從 CloudFront 傳回之錯誤訊息相關的詳細功能和選項資訊，請查看以下內容：

- 如需有關 CloudFront 主控台中自訂錯誤頁面設定的詳細資訊，請參閱[自訂錯誤頁面和錯誤快取](#)。
- 如需有關 CloudFront 主控台中錯誤快取最短 TTL 的詳細資訊，請參閱[錯誤快取最短 TTL \(秒\)](#)。

- 如需 CloudFront 快取的 HTTP 狀態碼清單，請參閱 [CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼](#)。

主題

- [當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤](#)
- [您尚未設定自訂錯誤頁面時 CloudFront 如何處理錯誤](#)
- [CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼](#)

當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤

如果您已設定自訂錯誤頁面，CloudFront 行為取決於請求的物件是否在邊緣快取中。

請求的物件不在邊緣快取中

CloudFront 會在下列所有情況皆成立時，繼續嘗試從您的原始伺服器取得請求的物件：

- 一個檢視器請求了一個物件。
- 該物件未在節點快取中。
- 您的原始伺服器會傳回 HTTP 4xx 或 5xx 狀態碼，且下列其中一項為真：
 - 您的原始伺服器會傳回 HTTP 5xx 狀態碼，而不傳回 304 狀態碼 (未修改) 或物件的更新版本。
 - 您的原始伺服器會傳回 HTTP 4xx 狀態碼，且不限於快取控制標頭，並包含在以下狀態碼清單中：[CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼](#)。
 - 您的原始伺服器會傳回包含 Cache-Control max-age 標頭或 Cache-Control s-maxage 標頭的 HTTP 4xx 狀態碼，且該狀態碼包含在以下狀態碼清單中：控制 [CloudFront 根據 Cache-Control 標頭快取的 HTTP 4xx 狀態碼](#)。

CloudFront 會執行下列作業：

1. 在收到檢視器請求的 CloudFront 邊緣快取中，CloudFront 會檢查您的分佈設定，並取得自訂錯誤頁面的路徑，此頁面對應於原始伺服器所傳回的狀態碼。
2. CloudFront 在有相符自訂錯誤頁面路徑的路徑模式之分佈中找到第一個快取行為。
3. CloudFront 節點將自訂錯誤頁面的請求傳送到快取行為中指定的原始伺服器。
4. 原始伺服器傳回自訂錯誤頁面至節點。
5. CloudFront 將自訂錯誤頁面傳回至提出請求的檢視器，並同時快取自訂錯誤頁面達以下最大值：
 - 由錯誤快取最短 TTL (預設為 10 秒) 指定的時間數

- 當第一個請求產生錯誤時，由原始伺服器傳回，並由 `Cache-Control max-age` 標頭或 `Cache-Control s-maxage` 標頭指定的時間量
6. 在快取時間 (於步驟 5 中決定) 經過之後，CloudFront 會藉由將另一個請求轉送到您的原始伺服器，再次嘗試取得請求的物件。CloudFront 會根據錯誤快取最短 TTL 指定的間隔，繼續進行重試。

Note

如果您也為相同的自訂錯誤頁面設定快取行為，CloudFront 會改用快取行為 TTL。在此情況下，CloudFront 會針對步驟 5 和 6 執行下列動作：

- CloudFront 將自訂錯誤頁面傳回給提出請求的檢視器後，CloudFront 會檢查快取行為 TTL (例如您將預設 TTL 設定為 5 秒)。然後 CloudFront 會快取自訂錯誤頁面至該上限。
- 經過 5 秒後，CloudFront 會再次從原始伺服器擷取自訂錯誤頁面。CloudFront 會根據快取行為 TTL 指定的間隔繼續進行重試。

如需更多詳細資訊，請參閱 [TTL 設定](#) 中的快取行為。

請求的物件在邊緣快取中

當下列所有情況皆成立時，CloudFront 會繼續提供目前在邊緣快取中的物件：

- 一個檢視器請求了一個物件。
- 物件在節點快取但已過期
- 您的原始伺服器會傳回 HTTP 5xx 狀態碼，而不傳回 304 狀態碼 (未修改) 或物件的更新版本。

CloudFront 會執行下列作業：

1. 如果您的原始伺服器傳回 5xx 狀態碼，即使物件已過期，CloudFront 仍會提供物件。在錯誤快取最短 TTL 期間，CloudFront 會藉由從邊緣快取提供物件來繼續回應檢視器的請求。

如果您的原始伺服器傳回 4xx 狀態碼，CloudFront 會將狀態碼 (而非請求的物件) 傳回給檢視器。

2. 在錯誤快取最低 TTL 經過之後，CloudFront 會藉由將另一個請求轉送到您的原始伺服器，再次嘗試取得請求的物件。請注意，如果未經常請求物件，CloudFront 可能會在您的原始伺服器仍然傳回 5xx 回應時，將該物件從邊緣快取移出。如需有關物件在 CloudFront 邊緣快取中留存時間長度的詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

您尚未設定自訂錯誤頁面時 CloudFront 如何處理錯誤

如果您沒有設定自訂錯誤頁面，CloudFront 行為取決於請求的物件是否在邊緣快取中。

主題

- [請求的物件不在邊緣快取中](#)
- [請求的物件在邊緣快取中](#)

請求的物件不在邊緣快取中

CloudFront 會在下列所有情況皆成立時，繼續嘗試從您的原始伺服器取得請求的物件：

- 一個檢視器請求了一個物件。
- 該物件未在節點快取中。
- 您的原始伺服器會傳回 HTTP 4xx 或 5xx 狀態碼，且下列其中一項為真：
 - 您的原始伺服器會傳回 HTTP 5xx 狀態碼，而不傳回 304 狀態碼 (未修改) 或物件的更新版本。
 - 您的原始伺服器會傳回 HTTP 4xx 狀態碼，且不限於快取控制標頭，並包含在以下狀態碼清單中：[CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼](#)
 - 您的原始伺服器會傳回包含 Cache-Control max-age 標頭或 Cache-Control s-maxage 標頭的 HTTP 4xx 狀態碼，且該狀態碼包含在以下狀態碼清單中：控制 [CloudFront 根據 Cache-Control 標頭快取的 HTTP 4xx 狀態碼](#)。

CloudFront 會執行下列作業：

1. CloudFront 會將 4xx 或 5xx 狀態碼傳回給檢視器，並同時在收到請求的邊緣快取中快取狀態碼達以下最大值：
 - 由錯誤快取最短 TTL (預設為 10 秒) 指定的時間數
 - 當第一個請求產生錯誤時，由原始伺服器傳回，並由 Cache-Control max-age 標頭或 Cache-Control s-maxage 標頭指定的時間量
2. 在快取時間 (於步驟 1 中決定) 期間，CloudFront 會以快取的 4xx 或 5xx 狀態碼，回應對相同物件的後續檢視器請求。
3. 在快取時間 (於步驟 1 中決定) 經過之後，CloudFront 會藉由將另一個請求轉送到您的原始伺服器，再次嘗試取得請求的物件。CloudFront 會根據錯誤快取最短 TTL 指定的間隔，繼續進行重試。

請求的物件在邊緣快取中

當下列所有情況皆成立時，CloudFront 會繼續提供目前在邊緣快取中的物件：

- 一個檢視器請求了一個物件。
- 物件在節點快取但已過期 這表示物件已過時。
- 您的原始伺服器會傳回 HTTP 5xx 狀態碼，而不傳回 304 狀態碼 (未修改) 或物件的更新版本。

CloudFront 會執行下列作業：

1. 如果您的原始伺服器傳回 5xx 錯誤碼，即使物件已過期，CloudFront 仍會提供物件。在錯誤快取最短 TTL (預設 10 秒) 期間，CloudFront 會藉由從邊緣快取提供物件來繼續回應檢視器的請求。

如果您的原始伺服器傳回 4xx 狀態碼，CloudFront 會將狀態碼 (而非請求的物件) 傳回給檢視器。

2. 在錯誤快取最低 TTL 經過之後，CloudFront 會藉由將另一個請求轉送到您的原始伺服器，再次嘗試取得請求的物件。如果未經常請求物件，CloudFront 可能會在您的原始伺服器仍然傳回 5xx 回應時，將該物件從邊緣快取移出。如需詳細資訊，請參閱[管理內容保持在快取中達多久時間 \(過期\)](#)。

Tip

- 如果您設定 `stale-if-error` 或 `Stale-While-Revalidate` 指令，可以指定過時物件在邊緣快取中可用的時間長度。這可讓您繼續為檢視器提供內容，即使您的原始伺服器無法使用。如需相關資訊，請參閱[提供過時 \(過期\) 內容](#)。
- CloudFront 只會提供已過時達到指定[最大 TTL](#) 值的物件。在此持續時間之後，物件將無法從邊緣快取使用。

CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼

視傳回的特定狀態碼，以及您的原始伺服器是否在回應中傳回特定標頭而定，CloudFront 會快取您的原始伺服器所傳回的 HTTP 4xx 和 5xx 狀態碼。

CloudFront 會快取您原始伺服器所傳回的下列 HTTP 4xx 和 5xx 狀態碼。如果您為 HTTP 狀態碼設定了自訂錯誤頁面，CloudFront 會快取自訂錯誤頁面。

 Note

如果您使用的是 [CachingDisabled](#) 受管快取政策，CloudFront 不會快取這些狀態碼或自訂錯誤頁面。

404	找不到
414	URI 請求過大。
500	內部伺服器錯誤
501	未導入
502	無效的閘道
503	服務無法使用
504	閘道逾時

CloudFront 根據 **Cache-Control** 標頭快取的 HTTP 4xx 狀態碼

如果您的原始伺服器傳回 `Cache-Control max-age` 或 `Cache-Control s-maxage` 標頭，則 CloudFront 只會快取您原始伺服器所傳回的下列 HTTP 4xx 狀態碼。如果您為其中一個 HTTP 狀態碼設定了自訂錯誤頁面 (而且您的原始伺服器傳回其中一個快取控制標頭)，則 CloudFront 會快取自訂錯誤頁面。

400	錯誤的請求。
403	禁止

405	方法不允許
412 ¹	先決條件失敗
415 ¹	不支援的媒體類型

¹CloudFront 不支援為這些 HTTP 狀態碼建立自訂錯誤頁面。

產生自訂錯誤回應

如果透過 CloudFront 提供的物件出於某些原因無法使用，則 Web 伺服器通常會傳回相關的 HTTP 狀態碼到 CloudFront，以指明此情形。例如若檢視器請求無效的 URL，則 Web 伺服器會傳回 HTTP 404 (找不到) 狀態碼到 CloudFront，然後 CloudFront 會傳回該狀態碼給檢視器。您可以建立 CloudFront 傳回給檢視器的自訂錯誤回應，而不是使用此預設錯誤回應。

如果您設定 CloudFront 傳回 HTTP 狀態碼的自訂錯誤頁面，但自訂錯誤頁面不可用，則 CloudFront 傳回 CloudFront 從包含自訂錯誤頁面的原始伺服器接收到的狀態碼給檢視器。例如，假設您的自訂原始伺服器會傳回一個 500 狀態碼且您已配置 CloudFront 以從 Amazon S3 儲存貯體取得 500 狀態碼的自訂錯誤頁面。不過有人不小心從 Amazon S3 儲存貯體刪除了自訂錯誤頁面。CloudFront 會傳回 HTTP 404 狀態碼 (找不到) 到請求物件的檢視器。

當 CloudFront 傳回自訂錯誤頁面給檢視器時，您支付自訂錯誤頁面的標準 CloudFront 費用，而非請求的物件費用。如需 CloudFront 費用的詳細資訊，請參閱 [Amazon CloudFront 定價](#)。

主題

- [設定錯誤回應行為](#)
- [針對特定的 HTTP 狀態碼建立自訂錯誤頁面](#)
- [將物件和自訂錯誤頁面存放在不同位置](#)
- [變更 CloudFront 傳回的回應碼](#)
- [控制 CloudFront 快取錯誤的時間長度](#)

設定錯誤回應行為

您可透過幾種選項來管理 CloudFront 在發生錯誤時的回應方式。若要設定自訂錯誤回應，您可以使用 CloudFront 主控台、CloudFront API 或 CloudFormation。無論您選擇如何更新組態，請考慮下列提示和建議：

- 在 CloudFront 可存取的位置中儲存自訂錯誤頁面。我們建議您將它們存放在 Amazon S3 儲存貯體中，並且不要將它們儲存在與網站或應用程式內容之其餘部分相同的位置。如果您將自訂錯誤頁面儲存在與您的網站或應用程式相同的原始伺服器，且原始伺服器開始傳回 5xx 錯誤，則 CloudFront 無法取得自訂錯誤頁面，因為原始伺服器無法使用。如需詳細資訊，請參閱 [將物件和自訂錯誤頁面存放在不同位置](#)。
- 請確定 CloudFront 具有取得自訂錯誤頁面的許可。如果自訂錯誤頁面存放於 Amazon S3 中，頁面必須可以公開存取，或者您必須設定 CloudFront [原始存取控制 \(OAC\)](#)。如果自訂錯誤頁面存放在自訂原始伺服器中，則必須可以公開存取頁面。
- (選用) 如果需要，請設定您的原始伺服器以新增 Cache-Control 或 Expires 標頭以及自訂錯誤頁面。您也可以使用 Error Caching Minimum TTL (快取最小 TTL 時發生錯誤) 設定來控制 CloudFront 快取自訂錯誤頁面的時間長度。如需詳細資訊，請參閱[控制 CloudFront 快取錯誤的時間長度](#)。

設定自訂錯誤回應

若要在 CloudFront 主控台中設定自訂錯誤回應，您必須擁有 CloudFront 分發。在主控台中，自訂錯誤回應的組態設定僅適用於現有分發。若要了解如何建立分發，請參閱[開始使用 CloudFront 標準分佈](#)。

Console

若要設定自訂錯誤回應 (主控台)

1. 登入 AWS 管理主控台，並在位於的 CloudFront 主控台中開啟分佈頁面<https://console.aws.amazon.com/cloudfront/v4/home#distributions>。
2. 在分發清單中，選擇要更新的分發。
3. 選擇 Error Pages (錯誤頁面) 標籤，然後選擇 Create Custom Error Response (建立自訂錯誤回應)。
4. 輸入適用的值。如需詳細資訊，請參閱[自訂錯誤頁面和錯誤快取](#)。
5. 輸入所需的值後，選擇 Create (建立)。

CloudFront API or CloudFormation

若要使用 CloudFront API 或 設定自訂錯誤回應 CloudFormation，請在分佈中使用 CustomErrorResponse 類型。如需詳細資訊，請參閱下列內容：

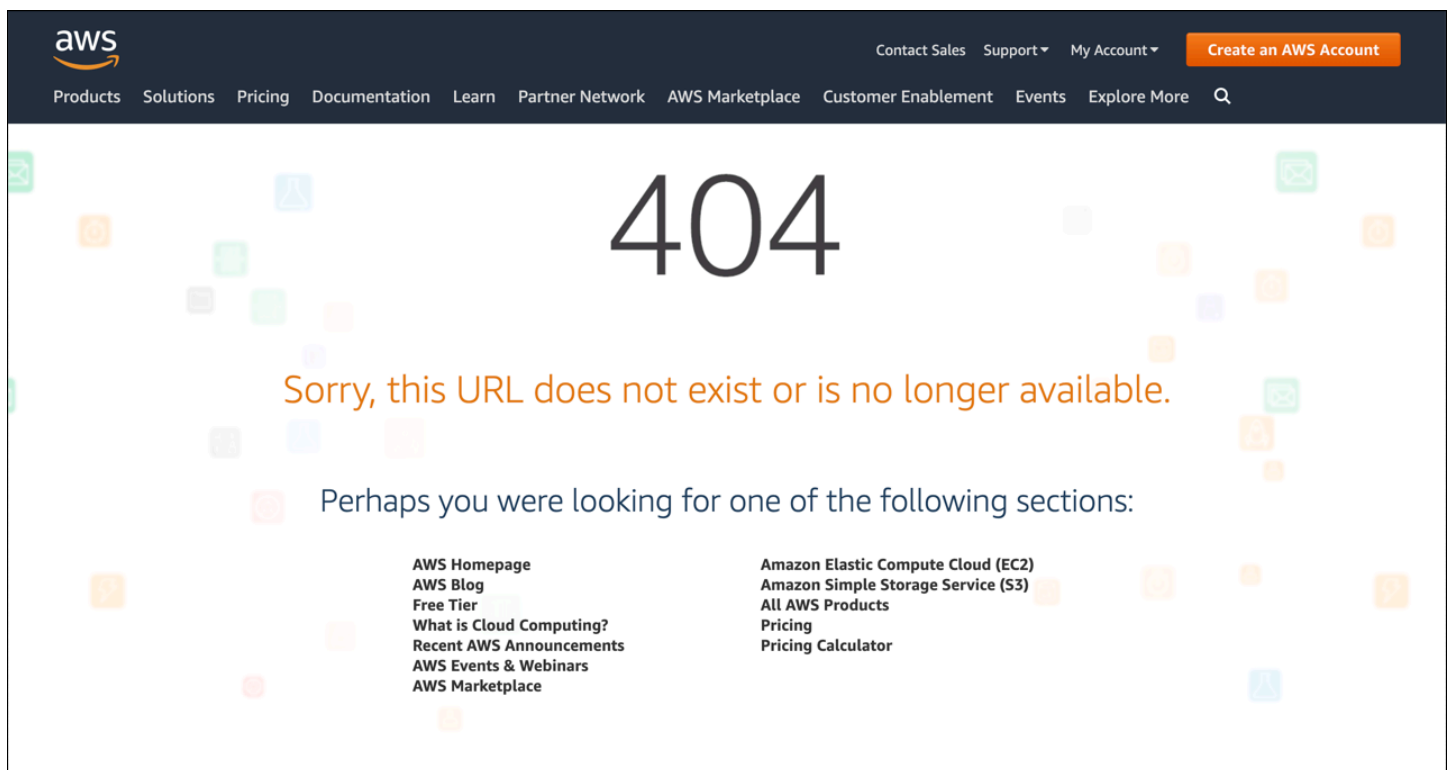
- AWS CloudFormation 使用者指南中的 [AWS::CloudFront::Distribution CustomErrorResponse](#)
- 《Amazon CloudFront API 參考》中的 [CustomErrorResponse](#)

針對特定的 HTTP 狀態碼建立自訂錯誤頁面

若您想要顯示自訂錯誤訊息而非預設訊息，比方與您的網站其餘部分使用相同格式設定的頁面，則可以讓 CloudFront 向檢視器傳回包含自訂錯誤訊息的物件 (如 HTML 檔案)。

若要指定您想要傳回的檔案，以及指明檔案應該傳回的錯誤，您可以更新 CloudFront 分發以指定這些值。如需詳細資訊，請參閱 [設定錯誤回應行為](#)。

例如，以下是自訂錯誤頁面：



您可以為每個支援的 HTTP 狀態碼指定不同的物件，或您可以為所有支援的狀態碼使用相同的物件。您可以選擇為某些狀態碼指定自訂錯誤頁面，而不是為其他狀態碼指定。

由於各種原因，透過 CloudFront 提供的物件無法使用。這些分為兩大類：

- 用戶端錯誤顯示請求有問題。例如，有指定名稱的物件不可用，或使用者沒有在 Amazon S3 儲存貯體中取得物件所需的許可。當用戶端發生錯誤，原始伺服器會在 4xx 範圍中傳回 HTTP 狀態碼給 CloudFront。
- 伺服器錯誤顯示原始伺服器有問題。例如，HTTP 伺服器是忙碌或不可用。當伺服器發生錯誤，您的原始伺服器會在 5xx 範圍中傳回 HTTP 狀態碼至 CloudFront，或特定期間內 CloudFront 不從您的原始伺服器取得回應並假設一個 504 狀態碼 (閘道逾時)。

針對 HTTP 狀態碼，CloudFront 可以傳回自訂錯誤頁面，其中包含下列項目：

- 400, 403, 404, 405, 414, 416
- 500, 501, 502, 503, 504

備註

- 如果 CloudFront 偵測到請求可能不安全，CloudFront 會傳回 400 (錯誤請求) 錯誤，而不是自訂錯誤頁面。
- 您可以建立自訂錯誤頁面以取得 HTTP 狀態碼 416 (不滿足請求範圍)，當原始伺服器傳回狀態碼 416 給 CloudFront 時，您可以變更 CloudFront 傳回給檢視器的 HTTP 狀態碼。如需詳細資訊，請參閱[變更 CloudFront 傳回的回應碼](#)。然而，CloudFront 不會快取狀態碼 416 回應，因此，即使您為狀態碼 416 指定 Error Caching Minimum TTL (快取最小 TTL 時發生錯誤) 的值，CloudFront 也不會使用它。
- 在某些情況下，CloudFront 不會傳回 HTTP 503 狀態碼的自訂錯誤頁面，即使您將 CloudFront 設定為此。如果 CloudFront 錯誤碼為 Capacity Exceeded 或 Limit Exceeded，則 CloudFront 會將 503 狀態碼傳回給檢視器，而不使用您的自訂錯誤頁面。
- 如果您建立了自訂錯誤頁面，CloudFront 將針對下列回應代碼傳回 Connection: close 或 Connection: keep-alive：
 - CloudFront 會對以下狀態碼傳回 Connection: close：
400、405、414、416、500、501
 - CloudFront 會對以下狀態碼傳回 Connection: keep-alive：
403、404、502、503、504

如需 CloudFront 如何從原始伺服器處理錯誤回應的詳細說明，請參閱 [CloudFront 如何處理來自原始伺服器的 HTTP 4xx 和 5xx 狀態碼](#)。

將物件和自訂錯誤頁面存放在不同位置

如果您要在不同位置存放物件和自訂錯誤頁面，則您的分佈必須包含下列為屬實的快取行為：

- Path Pattern (路徑模式) 的值與自訂錯誤訊息的路徑相符。例如，假設您已在 Amazon S3 儲存貯體名為 `/4xx-errors` 的目錄中儲存了 4xx 錯誤的自訂錯誤頁面。您的分佈必須包含路徑模式路由自訂錯誤頁面請求至該位置的快取行為，例如 `/4xx-errors/*`。
- Origin (原始伺服器) 的數值將 Origin ID (原始伺服器 ID) 的數值指定給包含自訂錯誤頁面的原始伺服器。

如需詳細資訊，請參閱[快取行為設定](#)。

變更 CloudFront 傳回的回應碼

您可以將 CloudFront 設定為傳回不同 HTTP 狀態碼給檢視器，而非 CloudFront 從原始伺服器接收的狀態碼。例如，如果您的原始伺服器傳回 500 狀態碼到 CloudFront，您可能會希望 CloudFront 傳回自訂錯誤頁面和 200 狀態碼 (OK) 給檢視器。基於各種原因，您想要讓 CloudFront 傳回狀態碼到檢視器，該檢視器與原始伺服器傳回到 CloudFront 的檢視器不同：

- 有些網際網路裝置 (例如，一些防火牆和公司代理) 會攔截 HTTP 4xx 和 5xx 狀態碼並禁止回應傳回給檢視器。在此案例中，如果您替換 200，回應不會攔截。
- 如果您不在意區別不同的用戶端錯誤或伺服器錯誤，您可以指定 400 或 500 作為 CloudFront 為所有 4xx 或 5xx 狀態碼傳回的值。
- 您會希望傳回 200 狀態碼 (OK) 與靜態網站，如此您的客戶便不會知道您網站已關閉。

如果您啟用 [CloudFront 標準日誌](#) 並設定 CloudFront 來變更回應中的 HTTP 狀態碼，則日誌中的 `sc-status` 欄位值會包含您指定的狀態碼。不過，`x-edge-result-type` 資料行的值不會受到影響。它包含來自原始伺服器回應的結果類型。例如，當原始伺服器傳回 200 (找不到) 給 CloudFront 時，假設您配置 CloudFront 傳回 404 的狀態碼給檢視器。當原始伺服器回應 404 狀態碼的請求時，在日誌中 `sc-status` 資料行的值會是 200，但 `x-edge-result-type` 資料行的值則會是 `Error`。

您可以配置 CloudFront 傳回下列任何 HTTP 狀態碼以及自訂錯誤頁面：

- 200
- 400, 403, 404, 405, 414, 416
- 500、501、502、503、504

控制 CloudFront 快取錯誤的時間長度

根據預設，CloudFront 會每 10 秒快取錯誤回應。CloudFront 然後將物件的下一個請求提交給您的原始伺服器，以查看導致錯誤的問題是否已解決，並且請求的物件是否可使用。

您可以指定錯誤快取持續時間 錯誤快取最短 TTL 給每個 CloudFront 快取的 4xx 和 5xx 狀態碼。(如需詳細資訊，請參閱 [CloudFront 快取的 HTTP 4xx 和 5xx 狀態碼](#)。) 當您指定持續時間時，請注意以下事項：

- 如果您指定的錯誤快取持續時間不長，則 CloudFront 轉送到原始伺服器的請求，會比如果您指定較長的持續時間時來的多。針對 5xx 錯誤，這可能會讓原本導致原始伺服器傳回錯誤的問題加劇。
- 當您的原始伺服器傳回物件的錯誤時，CloudFront 使用錯誤回應或自訂錯誤頁面來回應物件的請求，直到錯誤快取持續時間過後。如果您指定很長的錯誤快取持續時間，則 CloudFront 可能需要很長的時間使用錯誤回應或自訂錯誤頁面來持續回應請求，到物件再次可用之後。

Note

您可以建立自訂錯誤頁面以取得 HTTP 狀態碼 416 (不滿足請求範圍)，當原始伺服器傳回狀態碼 416 給 CloudFront 時，您可以變更 CloudFront 傳回給檢視器的 HTTP 狀態碼 (如需詳細資訊，請參閱「[變更 CloudFront 傳回的回應碼](#)」。) 然而，CloudFront 不會快取狀態碼 416 回應，因此，即使您為狀態碼 416 指定 Error Caching Minimum TTL (快取最小 TTL 時發生錯誤) 的值，CloudFront 也不會使用它。

如果您想要控制 CloudFront 快取個別物件錯誤的時間長度，您可以配置原始伺服器新增適用的標頭給該物件的錯誤回應。

如果原始伺服器新增 Cache-Control: max-age 或 Cache-Control: s-maxage 指示詞，或 Expires 標頭：CloudFront 將快取在標頭或 Error Caching Minimum TTL (錯誤快取最短 TTL) 中較大的錯誤回應。

Note

Cache-Control: max-age 和 Cache-Control: s-maxage 值不得大於為擷取的錯誤頁面所設定之快取行為的 Maximum TTL (最大 TTL) 值。

如果原始伺服器新增錯誤代碼 404Cache-Control: no-storeCache-Control: no-cache、410、414 或 501 的、或 Cache-Control: private指令，CloudFront 不會快取錯誤回應。對於所有其他錯誤代碼，CloudFront 會忽略 no-store、和 private指令no-cache，並快取錯誤快取最低 TTL 值的錯誤回應。

如果原始伺服器新增其他 Cache-Control 指示詞或未新增標頭：CloudFront 將快取Error Caching Minimum TTL (錯誤快取最短 TTL)值的錯誤回應。

如果物件的 4xx 或 5xx 狀態碼的到期時間超過您想要的時間，且物件可以再次使用，則您可以使用請求的物件 URL 使快取的錯誤碼失效。如果原始伺服器傳回多個物件的錯誤回應，則您需要個別使每一個物件失效。如需有關使物件失效的詳細資訊，請參閱[使檔案失效以移除內容](#)。

如果您已為 S3 儲存貯體原始伺服器啟用快取，且您在 CloudFront 分佈中設定錯誤快取最短 TTL 為 0 秒，則在發生 S3 原始伺服器錯誤的情況下，您仍會看到錯誤快取最短 TTL 為 1 秒。CloudFront 會這麼做是為了保護原始伺服器免受 DDoS 攻擊。這不適用於其他類型的原始伺服器。

新增、移除或取代 CloudFront 分配的內容

本節說明如何確保 CloudFront 能夠存取您要向檢視器提供的內容，如何在您的網站或應用程式中指定物件，以及如何移除或取代內容。

主題

- [新增和存取 CloudFront 分佈的內容](#)
- [使用檔案版本控制來更新或移除具有 CloudFront 分佈的內容](#)
- [自訂 CloudFront 中檔案的 URL 格式](#)
- [指定預設根物件](#)
- [使檔案失效以移除內容](#)
- [提供壓縮檔案](#)

新增和存取 CloudFront 分佈的內容

當您希望 CloudFront 分配內容 (物件) 時，您會將檔案新增至您指定用於分配的原始伺服器之一，並公開該檔案的 CloudFront 連結。CloudFront 節點不會從原始伺服器擷取新檔案，除非節點收到了對檔案發出的檢視器請求。如需詳細資訊，請參閱 [CloudFront 如何交付內容](#)。

當您新增要讓 CloudFront 分配的檔案時，請確保將其新增到分配中所指定的任一個 Amazon S3 儲存貯體，或者若是自訂原始伺服器，則新增到指定的網域內的某個目錄。此外，請確認適用的快取行為中路徑模式是否傳送請求到正確原始伺服器。

例如，假設快取行為的路徑模式是 *.html。若您尚未配置任何其他快取行為以將請求轉送到該原始伺服器，CloudFront 便只會轉送 *.html 檔案。在此情況下，舉例而言，CloudFront 絕不會分配您上傳至原始伺服器的 .jpg 檔案，因為您尚未建立任何包括 .jpg 檔案的快取行為。

CloudFront 伺服器不會就其所提供的物件判斷 MIME 類型。當您上傳檔案到原始伺服器時，建議您為該檔案設定 Content-Type 標頭欄位。

使用檔案版本控制來更新或移除具有 CloudFront 分佈的內容

若要更新 CloudFront 為您設定的現有內容，建議您在檔案名稱或資料夾名稱中使用版本識別碼。這可協助您控制管理 CloudFront 提供的內容。

使用版本控制的檔案名稱更新現有檔案

當您更新 CloudFront 分佈中的現有檔案時，建議您在檔案名稱或目錄名稱中包括某一類的版本識別符，以便更有效控管您的內容。此識別碼可能是日期時間戳記、序號或區分兩個相同物件版本的其他方法。

例如，您可能會命名圖形檔案為 image_1.jpg，而不是命名為 image.jpg。當您想要開始提供檔案的新版本，則命名新檔案名為 image_2.jpg 並更新在 Web 應用程式或網站的連結以指向 image_2.jpg。或者，您可能會將所有圖形放在 images_v1 目錄中，且當您想要開始提供一或多個圖形的新版本時，您可建立新的 images_v2 目錄，並更新連結以指向該目錄。使用版本控制，在 CloudFront 開始提供新版本之前，您不必等到物件過期，且您不需要支付物件失效。

即使您對檔案進行版本控制，仍建議您設定過期日期。如需詳細資訊，請參閱 [管理內容保持在快取中達多久時間 \(過期\)](#)。

Note

指定版本控制的檔案名稱或目錄名稱與 Amazon S3 物件版本控制無關。

移除內容，使 CloudFront 不會對其進行分配

您可以從原始伺服器移除您不再希望納入 CloudFront 分佈中的檔案。不過，CloudFront 將繼續向檢視器顯示邊緣快取中的內容，直到檔案過期。

若您想要立即移除檔案，就必須執行以下其中一項：

- 使用檔案版本控制。當您使用版本控制時，檔案的不同版本將具有不同的名稱，可供您用於 CloudFront 分佈中以更改要傳回給檢視器的檔案。如需更多詳細資訊，請參閱 [使用版本控制的檔案名稱更新現有檔案](#)。
- 使該檔案失效。如需更多詳細資訊，請參閱 [使檔案失效以移除內容](#)。

自訂 CloudFront 中檔案的 URL 格式

在以您希望 CloudFront 提供給檢視器的物件設定原始伺服器之後，您必須在您的網站或應用程式中使用正確的 URL 參照這些物件，以便 CloudFront 提供這些物件。

針對網頁或 Web 應用程式中的物件，您在 URLs 中使用的網域名稱可以是下列任一種：

- 建立分佈時，CloudFront 會自動指派網域名稱，例如 `d111111abcdef8.cloudfront.net`
- 您自己的網域名稱，例如 `example.com`

例如，您可以使用下列其中一個 URLs 傳回檔案 `image.jpg`：

```
https://d111111abcdef8.cloudfront.net/images/image.jpg
```

```
https://example.com/images/image.jpg
```

可以使用相同的 URL 格式無論在 Amazon S3 儲存貯體或自訂原始伺服器存放內容，像是自己的 Web 伺服器其中之一。

Note

URL 格式部分取決於您在分佈中為 Origin Path (原始伺服器路徑) 指定的值。這個值可讓 CloudFront 頂端目錄路徑適用於物件。如需有關建立分佈時設定原始伺服器路徑的詳細資訊，請參閱[原始伺服器路徑](#)。

如需 URL 格式的詳細資訊，請參閱下列各節：

使用您自己的網域名稱 (example.com)

您可以[新增一個更容易的使用替代網域名稱](#) (例如 `example.com`)，而非使用 CloudFront 在建立分佈時指派給您的預設網域名稱。透過使用 CloudFront 設定自己的網域名稱，您可以在分佈中使用像這樣適用於物件的 URL：

```
https://example.com/images/image.jpg
```

如果您計劃在檢視器與 CloudFront 之間使用 HTTPS，請參閱[使用備用網域名稱和 HTTPS](#)。

在 URL 中使用結尾斜線 (/)

當您在 CloudFront 分佈中為目錄指定 URL 時，請選擇要一律使用結尾斜線或一律不使用結尾斜線。例如，只選擇下列其中一種格式來適用於所有 URL：

```
https://d111111abcdef8.cloudfront.net/images/
```

```
https://d111111abcdef8.cloudfront.net/images
```

它為什麼重要？

這兩種格式都可以連結到 CloudFront 物件，但是當您以後要讓目錄失效，連結的一致性可協助防止問題發生。CloudFront 完全按照定義存放 URL，包括結尾斜線。因此，如果格式不一致，則需要使用或不使用斜線來讓目錄 URL 失效，以確保 CloudFront 移除目錄。

必須讓兩種 URL 格式都失效是麻煩事，且還會導致額外成本。這是因為如果您必須加倍失效處理以涵蓋這兩種類型的 URL，則可能會超出該月允許的免費失效處理數上限。如果發生這種情況，則需要支付所有的失效處理，即使在 CloudFront 中每個目錄 URL 只有一個格式。

建立適用於受限制內容的簽署 URL

如果您想要限制存取內容，可以建立簽署 URL。例如，如果只想分配您的內容給已授權的使用者，可以建立僅適用於特定時段內有效的 URL，或只從指定 IP 位址的 URL。如需更多詳細資訊，請參閱 [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。

指定預設根物件

當使用者 (檢視器) 請求分佈的根 URL，而不是請求該分佈中的物件時，您可以設定由 CloudFront 傳回特定的物件 (預設根物件)。您可以使用預設根物件以避免暴露分佈的內容。

內容

- [如何指定預設根物件](#)
- [預設根物件的運作方式](#)
- [未定義根物件的情況下 CloudFront 的運作方式](#)

如何指定預設根物件

為了避免暴露分佈的內容或傳回錯誤，請為您的分佈指定預設根物件。您可以指定確切的檔案名稱或檔案的路徑。例如，如果您的根物件是 `index.html` 檔案，您可以指定該檔案名稱。如果您的 `index.html` 檔案位於另一個資料夾中，請改為指定路徑，例如 `exampleFolderName/index.html`。如果您設定預設根物件的路徑，檢視器對分佈根 URL 的請求會從該路徑傳回指定的檔案。您可以使用檔案路徑在原始伺服器更靈活地組織內容，因為您的預設根物件可以位於資料夾，而不是根層級。

指定分佈的預設根物件

1. 上傳預設根物件到您分佈指向的原始伺服器。

檔案可以是由 CloudFront 支援的任何類型。針對檔案名稱上限制條件的清單，請參閱《Amazon CloudFront API 參考》之 [DistributionConfig](#) 中的 DefaultRootObject 元素說明。

 Note

如果預設根物件的檔案名稱太長或包含無效字元，則 CloudFront 會傳回錯誤 HTTP 400 Bad Request - InvalidDefaultRootObject。此外，CloudFront 會快取程式碼 10 秒 (預設情況下)，並將結果寫入存取日誌。

2. 確認物件授予 CloudFront 的許可至少包括讀取權限。

如需 Amazon S3 許可的詳細資訊，請參閱 Amazon Simple Storage Service 使用者指南中的 [Amazon S3 中的身分和存取管理](#)。

3. 請更新您的分佈，以使用 CloudFront 主控台或 CloudFront API 參照預設根物件。

如要使用 CloudFront 主控台指定預設根物件：

- a. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
- b. 在上方窗格中的分佈清單裡，請選取分佈來更新。
- c. 在 Settings (設定) 窗格中的 General (一般) 標籤，選擇 Edit (編輯)。
- d. 在編輯設定對話方塊的預設根物件欄位中，輸入預設根物件的檔案名稱或路徑。

 Tip

您的字串不能以正斜線 (/) 開頭。僅指定物件名稱或物件的路徑。例如，使用 index.html 或 *exampleFolderName*/index.html。指定 */exampleFolderName*/index.html 或 /index.html 可能會導致 [403 存取遭拒錯誤](#)。

- e. 選擇儲存變更。

若要更新使用 CloudFront API 的組態，請在分佈中指定 DefaultRootObject 元素的值。如需使用 CloudFront API 指定預設根物件的詳細資訊，請參閱《Amazon CloudFront API 參考》中的 [UpdateDistribution](#)。

4. 確認您已藉由請求根 URL 來啟用預設根物件。如果您的瀏覽器不會顯示預設根物件，請執行以下步驟：
 - a. 確認是否藉由檢視 CloudFront 主控台的分佈狀態來完全部署您的分佈。
 - b. 重複步驟 2 和 3 來驗證您是否已授予正確的許可和驗證您是否正確地更新分佈的組態來指定預設根物件。

預設根物件的運作方式

假設以下請求指向物件 `image.jpg`：

```
https://d1111111abcdef8.cloudfront.net/image.jpg
```

相對地，以下請求指向同一分佈的根 URL，而非如前述範例指向特定物件：

```
https://d1111111abcdef8.cloudfront.net/
```

當您定義預設根物件時，呼叫分佈根的最終使用者請求會傳回預設的根物件。例如，如果您指定檔案 `index.html` 為預設根物件，則請求：

```
https://d1111111abcdef8.cloudfront.net/
```

傳回：

```
https://d1111111abcdef8.cloudfront.net/index.html
```

Note

CloudFront 不會判斷具有多條結尾斜線 (`https://d1111111abcdef8.cloudfront.net///`) 的 URL 是否等於 `https://d1111111abcdef8.cloudfront.net/`。您的原始伺服器會進行這項比較。

如果您定義預設根物件時，適用於分佈子目錄的最終使用者請求不會傳回預設的根物件。例如，假設 `index.html` 是您的預設根物件，則 CloudFront 會收到在 CloudFront 分佈下適用於 `install` 目錄的最終使用者請求：

```
https://d1111111abcdef8.cloudfront.net/install/
```


CloudFront 不會傳回預設根物件，即使 `index.html` 的副本出現在 `install` 目錄中。不過，如果您已指定預設根物件的路徑，(`install/index.html`) CloudFront 會傳回 `install` 目錄最終使用者請求的預設根物件

如果您配置分佈來允許 CloudFront 支援的所有 HTTP 方法，則預設根物件便套用到所有方法。例如，如果您的預設根物件是 `index.php`，而且您編寫應用程式將 POST 請求提交至網域 (`https://example.com`) 的根，則 CloudFront 會將請求傳送至 `https://example.com/index.php`。

CloudFront 預設根物件的行為與 Amazon S3 索引文件的行為不同。當您配置 Amazon S3 儲存貯體做為網站，並指定索引文件時，Amazon S3 傳回索引文件，即使使用者在儲存貯體中請求子目錄。(索引文件的副本必須出現在每個子目錄)。有關將 Amazon S3 儲存貯體設定為網站以及索引文件的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [AmazonS3 上的託管網站](#) 一章。

Important

請記住，預設根物件僅適用於您的 CloudFront 分佈。您仍然需要為原始伺服器管理安全性。例如，如果您使用 Amazon S3 原始伺服器，您仍然需要設定適當的 Amazon S3 儲存貯體 ACL，以確保在儲存貯體上您想要的存取層級。

未定義根物件的情況下 CloudFront 的運作方式

如果您不定義預設根物件，則分佈根的請求會通過您的原始伺服器。如果您使用 Amazon S3，有可能傳回以下任何情況：

- Amazon S3 儲存貯體內容的清單 — 在任何以下條件，使用 CloudFront 存取分佈的任何人都可看到您原始伺服器的內容：
 - 您的儲存貯體未設定正確。
 - 儲存貯體上的 Amazon S3 許可與您的分佈相關連，且在儲存貯體中物件上把存取權限給予每個人。
 - 最終使用者使用原始伺服器根 URL 存取原始伺服器。
- 原始伺服器的私有內容清單 — 如果您配置原始伺服器做為私有分佈 (只有您與 CloudFront 有存取權限)，有登入資料通過 CloudFront 來存取分佈的任何人都可以看到與分佈相關連的 Amazon S3 儲存貯體內容。在這種情況下，使用者無法透過您的原始伺服器根 URL 來存取您的內容。如需有關分佈私有內容的詳細資訊，請參閱 [the section called “限制已簽署 URL 和已簽署 Cookie 的內容”](#)。
- Error 403 Forbidden — CloudFront 傳回此錯誤，如果許可在與分佈相關連的 Amazon S3 儲存貯體中，或許可在拒絕存取 CloudFront 和所有人的儲存貯體裡物件上。

使檔案失效以移除內容

若您需要在檔案過期之前將其從 CloudFront 邊緣快取移除，則可執行以下其中一項：

- 透過節點快取使該檔案失效。下次若有檢視器請求該檔案，CloudFront 將傳回原始伺服器以擷取檔案的最新版本。
- 使用檔案版本控制以提供該檔案具有不同名稱的不同版本。如需更多詳細資訊，請參閱 [使用版本控制的檔案名稱更新現有檔案](#)。

主題

- [選擇要使檔案失效或者使用版本控制的檔案名稱](#)
- [確定要失效的檔案](#)
- [讓檔案失效時的注意事項](#)
- [使檔案失效](#)
- [並行失效請求上限](#)
- [支付檔案失效的費用](#)

選擇要使檔案失效或者使用版本控制的檔案名稱

若要控制從分佈中提供的檔案版本，您可使檔案失效或以版本控制的檔案名稱為其命名。如果您想經常更新檔案，建議您主要使用檔案版本控制，原因如下：

- 版本控制可讓您控制請求傳回哪些檔案，甚至是使用者什麼時候在本地或在企業快取代理之後快取一個版本。如果您使該檔案失效，使用者可能會繼續看到舊版本，直到檔案從這些快取中過期。
- CloudFront 存取日誌包含檔案的名稱，所以版本控制能讓您更輕鬆地分析檔案變更的結果。
- 版本控制讓您可向不同的使用者提供不同版本的檔案。
- 版本控制簡化了檔案修訂版本之間的往返復原。
- 版本控制更便宜。將新版本的檔案傳輸到節點仍需要支付 CloudFront 費用，而使檔案失效則不必支付任何費用。

如需檔案版本控制的詳細資訊，請參閱[使用版本控制的檔案名稱更新現有檔案](#)。

確定要失效的檔案

若您要使多個檔案失效，例如某個目錄中的所有檔案，或者檔名以相同字元開頭的所有檔案，則可在失效路徑的結尾包含 * 萬用字元。如需有關使用 * 萬用字元的詳細資訊，請參閱[Invalidation paths](#)。

若要使檔案失效，您可以指定個別檔案的路徑或以 * 萬用字元結尾的路徑，其可能會套用到一個或多個檔案，如以下範例所示：

- /images/image1.jpg
- /images/image*
- /images/

如果您想要使選定的檔案失效，但使用者不需要存取原始伺服器上的每個檔案，則您可以判斷檢視器向 CloudFront 要求了哪些檔案，並只讓這些檔案失效。若要判斷檢視器要求了哪些檔案，請啟用 CloudFront 存取記錄。如需存取日誌的詳細資訊，請參閱[存取日誌（標準日誌）](#)。

讓檔案失效時的注意事項

當您指定要失效的檔案時，請參閱下列資訊：

區分大小寫

失效路徑區分大小寫。例如，/images/image.jpg 和 /images/Image.jpg 指定兩個不同的檔案。

使用 Lambda 函數變更 URI

如果您的 CloudFront 分佈在檢視器請求事件中觸發 Lambda 函式，而且該函式變更了請求檔案的 URI，我們建議您使以下兩種 URI 失效，以便從 CloudFront 邊緣快取中移除檔案：

- 在檢視器請求中的 URI
- 函數予以變更之後的 URI

Example 範例

例如，假設您的 Lambda 函數變更了檔案的 URI，而其原本為：

```
https://d1111111abcdef8.cloudfront.net/index.html
```

到包含語言目錄的 URI：

`https://d1111111abcdef8.cloudfront.net/en/index.html`

若要使該檔案失效，您必須指定以下路徑：

- `/index.html`
- `/en/index.html`

如需詳細資訊，請參閱 [Invalidation paths](#)。

預設根物件

若要使預設根物件 (檔案) 失效，請以您為任何其他檔案指定路徑的相同方式指定其路徑。如需更多詳細資訊，請參閱 [預設根物件的運作方式](#)。

轉送 Cookie

如果您配置 CloudFront 轉送 Cookie 到原始伺服器，則 CloudFront 邊緣快取可能包含檔案的多個版本。當您使檔案失效後，CloudFront 會使該檔案的每個快取版本皆失效，無論其是否與 Cookie 關聯。您無法根據關聯的 Cookie 選擇性地使某些版本而不是其他版本失效。如需詳細資訊，請參閱 [根據 Cookie 快取內容](#)。

Forwarding headers (轉送標頭)

如果您配置 CloudFront 轉送標頭清單到原始伺服器，並且根據標頭的值進行快取，則 CloudFront 邊緣快取可能包含檔案的多個版本。當您使檔案失效後，CloudFront 會使該檔案的每個快取版本皆失效，無論其標頭值為何。您無法根據標頭值選擇性地使某些版本而不是其他版本失效。(如果您將 CloudFront 配置為將所有標頭轉送到原始伺服器，CloudFront 不會快取您的檔案。) 如需詳細資訊，請參閱 [根據請求標頭快取內容](#)。

轉送查詢字串

如果您配置 CloudFront 轉送查詢字串至原始伺服器，使檔案失效時您必須包含查詢字串，如以下範例所示：

- `/images/image.jpg?parameter1=a`
- `/images/image.jpg?parameter1=b`

如果用戶端請求包含針對同一檔案的五種不同查詢字串，則您可以使該檔案失效五次，每一查詢字串一次，或者也可在失效路徑中使用 * 萬用字元，如以下範例所示：

`/images/image.jpg*`

如需有關在失效路徑使用萬用字元的詳細資訊，請參閱 [Invalidation paths](#)。

如需查詢字串的詳細資訊，請參閱 [根據查詢字串參數快取內容](#)。

若要判斷哪些查詢字串使用中，您可以啟用 CloudFront 記錄功能。如需詳細資訊，請參閱 [存取日誌（標準日誌）](#)。

允許的上限

如需允許的失效數量上限詳細資訊，請參閱 [並行失效請求上限](#)。

Microsoft Smooth Streaming file (Microsoft Smooth Streaming 檔案)

當您已啟用對應快取行為的 Smooth Streaming 時，您不能使 Microsoft Smooth Streaming 格式中的媒體檔案失效。

Non-ASCII or unsafe characters in the path (路徑中非 ASCII 或不安全字元)

如果路徑包含非 ASCII 字元或 [RFC 1738](#) 中定義的不安全字元，請將這些字元進行 URL 編碼。切勿對路徑中的任何其他字元進行 URL 編碼，否則 CloudFront 不會使更新後檔案的舊版本失效。

Important

請勿在路徑中使用 ~ 字元。CloudFront 不支援此字元的失效，無論其是否為 URL 編碼。

Invalidation paths (失效路徑)

路徑與分佈有關。例如，要使在 `https://d1111111abcdef8.cloudfront.net/images/image2.jpg` 的檔案失效，您可以指定 `/images/image2.jpg`。

Note

在 [CloudFront 主控台](#) 中，您可以省略路徑中的前導斜線，如：`images/image2.jpg`。直接使用 CloudFront API 時，無效驗證路徑須以前綴斜線開頭。

您也可以使用 * 萬用字元使多個檔案同時失效。取代 0 或多個字元的 *，必須是失效路徑的最後一個字元。

Important

若要在失效中使用萬用字元 (*)，您必須將萬用字元放在路徑尾端。插入其他任何位置的星號 (*) 都會視為常值字元相符，而不是萬用字元失效。

若您使用 AWS Command Line Interface (AWS CLI) 讓檔案失效，且指定包含 * 萬用字元的路徑，則該路徑必須用引號 (") 括住，如 `"/*`。

路徑的長度上限為 4,000 個字元。

Example 範例：失效路徑

- 使某個目錄中的所有檔案失效：

`/directory-path/*` (目錄路徑)

- 若要使某個目錄、其所有子目錄以及該目錄與子目錄中的所有檔案失效：

`/directory-path*` (目錄路徑)

- 為了使所有名稱相同，但不同的檔案名稱延伸的檔案失效，例如 logo.jpg、logo.png 和 logo.gif：

`/directory-path` (目錄路徑) `/file-name` (檔案名稱) `.*`

- 無論檔案名稱延伸如何，若要使檔案名稱以相同字元開頭的目錄中的所有檔案 (例如 HLS 格式中影片的所有檔案) 失效：

`/directory-path` (目錄路徑) `/initial-characters-in-file-name` (檔案名稱中的初始字元) `*`

- 當您配置 CloudFront 根據查詢字串參數進行快取時，要使檔案的每個版本皆失效：

`/directory-path` (目錄路徑) `/file-name` (檔案名稱) `.file-name-extension` (檔案名稱延伸) `*`

- 使分佈中的所有檔案失效：

`/*`

如需使用 Lambda 函式變更 URI 的情況下使檔案失效的詳細資訊，請參閱 [Changing the URI Using a Lambda Function](#)。

如果失效路徑是一個目錄且如果您尚未標準化指定目錄的方法 (包含或不包含結尾斜線 (/))，我們建議您，讓包含及不包含結尾斜線的目錄皆失效，例如 `/images` 和 `/images/`。

Signed URLs (簽署的 URL)

如果您使用已簽章的 URL，則透過僅包含問號 (?) 前面的 URL 部分以使檔案失效。

使檔案失效

您可以使用 CloudFront 主控台來建立與執行失效，顯示您之前提交的失效清單，以及顯示有關個別失效的詳細資訊。您也可以複製現有的失效、編輯檔案路徑的清單和執行已編輯的失效。您可以從清單移除失效。

內容

- [使檔案失效](#)
- [複製、編輯和重新執行現有的失效](#)
- [取消失效](#)
- [列出失效](#)
- [顯示有關失效的資訊](#)

使檔案失效

若要使用 CloudFront 主控台使檔案失效，請執行以下步驟。

Console

使檔案失效 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 選擇您欲使其檔案失效的分佈。
3. 請選擇 Invalidations (失效) 索引標籤。
4. 請選擇建立失效。
5. 針對您要使其失效的檔案，每行輸入一個失效路徑。如需有關指定失效路徑的詳細資訊，請參閱[讓檔案失效時的注意事項](#)。



Important

請謹慎指定檔案路徑。開始後便無法取消失效請求。

6. 請選擇建立失效。

CloudFront API

如需了解使物件失效及顯示與失效相關資訊的詳細資訊，請參閱《Amazon CloudFront API 參考》的以下主題：

- [CreateInvalidation](#)
- [ListInvalidations](#)
- [GetInvalidation](#)

Note

若您對失效檔案使用 AWS Command Line Interface(AWS CLI)，且指定包含 * 萬用字元的路徑，則該路徑必須用引號 (") 括住，如以下範例：

```
aws cloudfront create-invalidation --distribution-id distribution_ID --paths  
"/*"
```

複製、編輯和重新執行現有的失效

您可以複製您之前建立的失效、更新失效路徑的清單，並執行已更新的失效。您不能複製現有的失效、更新失效路徑，接著儲存更新的失效且無需執行。

Important

如果您在失效進行中複製失效、更新失效路徑清單，並接著執行更新的失效，CloudFront 不會停止或刪除您複製的失效。若有任何失效路徑出現在原始伺服器本和複本中，CloudFront 會嘗試使檔案失效兩次，且這兩次失效均將計入您每個月免費的失效數上限。如果您已達到免費的失效數上限，即將支付每個檔案兩次失效的費用。如需更多詳細資訊，請參閱 [並行失效請求上限](#)。

複製、編輯和重新執行現有的失效

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 請選取包含您欲複製的失效的分佈。

3. 請選擇 Invalidations (失效) 索引標籤。
4. 選擇您想要複製的失效。

如果您不確定要複製哪個失效，可以選擇一個失效並選擇檢視詳細資訊以顯示有關失效的詳細資訊。

5. 選擇複製到新的。
6. 如果適用，請更新失效路徑清單。
7. 請選擇建立失效。

取消失效

當您提交失效請求到 CloudFront 時，CloudFront 幾秒鐘內便轉送請求到所有節點，且每個節點立即開始處理失效。因此，提交後便無法取消失效。

列出失效

您可以使用 CloudFront 主控台，顯示過去您已建立且執行分佈的 100 個失效的清單。如果您想要取得超過 100 個失效的清單，請使用 ListInvalidations API 操作。如需詳細資訊，請參閱 Amazon CloudFront API 參考中的 [ListInvalidations](#)。

列出失效

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 請選取欲顯示失效清單的分佈。
3. 請選擇 Invalidations (失效) 索引標籤。

Note

您無法從清單移除失效。

顯示有關失效的資訊

您可以顯示有關失效的詳細資訊，包括分佈 ID、失效 ID、失效的狀態、建立失效的日期和時間，以及失效路徑的完整清單。

顯示有關失效的資訊

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 請選取包含您欲顯示詳細資訊的失效的分佈。
3. 請選擇 Invalidations (失效) 索引標籤。
4. 選擇適用的失效 ID 或選擇失效 ID，然後選擇檢視詳細資訊。

並行失效請求上限

如果您使檔案逐個失效，則每個進行中的分佈可同時請求最多 3,000 個檔案失效。一次失效請求最多可達 3,000 個檔案，或針對單一檔案請求 3,000 次，或者總計不超過 3,000 個檔案的任何組合方式。例如，您可以提交 30 次失效請求，每次使 100 個檔案失效。只要 30 個失效請求都仍在進行中，您就無法提交任何更多的失效請求。如果超過上限，CloudFront 會傳回錯誤訊息。

如果您使用 * 萬用字元，則每次進行中都可以請求最多 15 個失效路徑。您也可以就每一進行中的分佈同時請求失效最多 3,000 個檔案；萬用字元失效請求允許的上限與檔案逐個失效的上限無關。

支付檔案失效的費用

您每月提交的前 1,000 個失效路徑免費；您只需支付每月超過 1,000 個的每一失效路徑。每一失效路徑可適用於單一檔案 (如 /images/logo.jpg) 或多個檔案 (如 /images/*)。包含 * 萬用字元的路徑即便導致 CloudFront 使數千個檔案失效，也算作一個路徑。

每月 1,000 個免費失效路徑的上限適用於您建立一個 AWS 帳戶 中所有分佈的失效路徑總數。例如，如果您使用 AWS 帳戶 john@example.com 建立三種分佈，且您在指定月份中提交每一分佈 600 個失效路徑 (總共是 1,800 個失效路徑)，AWS 會就總失效路徑與 1000 筆免費上限額度間的差異向您收取費用。在此範例中，AWS 會在該月向您收取 800 個失效路徑的費用。

無論您要使之失效的檔案有多少，單一檔案 (/images/logo.jpg) 還是所有與分佈關聯的檔案 (/*)，提交失效路徑的費用皆相同。由於您失效請求中的每個路徑都會向您收費，即使您將多個路徑綁定到單一請求中，每個路徑仍會個別計入計費目的。

如需無效定價的詳細資訊，請參閱 [Amazon CloudFront 定價](#)。如需有關失效路徑的詳細資訊，請參閱 [Invalidation paths](#)。

提供壓縮檔案

請求的物件壓縮後物件較小，下載更快 – 在某些情況下，壓縮後還不到原始檔案的四分之一大小。加速下載可讓瀏覽者享有更快速的網頁轉譯，特別是 JavaScript 和 CSS 檔案。此外，CloudFront 資料傳輸成本是根據提供的總資料量計算。提供壓縮物件的成本可能低於未壓縮物件。

主題

- [設定 CloudFront 來壓縮物件](#)
- [CloudFront 壓縮如何運作](#)
- [壓縮條件](#)
- [CloudFront 壓縮的檔案類型](#)
- [ETag 標頭轉換](#)

設定 CloudFront 來壓縮物件

若要設定 CloudFront 來壓縮物件，請更新您想提供壓縮物件的快取行為。

設定 CloudFront 壓縮物件 (主控台)

1. 登入 [CloudFront 主控台](#)。
2. 選擇您的分佈，然後選擇要編輯的行為。
3. 請將自動壓縮物件設定為是。
4. 使用[快取政策](#)來指定快取設定，並確保 Gzip 和 Brotli 壓縮格式皆已啟用。

備註

- 您必須使用[快取政策](#)來使用 Brotli 壓縮。Brotli 不支援舊版快取設定。
- 若要使用 [CloudFormation](#) 或 [CloudFront](#) API 啟用壓縮，請將 Compress、EnableAcceptEncodingGzip、EnableAcceptEncodingBrotli 參數設定為 true。

若要瞭解 CloudFront 如何壓縮物件，請參閱下一節。

CloudFront 壓縮如何運作

1. 一個檢視器請求了一個物件。檢視器會在請求中包含 Accept-Encoding HTTP 標頭，而標頭值則包含 gzip、br 或兩者。這表示檢視器支援壓縮物件。檢視器同時支援 Gzip 和 Brotli 時，CloudFront 會使用 Brotli。

Note

只有在使用 HTTPS 發送請求時，Chrome 和 Firefox 網頁瀏覽器才支援 Brotli 壓縮。這兩種瀏覽器不支援 HTTP 請求的 Brotli。

2. 在節點，CloudFront 會檢查快取中是否有所請求物件的壓縮副本。
3. CloudFront 會根據壓縮物件是否在快取中執行下列其中一項操作：
 - 如果快取中已有壓縮物件，則 CloudFront 會將物件傳送至檢視器，略過其餘步驟。
 - 如果快取中沒有壓縮物件，則 CloudFront 會將請求轉送至原始伺服器。

Note

如果物件的未壓縮副本已存在於快取中，CloudFront 會將該物件傳送至檢視器，而不會將請求轉送至原始伺服器。舉例來說，如果 CloudFront [先前略過壓縮作業](#)，就會發生這種情況。發生這種情況時，CloudFront 會快取未壓縮的物件並繼續提供物件，直到物件過期、移出或無效為止。

4. 如果原始伺服器傳回壓縮物件 (由 HTTP 回應中出現的 Content-Encoding 標頭所指示)，CloudFront 會將壓縮的物件傳送到檢視器，將其新增到快取並略過其餘步驟。CloudFront 不會再次壓縮物件。
5. 如果原始伺服器將未壓縮的物件傳回到 CloudFront，且 HTTP 回應中沒有 Content-Encoding 標頭，CloudFront 會判斷物件是否可壓縮。如需更多詳細資訊，請參閱 [壓縮條件](#)。
6. 如果物件可以壓縮，CloudFront 會壓縮物件並將其傳送至檢視器，然後新增到快取。
7. 如果相同物件有後續的檢視器請求，CloudFront 會傳回第一個快取版本。例如若檢視器請求使用 Gzip 壓縮的特定快取物件，且檢視器接受 Gzip 格式，則對相同物件的後續請求一律會傳回 Gzip 版本，即使檢視器同時接受 Brotli 和 Gzip。

某些自訂原始伺服器也可以壓縮物件。您的原始伺服器可能可以壓縮 CloudFront 無法壓縮的物件。如需更多詳細資訊，請參閱 [CloudFront 壓縮的檔案類型](#)。

壓縮條件

下列清單提供詳細資訊，說明 CloudFront 不會壓縮物件的情境。

請求使用 HTTP 1.0

如果對 CloudFront 的請求使用 HTTP 1.0，則 CloudFront 會移除 Accept-Encoding 標頭，並且不會壓縮回應中的物件。

Accept-Encoding 請求標頭

如果檢視器請求中缺少 Accept-Encoding 標頭，或者如果其中不包含 gzip 或 br 作為值，則 CloudFront 不會壓縮回應中的物件。如果 Accept-Encoding 標頭包含額外的值，例如 deflate，則 CloudFront 在將請求轉送到原始伺服器之前會先移除這些值。

當 CloudFront [設定為壓縮物件](#)時，它會在快取金鑰中和原始伺服器請求中自動包含 Accept-Encoding 標頭。

當您設定 CloudFront 來壓縮物件時，內容已快取

CloudFront 會在從原始伺服器取得物件時壓縮物件。當您設定 CloudFront 來壓縮物件時，CloudFront 不會壓縮已在節點快取的物件。此外，如果快取物件在某個邊緣節點中已過期，但 CloudFront 又將對此物件的另一個請求轉送到原始伺服器，則原始伺服器傳回 HTTP 狀態碼 304 時，CloudFront 不會壓縮物件。這表示邊緣節點已有最新版本的物件。如果您想讓 CloudFront 壓縮已在節點快取的物件，您必須讓這些物件失效。如需詳細資訊，請參閱 [使檔案失效以移除內容](#)。

已配置原始伺服器來壓縮物件

如果您設定 CloudFront 來壓縮物件，而原始伺服器也壓縮物件，則原始伺服器應會包含 Content-Encoding 標頭。此標頭會向 CloudFront 指出物件已壓縮。原始伺服器回應包含 Content-Encoding 標頭時，則無論標頭的值為何，CloudFront 都不會壓縮物件。CloudFront 會將回應傳送給檢視器，並且在節點快取物件。

CloudFront 壓縮的檔案類型

如需完整清單，請參閱 [CloudFront 壓縮的檔案類型](#)。

CloudFront 壓縮的物件大小

CloudFront 壓縮的物件大小介於 1,000 個位元組到 10,000,000 個位元組之間。

Content-Length 標頭

原始伺服器必須在回應中包含 Content-Length 標頭，以便 CloudFront 判斷物件的大小是否在 CloudFront 壓縮的範圍內。如果 Content-Length 標頭遺失、包含無效值，或包含 CloudFront

可壓縮大小範圍之外的值，CloudFront 不會壓縮物件。如需詳細資訊瞭解 CloudFront 如何處理可能超過大小範圍的大型物件，請參閱 [CloudFront 如何處理物件的部分請求 \(範圍 GET\)](#)。

回應的 HTTP 狀態碼

只有在回應的 HTTP 狀態碼為 200、403 或 404 時，CloudFront 才會壓縮物件。

回應沒有內文

當來自原始伺服器的 HTTP 回應沒有內文時，CloudFront 沒有可以壓縮的內容。

ETag 標頭

CloudFront 有時會在壓縮物件時修改 HTTP 回應中的 ETag 標頭。如需更多詳細資訊，請參閱 [the section called “ETag 標頭轉換”](#)。

CloudFront 略過壓縮作業

CloudFront 一般會盡力壓縮物件，在極少數情況下，CloudFront 會在 CloudFront 遇到高流量負載時略過壓縮物件。CloudFront 會根據多種因素 (包括主機容量) 決定是否略過壓縮作業。CloudFront 略過物件的壓縮作業時，會快取未壓縮物件並繼續將物件提供給檢視者，直到物件過期、移出或失效為止。

CloudFront 壓縮的檔案類型

如果您設定 CloudFront 來壓縮物件，則 CloudFront 僅會壓縮 Content-Type 回應標頭中具有下列值的物件：

- application/dash+xml
- application/eot
- application/font
- application/font-sfnt
- application/javascript
- application/json
- application/opentype
- application/otf
- application/pdf
- application/pkcs7-mime
- application/protobuf

- application/rss+xml
- application/truetype
- application/ttf
- application/vnd.apple.mpegurl
- application/vnd.mapbox-vector-tile
- application/vnd.ms-fontobject
- application/wasm
- application/xhtml+xml
- application/xml
- application/x-font-opentype
- application/x-font-truetype
- application/x-font-ttf
- application/x-httpd-cgi
- application/x-javascript
- application/x-mpegurl
- application/x-opentype
- application/x-otf
- application/x-perl
- application/x-ttf
- font/eot
- font/opentype
- font/otf
- font/ttf
- image/svg+xml
- text/css
- text/csv
- text/html
- text/javascript
- text/js
- text/plain

- text/richtext
- text/tab-separated-values
- text/xml
- text/x-component
- text/x-java-source
- text/x-script
- vnd.apple.mpegurl

ETag 標頭轉換

如果來自原始伺服器的未壓縮物件包含有效的強 ETag HTTP 標頭，CloudFront 壓縮該物件時，CloudFront 會將強 ETag 標頭值轉換為弱 ETag，並將弱 ETag 值傳回給檢視器。檢視器可以儲存弱值 ETag，並使用它來傳送具有 If-None-Match HTTP 標頭的條件式請求。這可讓檢視器 CloudFront 和原始伺服器將物件的壓縮和未壓縮版本視為語義上等效的物件，從而減少不必要的資料傳輸。

有效的強式 ETag 標頭值以雙引號字元 (") 開頭和結尾。若要將強值 ETag 轉換為弱值，CloudFront 會將字元 W/ 新增至強值 ETag 的開頭。

當來自原始伺服器的物件包含弱標頭值 ETag (以字元 W/ 開頭的值) 時，CloudFront 不會修改此值，而是按照從原始伺服器接收的原樣傳回給檢視器。

當原始伺服器的物件包含無效的 ETag 標頭值 (非以 " 或 W/ 開頭的值) 時，CloudFront 會移除 ETag 標頭，並將不含 ETag 回應標頭的物件傳回給檢視器。

如需詳細資訊，請參閱 MDN Web 文件中的下列頁面：

- [指示詞](#) (ETag HTTP 標頭)
- [弱驗證](#) (HTTP 條件式請求)
- [If-None-Match HTTP 標頭](#)

使用 AWS WAF 保護

您可以使用 [AWS WAF](#) 來保護 CloudFront 分佈與原始伺服器。AWS WAF 是一種 Web 應用程式防火牆，可讓您保護 Web 應用程式和 API，以便在請求到達伺服器之前封鎖請求。如需詳細資訊，請參閱[使用 CloudFront 和 AWS WAF 加速和保護您的網站](#)，以及[實作 AWS WAF 的指導方針](#)。

若要啟用 AWS WAF 保護，您可以：

- 使用 CloudFront 主控台中的一鍵式保護。一鍵式保護將建立 AWS WAF 網頁存取控制清單 (Web ACL)、設定規則以保護您的伺服器免受常見網路威脅侵害，並為您連接網頁 ACL 至 CloudFront 分佈。本節中的主題假設使用一鍵式保護。
- 使用您在 AWS WAF 主控台或使用 AWS WAF API 建立的預先設定 Web ACL (存取控制清單)。如需詳細資訊，請參閱 AWS WAF 開發人員指南中的 [Web 存取控制清單 \(ACL\)](#) 和 AWS WAF API 參考中的 [AssociateWebACL](#)

您可以在以下情況啟用 AWS WAF：

- 建立分發
- 使用安全性儀表板編輯現有分佈的安全性設定

當您使用一鍵式保護時，CloudFront 會套用一組 AWS 建議的保護，包括：

- 根據 Amazon 內部威脅情報，封鎖 IP 位址免於遭受潛在威脅。
- 如 [OWASP 前 10](#) 所述，防範 Web 應用程式中發現的最常見漏洞。
- 抵禦發現應用程式漏洞的惡意行為者。

Important

如果您想要在 CloudFront 安全性儀表板中檢視安全指標，則必須啟用 AWS WAF。若 AWS WAF 未啟用，您只能使用安全性儀表板來啟用 AWS WAF 或設定 CloudFront 地理限制。如需更多儀表版的相關資訊，請參閱此章節後續的 [在 CloudFront 安全儀表板中管理 AWS WAF 安全性保護](#)。

主題

- [為分佈啟用 AWS WAF](#)
- [在 CloudFront 安全儀表板中管理 AWS WAF 安全性保護](#)
- [設定速率限制](#)
- [停用 AWS WAF 安全保護](#)

為分佈啟用 AWS WAF

您可以在建立分佈時啟用 AWS WAF，也可以啟用現有存取控制清單 (ACL) 的安全保護。

如果您為 CloudFront 分佈啟用 AWS WAF，也可以啟用機器人控制功能，並依機器人類別設定安全保護。

主題

- [為新分佈啟用 AWS WAF](#)
- [使用現有的 Web ACL](#)
- [啟用機器人控制功能](#)
- [依機器人類別設定保護](#)

為新分佈啟用 AWS WAF

下列程序顯示如何在建立 CloudFront 分佈時啟用 AWS WAF。

為新分佈啟用 AWS WAF

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇分佈，然後選擇建立分佈。
3. 如有需要，請遵循 [建立分發](#) 中的步驟。
4. 在 Web Application Firewall 區段中，選擇編輯，然後選擇啟用安全保護。
5. 完成下列欄位：
 - 使用監控模式：當您想要先收集資料以測試保護的工作原理時，請啟用監控模式。在您啟用監控模式時，如果保護處於作用中狀態，則不會封鎖請求。而是監控模式會收集請求相關資料，如果保護處於作用中狀態，則會封鎖這些請求。當您準備好開始封鎖時，可以在安全性頁面上啟用封鎖。

- 額外保護：選擇您要啟用的任何選項。如果您啟用速率限制，請參閱[the section called “設定速率限制”](#)取得詳細資訊。
- 價格估算：您可以開啟區段以顯示欄位，您可以在欄位中輸入不同的請求/月份數，並查看新的估計值。

6. 檢閱剩餘的分佈設定，然後選擇建立分佈。

建立分佈後，CloudFront 會建立一個安全性儀表板。您可以使用此儀表板來停用或啟用 AWS WAF。如果您尚未啟用 AWS WAF，儀表板中的圖表和圖形會保持空白。

使用現有的 Web ACL

如果您目前有 Web ACL，則可以使用它，而不是 AWS WAF 提供的保護。

使用現有 AWS WAF 組態

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 執行以下任意一項：
 - a. 選擇建立分佈並依照 [建立分發](#) 中的步驟執行，然後返回本主題。
 - b. 選擇現有的組態，然後選擇安全性標籤。
3. 在 Web Application Firewall (WAF) 區段中，選擇編輯，然後選擇啟用安全保護。
4. 選擇 使用現有 WAF 組態。只有在您已設定 Web ACL 時，才會顯示此選項。
5. 從 選擇 Web ACL 表格中選擇您現有的 Web ACL。
6. 檢閱剩餘的分佈設定，然後選擇建立分佈。

啟用機器人控制功能

如果您為 CloudFront 分佈啟用 AWS WAF，您可以在 CloudFront 主控台的安全性儀表板下檢視指定時間範圍的機器人請求。您也可以在此處啟用或停用機器人控制功能。

當您啟用機器人控制功能時，會產生費用。安全性儀表板提供成本估算。

如果您啟用機器人控制功能，安全性儀表板會依每個機器人類型和類別顯示機器人流量。如果您停用機器人控制功能，會根據請求取樣顯示機器人流量。

啟用機器人控制功能

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇分佈，然後選擇您要變更的分佈。
3. 選擇 Security (安全) 標籤。
4. 向下捲動至指定時間範圍的機器人請求區段，然後選擇啟用機器人控制功能。
5. 在機器人控制功能對話方塊的組態下，選取為一般機器人啟用機器人控制功能核取方塊。
6. 選擇儲存變更。

依機器人類別設定保護

當您啟用機器人控制功能時，您可以選擇設定每個未經驗證的機器人如何依據機器人類別進行處理。例如，您可以將 HTTP 程式庫機器人設定為監控模式，並將挑戰指派給連結檢查程式。

Note

已知為 AWS 常見且可驗證的機器人 (例如已知的搜尋引擎爬蟲程式) 不受您在此處設定的動作所限制。機器人控制功能會在將經過驗證的機器人標記為已驗證之前，確認其來源是否來自其聲明的來源。

若要設定機器人類別的保護

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇分佈，然後選擇您要變更的分佈。
3. 選擇 Security (安全) 標籤。
4. 在按機器人分類請求圖表中，指向未驗證機器人動作欄中的任何項目，然後選擇鉛筆圖示以進行編輯。
5. 在導覽窗格清單中，請選擇下列其中一個：
 - 封鎖
 - 允許
 - 監控模式
 - CAPTCHA
 - 挑戰

6. 選取清單旁邊的核取記號以儲存變更。

在 CloudFront 安全儀表板中管理 AWS WAF 安全性保護

CloudFront 會為您的每個分佈建立安全性儀表板。使用 CloudFront 主控台的安全儀表板。儀表板可讓您在單一位置使用 CloudFront 以及 AWS WAF 來監控和管理 Web 應用程式的常見安全性保護。儀表板提供下列任務和資料：

- **安全性組態：**您可以啟用和停用 AWS WAF 保護，並查看任何應用程式特有的保護，例如 WordPress 保護。
- **安全性趨勢：**包括允許和封鎖的請求、挑戰和 CAPTCHA 請求，以及主要攻擊類型。您可以查看流量比例以及它們隨時間變化的情況。例如，如果所有請求提升 3%，但允許的請求增加了 14%，這代表您在目前期間允許更大部分流量通過。
- **機器人請求：**您可以查看有多少流量來自機器人、有哪些類型的機器人 (已驗證與未驗證)，以及機器人類型的百分比分配 (已驗證與未驗證) 如何隨時間變化。如需有關啟用機器人控制功能的詳細資訊，請參閱 [啟用機器人控制功能](#)。
- **請求日誌：**日誌資料可有助於回答有關安全性趨勢或機器人請求的問題。您可以在不撰寫查詢的情況下搜尋日誌，並檢視彙整圖表，以協助判斷篩選的日誌集是否主要由 HTTP 方法、IP 位址、URI 路徑或國家/地區的子集來驅動。您可以將滑鼠游標暫留在圖表中的值上，並封鎖 IP 位址和國家/地區。如需更多詳細資訊，請參閱 [啟用 AWS WAF 日誌](#)。
- **地理限制管理：**CloudFront 和 AWS WAF 提供地理限制功能。CloudFront 免費提供地理限制，但安全性儀表板中不會顯示 CloudFront 地理限制的指標。若要查看封鎖國家/地區的請求指標，您必須使用 AWS WAF 地理限制。若要這樣做，請將滑鼠游標停留在安全性儀表板的國家/地區列，然後封鎖國家/地區。如需更多詳細資訊，請參閱 [使用 CloudFront 地理限制](#)。
 - 如果您之前在 CloudFront 主控台以外建立自訂 AWS WAF 規則來封鎖國家/地區，則可能無法使用封鎖選項。

主題

- [必要條件](#)
- [啟用 AWS WAF 日誌](#)

必要條件

如果您想要在 CloudFront 安全性儀表板中檢視安全指標，則必須啟用 AWS WAF。若您未啟用 AWS WAF，就只能使用安全性儀表板來啟用 AWS WAF 或設定 CloudFront 地理限制。

如需啟用 AWS WAF 的詳細資訊，請參閱[為分佈啟用 AWS WAF](#)。

啟用 AWS WAF 日誌

AWS WAF 日誌資料可協助您隔離特定的流量模式。例如，日誌可以顯示特定流量來自何處或其作用。

如果您啟用 CloudWatch 的 AWS WAF 記錄，則 CloudFront 安全性儀表板會查詢、彙總和顯示來自 CloudWatch 日誌的深入解析。我們不會針對安全性儀表板的使用收取費用，但 CloudWatch 定價適用於透過儀表板查詢的日誌。如需詳細資訊，請參閱[Amazon CloudWatch 定價](#)。

啟用日誌

1. 在每月請求數方塊中輸入預期的請求數量，以預估啟用的成本。
2. 選取啟用 AWS WAF 日誌核取方塊。
3. 選擇啟用。

CloudFront 會建立 CloudWatch 日誌群組，並更新您的 AWS WAF 組態，以便開始記錄到 CloudWatch。首次使用時，日誌資料可能需要幾分鐘才會出現。圖表的請求區段會列出每個請求。在個別請求下方，長條圖會依據 HTTP 方法、熱門 URI 路徑、熱門 IP 位址和熱門國家/地區彙總資料。圖表可以幫助您找出模式。例如，您可能看到來自單一 IP 位址的請求數量不成比例，或是您先前在日誌中未看到的國家/地區的資料。您可以依據國家/地區、主機標頭和其他屬性篩選請求，以協助尋找不想要的流量。找出該流量後，請將滑鼠游標暫留在個別請求或圖表項目上，並封鎖 IP 位址或國家/地區。

Note

顯示的指標是以 Web ACL 為基礎。因此如果您將相同的 Web ACL 與多個分佈建立關聯，就會看到 Web ACL 的所有指標，而不只是針對該分佈處理的 AWS WAF 請求。

設定速率限制

速率限制是您在設定安全保護時可能會收到的其中一項建議。

CloudFront 一律在監控模式下啟用速率限制。啟用監控模式時，CloudFront 會擷取指標，讓您了解是否已超過在速率限制欄位中設定的速率、頻率以及程度。

儲存分佈後，CloudFront 會開始根據 速率限制 欄位中的數量收集資料。

您可以在任何 CloudFront 分佈的安全性標籤上的安全性 - Web Application Firewall (WAF) 區段中啟用或管理速率限制設定。

Note

只有在您為分佈指定非 S3 自訂原始伺服器時，速率限制選項才會出現在 CloudFront 主控台中。否則，您只會看到為分佈啟用的核心保護。如需原始伺服器類型的詳細資訊，請參閱[將各種原始伺服器與 CloudFront 分佈搭配使用](#)。

設定速率限制

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇分佈，然後選擇您要變更的分佈。
3. 選擇 Security (安全) 標籤。
4. 在安全性 - Web Application Firewall (WAF) 區段中，選擇編輯。
5. 在額外保護下，選取速率限制。您可以選擇性變更速率限制。微調速率後，選擇儲存變更。
6. 在安全 - Web Application Firewall (WAF) 區段的速率限制旁，您可以選擇監控模式，然後選擇啟用封鎖以停用監控模式。CloudFront 會開始封鎖超過指定速率限制的請求。

如需啟用 AWS WAF 和速率限制的詳細資訊，請參閱[統一 CDN 和安全體驗的 CloudFront 安全儀表板簡介](#)部落格文章。

停用 AWS WAF 安全保護

如果您的分佈不需要 AWS WAF 安全保護，您可以使用 CloudFront 主控台停用此功能。

如果您先前已啟用 AWS WAF 保護，但未選擇現有的 WAF 組態 (也稱為一鍵式保護)，CloudFront 會自動為您建立 Web ACL。對於以此方式建立 Web ACL，CloudFront 主控台會取消資源的關聯，並刪除 Web ACL。

取消 Web ACL 的關聯與將其刪除不同。取消關聯會從您的分佈中移除 Web ACL，但不會從您的 AWS 帳戶中刪除。如需詳細資訊，請參閱 AWS WAF、AWS Firewall Manager 和 AWS Shield Advanced 開發人員指南中的[將 Web ACL 與 AWS 資源建立關聯或取消關聯](#)。

請參閱下列程序，以停用 AWS WAF 保護並取消 Web ACL 與分佈的關聯。

停用 CloudFront 中的安全性保護

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇分佈，然後選擇您要變更的分佈。
3. 選擇安全性索引標籤，然後選擇編輯。
4. 在 Web Application Firewall (WAF) 區段中，選擇停用 AWS WAF 保護。
5. 選擇儲存變更。

備註

- 如果停用 AWS WAF 安全性保護，但仍想要從 AWS 帳戶刪除 Web ACL，您可以手動將其刪除。依照程序[刪除 Web ACL](#)。在 AWS WAF & Shield 主控台中，對於 Web ACL 頁面，您必須選擇全域 (CloudFront) 清單來尋找 Web ACL。
- 從 CloudFront 主控台刪除分佈時，如果您選擇一鍵式保護，CloudFront 也會嘗試刪除 Web ACL。這是盡最大努力的結果，且不能保證。如需更多詳細資訊，請參閱[刪除分發](#)。

設定安全存取和限制對內容的存取

CloudFront 提供數個選項來保護其傳遞內容的安全。以下是一些可用 CloudFront 來保護和限制內容存取的方法：

- 設定 HTTPS 連線
- 防止特定地理位置的使用者存取內容
- 請求使用者使用 CloudFront 已簽署的 URL 或已簽署的 Cookie 來存取內容
- 設定特定內容欄位的欄位層級加密
- 使用 AWS WAF 控制對內容的存取

您也應該為您的基礎設施和應用程式實作 DDoS 彈性架構。如需詳細資訊，請參閱 [AWS DDoS 備援的最佳實務](#)。

如需其他資訊，請參閱以下內容：

- [使用 CloudFront 保護您的內容交付](#)
- [Amazon OpenSearch Service 上的 SIEM](#)

主題

- [透過 CloudFront 使用 HTTPS](#)
- [使用備用網域名稱和 HTTPS](#)
- [相互 TLS \(mTLS\) 檢視器](#)
- [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)
- [限制對 AWS 原始伺服器的存取](#)
- [限制對 Application Load Balancers 的存取](#)
- [限制您內容的地理分佈](#)
- [使用欄位層級加密來協助保護敏感資料](#)

透過 CloudFront 使用 HTTPS

您可以設定 CloudFront 請求檢視器使用 HTTPS，如此當 CloudFront 與檢視器進行通訊時，連線即可加密。您也可以設定 CloudFront 使用 HTTPS 與原始伺服器連線，如此當 CloudFront 與原始伺服器進行通訊時，連線即可加密。

如果您設定 CloudFront 請求 HTTPS 皆與檢視器和原始伺服器通訊，以下是 CloudFront 接收請求時會發生的情況：

1. 檢視器提交一個 HTTPS 請求給 CloudFront。檢視器和 CloudFront 之間有一些 SSL/TLS 溝通。最後，檢視器以加密格式提交請求。
2. 如果 CloudFront 節點包含快取的回應，則 CloudFront 會加密回應，並將其傳回檢視器，由檢視器進行解密。
3. 如果 CloudFront 節點未包含快取的回應，則 CloudFront 會執行與原始伺服器的 SSL/TLS 溝通，當溝通完成時，以加密格式將請求轉傳給原始伺服器。
4. 原始伺服器會解密請求、處理請求 (產生回應)、加密回應，並將回應傳回 CloudFront。
5. CloudFront 會解密回應、重新加密，以及將其轉傳給檢視器。CloudFront 還會在節點快取回應，以便在下次請求時可以使用該回應。
6. 檢視器解密回應。

無論您的原始伺服器是 Amazon S3 儲存貯體、MediaStore 或自訂原始伺服器，例如 HTTP/S 伺服器，程序基本上都以相同的方式運作：

Note

為了協助阻止 SSL 重新交涉類型攻擊，CloudFront 並不支援檢視器和原始伺服器請求的重新交涉。

或者，您可以為 CloudFront 分佈開啟交互身分驗證。如需詳細資訊，請參閱[相互 TLS \(mTLS\) 檢視器](#)。

如需如何在檢視器和 CloudFront 之間，以及在 CloudFront 和原始伺服器之間使用 HTTPS 的相關資訊，請參閱下列主題。

主題

- [檢視器和 CloudFront 之間的通訊需使用 HTTPS](#)

- [要求 CloudFront 與自訂原始伺服器之間的通訊使用 HTTPS](#)
- [請求使用 HTTPS 在 CloudFront 與您的 Amazon S3 原始伺服器之間進行通訊](#)
- [檢視器和 CloudFront 之間支援的通訊協定和密碼](#)
- [CloudFront 和原始伺服器之間支援的通訊協定和密碼](#)

檢視器和 CloudFront 之間的通訊需使用 HTTPS

您可以在 CloudFront 分發中設定一或多個快取行為以獲得檢視器與 CloudFront 之間進行通訊所需的 HTTPS。您也可以設定一或多個快取行為以允許 HTTP 和 HTTPS，以便 CloudFront 為某些物件使用 HTTPS，而非為其他物件。組態步驟取決於您在物件 URL 中使用哪個網域名稱：

- 如果您使用 CloudFront 指派給分發的網域名稱，例如 d111111abcdef8.cloudfront.net，請您變更一或多個快取行為的 檢視器通訊協定原則設定，使用 HTTPS 通訊。在該組態，CloudFront 提供 SSL/TLS 憑證。

若要使用 CloudFront 主控台來變更 檢視器通訊協定原則的值，請參閱本節稍後說明的步驟。

如需如何使用 CloudFront API 變更 ViewerProtocolPolicy 元素值的詳細資訊，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

- 如果您使用自己的網域名稱，如 example.com，則您需要變更多種 CloudFront 設定。您也需要使用 AWS Certificate Manager (ACM) 提供的 SSL/TLS 憑證，或是從第三方憑證授權機構或 IAM 憑證存放區將憑證匯入至 ACM。如需詳細資訊，請參閱 [使用備用網域名稱和 HTTPS](#)。

Note

如果您想要確保檢視器從 CloudFront 取得的物件是否加密，當 CloudFront 從您的原始伺服器取得他們時，請一律使用 CloudFront 與原始伺服器間的 HTTPS。如果您最近在 CloudFront 與原始伺服器之間從 HTTP 變更到 HTTPS，我們建議您使 CloudFront 節點中的物件失效。CloudFront 會將物件傳回到檢視器，無論檢視器使用的通訊協定 (HTTP 或 HTTPS) 與 CloudFront 用來取得物件的通訊協定是否相符。如需有關移除或替換分佈中物件的詳細資訊，請參閱[新增、移除或取代 CloudFront 分配的內容](#)。

檢視器需要 HTTPS

若要在一個或多個快取行為的檢視器與 CloudFront 之間使用 HTTPS，請執行下列程序。

若要將 CloudFront 設定為在檢視器與 CloudFront 之間請求使用 HTTPS

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 CloudFront 主控台的頂端窗格中，請選擇您要更新的分發 ID。
3. 在行為索引標籤中，選擇您要更新的快取行為，然後選擇編輯。
4. 請針對檢視器通訊協定政策指定下列其中一個值：

重新導向 HTTP 到 HTTPS

檢視器可以使用這兩種通訊協定。自動將 HTTP GET 和 HEAD 請求重新導向到 HTTPS 請求。CloudFront 傳回 HTTP 狀態碼 301 (永久移動) 與新的 HTTPS URL。然後檢視器使用 HTTPS URL 重新將請求提交到 CloudFront。

Important

如果您使用 HTTP 通過 HTTP 將 POST、PUT、DELETE、OPTIONS 或 PATCH 傳送到 HTTPS 快取行為和 HTTP 1.1 或更新的請求協定版本，則 CloudFront 請求將重新導向到有 HTTP 狀態碼 307 (暫時重新導向) 的 HTTPS 位置。這可確保再次將請求傳送到使用相同方法與內容承載的新位置。

如果您通過 HTTP 將 POST、PUT、DELETE、OPTIONS 或 PATCH 請求傳送到具有請求通訊版本低於 HTTP 1.1 的 HTTPS，則 CloudFront 會傳回 HTTP 狀態碼 403 (禁止)。

當檢視器執行一個 HTTP 請求供其重新導向到 HTTPS 請求時，CloudFront 會收取這兩個請求的費用。針對 HTTP 請求，只會收取 CloudFront 傳回到檢視器的請求與標頭之費用。針對 HTTPS 請求，會收取該請求與原始伺服器傳回的標頭及物件之費用。

僅限 HTTPS

檢視器只能在使用 HTTPS 的情況下存取您的內容。如果檢視器傳送 HTTP 請求，而不是 HTTPS 請求，則 CloudFront 傳回 HTTP 狀態碼 403 (禁止)，且不會傳回物件。

5. 選擇儲存變更。
6. 對於您希望在檢視器和 CloudFront 間需要使用 HTTPS 的每一個額外快取行為，重複步驟 3 至 5。
7. 您使用生產環境中已更新的組態之前，請先確認以下項目：

- 每個快取行為中的路徑模式僅適用於您想要檢視器使用 HTTPS 的請求。
- 以您想要 CloudFront 評估的順序列出快取行為。如需詳細資訊，請參閱 [路徑模式](#)。
- 快取行為會將請求路由到正確的原始伺服器。

要求 CloudFront 與自訂原始伺服器之間的通訊使用 HTTPS

您可以要求 CloudFront 與自訂原始伺服器之間的通訊使用 HTTPS。

Note

如果您的原始伺服器是設定為網站端點的 Amazon S3 儲存貯體，則無法設定 CloudFront 使用 HTTPS 來與原始伺服器通訊，因為 Amazon S3 不支援網站端點的 HTTPS。

若要在 CloudFront 與原始伺服器之間要求使用 HTTPS，請依照本主題中的程序執行下列步驟：

1. 在您的分佈中，針對原始伺服器變更其 Origin Protocol Policy (原始伺服器通訊協定政策) 設定。
2. 在您的原始伺服器安裝 SSL/TLS 憑證（當您使用 Amazon S3 原始伺服器或其他特定 AWS 原始伺服器時，不需要此操作）。

主題

- [自訂原始伺服器需要 HTTPS](#)
- [在您的自訂原始伺服器上安裝 SSL/TLS 憑證](#)

自訂原始伺服器需要 HTTPS

下列程序說明如何設定 CloudFront 使用 HTTPS 與 ELB 負載平衡器、Amazon EC2 執行個體或其他自訂原始伺服器通訊。如需使用 CloudFront API 更新分佈的詳細資訊，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

設定 CloudFront 在 CloudFront 和您的自訂原始伺服器間需使用 HTTPS

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 CloudFront 主控台的頂端窗格中，請選擇您要更新的分發 ID。

3. 在行為索引標籤中，選擇您要更新的原始伺服器，然後選擇編輯。
4. 更新下列設定：

原始伺服器通訊協定政策

針對您分佈中適用的原始伺服器，變更其 Origin Protocol Policy (原始伺服器通訊協定政策)：

- 僅限 HTTPS – CloudFront 只會使用 HTTPS 來和自訂原始伺服器進行通訊。
- 比對檢視器 – CloudFront 會根據檢視器請求的通訊協定，使用 HTTP 或 HTTPS 與您的原始伺服器進行通訊。例如，如果您為原始伺服器通訊協定原則選擇比對檢視器，且檢視器使用 HTTPS 從 CloudFront 要求一個物件，則 CloudFront 也使用 HTTPS 將請求轉送到原始伺服器。

只有針對 Viewer Protocol Policy (檢視器通訊協定政策) 指定 Redirect HTTP to HTTPS (將 HTTP 重新導向至 HTTPS) 或 HTTPS Only (僅限 HTTPS) 時，才能選擇 Match Viewer (配合檢視器)。

請注意 CloudFront 僅會在檢視器使用 HTTP 和 HTTPS 協定發出請求時，快取物件一次。

原始伺服器 SSL 通訊協定

針對分佈中適用的原始伺服器，選擇 Origin SSL Protocols (原始伺服器 SSL 通訊協定)。SSLv3 通訊協定較不安全，因此我們建議您只有在原始伺服器不支援 TLSv1 或新版本時，選擇 SSLv3。TLSv1 交握與 SSLv3 有回溯相容與正向相容，但 TLSv1.1 和後續版本則無。當您選擇 SSLv3 時，CloudFront 只會傳送 SSLv3 交握請求。

5. 選擇儲存變更。
6. 對於您希望在 CloudFront 和自訂原始伺服器間需要使用 HTTPS 的每一個額外原始伺服器，重複步驟 3 至 5。
7. 您使用生產環境中已更新的組態之前，請先確認以下項目：
 - 每個快取行為中的路徑模式僅適用於您想要檢視器使用 HTTPS 的請求。
 - 以您想要 CloudFront 評估的順序列出快取行為。如需詳細資訊，請參閱 [路徑模式](#)。
 - 快取行為會將請求轉傳到您已變更其 Origin Protocol Policy (原始伺服器通訊協定政策) 的原始伺服器。

在您的自訂原始伺服器上安裝 SSL/TLS 憑證

在自訂原始伺服器上，您可以從以下來源使用 SSL/TLS 憑證：

- 如果您的原始伺服器是 ELB 負載平衡器，您可以使用 AWS Certificate Manager (ACM) 提供的憑證。您也可以使用信任第三方憑證授權單位簽署的憑證並匯入 ACM。
- 針對 ELB 負載平衡器以外的原始伺服器，您必須使用信任第三方憑證授權單位 (CA) 簽署的憑證，例如 Comodo、DigiCert 或 Symantec。

從原始伺服器傳回的憑證必須包含下列其中一個網域名稱：

- 原始伺服器的 Origin domain (原始伺服器網域) 欄位 (CloudFront API 中的 DomainName 欄位) 中的網域名稱。
- 如果快取行為已設定為轉發 Host 標頭至原始伺服器，則為 Host 標頭中的網域名稱。

當 CloudFront 使用 HTTPS 與您的原始伺服器通訊時，CloudFront 驗證信任憑證授權機構發出的憑證。CloudFront 支援與 Mozilla 相同的憑證授權機構。如需目前的清單，請參閱 [Mozilla Included CA Certificate List](#)。您無法使用自我簽署憑證進行 CloudFront 和原始伺服器之間的 HTTPS 通訊。

Important

如果該原始伺服器傳回過期憑證、無效憑證，或自我簽署憑證，或者，如果原始伺服器傳回順序錯誤的憑證鍊，則 CloudFront 失去 TCP 連線、傳回 HTTP 狀態碼 502 (無效的閘道) 至檢視者，並將 X-Cache 標頭設定為 Error from cloudfront。此外，若完整的憑證鏈結 (包含中繼憑證) 不存在，CloudFront 便會卸除 TCP 連線。

請求使用 HTTPS 在 CloudFront 與您的 Amazon S3 原始伺服器之間進行通訊

當您的原始伺服器是 Amazon S3 bucket 儲存貯體，您能否選擇使用 HTTPS 與 CloudFront 通訊，取決於您使用儲存貯體的方式。如果設定 Amazon S3 儲存貯體為網站端點，則您無法設定 CloudFront 使用 HTTPS 與原始伺服器進行通訊，因為 Amazon S3 在該組態中不支援 HTTPS 連線。

如果您的原始伺服器是支援 HTTPS 通訊的 Amazon S3 儲存貯體，則 CloudFront 會使用檢視器用來提交請求的通訊協定，來將請求轉送到 S3。[通訊協定 \(僅限自訂原始伺服器\)](#) 設定的預設為比對檢視器，而且無法變更。不過如果您為 Amazon S3 原始伺服器啟用原始存取控制 (OAC)，CloudFront 和 Amazon S3 之間使用的通訊將取決於您的設定。如需詳細資訊，請參閱[建立新的原始存取控制](#)。

如果您要使用 HTTPS 進行 CloudFront 與 Amazon S3 之間的通訊，必須變更 檢視器通訊協定原則 的值為重新導向 HTTP 到 HTTPS 或 僅限 HTTP。本節稍後的步驟，說明如何使用 CloudFront 主控台來

變更檢視器通訊協定原則。如需使用 CloudFront API 來更新分佈 ViewerProtocolPolicy 元素的詳細資訊，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

當您搭配支援 HTTPS 通訊的 Amazon S3 儲存貯體使用 HTTPS 時，Amazon S3 會提供 SSL/TLS 憑證，您則不需要。

Amazon S3 原始伺服器需要 HTTPS

下列程序示範如何設定 CloudFront，以要求使用 HTTPS 與您的 Amazon S3 原始伺服器通訊。

若要將 CloudFront 設定為請求使用 HTTPS 連線到您的 Amazon S3 原始伺服器

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在 CloudFront 主控台的頂端窗格中，請選擇您要更新的分發 ID。
3. 在 Behaviors (行為) 索引標籤中，選擇您要更新的快取行為，然後選擇 Edit (編輯)。
4. 請針對 Viewer Protocol Policy (檢視器通訊協定政策) 指定下列其中一個值：

重新引導 HTTP 到 HTTPS

檢視器可以使用這兩種通訊協定，但會自動重新導向 HTTP 請求到 HTTPS 請求。CloudFront 傳回 HTTP 狀態碼 301 (永久移動) 與新的 HTTPS URL。然後檢視器使用 HTTPS URL 重新將請求提交到 CloudFront。

Important

CloudFront 不會從 HTTP 將 DELETE、OPTIONS、PATCH、POST 或 PUT 請求重新引導到 HTTPS。如果您配置快取行為重新導向至 HTTPS，則 CloudFront 回應該快取行為的 HTTP DELETE、OPTIONS、PATCH、POST 或 PUT 請求，此快取行為具有 HTTP 狀態碼 403 (禁止)。

當檢視器執行一個 HTTP 請求供其重新導向到 HTTPS 請求時，CloudFront 會收取這兩個請求的費用。針對 HTTP 請求，只會收取 CloudFront 傳回到檢視器的請求與標頭之費用。針對 HTTPS 請求，會收取該請求與標頭及原始伺服器傳回的物件之費用。

僅限 HTTPS

檢視器只能在使用 HTTPS 的情況下存取您的內容。如果檢視器傳送 HTTP 請求，而不是 HTTPS 請求，則 CloudFront 傳回 HTTP 狀態碼 403 (禁止)，且不會傳回物件。

5. 請選擇 Yes, Edit (是, 編輯)。
6. 為每個額外快取行為, 供您要在檢視器與 CloudFront 之間以及 CloudFront 與 S3 之間使用 HTTPS, 請重複步驟 3 到 5。
7. 您使用生產環境中已更新的組態之前, 請先確認以下項目:
 - 每個快取行為中的路徑模式僅適用於您想要檢視器使用 HTTPS 的請求。
 - 以您想要 CloudFront 評估的順序列出快取行為。如需詳細資訊, 請參閱 [路徑模式](#)。
 - 快取行為會將請求路由到正確的原始伺服器。

檢視器和 CloudFront 之間支援的通訊協定和密碼

當您在[檢視器和 CloudFront 分佈之間請求 HTTPS](#)時, 您必須選擇[安全性政策](#), 以決定下列設定。

- CloudFront 用來與檢視者通訊的最低 SSL/TLS 通訊協定。
- CloudFront 可用來加密與檢視器通訊的密碼。

若要選擇安全政策, 請指定適用於 [安全政策 \(最低 SSL/TLS 版本\)](#) 的適用值。下表列出 CloudFront 可用於每個安全原則的協定與密碼。

檢視器必須支援至少一個受支援的密碼, 建立與 CloudFront 的 HTTPS 連線。CloudFront 從檢視器支援的密碼中依列出的順序選擇密碼。另請參閱[OpenSSL、S2n 和 RFC 密碼名稱](#)。

	安全政策								
	SSLv3	TLSv1	TLSv1_6	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025

支援的 SSL/TLS 通訊協定

TLSv1.3	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLSv1.2	◆	◆	◆	◆	◆	◆	◆	◆	
TLSv1.1	◆	◆	◆	◆					
TLSv1	◆	◆	◆						
SSLv3	◆								

	安全政策								
	SSLv3	TLSv1	TLSv1.2	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025

支援的 TLSv1.3 密碼

TLS_AES_128_GCM_SHA256	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_AES_256_GCM_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_CHACHA20_POLY1305_SHA256	◆	◆	◆	◆	◆	◆	◆		◆

支援的 ECDSA 密碼

ECDHE-ECDSA-AES128-GCM-SHA256	◆	◆	◆	◆	◆	◆	◆	◆	
ECDHE-ECDSA-AES128-SHA256	◆	◆	◆	◆	◆	◆			
ECDHE-ECDSA-AES128-SHA	◆	◆	◆	◆					
ECDHE-ECDSA-AES256-GCM-SHA384	◆	◆	◆	◆	◆	◆	◆	◆	
ECDHE-ECDSA-CHACHA20-POLY1305	◆	◆	◆	◆	◆	◆	◆		
ECDHE-ECDSA-AES256-SHA384	◆	◆	◆	◆	◆	◆			

	安全政策								
	SSLv3	TLSv1	TLSv1.0_6	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025
ECDHE-ECDSA-AES256-SHA	◆	◆	◆	◆					

支援的 RSA 密碼

ECDHE-RSA-AES128-GCM-SHA256	◆	◆	◆	◆	◆	◆	◆	◆	
ECDHE-RSA-AES128-SHA256	◆	◆	◆	◆	◆	◆			
ECDHE-RSA-AES128-SHA	◆	◆	◆	◆					
ECDHE-RSA-AES256-GCM-SHA384	◆	◆	◆	◆	◆	◆	◆	◆	
ECDHE-RSA-CHACHA20-POLY1305	◆	◆	◆	◆	◆	◆	◆		
ECDHE-RSA-AES256-SHA384	◆	◆	◆	◆	◆	◆			
ECDHE-RSA-AES256-SHA	◆	◆	◆	◆					
AES128-GCM-SHA256	◆	◆	◆	◆	◆				
AES256-GCM-SHA384	◆	◆	◆	◆	◆				

	安全政策								
	SSLv3	TLSv1	TLSv1.2	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025
AES128-SHA256	◆	◆	◆	◆	◆				
AES256-SHA	◆	◆	◆	◆					
AES128-SHA	◆	◆	◆	◆					
DES-CBC3-SHA	◆	◆							
RC4-MD5	◆								

OpenSSL、S2n 和 RFC 密碼名稱

OpenSSL and [s2n](#) 會使用與 TLS 標準不同的密碼名稱 ([RFC 2246](#)、[RFC 4346](#)、[RFC 5246](#) 和 [RFC 8446](#))。下表將 OpenSSL 和 s2n 名稱對應到每個密碼的 RFC 名稱。

CloudFront 支援傳統和量子安全金鑰交換。對於使用橢圓曲線的傳統金鑰交換，CloudFront 支援下列項目：

- prime256v1
- X25519
- secp384r1

對於保護量子的金鑰交換，CloudFront 支援下列項目：

- X25519MLKEM768
- SecP256r1MLKEM768

Note

TLS 1.3 僅支援 Quantum 安全金鑰交換。TLS 1.2 和舊版不支援規定人數安全金鑰交換。

如需詳細資訊，請參閱下列主題：

- [後量子密碼學](#)
- [密碼編譯演算法和 AWS 服務](#)
- [TLS 1.3 中的混合金鑰交換](#)

如需 CloudFront 憑證需求的詳細資訊，請參閱 [與 CloudFront 搭配使用 SSL/TLS 憑證的要求](#)。

OpenSSL 和 s2n 密碼名稱	RFC 密碼名稱
支援的 TLSv1.3 密碼	
TLS_AES_128_GCM_SHA256	TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384	TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_SHA256	TLS_CHACHA20_POLY1305_SHA256
支援的 ECDSA 密碼	
ECDHE-ECDSA-AES128-GCM-SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
ECDHE-ECDSA-AES128-SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
ECDHE-ECDSA-AES128-SHA	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
ECDHE-ECDSA-AES256-GCM-SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
ECDHE-ECDSA-CHACHA20-POLY1305	TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
ECDHE-ECDSA-AES256-SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
ECDHE-ECDSA-AES256-SHA	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA

OpenSSL 和 s2n 密碼名稱	RFC 密碼名稱
支援的 RSA 密碼	
ECDHE-RSA-AES128-GCM-SHA256	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
ECDHE-RSA-AES128-SHA256	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
ECDHE-RSA-AES128-SHA	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
ECDHE-RSA-AES256-GCM-SHA384	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
ECDHE-RSA-CHACHA20-POLY1305	TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
ECDHE-RSA-AES256-SHA384	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
ECDHE-RSA-AES256-SHA	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
AES128-GCM-SHA256	TLS_RSA_WITH_AES_128_GCM_SHA256
AES256-GCM-SHA384	TLS_RSA_WITH_AES_256_GCM_SHA384
AES128-SHA256	TLS_RSA_WITH_AES_128_CBC_SHA256
AES256-SHA	TLS_RSA_WITH_AES_256_CBC_SHA
AES128-SHA	TLS_RSA_WITH_AES_128_CBC_SHA
DES-CBC3-SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA
RC4-MD5	TLS_RSA_WITH_RC4_128_MD5

檢視器和檢視器之間支援的簽名結構描述和 CloudFront

CloudFront 支援下列簽名結構描述，用於與檢視器之間的連結和 CloudFront。

簽章結構描述	安全政策								
	SSLv3	TLSv1	TLSv1.6	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025
TLS_SIGNATURE_RSA_PSS_SHA256	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_PSS_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_PSS_SHA512	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_SAE_SHA256	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_SAE_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_SAE_SHA512	◆	◆	◆	◆	◆	◆	◆	◆	◆

	安全政策								
簽章結構描述	SSLv3	TLSv1	TLSv1.6	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025
TLS_SIGNATURE_RSA_SHA256	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_SHA512	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_ECDSA_SHA256	◆	◆	◆	◆	◆	◆	◆		
TLS_SIGNATURE_ECDSA_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_ECDSA_SHA512	◆	◆	◆	◆	◆	◆	◆	◆	◆

	安全政策								
簽章結構描述	SSLv3	TLSv1	TLSv1.6	TLSv1.016	TLSv1.018	TLSv1.019	TLSv1.021	TLSv1.025	TLSv1.3_2025
TLS_SIGNATURE_ECDSA_SHA224	◆	◆	◆	◆	◆	◆	◆		
TLS_SIGNATURE_ECDSA_SECP256R1_SHA256	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_ECDSA_SECP384R1_SHA384	◆	◆	◆	◆	◆	◆	◆	◆	◆
TLS_SIGNATURE_RSA_PKCS1_SHA1	◆	◆	◆	◆					
TLS_SIGNATURE_ECDSA_SHA1	◆	◆	◆	◆					

CloudFront 和原始伺服器之間支援的通訊協定和密碼

如果您選擇[請求在 CloudFront 與原始伺服器之間使用 HTTPS](#)，則可以決定[允許使用哪種 SSL/TLS 通訊協定](#)來進行安全的連線，然後 CloudFront 可以使用下方表格中列出的任何 ECDSA 或 RSA 密碼，連線至原始伺服器。您的原始伺服器必須支援至少一種 CloudFront 密碼，才能建立與原始伺服器的 HTTPS 連線。

OpenSSL and s2n 會使用與 TLS 標準不同的密碼名稱 ([RFC 2246](#)、[RFC 4346](#)、[RFC 5246](#) 和 [RFC 8446](#))。下表將包含每種密碼的 OpenSSL 和 s2n 名稱以及 RFC 名稱。

對於具有橢圓曲線金鑰交換演算法的密碼，CloudFront 支援下列橢圓曲線：

- prime256v1
- secp384r1
- X25519

OpenSSL 和 s2n 密碼名稱	RFC 密碼名稱
支援的 ECDSA 密碼	
ECDHE-ECDSA-AES256-GCM-SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
ECDHE-ECDSA-AES256-SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
ECDHE-ECDSA-AES256-SHA	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
ECDHE-ECDSA-AES128-GCM-SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
ECDHE-ECDSA-AES128-SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
ECDHE-ECDSA-AES128-SHA	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
支援的 RSA 密碼	
ECDHE-RSA-AES256-GCM-SHA384	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
ECDHE-RSA-AES256-SHA384	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
ECDHE-RSA-AES256-SHA	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

OpenSSL 和 s2n 密碼名稱	RFC 密碼名稱
ECDHE-RSA-AES128-GCM-SHA256	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
ECDHE-RSA-AES128-SHA256	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
ECDHE-RSA-AES128-SHA	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
AES256-SHA	TLS_RSA_WITH_AES_256_CBC_SHA
AES128-SHA	TLS_RSA_WITH_AES_128_CBC_SHA
DES-CBC3-SHA	TLS_RSA_WITH_3DES_EDE_CBC_SHA
RC4-MD5	TLS_RSA_WITH_RC4_128_MD5

CloudFront 與原始伺服器之間支援的簽名方案

CloudFront 支援下列簽署設定，用於 CloudFront 和原始伺服器之間的連線。

- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA256
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA384
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA512
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA224
- TLS_SIGNATURE_SCHEME_ECDSA_SHA256
- TLS_SIGNATURE_SCHEME_ECDSA_SHA384
- TLS_SIGNATURE_SCHEME_ECDSA_SHA512
- TLS_SIGNATURE_SCHEME_ECDSA_SHA224
- TLS_SIGNATURE_SCHEME_RSA_PKCS1_SHA1
- TLS_SIGNATURE_SCHEME_ECDSA_SHA1

使用備用網域名稱和 HTTPS

若您想要在檔案的 URL 中使用自己的網域名稱 (例如 `https://www.example.com/image.jpg`)，並希望您的檢視器使用 HTTPS，則必須完成下列主題的步驟。(如果您在 URL 中使用預設的 CloudFront 分佈網域名稱，例如 `https://d1111111abcdef8.cloudfront.net/image.jpg`，請改為遵循下列主題中的指示：[檢視器和 CloudFront 之間的通訊需使用 HTTPS](#)。)

Important

當您新增憑證到分佈，CloudFront 會立即傳播憑證到其所有節點。隨著新節點變為可用，CloudFront 也會傳播憑證到這些位置。您無法限制 CloudFront 傳播憑證的節點。

主題

- [選擇 CloudFront 提供 HTTPS 請求的方式](#)
- [與 CloudFront 搭配使用 SSL/TLS 憑證的要求](#)
- [搭配 CloudFront 使用 SSL/TLS 憑證的配額 \(僅限檢視器與 CloudFront 之間的 HTTPS\)](#)
- [設定備用網域名稱和 HTTPS](#)
- [判斷 SSL/TLS RSA 憑證中公有金鑰的大小](#)
- [增加 SSL/TLS 憑證的配額](#)
- [輪換 SSL/TLS 憑證](#)
- [從自訂 SSL/TLS 憑證還原到預設 CloudFront 憑證](#)
- [若要從具有專用 IP 位址的自訂 SSL/TLS 憑證切換到 SNI](#)

選擇 CloudFront 提供 HTTPS 請求的方式

如果您想要檢視器使用 HTTPS 與使用檔案的備用網域名稱，請選擇以下其中一個選項以決定 CloudFront 如何提供 HTTPS 請求：

- 使用[伺服器名稱指示 \(SNI\)](#) – 推薦
- 在每個節點使用專用的 IP 位址

本節說明每個選項的運作方式。

使用 SNI 提供 HTTPS 請求 (適用於大部分用戶端)

[伺服器名稱指示 \(SNI\)](#) 是 TLS 通訊協定的延伸，2010 年之後推出的瀏覽器 and 用戶端支援此選項。如果設定 CloudFront 使用 SNI 來提供 HTTPS 請求，則 CloudFront 會將您的替代網域名稱，與每個節點中的 IP 位址建立關聯。當檢視器提交內容的 HTTPS 請求時，DNS 會將請求路由到正確節點的 IP 位址。您網域名稱的 IP 位址在 SSL/TLS 交握溝通期間決定；IP 位址並非專用於您的分佈。

SSL/TLS 溝通發生在建立 HTTPS 連線的初期。如果 CloudFront 無法立即判斷請求是否針對哪些網域，它就會失去連線。當 SNI 支援的檢視器提交內容的 HTTPS 請求時，情況如下：

1. 檢視器會自動從請求 URL 取得網域名稱，並將其新增至 TLS 用戶端 hello 訊息的 SNI 延伸模組。
2. 當 CloudFront 收到 TLS 用戶端 hello 時，它會使用 SNI 延伸模組中的網域名稱來尋找相符的 CloudFront 分佈，並傳回相關聯的 TLS 憑證。
3. 檢視器和 CloudFront 執行 SSL/TLS 交涉。
4. CloudFront 將請求的內容傳回到檢視器。

關於支援 SNI 的瀏覽器，最新的清單請參閱 Wikipedia 條目 [Server Name Indication](#) (伺服器名稱指示)。

如果您想要使用 SNI，但某些使用者的瀏覽器不支援 SNI，則您有幾個選項：

- 設定 CloudFront 使用專用 IP 位址而不是 SNI，提供 HTTPS 請求。如需詳細資訊，請參閱 [使用專用 IP 位址提供 HTTPS 請求 \(適用於所有用戶端\)](#)。
- 使用 CloudFront SSL/TLS 憑證，不使用自訂憑證。這需要您在檔案的 URL 中，使用您分佈的 CloudFront 網域名稱，例如，`https://d1111111abcdef8.cloudfront.net/logo.png`。

如果您使用預設 CloudFront 憑證，檢視器必須支援 SSL 協定 TLSv1 或更新版本。CloudFront 不支援具有預設 CloudFront 憑證的 SSLv3。

您也必須將 CloudFront 使用的 SSL/TLS 憑證，從自訂憑證變更為預設的 CloudFront 憑證：

- 如果您尚未使用分佈來分配內容，可以只變更組態。如需詳細資訊，請參閱 [更新分佈](#)。
- 如果您已經使用分佈來分配內容，則您必須建立新的 CloudFront 分配和變更檔案的 URL，以降低或消除內容不可用的時間。如需詳細資訊，請參閱 [從自訂 SSL/TLS 憑證還原到預設 CloudFront 憑證](#)。
- 如果您可以控制您的使用者使用哪些瀏覽器，請讓他們升級到支援 SNI 的瀏覽器。
- 使用 HTTP，而非 HTTPS。

使用專用 IP 位址提供 HTTPS 請求 (適用於所有用戶端)

伺服器名稱指示 (SNI) 是將請求關聯至網域的一個方法。另一個方式是使用專用 IP 位址。如果您有使用者無法升級到 2010 年後推出的瀏覽器或用戶端，您可以使用專用 IP 位址來提供 HTTPS 請求。關於支援 SNI 的瀏覽器，最新的清單請參閱 Wikipedia 條目 [Server Name Indication](#) (伺服器名稱指示)。

Important

如果您設定 CloudFront 使用專用 IP 位址提供 HTTPS 請求，您需要支付額外的每月費用。當您把 SSL/TLS 憑證與分佈相關聯並啟用此分佈時，即開始收費。如需 CloudFront 定價的詳細資訊，請參閱 [Amazon CloudFront 定價](#)。除此之外：請參閱 [Using the Same Certificate for Multiple CloudFront Distributions](#)。

當您設定 CloudFront 為使用專用 IP 位址來提供 HTTPS 請求，CloudFront 會將您的憑證與每個 CloudFront 邊緣節點中的專用 IP 位址建立關聯。當檢視器提交內容的 HTTPS 請求時，情況如下：

1. DNS 將請求路由到適用節點中分佈的 IP 位址。
2. 如果用戶端請求在 ClientHello 訊息中提供 SNI 延伸，CloudFront 會搜尋與該 SNI 相關聯的分佈。
 - 如果有相符項目，CloudFront 會使用 SSL/TLS 憑證回應請求。
 - 如果找不到相符項目，CloudFront 將改用 IP 位址識別您的分佈，並判斷要將哪些 SSL/TLS 憑證傳回到檢視器。
3. 檢視器和 CloudFront 使用 SSL/TLS 憑證執行 SSL/TLS 溝通。
4. CloudFront 將請求的內容傳回到檢視器。

這種方式適用於每個 HTTPS 請求，無論使用者使用的是瀏覽器或其他檢視器。

Note

專用 IP 不是靜態 IP，而且可能會隨著時間而變更。為邊緣節點傳回的 IP 位址會從 [CloudFront 邊緣伺服器清單](#) 的 IP 位址範圍動態分配。

CloudFront 邊緣伺服器的 IP 位址範圍可能變更。若要收到 IP 地址變更的通知，[請透過 Amazon SNS 訂閱 AWS 公有 IP 地址變更](#)。

請求使用三個或更多專用 IP SSL/TLS 憑證的許可

如果您需要三個或多個 SSL/TLS 專用的 IP 憑證與 CloudFront 永久關聯的許可，請執行下列程序。如需 HTTPS 請求的詳細資訊，請參閱 [選擇 CloudFront 提供 HTTPS 請求的方式](#)。

Note

此程序用於使用 3 個或多個跨 CloudFront 分佈的專用 IP 憑證時。預設值為 2。請注意，您不可以繫結超過 1 個 SSL 憑證到分佈。

您一次只能將一個 SSL/TLS 憑證與 CloudFront 分佈相關聯。此數字為您可以跨所有 CloudFront 分佈使用的專用 IP SSL 憑證之總和。

請求使用具有一個 CloudFront 分佈的三個或多個憑證之許可

1. 請前往[支援中心](#)並建立案例。
2. 請指出您需要許可才能使用的憑證有多少，以及說明請求中的情況。我們會儘快更新您的帳戶。
3. 請繼續下一個程序。

與 CloudFront 搭配使用 SSL/TLS 憑證的要求

本主題描述 SSL/TLS 憑證的需求。它們適用於以下兩種情況 (除非另有說明)：

- 適用於檢視器和 CloudFront 之間使用 HTTPS 的憑證
- 適用於 CloudFront 與原始伺服器之間使用 HTTPS 的憑證

主題

- [憑證發行者](#)
- [AWS 區域 適用於 的 AWS Certificate Manager](#)
- [憑證格式](#)
- [中繼憑證](#)
- [Key type](#)
- [私有金鑰](#)
- [許可](#)

- [憑證金鑰的大小](#)
- [支援的憑證類型](#)
- [憑證過期日期和續約](#)
- [CloudFront 分佈和憑證中的網域名稱](#)
- [最低 SSL/TLS 通訊協定版本](#)
- [支援的 HTTP 版本](#)

憑證發行者

我們建議您使用由 [AWS Certificate Manager \(ACM\)](#) 出具的憑證。如需有關如何從 ACM 取得憑證的資訊，請參閱 [AWS Certificate Manager 使用者指南](#)。若要搭配 CloudFront 分佈使用 ACM 憑證，請確定您是在美國東部 (維吉尼亞北部) 區域 (us-east-1) 請求 (或匯入) 憑證。

CloudFront 支援與 Mozilla 相同的憑證授權機構 (CA)，因此如果您不使用 ACM，請使用由 [Mozilla 所含憑證授權機構憑證清單](#) 上的 CA 發行的憑證。

您為 CloudFront 分佈指定的原始伺服器所使用的 TLS 憑證也需要從 Mozilla 包含 CA 憑證清單上的 CA 發行。

如需有關如何取得和安裝憑證的詳細資訊，請參閱 HTTP 伺服器軟體的文件和 CA 的文件。

AWS 區域 適用於 的 AWS Certificate Manager

若要在 AWS Certificate Manager (ACM) 中使用憑證來要求檢視器和 CloudFront 之間的 HTTPS，請務必在美國東部 (維吉尼亞北部) 區域 () 請求 (或匯入) 憑證 us-east-1。

如果您想要在 CloudFront 和原始伺服器之間要求 HTTPS，而且您正在使用 ELB 中的負載平衡器做為原始伺服器，您可以在任何 中請求或匯入憑證 AWS 區域。

憑證格式

憑證必須為 X.509 PEM 格式。如果您使用 AWS Certificate Manager，這是預設格式。

中繼憑證

如果您使用第三方憑證授權機構 (CA)，請在列出位於 .pem 檔案的憑證鏈中的所有中繼憑證，從為您網域簽屬憑證的 CA 開始。一般而言，您可以在 CA 網站上找到以適當鏈結順序列出中繼和根憑證的檔案。

⚠ Important

不包括下列項目：根憑證、不在信任路徑的中繼憑證，或您 CA 的公有金鑰憑證。

範例如下：

```
-----BEGIN CERTIFICATE-----  
Intermediate certificate 2  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
Intermediate certificate 1  
-----END CERTIFICATE-----
```

Key type

CloudFront 支援 RSA 和公有/私有金鑰對。

CloudFront 支援使用 RSA 和 ECDSA 憑證與檢視器和原始伺服器進行 HTTPS 連線。搭配 [AWS Certificate Manager \(ACM\)](#)，您可以請求和匯入 RSA 或 ECDSA 憑證，然後將它們與 CloudFront 分佈建立關聯。

如需可在 HTTPS 連線中交涉之 CloudFront 所支援的 RSA 和 ECDSA 密碼清單，請參閱[the section called “檢視器和 CloudFront 之間支援的通訊協定和密碼”](#)和[the section called “CloudFront 和原始伺服器之間支援的通訊協定和密碼”](#)。

私有金鑰

如果您從第三方憑證授權單位 (CA) 使用憑證，請注意以下事項：

- 私有金鑰必須符合憑證中的公有金鑰。
- 私有金鑰必須是 PEM 格式。
- 無法用密碼加密私有金鑰。

如果 AWS Certificate Manager (ACM) 提供憑證，ACM 不會釋出私有金鑰。私有金鑰存放在 ACM 中，供與 ACM 整合的 AWS 服務使用。

許可

您必須具有使用和匯入 SSL/TLS 憑證的許可。如果您使用的是 AWS Certificate Manager (ACM)，我們建議您使用 AWS Identity and Access Management 許可來限制對憑證的存取。如需詳細資訊，請參閱 AWS Certificate Manager 使用者指南中的 [Identity and Access Management](#)。

憑證金鑰的大小

CloudFront 支援的憑證金鑰大小取決於金鑰和憑證的類型。

對於 RSA 憑證：

CloudFront 支援 1024 位元、2048 位元、3072 位元和 4096 位元 RSA 金鑰。與 CloudFront 搭配使用的 RSA 憑證，其金鑰長度上限為 4096 位元。

請注意，ACM 發行的 RSA 憑證最多擁有 2048 位元金鑰。若要使用 3072 位元或 4096 位元的 RSA 憑證，您需要從外部取得憑證並將其匯入 ACM，之後即可和 CloudFront 搭配使用。

如需有關如何判斷 RSA 金鑰大小的詳細資訊，請參閱[判斷 SSL/TLS RSA 憑證中公有金鑰的大小](#)。

對於 ECDSA 憑證：

CloudFront 支援 256 位元金鑰。若要在 ACM 中使用 ECDSA 憑證，以便在檢視器和 CloudFront 之間請求使用 HTTPS，請使用 prime256v1 橢圓曲線。

支援的憑證類型

CloudFront 支援由受信任憑證授權單位發行的所有憑證類型。

憑證過期日期和續約

如果您使用從第三方憑證授權機構 (CA) 取得的憑證，則必須監控憑證過期日期，並在憑證過期之前續約您匯入 AWS Certificate Manager (ACM) 或上傳至 AWS Identity and Access Management 憑證存放區的憑證。

Important

為了避免憑證過期問題，請在目前憑證的 NotAfter 值之前至少 24 小時續約或重新匯入憑證。如果您的憑證在 24 小時內過期，請向 ACM 請求新憑證或將新憑證匯入 ACM。接著，將新憑證與 CloudFront 分佈建立關聯。

當您的憑證續約或重新匯入正在進行時，CloudFront 可能會繼續使用先前的憑證。這是一個非同步程序，最多可能需要 24 小時的時間，CloudFront 才會顯示您的變更。

如果使用 ACM 提供的憑證，ACM 會為您管理憑證續約。如需詳細資訊，請參閱 AWS Certificate Manager 使用者指南中的[受管續約](#)。

CloudFront 分佈和憑證中的網域名稱

當您使用自訂原始伺服器時，原始伺服器上的 SSL/TLS 憑證包含 Common Name (通用名稱) 欄位中的網域名稱，且在 Subject Alternative Names (主體別名) 欄位中可能還有更多網域名稱。(CloudFront 在憑證網域名稱中支援萬用字元)。

憑證中其中一個網域名稱必須符合您指定給原始網域名稱的網域名稱。如果沒有相符的網域名稱，CloudFront 會將 HTTP 狀態碼 502 (Bad Gateway) 傳回到檢視器。

Important

如果將替代網域名稱新增到分佈，CloudFront 會檢查替代網域名稱是否涵蓋在您所連線的憑證範圍內。憑證必須涵蓋憑證主體別名 (SAN) 欄位中的備用網域名稱。這表示 SAN 欄位必須包含完全相符的替代網域名稱，或在您要新增之替代網域名稱的相同層級包含萬用字元。如需詳細資訊，請參閱[使用備用網域名稱的需求](#)。

最低 SSL/TLS 通訊協定版本

如果使用專用 IP 位址，請透過選擇安全政策來設定檢視器與 CloudFront 之間連線的最低 SSL/TLS 通訊協定版本。

如需詳細資訊，請參閱[所有分佈設定參考](#)主題中的[安全政策 \(最低 SSL/TLS 版本\)](#)。

支援的 HTTP 版本

如果您把一個憑證與多個 CloudFront 分發相關聯，則所有與憑證關聯的分發必須使用相同[支援的 HTTP 版本](#)的選項。您在建立或更新 CloudFront 分佈時請指定此選項。

搭配 CloudFront 使用 SSL/TLS 憑證的配額 (僅限檢視器與 CloudFront 之間的 HTTPS)

請注意以下將 SSL/TLS 憑證與 CloudFront 搭配使用的配額。這些配額僅適用於您使用匯入 ACM 的 AWS Certificate Manager (ACM) 佈建的 SSL/TLS 憑證，或上傳至 IAM 憑證存放區以在檢視器和 CloudFront 之間進行 HTTPS 通訊。

如需詳細資訊，請參閱[增加 SSL/TLS 憑證的配額](#)。

每個 CloudFront 分佈的憑證數目上限

您可以把一個 SSL/TLS 憑證的上限數量與每個 CloudFront 分佈相關聯。

您可以匯入 ACM 或上傳至 IAM 憑證存放區的憑證上限數量

如果您從第三方 CA 取得 SSL/TLS 憑證，您必須在以下其中一個位置存放憑證：

- AWS Certificate Manager – 如需 ACM 憑證數量的目前配額，請參閱 AWS Certificate Manager 使用者指南中的[配額](#)。列出的配額是包含您使用 ACM 佈建的憑證與匯入 ACM 的憑證之總和。
- IAM 憑證存放區 – 如需您可以上傳到 AWS 帳戶 IAM 憑證存放區之憑證數量的目前配額（先前稱為限制），請參閱《IAM [使用者指南](#)》中的 [IAM 和 STS 限制](#)。您可以在 Service Quotas 主控台請求提高配額。

每個 AWS 帳戶的憑證數量上限（僅限專用 IP 地址）

如果您想要使用專用 IP 位址提供 HTTPS 請求，請注意以下事項：

- 根據預設，CloudFront 會授予您在 AWS 帳戶中使用兩個憑證的許可，一個用於日常使用，另一個用於需要輪換多個分佈的憑證。
- 如果您需要兩個以上的自訂 SSL/TLS 憑證 AWS，您可以在 Service Quotas 主控台中請求更高的配額。

將相同的憑證用於不同 AWS 帳戶建立的 CloudFront 分佈

如果您使用的是第三方 CA，而且想要搭配使用不同 AWS 帳戶建立的多個 CloudFront 分佈使用相同的憑證，您必須將憑證匯入 ACM，或為每個 AWS 帳戶將其上傳至 IAM 憑證存放區一次。

如果您使用 ACM 提供的憑證，則不能配置 CloudFront 使用不同 AWS 帳戶建立的憑證。

針對 CloudFront 和其他服務使用相同的憑證 AWS

如果您從信任的憑證授權機構購買憑證，例如 Comodo、DigiCert 或 Symantec，您可以將相同的憑證用於 CloudFront 和其他 AWS 服務。如果將憑證匯入到 ACM，則只需要匯入一次，即可將憑證用於多個 AWS 服務。

如果您使用 ACM 提供的憑證，則在 ACM 中存放憑證。

將相同的憑證用於多個 CloudFront 分佈

您可以對用來提供 HTTPS 請求的任何或所有 CloudFront 分佈使用相同的憑證。注意下列事項：

- 您可以使用相同的憑證提供使用專用 IP 位址的請求，與提供使用 SNI 的請求。
- 您僅能把一個 SSL/TLS 憑證與每個 &CF; 分佈相關聯。
- 每個分佈必須包含一或多個也出現在憑證中 Common Name (通用名稱) 欄位或 Subject Alternative Names (主體別名) 欄位的備用網域名稱。
- 如果您使用專用 IP 位址提供 HTTPS 請求，且透過使用相同的 AWS 帳戶建立所有分佈，則可以藉由使用適用於所有分佈的相同憑證大幅降低您的成本。CloudFront 收取每個憑證的費用，而不是每個分佈。

例如，假設您使用相同的 AWS 帳戶建立三個分佈，並且針對所有三個分佈使用相同的憑證。您只需支付使用專用 IP 位址的費用一次。

不過，如果您使用專用 IP 位址提供 HTTPS 請求，且在不同的 AWS 帳戶中使用相同的憑證建立 CloudFront 分佈，則需支付每個帳戶使用專用 IP 位址的費用。例如，如果您使用三個不同的 AWS 帳戶來建立三個分佈，並且針對所有三個分佈使用相同的憑證，則每個帳戶都會支付使用專用 IP 地址的完整費用。

設定備用網域名稱和 HTTPS

若要使用 URL 中適用於檔案的備用網域名稱，與在檢視器和 CloudFront 之間使用 HTTPS，請執行適用的程序。

主題

- [取得 SSL/TLS 憑證](#)
- [匯入 SSL/TLS 憑證](#)
- [更新您的 CloudFront 分佈](#)

取得 SSL/TLS 憑證

如果您尚未擁有憑證，請取得一個 SSL/TLS 憑證。如需詳細資訊，請參閱適用的文件：

- 若要使用 AWS Certificate Manager (ACM) 提供的憑證，請參閱 [AWS Certificate Manager 使用者指南](#)。然後跳至 [更新您的 CloudFront 分佈](#)。

Note

我們建議您使用 ACM 在 AWS 受管資源上佈建、管理和部署 SSL/TLS 憑證。您必須在美國東部 (維吉尼亞北部) 區域請求 ACM 憑證。

- 若要從第三方憑證授權單位 (CA) 取得憑證，請參閱憑證授權單位提供的文件。當您有憑證時，請繼續進行下一個程序。

匯入 SSL/TLS 憑證

如果您從第三方 CA 取得憑證，請將憑證匯入到 ACM 或上傳至 IAM 憑證存放區：

ACM (推薦)

ACM 讓您從 ACM 主控台以及透過程式設計方式匯入第三方憑證。如需將憑證匯入 ACM 的詳細資訊，請參閱 AWS Certificate Manager 使用者指南中的[將憑證匯入 AWS Certificate Manager](#)。您必須在美國東部 (維吉尼亞北部) 區域匯入憑證。

IAM 憑證存放區

(不建議) 使用以下 AWS CLI 命令將您的第三方憑證上傳至 IAM 憑證存放區。

```
aws iam upload-server-certificate \  
    --server-certificate-name CertificateName \  
    --certificate-body file://public_key_certificate_file \  
    --private-key file://privatekey.pem \  
    --certificate-chain file://certificate_chain_file \  
    --path /cloudfront/path/
```

注意下列事項：

- AWS 帳戶 – 您必須使用用來建立 CloudFront 分佈的相同 AWS 帳戶，將憑證上傳至 IAM 憑證存放區。
- path 參數 – 當您將憑證上傳到 IAM 時，--path 參數 (憑證路徑) 的值必須以 /cloudfront/ 開頭，例如 /cloudfront/production/ 或 /cloudfront/test/。路徑必須以 / 結尾。
- 現有的憑證 – 您必須指定 --server-certificate-name 和 --path 參數的值 (不同於與現有憑證相關聯的值)。
- 使用 CloudFront 主控台 – 您在 中為 --server-certificate-name 參數指定的值 AWS CLI，例如 myServerCertificate，會出現在 CloudFront 主控台的 SSL 憑證清單中。

- 使用 CloudFront API – 請記下 AWS CLI 傳回的英數字元字串，例如 AS1A2M3P4L5E67SIIXR3J。這是您會在 IAMCertificateId 元素中指定的值。您不需要也由 CLI 傳回的 IAM ARN。

如需的詳細資訊 AWS CLI，請參閱 [AWS Command Line Interface 使用者指南](#) 和 [AWS CLI 命令參考](#)。

更新您的 CloudFront 分佈

若要更新分佈的設定，請執行以下程序：

設定適用於備用網域名稱的 CloudFront 分佈

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇您希望更新的分佈 ID。
3. 在 General (一般) 索引標籤上，選擇 Edit (編輯)。
4. 更新下列的值：

備用網域名稱 (CNAME)

選擇新增項目，以新增適用的備用網域名稱。使用逗號區隔網域名稱，或在新的一行輸入每個網域名稱。

自訂 SSL 憑證

從下拉式清單中選取憑證。

此處列出高達 100 個憑證。如果您有超過 100 個憑證，而您沒有看到想要新增的憑證，可以在欄位中輸入憑證 ARN 來選取。

如果您將憑證上傳至 IAM 憑證存放區但它沒有被列出，而您無法在欄位中輸入名稱來選取，請檢閱此程序 [匯入 SSL/TLS 憑證](#) 以確認您是否正確上傳憑證。

Important

在您把 SSL/TLS 憑證與 CloudFront 分佈相關聯後，請不要從 ACM 或 IAM 憑證存放區刪除憑證，直到您從所有分佈移除憑證且部署所有分佈。

5. 選擇儲存變更。

6. 將 CloudFront 設定為在檢視器和 CloudFront 之間請求使用 HTTPS：

- a. 在 Behaviors (行為) 索引標籤中，選擇您想要更新的快取行為，然後選擇 Edit (編輯)。
- b. 請針對 Viewer Protocol Policy (檢視器通訊協定政策) 指定下列其中一個值：

重新引導 HTTP 到 HTTPS

檢視器可以使用這兩種通訊協定，但會自動重新引導 HTTP 請求到 HTTPS 請求。CloudFront 會傳回 HTTP 狀態碼 301 (Moved Permanently) 以及新的 HTTPS URL。然後檢視器使用 HTTPS URL 重新將請求提交到 CloudFront。

Important

CloudFront 不會從 HTTP 將 DELETE、OPTIONS、PATCH、POST 或 PUT 請求重新引導到 HTTPS。如果將快取行為設定為重新引導至 HTTPS，則 CloudFront 會以 HTTP 狀態碼 DELETE，來回應此快取行為的 HTTP OPTIONS、PATCH、POST、PUT 或 403 (Forbidden) 請求。

當檢視器執行一個 HTTP 請求供其重新引導到 HTTPS 請求時，CloudFront 會收取這兩個請求的費用。針對 HTTP 請求，只會收取 CloudFront 傳回到檢視器的請求與標頭之費用。針對 HTTPS 請求，會收取該請求與標頭及原始伺服器傳回的檔案之費用。

僅限 HTTPS

檢視器只能在使用 HTTPS 的情況下存取您的內容。如果檢視器傳送 HTTP 請求，而不是 HTTPS 請求，則 CloudFront 會傳回 HTTP 狀態碼 403 (Forbidden)，且不會傳回檔案。

- c. 請選擇 Yes, Edit (是，編輯)。
 - d. 為每個額外快取行為，供您要在檢視器與 CloudFront 之間使用 HTTPS，請重複執行步驟 a 到 c。
- ## 7. 您使用生產環境中已更新的組態之前，請先確認以下項目：
- 每個快取行為中的路徑模式僅適用於您想要檢視器使用 HTTPS 的請求。
 - 以您想要 CloudFront 評估的順序列出快取行為。如需詳細資訊，請參閱 [路徑模式](#)。
 - 快取行為會將請求路由到正確的原始伺服器。

判斷 SSL/TLS RSA 憑證中公有金鑰的大小

如果使用 CloudFront 備用網域名稱和 HTTPS，SSL/TLS RSA 憑證中的公有金鑰大小上限為 4096 位元。(這是金鑰大小，而不是公有金鑰中字元的數量)。如果您使用 AWS Certificate Manager 做為憑證，雖然 ACM 支援較大的 RSA 金鑰，但無法搭配 CloudFront 使用較大的金鑰。

您可以透過執行以下 OpenSSL 命令判斷 RSA 公有金鑰的大小：

```
openssl x509 -in path and filename of SSL/TLS certificate -text -noout
```

其中：

- `-in` 會指定 SSL/TLS RSA 憑證的路徑和檔案名稱。
- `-text` 會讓 OpenSSL 以位元為單位顯示 RSA 公有金鑰的長度。
- `-noout` 避免 OpenSSL 顯示公有金鑰。

輸出範例：

```
Public-Key: (2048 bit)
```

增加 SSL/TLS 憑證的配額

您可以匯入 AWS Certificate Manager (ACM) 或上傳到 AWS Identity and Access Management (IAM) 的 SSL/TLS 憑證數量有配額。當您將 CloudFront 設定為使用專用 IP 地址來提供 HTTPS 請求 AWS 帳戶時，您可以搭配使用的 SSL/TLS 憑證數量也有配額。不過，您可以請求提高配額。

主題

- [增加匯入 ACM 的憑證配額](#)
- [增加上傳到 IAM 的憑證配額](#)
- [增加與專用 IP 位址搭配使用的憑證配額](#)

增加匯入 ACM 的憑證配額

如需可匯入 ACM 之憑證數量的配額，請參閱 AWS Certificate Manager 使用者指南中的[配額](#)。

若要請求提升配額，您可以使用 Service Quotas 主控台。如需詳細資訊，請參閱「Service Quotas 使用者指南」中的[請求提高配額](#)。

增加上傳到 IAM 的憑證配額

如需您可以上傳到 IAM 的憑證數量配額 (先前稱為限制)，請參閱 IAM 使用者指南中的 [IAM 和 STS 限制](#)。

若要請求提升配額，您可以使用 Service Quotas 主控台。如需詳細資訊，請參閱「Service Quotas 使用者指南」中的[請求提高配額](#)。

增加與專用 IP 位址搭配使用的憑證配額

如需使用專用 IP 地址提供 HTTPS 請求 AWS 帳戶時，每個憑證可使用的 SSL 憑證數量配額，請參閱 [SSL 憑證的配額](#)。

若要請求提升配額，您可以使用 Service Quotas 主控台。如需詳細資訊，請參閱「Service Quotas 使用者指南」中的[請求提高配額](#)。

輪換 SSL/TLS 憑證

當您的 SSL/TLS 憑證即將過期時，您需要輪換憑證以確保分佈的安全性，並避免檢視器的服務中斷。您可用下列方式輪換憑證：

- 對於 AWS Certificate Manager (ACM) 提供的 SSL/TLS 憑證，您不需要輪換它們。ACM 會自動為您管理憑證續約。如需詳細資訊，請參閱《AWS Certificate Manager 使用者指南》中的「[受管憑證續約](#)」。
- 如果您使用第三方憑證認證機構，而且已將憑證匯入 ACM (建議) 或上傳到 IAM 憑證存放區，則必須偶爾使用另一個憑證來替換某個憑證。

Important

- ACM 不會為您從第三方憑證認證機構取得並匯入到 ACM 的憑證管理憑證續約。
- 如果您設定 CloudFront 使用專用 IP 位址提供 HTTPS 請求，在更換憑證時，則可能需要支付額外按比例計算的費用來使用一或多個額外的憑證。我們建議您更新分佈，把額外的費用降到最低。

輪換 SSL/TLS 憑證

若要輪換憑證，請執行以下程序。在您更換憑證的同時以及處理完成後，檢視器皆能持續存取您的內容。

若要輪換 SSL/TLS 憑證

1. [增加 SSL/TLS 憑證的配額](#) 判斷您是否需要使用更多 SSL 憑證的許可。若是如此，請請求許可並等到授予許可時，再繼續步驟 2。
2. 將新憑證匯入到 ACM 或上傳至 IAM。如需詳細資訊，請參閱 Amazon CloudFront 開發人員指南中的[匯入 SSL/TLS 憑證](#)。
3. (僅限 IAM 憑證) 一次更新一個分佈，以使用新的憑證。如需詳細資訊，請參閱[更新分佈](#)。
4. (選用) 從 ACM 或 IAM 刪除先前的憑證。

Important

請勿刪除 SSL/TLS 憑證，直到您將其從所有分佈中移除，並直到您已更新的分佈已變更為 Deployed。

從自訂 SSL/TLS 憑證還原到預設 CloudFront 憑證

如果您已將 CloudFront 設定為在檢視器和 CloudFront 之間使用 HTTPS，且已將 CloudFront 設定為使用自訂 SSL/TLS 憑證，則可以變更設定以使用預設 CloudFront SSL/TLS 憑證。此程序取決於您是否已使用分佈來分配您的內容：

- 如果您尚未使用分佈來分配內容，可以只變更組態。如需詳細資訊，請參閱[更新分佈](#)。
- 如果您已經使用分佈來分配內容，則您必須建立新的 CloudFront 分配和變更檔案的 URL，以降低或消除內容不可用的時間。若要執行此操作，請執行以下程序。

還原為預設 CloudFront 憑證

以下程序顯示如何從自訂 SSL/TLS 憑證還為到預設 CloudFront 憑證。

還原為預設 CloudFront 憑證

1. 使用所需的組態建立新的 CloudFront 分佈。針對 SSL 憑證，選擇 預設 CloudFront 憑證 (*.cloudfront.net)。

如需詳細資訊，請參閱[建立分發](#)。

2. 針對您使用 CloudFront 所分佈的檔案，請更新應用程式中的 URL，以使用 CloudFront 指派給新分佈的網域名稱。例如，請將 `https://www.example.com/images/logo.png` 變更為 `https://d1111111abcdef8.cloudfront.net/images/logo.png`。
3. 刪除與自訂 SSL/TLS 憑證相關聯的分佈，或是更新分佈，將 SSL 憑證的值變更為預設 CloudFront 憑證 (*.cloudfront.net)。如需詳細資訊，請參閱[更新分佈](#)。

 Important

在您完成此步驟之前，AWS 會繼續向您收取使用自訂 SSL/TLS 憑證的費用。

4. (選用) 刪除您的自訂 SSL/TLS 憑證。
 - a. 執行 AWS CLI 命令 `list-server-certificates` 以取得您要刪除之憑證的憑證 ID。如需詳細資訊，請參閱 AWS CLI 命令參考中的 [list-server-certificates](#)。
 - b. 執行 AWS CLI 命令 `delete-server-certificate` 來刪除憑證。如需詳細資訊，請參閱 AWS CLI 命令參考中的 [delete-server-certificate](#)。

若要從具有專用 IP 位址的自訂 SSL/TLS 憑證切換到 SNI

如果您設定 CloudFront 使用有專用 IP 位址的自訂 SSL/TLS 憑證，您可以切換為使用有 SNI 的自訂 SSL/TLS 憑證，並消除與專用 IP 位址關聯的費用。

 Important

對 CloudFront 組態的此更新對支援 SNI 的檢視器沒有任何影響。檢視器可以在變更之前和之後存取您的內容，也可以在變更傳播到 CloudFront 節點位置時存取內容。不支援 SNI 的檢視器在變更之後不能存取您的內容。如需詳細資訊，請參閱[選擇 CloudFront 提供 HTTPS 請求的方式](#)。

從自訂憑證切換至 SNI

下列程序顯示如何從具有專用 IP 位址的自訂 SSL/TLS 憑證切換到 SNI。

若要從具有專用 IP 位址的自訂 SSL/TLS 憑證切換到 SNI

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇欲檢閱或更新的分佈 ID。
3. 請選擇 Distribution Settings (分佈設定)。
4. 在 General (一般) 索引標籤上，選擇 Edit (編輯)。
5. 在自訂 SSL 憑證 – 選用下，取消選取舊版用戶端支援。
6. 請選擇 Yes, Edit (是，編輯)。

相互 TLS (mTLS) 檢視器

相互 TLS 身分驗證 (相互傳輸層安全身分驗證 — mTLS) 是一種安全通訊協定，透過要求雙向憑證型身分驗證來延伸標準 TLS 身分驗證，其中用戶端和伺服器都必須證明其身分，才能建立安全連線。使用交互 TLS，您可以確保只有提供信任 TLS 憑證的用戶端才能存取您的 CloudFront 分佈。

運作方式

在標準 TLS 交握中，只有伺服器會呈現憑證，向用戶端證明其身分。使用交互 TLS，身分驗證程序會變成雙向。當用戶端嘗試連線到 CloudFront 分佈時，CloudFront 會在 TLS 交握期間請求用戶端憑證。在建立安全連線之前，用戶端必須呈現 CloudFront 針對您設定的信任存放區驗證的有效 X.509 憑證。

CloudFront 會在 AWS 節點執行此憑證驗證，從原始伺服器卸載身分驗證複雜性，同時維護 CloudFront 的全域效能優勢。您可以將 mTLS 設定為兩種模式：驗證模式 (需要所有用戶端呈現有效的憑證) 或選用模式 (在呈現時驗證憑證，但也允許沒有憑證的連線)。

使用案例

使用 CloudFront 的相互 TLS 身分驗證解決了傳統身分驗證方法不足的幾個關鍵安全案例：

- 裝置身分驗證與內容快取 - 您可以在允許存取韌體更新、遊戲下載或內部資源之前，對遊戲主控台、IoT 裝置或公司硬體進行身分驗證。每個裝置都包含唯一的憑證，可證明其真實性，同時受益於 CloudFront 的快取功能。
- API-to-API 身分驗證 - 您可以保護信任的業務合作夥伴、付款系統或微型服務之間的 machine-to-machine 通訊。憑證型身分驗證不需要共用秘密或 API 金鑰，同時為自動化資料交換提供強大的身分驗證。

主題

- [信任存放區和憑證管理](#)
- [為 CloudFront 分佈啟用交互 TLS](#)
- [關聯 CloudFront 連線函數](#)
- [設定其他設定](#)
- [快取政策並轉送至原始伺服器的檢視器 mTLS 標頭](#)
- [使用 CloudFront Connection Function 和 KVS 撤銷](#)
- [使用連線日誌的可觀測性](#)

信任存放區和憑證管理

建立和設定信任存放區是使用 CloudFront 實作交互 TLS 身分驗證的必要要求。信任存放區包含 CloudFront 在身分驗證程序期間用來驗證用戶端憑證的憑證授權機構 (CA) 憑證。

什麼是信任存放區？

信任存放區是 CA 憑證的儲存庫，CloudFront 會使用此儲存庫在交互 TLS 身分驗證期間驗證用戶端憑證。信任存放區包含根和中繼 CA 憑證，構成用於驗證用戶端憑證的信任鏈。

當您使用 CloudFront 實作交互 TLS 時，信任存放區會定義您信任哪些憑證授權機構來發行有效的用戶端憑證。CloudFront 會在 TLS 交握期間，針對您的信任存放區驗證每個用戶端憑證。只有呈現連結至信任存放區中其中一個 CAs 的憑證的用戶端才會成功驗證。

CloudFront 中的信任存放區是可與多個分佈建立關聯的帳戶層級資源。這可讓您在整個 CloudFront 部署中維持一致的憑證驗證政策，同時簡化 CA 憑證管理。

憑證授權單位支援

CloudFront 支援 AWS 私有憑證授權機構和第三方私有憑證授權機構發行的憑證。此彈性可讓您根據您的組織需求，使用現有的憑證基礎設施或利用 AWS 受管憑證服務。

- **AWS 私有憑證授權機構：**您可以使用 AWS Private CA 發行的憑證，該憑證提供受管私有憑證授權機構服務。此整合可簡化憑證生命週期管理，並提供與其他 AWS 服務的無縫整合。
- **第三方私有憑證授權機構：**您也可以使用現有私有憑證授權機構基礎設施的憑證，包括企業 CAs 或其他第三方憑證提供者。這可讓您維護目前的憑證管理程序，同時新增 CloudFront 的 mTLS 功能。

憑證需求和規格

信任存放區對其包含的 CA 憑證有特定要求：

CA 憑證格式要求

- 格式：PEM（隱私權增強郵件）格式
- 內容界限：憑證必須括在 -----BEGIN CERTIFICATE----- 和 -----END CERTIFICATE----- 邊界內
- 註解：必須以 # 字元開頭，且不能包含任何 - 字元
- 換行：憑證之間不允許空白行

支援的憑證規格

- 憑證類型：X.509v3
- 公有金鑰類型：
 - RSA 2048、RSA 4096
 - ECDSA：secp256r1
- 簽章演算法：
 - SHA256, SHA384, SHA512 搭配 RSA
 - SHA256, SHA384, SHA512 搭配 EC
 - SHA256, SHA384, SHA512 搭配 RSASSA-PSS 搭配 MGF1

憑證套件格式範例

多個憑證 (PEM 編碼)：

```
# Root CA Certificate
-----BEGIN CERTIFICATE-----
MIIDXTCCAkWgAwIBAgIJAKoK/0vD/XqiMA0GCSqGSIb3DQEBCwUAMEUxCzAJBgNV
BAYTAKFVMRMwEQYDVQQIDApTb211LVN0YXR1MSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTCwNzEyMTU0NzQ4WhcNMjcwNzEwMTU0NzQ4WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAAuuExKvY1xzHfYlsHiuowqpmzs7rEcuuy10uEszpFp+BtXh0ZuEtts9LP
-----END CERTIFICATE-----
# Intermediate CA Certificate
-----BEGIN CERTIFICATE-----
```

```

MIIDXTCCAkWgAwIBAgIJAKoK/OvD/XqjMA0GCSqGSIb3DQEBCwUAMEUxCzAJBgNV
BAYTAKFVMRMwEQYDVQQIDApTb211LVN0YXR1MSEwHwYDVQQKBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTcwNzEyMTU0NzQ4WhcNMjcwNzEwMTU0NzQ4WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAAuExKvY1xzHFylsHiuowqpmzs7rEcuuy10uEszpFp+BtXh0ZuEtts9LP
-----END CERTIFICATE-----

```

建立信任存放區

建立信任存放區之前，您必須將 PEM 格式的 CA 憑證套件上傳至 Amazon S3 儲存貯體。憑證套件應包含驗證用戶端憑證所需的所有信任根憑證和中繼 CA 憑證。

建立信任存放區時，CA 憑證套件只會從 S3 讀取一次。如果未來變更 CA 憑證套件，則必須手動更新信任存放區。信任存放區和 S3 CA 憑證套件之間不會維護同步。

先決條件

- 從您的憑證授權機構 (CA) 上傳到 Amazon S3 儲存貯體的憑證套件
- 建立 CloudFront 資源的必要許可

建立信任存放區（主控台）

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇信任存放區。
3. 選擇建立信任存放區。
4. 針對信任存放區名稱，輸入信任存放區的名稱。
5. 針對憑證授權機構 (CA) 套件，輸入 PEM 格式 CA 憑證套件的 Amazon S3 路徑。
6. 選擇建立信任存放區。

建立信任存放區 (AWS CLI)

```

aws cloudfront create-trust-store \
  --name MyTrustStore \
  --certificate-authority-bundle-s3-location Bucket=my-bucket,Key=ca-bundle.pem \
  --tags Items=[{Key=Environment,Value=Production}]

```

將信任存放區與分佈建立關聯

建立信任存放區之後，您必須將其與 CloudFront 分佈建立關聯，才能啟用交互 TLS 身分驗證。

先決條件

- 已啟用僅 HTTPS 檢視器通訊協定政策且 HTTP3 支援已停用的現有 CloudFront 分佈。

建立信任存放區的關聯（主控台）

CloudFront 主控台信任存放區有兩種關聯方式：透過信任存放區詳細資訊頁面或透過分佈設定頁面。

透過信任存放區詳細資訊頁面關聯信任存放區：

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇信任存放區。
3. 選擇您要關聯的信任存放區名稱。
4. 選擇關聯以分佈。
5. 設定可用的檢視器 mTLS 選項：
 - 用戶端憑證驗證模式：選擇必要和選用模式。在必要模式下，所有用戶端都需要提供憑證。在選用模式中，會驗證呈現憑證的用戶端，而未呈現憑證的用戶端則允許存取。
 - 公告信任存放區 CA 名稱：選擇是否要在 TLS 交握期間將信任存放區中的 CA 名稱公告給用戶端。
 - 忽略憑證過期日期：選擇是否允許憑證過期的連線（其他驗證條件仍適用）。
 - 連線函數：選用的連線函數可以相關聯，以根據其他自訂條件允許/拒絕連線。
6. 選取要與信任存放區建立關聯的一或多個分佈。只有停用 HTTP3 且具有僅限 HTTPS 快取行為的分佈，才能支援檢視器 mTLS。
7. 選擇關聯。

透過分佈設定頁面關聯信任存放區：

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。

2. 選取您要關聯的分佈
3. 在一般索引標籤下方的設定容器中，選擇右上角的編輯
4. 向下捲動至頁面底部，在連線容器中切換檢視器 mTLS 開啟
5. 設定可用的檢視器 mTLS 選項：
 - 用戶端憑證驗證模式：選擇必要和選用模式。在必要模式下，所有用戶端都需要提供憑證。在選用模式中，會驗證呈現憑證的用戶端，而未呈現憑證的用戶端則允許存取。
 - 公告信任存放區 CA 名稱：選擇是否要在 TLS 交握期間將信任存放區中的 CA 名稱公告給用戶端。
 - 忽略憑證過期日期：選擇是否允許憑證過期的連線（其他驗證條件仍適用）。
 - 連線函數：選用的連線函數可以建立關聯，以根據其他自訂條件允許/拒絕連線。
6. 選擇右下角的儲存變更。

建立信任存放區 (AWS CLI) 的關聯

信任存放區可以透過 `DistributionConfig.ViewerMtlsConfig` 屬性與分佈建立關聯。這表示我們首先需要擷取分佈組態，然後在後續的 `UpdateDistribution` 請求中提供 `ViewerMtlsConfig`。

```
// First fetch the distribution
aws cloudfront get-distribution {DISTRIBUTION_ID}

// Update the distribution config, for example:
Distribution config, file://distConf.json:
{
  ...other fields,
  ViewerMtlsConfig: {
    Mode: 'required',
    TrustStoreConfig: {
      AdvertiseTrustStoreCaNames: false,
      IgnoreCertificateExpiry: true,
      TrustStoreId: {TRUST_STORE_ID}
    }
  }
}

aws cloudfront update-distribution \
  --id {DISTRIBUTION_ID} \
  --if-match {ETAG} \
  --distribution-config file://distConf.json
```


管理信任存放區

檢視信任存放區詳細資訊

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇信任存放區。
3. 選擇信任存放區的名稱以檢視其詳細資訊頁面。

詳細資訊頁面顯示：

- 信任存放區名稱和 ID
- CA 憑證數量
- 建立日期和上次修改日期
- 關聯的分佈
- Tags (標籤)

修改信任存放區

若要取代 CA 憑證套件：

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇信任存放區。
3. 選擇信任存放區的名稱。
4. 選擇動作，然後選擇編輯。
5. 針對憑證授權單位 (CA) 套件，輸入更新後的 CA 套件 PEM 檔案的 Amazon S3 位置。
6. 選擇更新信任存放區。

刪除信任存放區

先決條件：您必須先取消信任存放區與所有 CloudFront 分佈的關聯。

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。

2. 在導覽窗格中，選擇信任存放區。
3. 選擇信任存放區的名稱。
4. 選擇刪除信任存放區。
5. 選擇刪除以確認刪除。

後續步驟

在建立信任存放區並將其與 CloudFront 分發建立關聯之後，您可以繼續在分發上啟用交互 TLS 身分驗證，並設定其他設定，例如將憑證標頭轉送至原始伺服器。如需在分佈上啟用 mTLS 的詳細說明，請參閱 [為 CloudFront 分佈啟用交互 TLS](#)。

為 CloudFront 分佈啟用交互 TLS

先決條件和要求

CloudFront 的交互 TLS 驗證模式要求所有用戶端在 TLS 交握期間呈現有效的憑證，並拒絕沒有有效憑證的連線。在 CloudFront 分佈上啟用交互 TLS 之前，請確定您有：

- 使用憑證授權單位憑證建立信任存放區
- 將信任存放區與您的 CloudFront 分佈建立關聯
- 確保所有分佈快取行為都使用僅限 HTTPS 的檢視器通訊協定政策
- 確保您的分佈使用 HTTP/2（預設設定，HTTP/3 不支援檢視器 mTLS）

Note

相互 TLS 身分驗證需要檢視器和 CloudFront 之間的 HTTPS 連線。您無法在具有支援 HTTP 連線之任何快取行為的分佈上啟用 mTLS。

啟用交互 TLS（主控台）

對於新分佈

在 CloudFront 主控台中建立新分佈的過程中，無法設定檢視器 mTLS。首先透過任何方式（主控台、CLI、API）建立分佈，然後編輯分佈設定，以根據下列現有的分佈說明啟用檢視器 mTLS。

對於現有的分佈

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 從分佈清單中，選取您要修改的分佈。
3. 確定針對所有快取行為，檢視器通訊協定政策設定為將 HTTP 重新導向至 HTTPS 或僅限 HTTPS。（您可以選擇快取行為索引標籤，以使用 HTTP 通訊協定政策檢視和更新任何快取行為。）
4. 選擇一般索引標籤。
5. 在 Settings (設定) 區段中，選擇 Edit (編輯)。
6. 在連線區段中，尋找檢視器交互身分驗證 (mTLS)。
7. 將啟用交互身分驗證切換為開啟。
8. 對於用戶端憑證驗證模式，選取必要（所有用戶端都必須顯示憑證）或選用（用戶端可以選擇性顯示憑證）。
9. 針對信任存放區，選取您先前建立的信任存放區。
10. （選用）如果您希望 CloudFront 在 TLS 交握期間將 CA 名稱傳送給用戶端，請切換公告信任存放區 CA 名稱。
11. （選用）如果您想要允許憑證過期的連線，請切換忽略憑證過期日期。
12. 選擇儲存變更。

啟用交互 TLS (AWS CLI)

對於新分佈

下列範例示範如何建立包含 mTLS 設定的分佈組態檔案 (distribution-config.json)：

```
{
  "CallerReference": "cli-example-1",
  "Origins": {
    "Quantity": 1,
    "Items": [
      {
        "Id": "my-origin",
        "DomainName": "example.com",
        "CustomOriginConfig": {
          "HTTPPort": 80,
```

```

        "HTTPSPort": 443,
        "OriginProtocolPolicy": "https-only"
    }
}
],
},
"DefaultCacheBehavior": {
    "TargetOriginId": "my-origin",
    "ViewerProtocolPolicy": "https-only",
    "MinTTL": 0,
    "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
            "Forward": "none"
        }
    }
},
"ViewerCertificate": {
    "CloudFrontDefaultCertificate": true
},
"ViewerMtlsConfig": {
    "Mode": "required",
    "TrustStoreConfig": {
        "TrustStoreId": {TRUST_STORE_ID},
        "AdvertiseTrustStoreCaNames": true,
        "IgnoreCertificateExpiry": true
    }
},
"Enabled": true
}

```

使用下列範例命令建立已啟用 mTLS 的分佈：

```
aws cloudfront create-distribution --distribution-config file://distribution-config.json
```

對於現有的分佈

使用下列範例命令取得目前的分佈組態：

```
aws cloudfront get-distribution-config --id E1A2B3C4D5E6F7 --output json > dist-config.json
```

編輯 檔案以新增 mTLS 設定。將下列範例區段新增至您的分佈組態：

```
"ViewerMtlsConfig": {
  "Mode": "required",
  "TrustStoreConfig": {
    "TrustStoreId": {TRUST_STORE_ID},
    "AdvertiseTrustStoreCaNames": true,
    "IgnoreCertificateExpiry": true
  }
}
```

從檔案移除 ETag 欄位，但分別儲存其值。

使用下列範例命令，使用新組態更新分佈：

```
aws cloudfront update-distribution \
  --id E1A2B3C4D5E6F7 \
  --if-match YOUR-ETAG-VALUE \
  --distribution-config file://dist-config.json
```

檢視器通訊協定政策

使用交互 TLS 時，所有分佈快取行為都必須使用僅限 HTTPS 的檢視器通訊協定政策進行設定：

- 將 HTTP 重新導向至 HTTPS - 在執行憑證驗證之前將 HTTP 請求重新導向至 HTTPS。
- 僅限 HTTPS - 僅接受 HTTPS 請求並執行憑證驗證。

Note

交互 TLS 不支援 HTTP 和 HTTPS 檢視器通訊協定政策，因為 HTTP 連線無法執行憑證驗證。

後續步驟

在 CloudFront 分佈上啟用檢視器 TLS 之後，您可以關聯連線函數來實作自訂憑證驗證邏輯。Connection Functions 可讓您使用自訂驗證規則、憑證撤銷檢查和記錄來擴展內建 mTLS 身分驗證功能。如需建立和關聯連線函數的詳細資訊，請參閱 [關聯 CloudFront 連線函數](#)。

關聯 CloudFront 連線函數

CloudFront Connection Functions 可讓您在 TLS 交握期間實作自訂憑證驗證邏輯，提供內建 mTLS 身分驗證功能的延伸。

什麼是連線函數？

連線函數是在用戶端憑證經過驗證後，在 TLS 交握期間執行的 JavaScript 函數。已驗證的用戶端憑證會傳遞至 Connection Function，此時 Connection Function 可以對是否授予存取權進行額外判斷。如需連線函數的詳細資訊，請參閱 [使用 CloudFront Functions 在邊緣進行自訂](#)。

Connection Functions 如何使用 mTLS

當用戶端嘗試建立與 CloudFront 分佈的 mTLS 連線時，會發生下列順序：

1. 用戶端使用 CloudFront 節點啟動 TLS 交握。
2. CloudFront 請求和接收用戶端憑證。
3. CloudFront 會對信任存放區執行標準憑證驗證。
4. 如果憑證通過標準驗證，CloudFront 會叫用您的連線函數。如果在 ViewerMtlsConfig 中啟用了 IgnoreCertificateExpiry，則您的過期但有效憑證也會傳遞至連線函數。如果用戶端憑證無效，將不會叫用連線函數。
5. 您的 Connection Function 會收到剖析的憑證資訊和連線詳細資訊。
6. 您的函數會根據自訂邏輯做出允許/拒絕決策。
7. CloudFront 會根據您的決定完成或終止 TLS 連線。

驗證模式和選用模式（當用戶端提供憑證時）都會叫用連線函數。

請求提高連線函數配額

請求提高的連線函數配額 AWS 帳戶。

請求增加連線函數配額

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽視窗中，選擇函數。
3. 選擇連線函數索引標籤

4. 針對請求，請選擇連結以聯絡 CloudFront 支援工程。
5. CloudFront 支援工程審查您的請求。審核程序最多可能需要兩天的時間。

請求獲得核准後，您可以在帳戶中建立連線函數，並在使用交互 TLS 時將其與一或多個分佈建立關聯。

建立連線函數

您可以使用 CloudFront 主控台或 CLI AWS 建立連線函數。

建立連線函數（主控台）

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽視窗中，選擇函數。
3. 選擇連線函數索引標籤，然後選擇建立連線函數。
4. 輸入您 AWS 帳戶中唯一的函數名稱。
5. 選擇繼續。
6. 在函數編輯器中，撰寫要驗證憑證的 JavaScript 程式碼。函數處理常式必須呼叫允許或拒絕。
7. 選用：KeyValue 存放區可以與連線函數建立關聯，以實作撤銷控制。
8. 選擇儲存變更。

建立連線函數 (AWS CLI)

下列範例示範如何建立連線函數：

將函數程式碼寫入不同的檔案中，例如 code.js：

```
function connectionHandler(connection) {  
    connection.allow();  
}
```

```
aws cloudfront create-connection-function \  
  --name "certificate-validator" \  
  --connection-function-config '{  
    "Comment": "Client certificate validation function",  
    "Runtime": "cloudfront-js-2.0"
```

```
}' \  
--connection-function-code fileb://code.js
```

Connection Function 程式碼結構

Connection Functions 實作 `connectionHandler` 函數，該函數會接收包含憑證和連線資訊的連線物件。您的函數必須使用 `connection.allow()` 或 `connection.deny()` 來決定連線。

基本連線函數範例

下列範例顯示簡單的連線函數，可驗證用戶端憑證的主旨欄位：

```
function connectionHandler(connection) {  
  // Only process if a certificate was presented  
  if (!connection.clientCertificate) {  
    console.log("No certificate presented");  
    connection.deny();  
  }  
  
  // Check the subject field for specific organization  
  const subject = connection.clientCertificate.certificates.leaf.subject;  
  if (!subject.includes("O=ExampleCorp")) {  
    console.log("Certificate not from authorized organization");  
    connection.deny();  
  } else {  
    // All checks passed  
    console.log("Certificate validation passed");  
    connection.allow();  
  }  
}
```

連線物件上可用的用戶端憑證屬性完整規格可在此處取得：

```
{  
  "connectionId": "Fdb-Eb7L9gVn2cFakz7wWyBJIDAD4-oN06g8r3vXDV132BtnIVtqDA==", // Unique  
  identifier for this TLS connection  
  "clientId": "203.0.113.42", // IP address of the connecting client (IPv4 or IPv6)  
  "clientCertificate": {  
    "certificates": {  
      "leaf": {  
        "subject": "CN=client.example.com,O=Example Corp,C=US", // Distinguished Name  
        (DN) of the certificate holder
```



```
    "issuer": "CN=Example Corp Intermediate CA,O=Example Corp,C=US", //
Distinguished Name (DN) of the certificate authority that issued this certificate
    "serialNumber": "4a:3f:5c:92:d1:e8:7b:6c", // Unique serial number assigned by
the issuing CA (hexadecimal)
    "validity": {
        "notBefore": "2024-01-15T00:00:00Z", // Certificate validity start date (ISO
8601 format)
        "notAfter": "2025-01-14T23:59:59Z"    // Certificate expiration date (ISO 8601
format)
    },
    "sha256Fingerprint": "a1b2c3d4e5f6...abc123def456", // SHA-256 hash of the
certificate (64 hex characters)
  },
},
},
}
```

關聯連線函數

建立連線函數之後，您必須將其發佈至 LIVE 階段，並將其與您的分佈建立關聯。

發佈和關聯連線函數（主控台）

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇函數
3. 選擇連線函數索引標籤，然後選取您的連線函數。
4. 選擇發佈以將其移至 LIVE 階段。
5. 在發佈區段下方的關聯分佈表格中選擇新增關聯。
6. 選取已啟用檢視器 mTLS 的分佈，以便與之建立關聯。

或者，也可以從分佈詳細資訊頁面關聯已發佈的連線函數。

1. 導覽至列出所有分發的主控台首頁。
2. 選取您要關聯的分佈。
3. 選擇一般索引標籤。
4. 在 Settings (設定) 區段中，選擇 Edit (編輯)。
5. 在連線區段中，尋找檢視器交互身分驗證 (mTLS)。

6. 針對連線函數，選取您的函數。
7. 選擇儲存變更。

建立連線函數 (AWS CLI) 的關聯

下列範例示範如何將 Connection Function 與分佈建立關聯：

```
// DistributionConfig:
{
  ...other settings,
  "ConnectionFunctionAssociation": {
    "Id": "cf_30c2CV2e1HwCoInb3LtcaUJkZeD"
  }
}
```

Connection Functions 的使用案例

Connection Functions 啟用數個進階 mTLS 使用案例：

- 憑證屬性驗證 - 驗證用戶端憑證中的特定欄位，例如組織單位要求或主體替代名稱模式。
- 憑證撤銷檢查 - 使用 KeyValueStore 實作自訂憑證撤銷檢查，以存放已撤銷的憑證序號。
- 以 IP 為基礎的憑證政策 - 根據用戶端 IP 地址或地理限制套用不同的憑證政策。
- 多租戶驗證 - 實作租戶特定的驗證規則，其中根據主機名稱或憑證屬性套用不同的憑證要求。

Note

在 TLS 交握期間，連線函數每個用戶端連線執行一次。
Connection Functions 只能允許或拒絕連線，不能修改 HTTP 請求/回應。
只有 LIVE 階段函數（已發佈）可以與分佈建立關聯。
每個分佈最多可以有一個連線函數。

後續步驟

將連線函數與 CloudFront 分佈建立關聯後，您可以設定選用設定來自訂 mTLS 實作的行為。如需設定其他設定的詳細指示，例如選用的用戶端憑證驗證模式，請參閱 [設定其他設定](#)。

設定其他設定

啟用基本交互 TLS 身分驗證之後，您可以設定其他設定，以針對特定使用案例和需求自訂身分驗證行為。

用戶端憑證驗證選用模式

CloudFront 提供替代的選用用戶端憑證驗證模式，可驗證呈現的用戶端憑證，但允許存取不存在憑證的用戶端。

選用模式行為

- 授予具有有效憑證的用戶端連線（拒絕無效的憑證）。
- 允許在沒有憑證的情況下連線至用戶端
- 允許透過單一分佈的混合用戶端身分驗證案例。

選用模式非常適合逐步遷移至 mTLS 身分驗證、支援具有憑證的用戶端和沒有憑證的用戶端，或與舊版用戶端保持回溯相容性。

Note

在選用模式中，即使用戶端不存在憑證，連線函數仍會被叫用。這可讓您實作自訂邏輯，例如記錄用戶端 IP 地址，或根據是否顯示憑證套用不同的政策。

設定選用模式（主控台）

1. 在您的分佈設定中，導覽至一般索引標籤，選擇編輯。
2. 捲動至連線容器中的檢視器交互身分驗證 (mTLS) 區段。
3. 針對用戶端憑證驗證模式，選取選用。
4. 儲存變更。

設定選用模式 (AWS CLI)

下列範例示範如何設定選用模式：

```
"ViewerMtlsConfig": {
```

```
"Mode": "optional",
...other settings
}
```

憑證授權單位公告

AdvertiseTrustStoreCaNames 欄位控制 CloudFront 在 TLS 交握期間是否將信任的 CA 名稱清單傳送給用戶端，協助用戶端選取適當的憑證。

設定 CA 公告（主控台）

1. 在您的分佈設定中，導覽至一般索引標籤，選擇編輯。
2. 捲動至連線容器中的檢視器交互身分驗證 (mTLS) 區段。
3. 選取或取消選取公告信任存放區 CA 名稱核取方塊。
4. 選擇儲存變更。

設定 CA 公告 (AWS CLI)

下列範例示範如何啟用 CA 公告：

```
"ViewerMtlsConfig": {
  "Mode": "required", // or "optional"
  "TrustStoreConfig": {
    "AdvertiseTrustStoreCaNames": true,
    ...other settings
  }
}
```

憑證過期處理

IgnoreCertificateExpiry 屬性會決定 CloudFront 如何回應過期的用戶端憑證。根據預設，CloudFront 會拒絕過期的用戶端憑證，但您可以視需要將其設定為接受憑證。對於憑證過期且無法立即更新的裝置，通常會啟用此功能。

設定憑證過期處理（主控台）

1. 在您的分佈設定中，導覽至一般索引標籤，選擇編輯。
2. 捲動至連線容器的檢視器交互身分驗證 (mTLS) 區段。

3. 選取或取消選取忽略憑證過期日期核取方塊。
4. 選擇儲存變更。

設定憑證過期處理 (AWS CLI)

下列範例顯示如何忽略憑證過期：

```
"ViewerMtlsConfig": {  
  "Mode": "required", // or "optional"  
  "TrustStoreConfig": {  
    "IgnoreCertificateExpiry": false,  
    ...other settings  
  }  
}
```

Note

IgnoreCertificateExpiry 僅適用於憑證 有效期限。所有其他憑證驗證檢查仍然適用（信任鏈、簽章驗證）。

後續步驟

設定其他設定後，您可以設定標頭轉送，將憑證資訊傳遞至原始伺服器、使用 Connection Functions 和 KeyValueStore 實作憑證撤銷，以及啟用連線日誌以進行監控。如需轉送憑證資訊至原始伺服器的詳細資訊，請參閱[轉送標頭至原始伺服器](#)。

快取政策並轉送至原始伺服器的檢視器 mTLS 標頭

使用交互 TLS 身分驗證時，CloudFront 可以從用戶端憑證擷取資訊，並將其作為 HTTP 標頭轉送到您的原始伺服器。這可讓您的原始伺服器存取憑證詳細資訊，而無需實作憑證驗證邏輯。

下列標頭可供用來建立快取行為：

標頭名稱	Description	範例值
CloudFront-Viewer-Cert-Serial-Number	憑證序號的十六進位表示	4a : 3f : 5c : 92 : d1 : e8 : 7b : 6c

標頭名稱	Description	範例值
CloudFront-Viewer-Cert-Issuer	RFC2253 發行者辨別名稱 (DN) 的字串表示	CN=rootcamtls.com , OU=rootCA , O=mTLS , L=Seattle , ST=Washington , C=US
CloudFront-Viewer-Cert-Subject	主旨辨別名稱 (DN) 的 RFC2253 字串表示	CN=client_.com , OU=client-3 , O=mTLS , ST=Washington , C=US
CloudFront-Viewer-Cert-Present	1 (存在) 或 0 (不存在) 表示憑證是否存在。在必要模式下，此值一律為 1。	1
CloudFront-Viewer-Cert-Sha256	用戶端憑證的 SHA256 雜湊	01fbf94fef5569753420c349f49adbfd80af5275377816e3ab1fb371b29cb586

對於原始伺服器請求，除了上述標頭可用於快取行為之外，還提供兩個額外的標頭：

標頭名稱	Description	範例值
CloudFront-Viewer-Cert-Validity	notBefore 和 notAfter date 的 ISO8601 格式	CloudFront-Viewer-Cert-Validity : NotBefore=2023-09-21T01 : 50 : 17Z ; NotAfter=2024-09-20T01 : 50 : 17Z
CloudFront-Viewer-Cert-Pem	分葉憑證的 URL 編碼 PEM 格式	CloudFront-Viewer-Cert-Pem : -----BEGIN%20CERTIFICATE-----%0AMIIG<...reduced...>NmrUlw%0A-----END%20CERTIFICATE-----%0A

設定標頭轉送

主控台

在驗證模式中，CloudFront 會自動將 CloudFront-Viewer-Cert-* 標頭新增至所有檢視器請求。若要將這些標頭轉送到您的原始伺服器：

1. 在主清單分佈頁面中，選取已啟用檢視器 mTLS 的分佈，然後前往行為索引標籤
2. 選取快取行為，然後選擇編輯
3. 在原始伺服器請求政策區段中，選擇建立政策或選取現有政策
4. 確保原始伺服器請求政策中包含下列標頭：
 - CloudFront-Viewer-Cert-Serial-Number
 - CloudFront-Viewer-Cert-Issuer
 - CloudFront-Viewer-Cert-Subject
 - CloudFront-Viewer-Cert-Present
 - Cloudfront-Viewer-Cert-Sha256
 - CloudFront-Viewer-Cert-Validity
 - CloudFront-Viewer-Cert-Pem
5. 選擇建立（適用於新政策）或儲存變更（適用於現有政策）
6. 在快取行為中選取政策並儲存變更

使用 AWS CLI

下列範例示範如何建立原始伺服器請求政策，其中包含用於驗證模式的 mTLS 標頭：

```
aws cloudfront create-origin-request-policy \  
  --origin-request-policy-config '{  
    "Name": "MTLSHeadersPolicy",  
    "HeadersConfig": {  
      "HeaderBehavior": "whitelist",  
      "Headers": {  
        "Quantity": 5,  
        "Items": [  
          "CloudFront-Viewer-Cert-Serial-Number",  
          "CloudFront-Viewer-Cert-Issuer",  
          "CloudFront-Viewer-Cert-Subject",
```

```
        "CloudFront-Viewer-Cert-Validity",
        "CloudFront-Viewer-Cert-Pem"
    ]
},
"CookiesConfig": {
    "CookieBehavior": "none"
},
"QueryStringConfig": {
    "QueryStringBehavior": "none"
}
}'
```

標頭處理考量

使用憑證標頭時，請考慮下列最佳實務：

- 標頭驗證：驗證原始伺服器的憑證標頭值作為額外的安全措施
- 標頭大小限制：PEM 憑證標頭可以很大，確保您的原始伺服器可以處理它們
- 快取考量：在快取金鑰中使用憑證標頭會增加快取分段
- 跨來源請求：如果您的應用程式使用 CORS，則您可能需要將其設定為允許憑證標頭

後續步驟

設定標頭轉送之後，您可以使用 CloudFront Connection Functions 和 KeyValueTypeStore 實作憑證撤銷檢查。如需實作撤銷檢查的詳細資訊，請參閱 [使用 CloudFront Connection Function 和 KVS 撤銷](#)。

使用 CloudFront Connection Function 和 KVS 撤銷

您可以結合 CloudFront Connection Functions 與 KeyValueTypeStore，實作交互 TLS 身分驗證的憑證撤銷檢查。此方法提供可擴展的即時憑證撤銷機制，可補充 CloudFront 的內建憑證驗證。

Connection Functions 是在 CloudFront 節點建立 TLS 連線期間執行的 JavaScript 函數，可讓您實作 mTLS 身分驗證的自訂憑證驗證邏輯。如需連線函數的詳細資訊，請參閱 [關聯 CloudFront 連線函數](#)。

憑證撤銷如何與 Connection Functions 搭配使用

CloudFront 的標準憑證驗證會驗證憑證鏈、簽章和過期，但不包含內建憑證撤銷檢查。透過使用連線函數，您可以在 TLS 交握期間實作自訂撤銷檢查。

憑證撤銷程序的運作方式如下：

1. 在 CloudFront KeyValueStore 中存放撤銷的憑證序號。
2. 當用戶端提供憑證時，就會叫用您的連線函數。
3. 函數會根據 KeyValueStore 檢查憑證的序號。
4. 如果在存放區中找到序號，則會撤銷憑證。
5. 您的函數拒絕已撤銷憑證的連線。

此方法提供跨 CloudFront 全球邊緣網路的near-real-time撤銷檢查。

為已撤銷的憑證設定 KeyValueStore

首先，建立 KeyValueStore 以存放已撤銷憑證的序號：

建立 KeyValueStore（主控台）

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇鍵值存放區。
3. 選擇建立索引鍵值存放區。
4. 輸入金鑰值存放區的名稱（例如，已撤銷憑證）。
5. (選用) 新增描述。
6. 選擇建立索引鍵值存放區。

建立 KeyValueStore (AWS CLI)

下列範例示範如何建立 KeyValueStore：

```
aws cloudfront create-key-value-store \  
  --name "revoked-certificates" \  
  --comment "Store for revoked certificate serial numbers"
```

匯入已撤銷的憑證序號

建立 KeyValueStore 之後，您需要匯入已撤銷憑證的序號：

準備撤銷資料

使用您撤銷的憑證序號建立 JSON 檔案：

```
{
  "data": [
    {
      "key": "ABC123DEF456",
      "value": ""
    },
    {
      "key": "789XYZ012GHI",
      "value": ""
    }
  ]
}
```

從 S3 匯入

1. 將 JSON 檔案上傳至 S3 儲存貯體
2. 將檔案匯入至 KeyValueStore：

```
aws cloudfront create-key-value-store \
  --name "revoked-certificates" \
  --import-source '{
    "SourceType": "S3",
    "SourceARN": "arn:aws:s3:::amzn-s3-demo-bucket1/revoked-serials.json"
  }'
```

建立用於撤銷檢查的連線函數

建立連線函數，以檢查憑證序號與 KeyValueStore：

Connection Function 程式碼範例

下列範例顯示執行憑證撤銷檢查的連線函數：

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
```

```

const kvsHandle = cf.kvs();

// Get client certificate serial number
const clientSerialNumber =
connection.clientCertificate.certificates.leaf.serialNumber;

// Check if the serial number exists in the KeyValueStore
const isRevoked = await kvsHandle.exists(clientSerialNumber.replaceAll(':', ''));

if (isRevoked) {
    console.log(`Certificate ${clientSerialNumber} is revoked. Denying
connection.`);
    connection.logCustomData(`REVOKED:${clientSerialNumber}`);
    connection.deny();
} else {
    console.log(`Certificate ${clientSerialNumber} is valid. Allowing
connection.`);
    connection.allow();
}
}

```

建立連線函數 (AWS CLI)

下列範例示範如何使用 KeyValueStore 關聯建立連線函數：

```

aws cloudfront create-connection-function \
  --name "revocation-checker" \
  --connection-function-config '{
    "Comment": "Certificate revocation checking function",
    "Runtime": "cloudfront-js-2.0",
    "KeyValueStoreAssociations": {
      "Quantity": 1,
      "Items": [
        {
          "KeyValueStoreARN": "arn:aws:cloudfront::123456789012:key-value-
store/revoked-certificates"
        }
      ]
    }
  }' \
  --connection-function-code fileb://revocation-checker.js

```

將函數與您的分佈建立關聯

建立和發佈連線函數之後，請將其與啟用 mTLS 的 CloudFront 分佈建立關聯，如 [關聯 CloudFront 連線函數](#) 一節中所述。

使用連線日誌的可觀測性

CloudFront 連線日誌提供交互 TLS 身分驗證事件的詳細可見性，可讓您監控憑證驗證、追蹤連線嘗試，以及疑難排解身分驗證問題。

什麼是連線日誌？

連線日誌會擷取有關啟用交互 TLS 分佈的 TLS 交握和憑證驗證的詳細資訊。與記錄 HTTP 請求資訊的標準存取日誌不同，連線日誌特別著重於 TLS 連線建立階段，包括：

- 連線狀態（成功/失敗）
- 用戶端憑證詳細資訊
- TLS 通訊協定和密碼資訊
- 連線時間指標
- 來自 Connection Functions 的自訂資料

這些日誌提供憑證型身分驗證事件的完整可見性，協助您監控安全性、疑難排解問題，並符合合規要求。

啟用連線日誌

連線日誌僅適用於已啟用交互 TLS 身分驗證的分佈。您可以將連線日誌傳送至多個目的地，包括 CloudWatch Logs、Amazon Data Firehose 和 Amazon S3。

先決條件

啟用連線日誌之前：

- 為您的 CloudFront 分佈設定交互 TLS
- 為您的 CloudFront 分佈啟用連線日誌
- 確定您擁有所選記錄目的地的必要許可
- 對於跨帳戶交付，請設定適當的 IAM 政策

啟用連線日誌（主控台）

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 從分發清單中，選取已啟用 mTLS 的分發。
3. 選擇 Logging (日誌記錄) 索引標籤。
4. 選擇新增。
5. 選取要接收日誌的服務：
 - CloudWatch Logs
 - Firehose
 - Amazon S3
6. 針對目的地，選取所選服務的資源：
 - 對於 CloudWatch Logs，輸入日誌群組名稱
 - 對於 Firehose，選取 Firehose 交付串流
 - 對於 Amazon S3，輸入儲存貯體名稱（選擇性使用字首）
7. (選用) 設定其他設定：
 - 欄位選擇：選取要包含的特定日誌欄位。
 - 輸出格式：從 JSON、Plain、w3c、Raw 或 Parquet（僅限 S3）中選擇。
 - 欄位分隔符號：指定如何分隔日誌欄位。
8. 選擇 Save changes (儲存變更)

啟用連線日誌 (AWS CLI)

下列範例示範如何使用 CloudWatch API 啟用連線日誌：

```
# Step 1: Create a delivery source
aws logs put-delivery-source \
  --name "cf-mtls-connection-logs" \
  --resource-arn "arn:aws:cloudfront::123456789012:distribution/E1A2B3C4D5E6F7" \
  --log-type CONNECTION_LOGS

# Step 2: Create a delivery destination
aws logs put-delivery-destination \
  --name "s3-destination" \
```

```
--delivery-destination-configuration \  
"destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket1"  
  
# Step 3: Create the delivery  
aws logs create-delivery \  
  --delivery-source-name "cf-mtls-connection-logs" \  
  --delivery-destination-arn "arn:aws:logs:us-east-1:123456789012:delivery-  
destination:s3-destination"
```

Note

使用 CloudWatch API 時，即使將日誌交付至其他區域，您也必須指定美國東部（維吉尼亞北部）區域 (us-east-1)。

連線日誌欄位

連線日誌包含每次 TLS 連線嘗試的詳細資訊：

欄位	Description	範例
eventTime stamp	建立連線或失敗時的 ISO 8601 時間戳記	1731620046814
connectio nId	TLS 連線的唯一識別符	oLHiEKbQSn8lkvJfA3 D4gFowK3_iZ0g4i5nM UjE1Akod8TuAzn5nzg==
connectio nStatus	mTLS 連線嘗試的狀態。	Success 或 Failed
clientIp	連線用戶端的 IP 地址	2001:0db8:85a3:000 0:0000:8a2e:0370:7334
clientPort	用戶端使用的連接埠	12137
serverIp	CloudFront 邊緣伺服器的 IP 地址	99.84.71.136
distribut ionId	CloudFront 分佈 ID	E2DX1SLDPK0123

欄位	Description	範例
distributionTenantId	CloudFront 分佈租用戶 ID (如適用)	dt_2te1Ura9X3R2iCGNjW123
tlsProtocol	使用的 TLS 通訊協定版本	TLSv1.3
tlsCipher	用於連線的 TLS 密碼套件	TLS_AES_128_GCM_SHA256
tlsHandshakeDuration	TLS 交握的持續時間，以毫秒為單位	153
tlsSni	來自 TLS 交握的伺服器名稱指示值	d111111abcdef8.cloudfront.net
clientLeafCertSerialNumber	用戶端憑證的序號	00:b1:43:ed:93:d2:d8:f3:9d
clientLeafCertSubject	用戶端憑證的主旨欄位	C=US, ST=WA, L=Seattle, O=Amazon.com, OU=CloudFront, CN=client.test.mtls.net
clientLeafCertIssuer	用戶端憑證的發行者欄位	C=US, ST=WA, L=Seattle, O=Amazon.com, OU=CloudFront, CN=test.mtls.net
clientLeafCertValidity	用戶端憑證的有效期	NotBefore=2025-06-05T23:28:21Z;NotAfter=2125-05-12T23:28:21Z
connectionLogCustomData	透過 Connection Functions 新增的自訂資料	REVOKED:00:b1:43:ed:93:d2:d8:f3:9d

連線錯誤代碼

```
Failed:ClientCertMaxChainDepthExceeded
Failed:ClientCertMaxSizeExceeded
Failed:ClientCertUntrusted
Failed:ClientCertNotYetValid
Failed:ClientCertExpired
Failed:ClientCertTypeUnsupported
Failed:ClientCertInvalid
Failed:ClientCertIntentInvalid
Failed:ClientCertRejected
Failed:ClientCertMissing
Failed:TcpError
Failed:TcpTimeout
Failed:ConnectionFunctionError
Failed:ConnectionFunctionDenied
Failed:Internal
Failed:UnmappedConnectionError
```

當連線失敗時，CloudFront 會記錄特定原因代碼：

Code	Description
ClientCertMaxChainDepthExceeded	超過憑證鏈深度上限
ClientCertMaxSizeExceeded	超過憑證大小上限
ClientCertUntrusted	憑證不受信任
ClientCertNotYetValid	憑證尚未有效
ClientCertExpired	憑證已過期
ClientCertTypeUnsupported	不支援憑證類型
ClientCertInvalid	憑證無效
ClientCertIntentInvalid	憑證意圖無效
ClientCertRejected	自訂驗證拒絕的憑證

Code	Description
ClientCertMissing	憑證遺失
TcpError	嘗試建立連線時發生錯誤
TcpTimeout	無法在逾時期間內建立連線
ConnectionFunctionError	在 Connection Function 執行期間擲回未攔截的例外狀況
內部 (Internal)	發生內部服務錯誤
UnmappedConnectionError	發生錯誤，其不屬於任何其他類別

使用已簽署 URL 和已簽署 Cookie 提供私有內容

許多透過網際網路分配內容的公司想要限制對所選使用者的文件、業務資料、媒體串流，或內容的存取許可，例如已付費的使用者。若要使用 CloudFront 安全地提供這類私有內容，請執行以下操作：

- 請求使用者使用特殊 CloudFront 已簽署的 URL 或已簽署的 Cookie 存取私有內容。
- 請求您的使用者使用 CloudFront URL 來存取您的內容，而不要使用直接在原始伺服器上存取內容的 URL (例如，Amazon S3 或私有 HTTP 伺服器)。CloudFront URL 非屬必要但建議使用，可防止使用者略過已簽署的 URL 或已簽署的 Cookie 中指定的限制。

如需詳細資訊，請參閱[限制檔案存取](#)。

如何提供私有內容

若要設定 CloudFront 以提供私有內容，請執行以下任務：

1. (選用建議) 請求您的使用者只能透過 CloudFront 存取內容。您使用的方法取決於您使用的是 Amazon S3 或自訂原始伺服器：
 - Amazon S3 – 請參閱 [the section called “限制對 Amazon S3 原始伺服器的存取”](#)。
 - 自訂原始伺服器 – 請參閱 [在自訂原始伺服器上限制存取檔案](#)。

自訂原始伺服器包括 Amazon EC2、設定為網站端點的 Amazon S3 儲存貯體、ELB 和您自己的 HTTP Web 伺服器。

2. 指定您要用來建立已簽署 URL 或已簽署 Cookie 的信任金鑰群組或可信簽署者。建議您使用信任的金鑰群組。如需詳細資訊，請參閱 [指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。
3. 撰寫您的應用程式，使用已簽章的 URL 或使用已設定簽章的 Cookie 的 Set-Cookie 標頭，以回應來自已獲授權使用者的請求。請遵循下列其中一個主題中的步驟進行：
 - [使用已簽署 URL](#)
 - [使用已簽署 Cookie](#)

如果您不確定要使用哪個方法，請參閱 [決定使用已簽署 URL 或已簽署 Cookie](#)。

主題

- [限制檔案存取](#)
- [指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)
- [決定使用已簽署 URL 或已簽署 Cookie](#)
- [使用已簽署 URL](#)
- [使用已簽署 Cookie](#)
- [用於 Base64 編碼和加密的 Linux 命令和 OpenSSL](#)
- [為已簽署 URL 建立簽章的程式碼範例](#)

限制檔案存取

您可以透過兩種方式控制使用者對私人內容的存取權：

- [限制存取 CloudFront 快取中的檔案](#)。
- 執行以下其中一項以在您的原始伺服器中限制存取檔案：
 - [為您的 Amazon S3 儲存貯體設定原始存取控制 \(OAC\)](#)。
 - [為私有 HTTP 伺服器 \(自訂原始伺服器\) 設定自訂標頭](#)。

限制存取 CloudFront 快取中的檔案

您可以設定 CloudFront，來請求使用者使用已簽署的 URL 或已簽署的 Cookie，以存取您的檔案。然後，您可以開發應用程式，以建立和分配已簽署的 URL 給驗證過的使用者，或傳送 Set-Cookie 標頭，這是經驗證之使用者設定的已簽署 Cookie。(若要提供一些使用者長期存取少量檔案，您還可以手動建立簽章的 URL。)

當您建立已簽章的 URL 或已簽章的 Cookie，以控制存取您的檔案時，可以指定下列限制：

- 結束日期和時間，之後 URL 不再有效。
- (選用) URL 生效的日期和時間。
- (選用) 可用於存取內容的電腦的 IP 位址或地址範圍。

使用公有-私有金鑰對中的私有金鑰對已簽署的 URL 或已簽署的 Cookie 的一部分進行雜湊和簽名。當有人使用已簽署的 URL 或已簽署的 Cookie 存取檔案時，CloudFront 會比較 URL 或 Cookie 的已簽署和未簽署部分。如果它們不相符，CloudFront 就不會提供檔案。

您必須使用 RSA 2048 或 ECDSA 256 私有金鑰來簽署 URL 或 Cookie。

限制對 Amazon S3 儲存貯體中檔案的存取

您可以選擇性地保護 Amazon S3 儲存貯體中的內容，以便使用者可以透過指定的 CloudFront 分佈，但無法使用 Amazon S3 URL 直接進行存取。這可防止某些人繞過 CloudFront 並利用 Amazon S3 URL 來取得您想要限制其存取的內容。使用簽章 URL 不需要此步驟，但我們建議執行此作業。

若要請求使用者透過 CloudFront URL 存取您的內容，請執行下列任務：

- 授予 CloudFront 原始存取控制許可，以讀取 S3 儲存貯體中的檔案。
- 建立原始存取控制，並將其與您的 CloudFront 分佈產生關聯。
- 移除其他人使用 Amazon S3 URL 讀取檔案的許可。

如需詳細資訊，請參閱[the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

在自訂原始伺服器上限制存取檔案

如果您使用自訂原始伺服器，您可以選擇性設定自訂標頭來限制存取。若要使 CloudFront 從自訂原始伺服器取得您的檔案，必須使用標準 HTTP (或 HTTPS) 請求來提供 CloudFront 檔案的存取權限。然而透過自訂標頭，您可以限制存取內容，讓使用者只能透過 CloudFront，而無法直接進行存取。使用簽章 URL 不需要此步驟，但我們建議執行此作業。

若要請求使用者透過 CloudFront 存取內容，請在您的 CloudFront 分佈中變更下列設定：

原始伺服器自訂標頭

設定 CloudFront 以將自訂標頭轉送至原始伺服器 請參閱 [設定 CloudFront 以將自訂標頭新增到原始伺服器請求](#)。

檢視器通訊協定政策

設定您的分佈以請求檢視器使用 HTTPS 來存取 CloudFront。請參閱 [檢視器通訊協定政策](#)。

原始伺服器通訊協定政策

將分佈設定為請求 CloudFront 使用與檢視器相同的協定將請求轉傳到原始伺服器。請參閱 [通訊協定 \(僅限自訂原始伺服器\)](#)。

完成這些變更之後，請在自訂原始伺服器上更新應用程式，以僅接受包含您已設定 CloudFront 要傳送之自訂標頭的請求。

檢視器通訊協定政策與原始伺服器通訊協定政策的組合可確保自訂標頭在傳輸過程中加密。但是，我們建議您定期執行以下任務來輪換 CloudFront 轉傳到您原始伺服器的自訂標頭：

1. 更新您的 CloudFront 分佈以開始將新標頭轉送至您的自訂原始伺服器。
2. 更新您的應用程式以接受新標頭，以確認請求來自 CloudFront。
3. 當檢視器請求不再包括要替換的標頭時，請更新應用程式至不再接受的舊標頭，以確認請求來自 CloudFront。

指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者

主題

- [在信任的金鑰群組（建議）和之間進行選擇 AWS 帳戶](#)
- [為您的簽署者建立金鑰對](#)
- [重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)
- [將簽署者新增至分佈](#)
- [輪換金鑰對](#)

若要建立已簽署 URL 或已簽署 Cookie，您需要簽署者。簽署者是您在 CloudFront 中建立的受信任金鑰群組，或是包含 CloudFront 金鑰對 AWS 的帳戶。建議您使用具有已簽署 URL 和已簽署 Cookie 信任的金鑰群組。如需詳細資訊，請參閱 [在信任的金鑰群組（建議）和之間進行選擇 AWS 帳戶](#)。

簽署者有兩個用途：

- 一旦您將簽署者新增至您的分佈，CloudFront 就會開始請求檢視器使用已簽署的 URL 或已簽署的 Cookie 來存取您的檔案。

- 當您建立已簽署 URL 或已簽署 Cookie 時，您可以使用簽署者金鑰對中的私有金鑰對 URL 或 Cookie 的一部分進行簽署。當有人請求受限制的檔案時，CloudFront 會將 URL 或 Cookie 中的簽章與未簽署的 URL 或 Cookie 進行比較，以確認檔案未遭竄改。CloudFront 也可以驗證 URL 或 Cookie 是否有效，也就是好比說到日期時間到期了嗎？

當您指定簽署者時，您也可以透過將簽署者新增至快取行為，以間接指定需要簽署 URL 或簽署 Cookie 的檔案。如果分佈只有一個快取行為，則使用者必須使用已簽署 URL 或已簽署 Cookie 來存取該分佈中的任何檔案。如果您建立多個快取行為，並將簽署者新增到一些快取行為，而非其他快取行為，則可以請求檢視器使用已簽署 URL 或已簽署 Cookie 來存取這些物件，而非其他檔案。

若要指定允許建立已簽署 URL 或已簽署 Cookie 的簽署者 (私有金鑰)，並將帳戶新增到 CloudFront 分佈中，請執行以下任務：

1. 決定要使用信任的金鑰群組或 AWS 帳戶 做為簽署者。建議您使用信任的金鑰群組。如需詳細資訊，請參閱 [在信任的金鑰群組（建議）和之間進行選擇 AWS 帳戶](#)。
2. 針對您在步驟 1 中選擇的簽署者，建立公有-私有金鑰對。如需詳細資訊，請參閱 [為您的簽署者建立金鑰對](#)。
3. 如果您使用 .NET 或 Java 建立已簽署 URL 或已簽署 Cookie，請重新格式化私有金鑰。如需詳細資訊，請參閱 [重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)。
4. 在您要為其建立已簽署 URL 或已簽署 Cookie 的分佈中，指定簽署者。如需詳細資訊，請參閱[將簽署者新增至分佈](#)。

在信任的金鑰群組（建議）和之間進行選擇 AWS 帳戶

若要使用已簽署 URL 或已簽署 Cookie，您需要 簽署者。簽署者可以是您在 CloudFront 中建立的可信金鑰群組，也可以是包含 CloudFront 金鑰對的 AWS 帳戶。基於下列原因，建議您使用信任的金鑰群組：

- 使用 CloudFront 金鑰群組時，您不需要使用 AWS 帳戶根使用者來管理 CloudFront 已簽章 URLs 公有金鑰。[AWS 最佳實務](#)建議您不需要使用根使用者。
- 透過 CloudFront 金鑰群組，您可以使用 CloudFront API 管理公有金鑰、金鑰群組和受信任的簽署者。您可以使用 API 自動化金鑰建立和金鑰輪換。當您使用 AWS 根使用者時，必須使用 AWS 管理主控台 來管理 CloudFront 金鑰對，因此您無法自動化程序。
- 由於您可以使用 CloudFront API 管理金鑰群組，因此您也可以使用 AWS Identity and Access Management (IAM) 許可原則來限制不同使用者可執行的動作。例如，您可以允許使用者上傳公有金

鑰，但不能刪除公有金鑰。或者，您可以允許使用者刪除公有金鑰，但只有在符合特定條件時，例如使用多因素驗證、從特定網路傳送請求，或是在特定日期和時間範圍內傳送請求。

- 透過 CloudFront 金鑰群組，您可以將較多數目的公有金鑰與您的 CloudFront 分佈產生關聯，讓您在使用和管理公有金鑰方面擁有更大的彈性。根據預設，您最多可以將四個金鑰群組與單一分佈產生關聯，而且一個金鑰群組中最多可以有五個公有金鑰。

當您使用 AWS 帳戶根使用者管理 CloudFront 金鑰對時，每個 AWS 帳戶最多只能有兩個作用中的 CloudFront 金鑰對。

為您的簽署者建立金鑰對

您用來建立 CloudFront 已簽署 URL 或已簽署 Cookie 的每個簽署者都必須具有公有-私有金鑰對。簽署者使用其私有金鑰來簽署 URL 或 Cookie，並且 CloudFront 會使用公有金鑰來驗證簽章。

建立金鑰對的方式視您使用信任的金鑰群組做為簽署者 (建議使用) 或是 CloudFront 金鑰對而定。如需詳細資訊，請參閱下列區段。您建立的金鑰對必須符合下列需求：

- 它必須是 SSH-2 RSA 2048 或 ECDSA 256 金鑰對。
- 它必須是 base64 編碼的 PEM 格式。

為了協助保護應用程式的安全，建議您定期輪換金鑰對。如需詳細資訊，請參閱 [輪換金鑰對](#)。

為信任的金鑰群組建立金鑰對 (建議使用)

若要為信任的金鑰群組建立金鑰對，請執行下列步驟：

1. 建立公有-私有金鑰對。
2. 將公有金鑰上傳至 CloudFront。
3. 將公有金鑰新增至 CloudFront 金鑰群組。

如需詳細資訊，請參閱下列程序。

建立一組金鑰對

Note

下列步驟會使用 OpenSSL 做為建立金鑰對的一種方式來示範。還有許多其他方法可以建立 RSA 或 ECDSA 金鑰對。

1. 請執行下列其中一個範例命令：

- 下列範例命令會使用 OpenSSL 產生長度為 2048 位元的 RSA 金鑰對，並儲存至名為 `private_key.pem` 的檔案。

```
openssl genrsa -out private_key.pem 2048
```

- 下列範例命令會使用 OpenSSL 產生具備 prime256v1 曲線的 ECDSA 金鑰對，並儲存至名為 `private_key.pem` 的檔案。

```
openssl ecparam -name prime256v1 -genkey -noout -out privatekey.pem
```

2. 產生的檔案同時包含公有和私有金鑰。下列範例命令會從名為 `private_key.pem` 的檔案擷取公有金鑰。

```
openssl rsa -pubout -in private_key.pem -out public_key.pem
```

您稍後會在下列程序中上傳公有金鑰 (在 `public_key.pem` 檔案中)。

將公有金鑰上傳至 CloudFront

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽功能表中，選擇公有金鑰。
3. 選擇建立公有金鑰。
4. 在建立公有金鑰視窗中，執行下列動作：
 - a. 在金鑰名稱中，輸入識別公有金鑰的名稱。
 - b. 對於 Key value (鍵值)，貼上公有金鑰。如果您遵循上述程序中的步驟，則公有金鑰位於名為 `public_key.pem` 的檔案中。若要複製並貼上公有金鑰的內容，您可以：
 - 在 macOS 或 Linux cat 命令列上使用這個命令，如下所示：

```
cat public_key.pem
```


複製該命令的輸出，然後將其貼 Key value (鍵值) 欄位中。

- 使用像「記事本」(在 Windows 上) 或「文字編輯」(在 macOS 上) 等純文字編輯器開啟 `public_key.pem` 檔案。複製檔案內容，然後將其貼到 Key value (鍵值) 欄位中。

c. (選用) 對於註解，請新增註解以描述公有金鑰。

完成時，請選擇新增。

5. 記錄公有金鑰 ID。您稍後在建立已簽署 URL 或已簽署 Cookie 時使用它，以做為 Key-Pair-Id 欄位的值。

將公有金鑰新增至金鑰群組

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽功能表中，選擇金鑰群組。
3. 選擇新增金鑰群組。
4. 在建立金鑰群組頁面上，執行下列動作：
 - a. 在金鑰群組名稱中，輸入識別金鑰群組的名稱。
 - b. (選用) 在註解中，輸入註解以描述金鑰群組。
 - c. 在公有金鑰中，選取要新增至金鑰群組的公有金鑰，然後選擇新增。針對您要新增至金鑰群組的每個公有金鑰重複此步驟。
5. 選擇建立金鑰對。
6. 記錄金鑰群組名稱。稍後您可以使用它來將金鑰群組與 CloudFront 分佈中的快取行為產生關聯。(在 CloudFront API 中，您可以使用金鑰群組 ID，將金鑰群組與快取行為產生關聯。)

建立 CloudFront 金鑰對 (不建議，需要 AWS 帳戶 根使用者)

Important

建議您建立信任金鑰群組的公有金鑰，而不是遵循下列步驟。如需建立已簽署 URL 和已簽署 Cookie 之公有金鑰的建議方式，請參閱[為信任的金鑰群組建立金鑰對 \(建議使用\)](#)。

您可以使用下列方式建立 CloudFront 金鑰對：

- 在 中建立金鑰對 AWS 管理主控台 並下載私有金鑰。請參見下列步驟：
- 使用諸如 OpenSSL 的應用程式建立 RSA 金鑰對，並將公有金鑰上傳到 AWS 管理主控台。如需有關建立 RSA 金鑰對的詳細資訊，請參閱 [為信任的金鑰群組建立金鑰對 \(建議使用\)](#)。

在 中建立 CloudFront 金鑰對 AWS 管理主控台

1. AWS 管理主控台 使用帳戶根使用者的登入資料登入 AWS 。

Important

IAM 使用者無法建立 CloudFront 金鑰對。您必須使用根使用者登入資料登入，才能建立金鑰對。

2. 選擇您的帳戶名稱，然後選擇我的安全登入資料。
3. 選擇 CloudFront 金鑰對。
4. 確認您沒有多個作用中的金鑰對。如果您已經有兩個使用中的金鑰對，則無法建立金鑰對。
5. 選擇建立新的金鑰對。

Note

您也可以選擇建立自己的金鑰對並上傳公有金鑰。CloudFront 金鑰對支援 1024、2048 或 4096 位元金鑰。

6. 在建立金鑰對的對話方塊中，選擇下載私有金鑰檔案，然後將檔案儲存在電腦上。

Important

在安全的位置儲存 CloudFront 金鑰對的私有金鑰，並設定該檔案的許可，以便只有所需的管理局員可以讀取該金鑰。如果有人取得您的私有金鑰，他們可以產生有效已簽章的 URL 和已簽章的 Cookie 並下載您的內容。您不能再次取得私有金鑰，因此如果您遺失或刪除它，則必須建立新的 CloudFront 金鑰對。

7. 記錄您的金鑰對的金鑰對 ID。(在 中 AWS 管理主控台，這稱為存取金鑰 ID。)您將在建立已簽署 URL 或已簽署 Cookie 時使用它。

重新格式化私有金鑰 (僅限 .NET 和 Java)

如果您正使用 .NET 或 Java 以建立已簽署 URL 或已簽署 Cookie，則無法在預設 PEM 格式中，使用金鑰對的私有金鑰來建立簽章。相反地，請執行下列動作：

- .NET 框架 – 將私有金鑰轉換為 .NET 框架使用的 XML 格式。有幾種工具可用。
- Java – 將私有金鑰轉換為 DER 格式。執行此操作的一種方法是使用以下 OpenSSL 命令。在下列命令中，`private_key.pem` 是包含 PEM 格式化的私有金鑰的檔案名稱，而且 `private_key.der` 是執行命令之後包含 DER 格式化的私有金鑰的檔案名稱。

```
openssl pkcs8 -topk8 -nocrypt -in private_key.pem -inform PEM -out private_key.der -outform DER
```

為了確保編碼器能正確運作，請將 Bouncy Castle Java 的密碼編譯 API 的 JAR 新增到專案中，然後再新增 Bouncy Castle 供應商。

將簽署者新增至分佈

簽署者是信任的金鑰群組 (建議使用) 或 CloudFront 金鑰對，可為分佈建立已簽署 URL 和已簽署 Cookie。若要在 CloudFront 分佈中使用已簽署 URL 或已簽署 Cookie，您必須指定簽署者。

簽署者與快取行為相關聯。在某些物件或不用於相同分佈之其他檔案下，這允許您要求簽章 URL 或簽章 Cookie。只有和對應快取行為相關聯的檔案，分佈才會要求已簽署的 URL 或 Cookie。

同樣地，簽署者只能為與對應快取行為相關聯的檔案簽署 URL 或 Cookie。例如，如果您有一個快取行為下的簽署者和不同快取行為下的不同簽署者，則這兩個簽署者都不能為與其他快取行為的相關檔案建立簽署 URL 或 Cookie。

Important

在您將簽署者新增至分佈之前，請執行下列動作：

- 請仔細定義快取行為中的路徑模式，以及快取行為順序，這樣您就不會讓使用者意外地存取您的內容，也不會讓他們存取您希望所有人都能使用的內容。

例如，假設請求與兩個快取行為的路徑模式相符合。第一個快取行為不需要已簽章的 URL 或已簽章的 Cookie，而第二個快取行為則需要。使用者將能夠在不使用已簽署的 URL 或已簽署的 Cookie 的情況下存取檔案，因為 CloudFront 處理與第一個符合相關聯的快取行為。

如需瞭解歐冠路徑模式的詳細資訊，請參閱 [路徑模式](#)。

- 對於您已經用來分佈內容的分配內容，請在新增簽署者之前，確定您已準備好開始產生已簽署 URL 和已簽署 Cookie。當您新增簽署者時，CloudFront 會拒絕未包含有效的已簽署 URL 或已簽署 Cookie 請求。

您可以使用 CloudFront 主控台或 CloudFront API 會將簽署者新增至您的分佈。

Console

下列步驟說明如何將信任的金鑰群組新增為簽署者。您也可以新增 AWS 帳戶 做為信任的簽署者，但不建議這麼做。

使用主控台將簽署者新增至分佈

1. 記錄要用作信任的簽署者之金鑰群組的金鑰群組 ID。如需詳細資訊，請參閱 [為信任的金鑰群組建立金鑰對 \(建議使用\)](#)。
2. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
3. 使用已簽署 URL 或已簽署 Cookie，選擇要保護其檔案的分佈。

Note

若要將簽署者新增至新的發行套件，您可以指定建立發行套件時步驟 6 所述的相同設定。

4. 選擇 Behaviors (動作) 索引標籤。
5. 選取快取行為，其路徑模式符合您要使用已簽署 URL 或已簽署 Cookie 保護的檔案，然後選擇編輯。
6. 在編輯行為頁面上，執行下列動作：
 - a. 針對限制檢視器存取 (使用已簽署 URL 或已簽署 Cookie)，按一下是。
 - b. 針對 Trusted Key Groups or Trusted Signer (信任的金鑰群組或可信簽署者)，選擇 Trusted Key Groups (信任的金鑰群組)。
 - c. 在信任的金鑰群組中，選擇要新增的金鑰群組，然後選擇新增。如果您要新增多個金鑰群組，請重複此步驟。
7. 選擇是，編輯以更新快取行為。

API

您可以使用 CloudFront API 將信任的金鑰群組新增為簽署者。您可以將簽署者新增至現有分佈或新分佈。在任一情況下，請在 `TrustedKeyGroups` 元素中指定值。

您也可以新增 AWS 帳戶 做為信任的簽署者，但不建議這麼做。

請參閱 Amazon CloudFront API 參考中的下列主題：

- 更新現有的分佈 - [UpdateDistribution](#)
- 建立新的分佈 - [CreateDistribution](#)

輪換金鑰對

建議您定期輪換 (變更) 已簽署網址和已簽署 Cookie 的金鑰對。若要輪換您用來建立已簽署 URL 或已簽署 Cookie 的金鑰對，而不會使尚未到期的無效 URL 或 Cookie，請執行以下任務：

1. 建立新的金鑰對，並將公有金鑰新增至金鑰群組。如需詳細資訊，請參閱 [為信任的金鑰群組建立金鑰對 \(建議使用\)](#)。
2. 如果您在上一個步驟中建立了新的金鑰群組，[請以簽署者的身分將金鑰群組新增到分佈中](#)。

Important

請勿從金鑰群組中移除任何現有的公有金鑰，或從分佈中移除任何金鑰群組。請僅加入新的。

3. 使用新金鑰對中的私有金鑰來更新應用程式以建立簽章。確認已簽署 URL 或使用新私有金鑰簽署的 Cookie 正在運作。
4. 等待直到已簽署 URL 或 Cookie 過期，而使用先前的金鑰對。然後從金鑰群組中移除舊的公有金鑰。如果您在步驟 2 中建立新的金鑰群組，請從分佈中移除舊的金鑰群組。

決定使用已簽署 URL 或已簽署 Cookie

CloudFront 已簽署的 URL 和已簽署的 Cookie 提供了相同的基本函數：允許您控制哪些使用者可以存取您的內容。如果您想要透過 CloudFront 提供私有內容，並且正在嘗試決定是否使用已簽署的 URL 或已簽署的 Cookie，請考慮以下事項。

在以下案例使用已簽章的 URL：

- 您想要限制對個別檔案的存取，例如，適用於您的應用程式安裝下載。
- 您的使用者正在使用不支援 Cookie 的用戶端 (例如，自訂 HTTP 用戶端)。

在以下案例使用已簽章的 Cookie：

- 您想要提供對多個限制檔案的存取，例如，HLS 格式視訊的所有檔案或網站中訂閱者區域的所有檔案。
- 您不想變更目前的 URL。

如果您目前未使用簽章的 URL，並且您的 (未簽署) URL 包含以下任何查詢字串參數，則不能使用已簽章的 URL 或已簽章的 Cookie：

- Expires
- Policy
- Signature
- Key-Pair-Id

CloudFront 假設包含任何這些查詢字串參數的 URL 都是簽署的 URL，因此不會查看已簽署的 Cookie。

使用已簽署 URL 和已簽署 Cookie

已簽署的網址優先於已簽署的 Cookie。如果您同時使用已簽署的 URL 和已簽署的 Cookie 來控制對相同檔案的存取，並且檢視器使用已簽署的 URL 來請求檔案，CloudFront 將判斷是否只根據已簽署的 URL 將該檔案傳回給檢視器。

使用已簽署 URL

已簽章的 URL 包含附加資訊，例如到期日期和時間，以便您更有效地控制對內容的存取。此附加資訊顯示在政策聲明中，該政策聲明基於標準政策或自訂政策。標準和自訂政策之間的差異將在接下來的兩節中說明。

Note

您可使用標準政策建立一些簽章的 URL，並使用自訂政策為相同的分佈建立一些簽章的 URL。

主題

- [決定對已簽署 URL 使用標準或自訂政策](#)
- [已簽署 URL 的工作方式](#)
- [決定已簽署 URL 的有效時間](#)
- [CloudFront 何時檢查已簽署 URL 中的到期日期和時間](#)
- [範例程式碼和第三方工具](#)
- [使用標準政策建立已簽署 URL](#)
- [使用自訂政策建立已簽署 URL](#)

決定對已簽署 URL 使用標準或自訂政策

當您建立已簽章的 URL 時，您將編寫一個 JSON 格式的政策聲明來指定對已簽章的 URL 的限制，例如 URL 的有效時間。您可以使用標準政策或自訂政策。以下是標準和自訂政策的比較：

描述	標準政策	自訂政策
您可以重複使用多個檔案的政策聲明。要重複使用政策聲明，您必須在 Resource 物件中使用萬用字元。如需詳細資訊，請參閱 您在使用自訂政策的已簽署 URL 政策陳述式中指定的值 。	否	是
您可以指定使用者可以開始存取您的內容的日期和時間。	否	是 (選用)
您可以指定使用者無法再存取您的內容的日期和時間。	是	是
您可以指定可以存取您的內容的使用者的 IP 位址或 IP 位址範圍。	否	是 (選用)
該已簽章的 URL 包含該政策的 base64 編碼版本，這會導致較長的 URL。	否	是

如需使用標準政策建立已簽署的 URL 的詳細資訊，請參閱 [使用標準政策建立已簽署 URL](#)。

如需使用自訂政策建立已簽署的 URL 的詳細資訊，請參閱 [使用自訂政策建立已簽署 URL](#)。

已簽署 URL 的工作方式

以下概述您如何為已簽署的 URL 設定 CloudFront 和 Amazon S3，以及當使用者使用已簽署的 URL 要求檔案時，CloudFront 如何回應。

1. 在您的 CloudFront 分佈中，指定一或多個信任的金鑰群組，其中包含 CloudFront 可用來驗證 URL 簽章的公有金鑰。您可以使用對應的私有金鑰來簽署 URL。

CloudFront 支援具有 RSA 2048 和 ECDSA 256 金鑰簽章的已簽署 URL。

如需詳細資訊，請參閱[指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

2. 開發應用程式以判斷使用者是否應該有權存取您的內容，並為您所要限制存取的應用程式的檔案或某些部分建立的簽署 URL。如需詳細資訊，請參閱下列主題：

- [使用標準政策建立已簽署 URL](#)
- [使用自訂政策建立已簽署 URL](#)

3. 使用者請求檔案用於所要請求的簽章 URL。
4. 您的應用程式會驗證使用者是否有權存取檔案：他們已經登入，他們已經支付存取內容的費用，或者他們已經滿足其他一些存取要求。
5. 您的應用程式會建立和傳回已簽章的 URL 給使用者。
6. 已簽章的 URL 允許使用者下載或串流內容。

這個步驟是自動的；使用者通常不需要執行任何額外操作來存取內容。例如，如果使用者在 Web 瀏覽器中存取您的內容時，則應用程式會將已簽章的 URL 傳回到瀏覽器。瀏覽器立即使用已簽署的 URL 存取 CloudFront 邊緣快取中的檔案，無需使用者的任何干預。

7. CloudFront 使用公有金鑰來驗證簽章，並確認 URL 尚未遭到篡改。如果簽章無效，請求會遭到拒絕。

如果簽章有效，CloudFront 將查看 URL 中的政策聲明 (或如果使用的是標準政策，則建構一個聲明) 以確認請求仍然有效。例如，如果您為 URL 指定了開始和結束日期和時間，CloudFront 會確認使用者在您希望允許存取的時間段內嘗試存取您的內容。

如果請求符合政策聲明中的請求，CloudFront 會執行標準操作：決定檔案是否已經在邊緣上快取，如果需要，將請求轉傳到原始伺服器，並將該檔案傳回給使用者。

 Note

如果不帶正負號的 URL 包含查詢字串參數，請務必將這些參數包含在您已簽署 URL 部分中。如果在簽署已簽署 URL 之後，將查詢字串新增此 URL，則 URL 會傳回 HTTP 403 狀態。

決定已簽署 URL 的有效時間

您可以分配私有內容以僅在短時間內使用已簽章的有效 URL (可能只有區區幾分鐘)。有效期很短的簽章 URL 適合於為了特定目的而將內容即時分佈給使用者，例如按需要對客戶作電影租賃或音樂下載的分佈。如果已簽章的 URL 僅在短時間內有效，您可能會希望使用您開發的應用程式自動產生他們。當使用者開始下載檔案或開始播放媒體檔案時，CloudFront 會將 URL 中的到期時間與目前時間進行比較，以判斷 URL 是否仍然有效。

您還可以使用有效期較長的 (可能會持續數年) 已簽章的 URL 分配私有內容。有效期較長的已簽章的 URL 對於向已知使用者分佈私有內容非常有用，例如向投資者分佈業務計畫或向員工分佈培訓教材。您可以開發應用程式，為您產生這些長期簽署的 URL。

CloudFront 何時檢查已簽署 URL 中的到期日期和時間

CloudFront 會在 HTTP 請求時，在已簽署 URL 中檢查過期日期和時間。如果用戶端在到期前一刻才開始下載大型檔案，則即使在下載期間過期了，下載也應該要完成。如果 TCP 連線中斷並且用戶端在到期時間過後嘗試重新啟動下載，則下載將失敗。

如果用戶端使用範圍 GET 以取得較小型的檔案，則到期時間過後發生的任何 GET 請求都將失敗。如需範圍 GET 的詳細資訊，請參閱 [CloudFront 如何處理物件的部分請求 \(範圍 GET\)](#)。

範例程式碼和第三方工具

如需建立已簽署 URL 的雜湊和已簽署部分的範例程式碼，請參閱下列主題：

- [使用 Perl 建立 URL 簽章](#)
- [使用 PHP 建立 URL 簽章](#)
- [使用 C# 和 .NET 架構建立 URL 簽章](#)
- [使用 Java 建立 URL 簽章](#)

使用標準政策建立已簽署 URL

若要使用標準政策建立已簽署 URL，請完成以下步驟。

使用標準政策建立簽章的 URL

1. 如果您使用 .NET 或 Java 建立簽章的 URL，並且您還沒有將金鑰對的私有金鑰從預設的 .pem 格式重新格式化為與 .NET 或 Java 相容的格式，則現在執行此操作。如需詳細資訊，請參閱[重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)。
2. 串連下列值。您可以在已簽署 URL 範例中使用格式。

```
https://d111111abcdef8.cloudfront.net/  
image.jpg?color=red&size=medium&Expires=1767290400&Signature=nitfHRCrtziw02HwPfWw~yYDhUF5Ew  
j19DzZrvDh6hQ73lDx~-ar3UocvvRQVw6EkC~GdpGQyy0SKQim-  
TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkytL6f3fVYNGQI6&Key-  
Pair-Id=K2JCJMDEHXQW5F
```

移除所有空格 (包括標籤和換行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。所有值都有一個 String 類型。

1. ##### URL

如果您未使用簽署的 URL，包括您自我的查詢字串參數 (如果有)，則基本 URL 是您用來存取檔案的 CloudFront URL。在上述範例中，基本 URL 為 `https://d111111abcdef8.cloudfront.net/image.jpg`。如需有關分佈的 URL 格式的詳細資訊，請參閱 [自訂 CloudFront 中檔案的 URL 格式](#)。

- 下列 CloudFront URL 適用於分佈中的映像檔案 (使用 CloudFront 網域名稱)。請注意，`image.jpg` 位於 `images` 目錄中。在 URL 中檔案的路徑必須與 HTTP 伺服器或 Amazon S3 儲存貯體中的檔案的路徑相符。

```
https://d111111abcdef8.cloudfront.net/images/image.jpg
```

- 下列 CloudFront URL 包含查詢字串：

```
https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large
```

- 下列 CloudFront URL 適用於分佈中的圖片檔案。同時使用替代網域名稱。第二個包含查詢字串：

```
https://www.example.com/images/image.jpg
```

```
https://www.example.com/images/image.jpg?color=red
```

- 以下 CloudFront URL 用於分佈中使用備用網域名稱和 HTTPS 通訊協定的圖片檔案：

```
https://www.example.com/images/image.jpg
```

2. ?

? 代表基本 URL 後面所接的查詢字串參數。即使您未指定任何查詢參數，也請包含 ?。

Note

您可以依任何順序指定下列查詢參數。

3. ##### (#####)&

(選用) 您可以輸入自己的查詢字串參數。若要這樣做，請在每個參數之間新增 and (&)，例如 color=red&size=medium。您可以在 URL 內以任何順序指定查詢字串參數。

Important

您的查詢字串參數無法命名為 Expires、Signature 或 Key-Pair-Id。

4. Expires=**Unix ##### (#####) ##### (UTC)**

您希望 URL 停止允許存取檔案的日期和時間。

以 Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 指定過期日期和時間。例如，2026 年 1 月 1 日上午 10:00 UTC 會以 Unix 時間格式轉換為 1767290400，如本主題開頭的範例所示。

若要使用 epoch 時間，請為不晚於 9223372036854775807 的日期指定 64 位元整數 (2262 年 4 月 11 日星期五 23:47:16.854 UTC)。

如需世界協調時間的詳細資訊，請參閱 [RFC 3339、網際網路上的日期和時間：時間戳記](#)。

5. &Signature=#####

JSON 政策聲明的雜湊、簽章和 base64-encoded 版本。如需詳細資訊，請參閱 [為使用標準政策的已簽署 URL 建立簽章](#)。

6. &Key-Pair-Id=**CloudFront ##### ID#####**

CloudFront 公有金鑰的 ID，例如 K2JCJMDEHXQW5F。公有金鑰 ID 會告訴 CloudFront 要使用哪個公有金鑰來驗證已簽署的 URL。CloudFront 將簽章中的資訊與政策聲明中的資訊進行比較，以驗證該 URL 尚未遭到篡改。

此公有金鑰必須屬於分佈中信任的簽署者金鑰群組。如需詳細資訊，請參閱[指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

為使用標準政策的已簽署 URL 建立簽章

若要為使用標準政策的已簽署 URL 建立簽章，請完成下列程序。

主題

- [為使用標準政策的已簽署 URL 建立政策陳述式](#)
- [為使用標準政策的已簽署 URL 建立簽章](#)

為使用標準政策的已簽署 URL 建立政策陳述式

當您使用標準政策建立已簽章的 URL 時，Signature 參數是政策聲明的雜湊和簽章版本。對於使用標準政策的簽章的 URL，您不會將政策聲明包括在 URL 中，就像您使用自訂政策的已簽章的 URL 一樣。若要建立政策聲明，請執行下列程序。

為使用標準政策的已簽章的 URL 建立政策聲明

1. 建構政策聲明，使用下列 JSON 格式，並使用 UTF-8 字元編碼。完全按照規定包含所有標點符號和其他常值。如需有關 Resource 和 DateLessThan 參數的詳細資訊，請參閱[您在使用標準政策的已簽署 URL 的政策陳述式中指定的值](#)。

```
{
  "Statement": [
    {
      "Resource": "base URL or stream name",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": ending date and time in Unix time format and
          UTC
        }
      }
    }
  ]
}
```

```
]
}
```

2. 從政策陳述式中刪除所有空格 (包括標籤和新行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。

您在使用標準政策的已簽署 URL 的政策陳述式中指定的值

當您為標準政策建立政策聲明時，您可以指定以下值。

資源

Note

您只能為 Resource 指定一個值。

基本 URL 包括查詢字串 (如果有)，但不包括 CloudFront Expires、Signature 和 Key-Pair-Id 參數，例如：

```
https://d1111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

注意下列事項：

- 通訊協定 – 此值必須以 http:// 或 https:// 開頭。
- 查詢字串參數 – 如果沒有查詢字串參數，請省略問號。
- 替代網域名稱 – 如果在 URL 中指定了替代網域名稱 (CNAME)，則在引用網頁或應用程式中的檔案時，必須指定替代網域名稱。請勿為物件指定 Amazon S3 URL。

DateLessThan

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的到期日期和時間。例如，2026 年 1 月 1 日上午 10:00 UTC 以 Unix 時間格式轉換為 1767290400。

此值必須與已簽章的 URL 中的 Expires 查詢字串參數的值相符。不要將值括在引號中。

如需詳細資訊，請參閱 [CloudFront 何時檢查已簽署 URL 中的到期日期和時間](#)。

使用標準政策的已簽署 URL 範例政策陳述式

當您在已簽署 URL 中使用以下範例政策陳述式時，使用者在到 UTC 2026 年 1 月 1 日上午 10:00 之前可以存取檔案 `https://d111111abcdef8.cloudfront.net/horizon.jpg`：

```
{
  "Statement": [
    {
      "Resource": "https://d111111abcdef8.cloudfront.net/horizon.jpg?size=large&license=yes",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": 1767290400
        }
      }
    }
  ]
}
```

為使用標準政策的已簽署 URL 建立簽章

若要在已簽章的 URL 中建立 Signature 參數的值，請對您在 [為使用標準政策的已簽署 URL 建立政策陳述式](#) 中建立的政策聲明進行雜湊和簽署。

如需有關如何對政策聲明進行雜湊、簽章和編碼的詳細資訊和範例，請參閱：

- [用於 Base64 編碼和加密的 Linux 命令和 OpenSSL](#)
- [為已簽署 URL 建立簽章的程式碼範例](#)

選項 1：使用標準政策建立簽章

1. 使用 SHA-1 雜湊函數和產生的 RSA 或 ECDSA 私有金鑰對您在 [為使用標準政策的已簽章的 URL 建立政策聲明](#) 程序中建立的政策陳述式進行雜湊和簽署。使用不再包含空格之政策陳述式的版本。

對於雜湊函數所需的私有金鑰，使用其公有金鑰在活動的信任金鑰組中的私有金鑰進行分佈。

Note

用於雜湊和簽名政策聲明的方法取決於您的程式設計語言和平台。如需程式碼範例，請參閱 [為已簽署 URL 建立簽章的程式碼範例](#)。

2. 從雜湊和已簽署字串中移除所有空格 (包括索引標籤和新行字元)。
3. 使用 MIME base64 編碼的 Base64-encode 字串。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的 [第 6.8 節：Base64 Content-Transfer-Encoding](#)。
4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

5. 在 &Signature= 之後，將結果值附加到已簽章的 URL，然後返回到 [使用標準政策建立簽章的 URL](#) 以完成已簽章的 URL 的各個部分的連接。

使用自訂政策建立已簽署 URL

若要使用自訂政策建立簽章的 URL，請完成以下程序。

使用自訂政策建立簽章的 URL

1. 如果您使用 .NET 或 Java 建立簽章的 URL，並且您還沒有將金鑰對的私有金鑰從預設的 .pem 格式重新格式化為與 .NET 或 Java 相容的格式，則現在執行此操作。如需詳細資訊，請參閱 [重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)。
2. 串連下列值。您可以在已簽署 URL 範例中使用格式。

```
https://d1111111abcdef8.cloudfront.net/
image.jpg?color=red&size=medium&Policy=eyJANCIAGICEXAMPLEW1bnQiOiBbeyANCiAGICAgICJSZXNvdXJj
j19DzZrvDh6hQ73LDx~-ar3UocvvRQVw6EkC~GdpGQyy0SKQim-
```

```
TxAw7d8F5Kkai9HVx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkYtL6f3fVYNGQI6&Key-Pair-Id=K2JCJMDEHXQW5F
```

移除所有空格 (包括標籤和換行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。所有值都有一個 String 類型。

1. ##### URL

如果您未使用簽署的 URL，包括您自我的查詢字串參數 (如果有)，則基本 URL 是您用來存取檔案的 CloudFront URL。在上述範例中，基本 URL 為 `https://d111111abcdef8.cloudfront.net/image.jpg`。如需有關分佈的 URL 格式的詳細資訊，請參閱 [自訂 CloudFront 中檔案的 URL 格式](#)。

以下範例說明您為分佈指定的值。

- 下列 CloudFront URL 適用於分佈中的映像檔案 (使用 CloudFront 網域名稱)。請注意，`image.jpg` 位於 `images` 目錄中。在 URL 中檔案的路徑必須與 HTTP 伺服器或 Amazon S3 儲存貯體中的檔案的路徑相符。

```
https://d111111abcdef8.cloudfront.net/images/image.jpg
```

- 下列 CloudFront URL 包含查詢字串：

```
https://d111111abcdef8.cloudfront.net/images/image.jpg?size=large
```

- 下列 CloudFront URL 適用於分佈中的圖片檔案。兩種都使用備用網域名稱，第二個包含查詢字串：

```
https://www.example.com/images/image.jpg
```

```
https://www.example.com/images/image.jpg?color=red
```

- 以下 CloudFront URL 用於分佈中使用備用網域名稱和 HTTPS 通訊協定的圖片檔案：

```
https://www.example.com/images/image.jpg
```

2. ?

? 代表基本 URL 後面所接的查詢字串參數。即使您未指定任何查詢參數，也請包含 ?。

Note

您可以依任何順序指定下列查詢參數。

3. ##### (#####)&

(選用) 您可以輸入自己的查詢字串參數。若要這樣做，請在每個參數之間新增 and (&)，例如 color=red&size=medium。您可以在 URL 內以任何順序指定查詢字串參數。

Important

您的查詢字串參數無法命名為 Policy、Signature 或 Key-Pair-Id。

如果加入自己的參數，請在每個參數的後面附加一個 &，包括最後一個參數。

4. Policy=**base64 #####**

您的政策陳述式採用 JSON 格式，刪除了空格，然後使用 base64 編碼。如需詳細資訊，請參閱[為使用自訂政策的已簽署 URL 建立政策陳述式](#)。

政策陳述式可控制簽署的 URL 授予使用者的存取權限。它包括檔案的 URL、到期日期和時間、URL 變成有效的選擇性日期和時間，以及允許存取檔案的選用 IP 位址或 IP 位址範圍。

5. &Signature=#####

JSON 政策聲明的雜湊、簽章和 base64-encoded 版本。如需詳細資訊，請參閱[為使用自訂政策的已簽署 URL 建立簽章](#)。

6. &Key-Pair-Id=**CloudFront ##### ID#####**

CloudFront 公有金鑰的 ID，例如 K2JCJMDEHXQW5F。公有金鑰 ID 會告訴 CloudFront 要使用哪個公有金鑰來驗證已簽署的 URL。CloudFront 將簽章中的資訊與政策聲明中的資訊進行比較，以驗證該 URL 尚未遭到篡改。

此公有金鑰必須屬於分佈中信任的簽署者金鑰群組。如需詳細資訊，請參閱[指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

為使用自訂政策的已簽署 URL 建立政策陳述式

請完成以下步驟，以為使用自訂政策之簽署的 URL 建立政策陳述式。

如需以各種方式控制對檔案存取的政策聲明範例，請參閱 [the section called “使用自訂政策的已簽署 URL 範例政策陳述式”](#)。

為使用自訂政策的已簽章的 URL 建立政策聲明

1. 使用下列 JSON 格式建構政策聲明。使用您自己的值取代小於 (<) 和大於 (>) 符號及其中的描述。如需詳細資訊，請參閱[the section called “您在使用自訂政策的已簽署 URL 政策陳述式中指定的值”](#)。

```
{
  "Statement": [
    {
      "Resource": "<Optional but recommended: URL of the file>",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": <Required: ending date and time in Unix time
format and UTC>
        },
        "DateGreaterThan": {
          "AWS:EpochTime": <Optional: beginning date and time in Unix time
format and UTC>
        },
        "IpAddress": {
          "AWS:SourceIp": "<Optional: IP address>"
        }
      }
    }
  ]
}
```

注意下列事項：

- 您只能在政策中包含一個陳述式。
 - 使用 UTF-8 字元編碼。
 - 完全按照規定包含所有標點符號和參數名稱。不接受參數名稱的縮寫。
 - Condition 部分的參數順序不重要。
 - 如需有關 Resource、DateLessThan、DateGreaterThan 和 IpAddress 值的詳細資訊，請參閱 [the section called “您在使用自訂政策的已簽署 URL 政策陳述式中指定的值”](#)。
2. 從政策陳述式中刪除所有空格 (包括標籤和新行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。

3. 使用 MIME base64 編碼 Base64-encode 政策聲明。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的 [第 6.8 節：Base64 Content-Transfer-Encoding](#)。
4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

5. 在 Policy= 之後，將結果值附加到已簽章的 URL。
6. 透過政策聲明進行雜湊、簽名和 base64-encoding 來建立用於簽章 URL 的簽章。如需詳細資訊，請參閱 [the section called “為使用自訂準政策的已簽署 URL 建立簽章”](#)。

您在使用自訂政策的已簽署 URL 政策陳述式中指定的值

當您為自訂政策建立政策聲明時，您可以指定以下值。

資源

包括任何查詢字串，但不包括 CloudFront Policy、Signature 和 Key-Pair-Id 參數的 URL。
例如：

```
https://d1111111abcdef8.cloudfront.net/images/horizon.jpg?size=large&license=yes
```

您只能為 Resource 指定一個 URL 值。

Important

您可以省略政策中的 Resource 參數，但如此便代表任何擁有已簽署 URL 的人，都可以存取與您建立已簽署 URL 的金鑰對關聯的任何分佈中的所有檔案。

注意下列事項：

- 通訊協定 – 此值必須以 `http://`、`https://` 或 `*://` 開頭。
- 查詢字串參數 – 如果 URL 具有查詢字串參數，請勿使用反斜線字元 (\) 來逸出開始查詢字串的問題字元 (?)。例如：

```
https://d1111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

- 萬用字元 – 您可以在政策中的 URL 中使用萬用字元。支援下列萬用字元：
 - 星號 (*)，符合零或更多字元
 - 問號 (?)，剛好符合一個字元

在 CloudFront 將政策中的 URL 與 HTTP 要求中的 URL 相符時，政策中的 URL 會分為四個區段 - 通訊協定、網域、路徑和查詢字串)，如下所示：

```
[protocol]://[domain]/[path]\?[query string]
```

當您在政策的 URL 中使用萬用字元時，萬用字元比對只會套用在包含萬用字元的區段邊界內。例如，在政策中考量此 URL：

```
https://www.example.com/hello*world
```

在此範例中，星號萬用字元 (*) 僅適用於路徑區段中，因此它符合 URL `https://www.example.com/helloworld` 和 `https://www.example.com/hello-world`，但與 URL `https://www.example.net/hello?world` 不相符。

下列例外適用於萬用字元相符的區段邊界：

- 路徑區段中的結尾星號表示查詢字串區段中的星號。例如，`http://example.com/hello*` 等同於 `http://example.com/hello*\?*`。
- 網域區段域部分中的結尾星號表示路徑和查詢字串部分中都有星號。例如，`http://example.com*` 等同於 `http://example.com*/*\?*`。
- 政策中的 URL 可以省略通訊協定區段，並在網域區段中以星號開頭。在這種情況下，通訊協定部分暗中設為星號。例如，策略 `*example.com` 中的 URL 等同於 `*://example.com/`。
- 星號本身 ("Resource": "*") 符合任何 URL。

例如，策略 `https://d1111111abcdef8.cloudfront.net/*game_download.zip*` 中的值符合下列所有 URL：

- `https://d1111111abcdef8.cloudfront.net/game_download.zip`

- `https://d111111abcdef8.cloudfront.net/example_game_download.zip?license=yes`
- `https://d111111abcdef8.cloudfront.net/test_game_download.zip?license=temp`
- 替代網域名稱 – 如果在 URL 中指定了替代網域名稱 (CNAME)，則在引用網頁或應用程式中的檔案時，必須指定替代網域名稱。請勿在政策中為檔案指定 Amazon S3 URL。

DateLessThan

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的到期日期和時間。在政策中，不要將值括在引號中。如需世界協調時間的詳細資訊，請參閱[網際網路上的日期和時間：時間戳記](#)。

例如，2023 年 1 月 31 日上午 10:00 UTC 以 Unix 時間格式轉換為 1675159200。

這是 Condition 部分中唯一需要的參數。CloudFront 請求這個值，以防止使用者永久存取您的私有內容。

如需詳細資訊，請參閱[the section called “CloudFront 何時檢查已簽署 URL 中的到期日期和時間”](#)

DateGreaterThan (選用)

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的選用日期和時間。不允許使用者在指定的日期和時間之前存取檔案。不要將值括在引號中。

IpAddress (選用)

提出 HTTP 請求的用戶端 IP 位址。注意下列事項：

- 要允許任何 IP 位址存取該檔案，請省略 IpAddress 參數。
- 您可以指定一個 IP 位址或一個 IP 位址範圍。如果用戶端的 IP 位址位於兩個不同的範圍之一，則無法設定政策以允許存取。
- 若要允許從單一 IP 位址存取，您需要指定：

"IPv4 IP ##/32"

- 必須以標準 IPv4 CIDR 格式指定 IP 位址範圍 (例如 192.0.2.0/24)。如需詳細資訊，請參閱[《無類別域間路由 \(CIDR\)：網際網路地址指派和彙總計劃》](#)。

Important

不支援 IPv6 格式的 IP 位址，例如 2001:0db8:85a3::8a2e:0370:7334。

如果您使用的是包括 `IpAddress` 的自訂政策，請不要為分佈啟用 IPv6。如果您希望透過 IP 位址限制對某些內容的存取，並支援對其他內容的 IPv6 請求，則可以建立兩個分佈。如需詳細資訊，請參閱 [the section called “所有分佈設定”](#) 主題中的 [the section called “啟用 IPv6 \(檢視器請求\)”](#)。

使用自訂政策的已簽署 URL 範例政策陳述式

以下範例說明政策聲明說明了如何控制對特定檔案的存取、目錄中的所有檔案，或與金鑰對 ID 相關聯的所有檔案。此範例還會說明了如何控制來自個別 IP 位址或各種 IP 位址的存取，以及如何防止使用者在指定的日期和時間過期後使用簽章 URL。

如果複製並貼上這些範例中的任何一項，請刪除所有空格 (包括索引標籤和換行符號字元)、用您自我的值取代該值，並在大括弧 ({}) 的後面加上換行字元。

如需詳細資訊，請參閱[the section called “您在使用自訂政策的已簽署 URL 政策陳述式中指定的值”](#)。

主題

- [範例政策陳述式：從一個 IP 位址範圍存取一個檔案](#)
- [範例政策陳述式：從一個 IP 位址範圍存取目錄中的所有檔案](#)
- [範例政策陳述式：從一個 IP 位址存取與金鑰對 ID 相關的所有檔案](#)

範例政策陳述式：從一個 IP 位址範圍存取一個檔案

以下範例自訂政策在已簽署的 URL 中指定使用者在到 UTC 2023 年 1 月 31 日上午 10:00 之前可以從 192.0.2.0/24 範圍內的 IP 位址存取檔案 `https://d111111abcdef8.cloudfront.net/game_download.zip`：

```
{
  "Statement": [
    {
      "Resource": "https://d111111abcdef8.cloudfront.net/game_download.zip",
      "Condition": {
        "IpAddress": {
          "AWS:SourceIp": "192.0.2.0/24"
        },
        "DateLessThan": {
          "AWS:EpochTime": 1675159200
        }
      }
    }
  ]
}
```

```

    }
  }
}
]
}

```

範例政策陳述式：從一個 IP 位址範圍存取目錄中的所有檔案

以下範例自訂政策允許您為 training 目錄中的任何檔案建立簽章的 URL，如 Resource 參數中的 (*) 萬用字元所示。使用者在到 UTC 2023 年 1 月 31 日上午 10:00 之前可以從 192.0.2.0/24 範圍內的 IP 位址存取該檔案：

```

{
  "Statement": [
    {
      "Resource": "https://d111111abcdef8.cloudfront.net/training/*",
      "Condition": {
        "IpAddress": {
          "AWS:SourceIp": "192.0.2.0/24"
        },
        "DateLessThan": {
          "AWS:EpochTime": 1675159200
        }
      }
    }
  ]
}

```

您使用此政策的每個已簽章的 URL 都包括一個識別特定的檔案的基本 URL，例如：

<https://d111111abcdef8.cloudfront.net/training/orientation.pdf>

範例政策陳述式：從一個 IP 位址存取與金鑰對 ID 相關的所有檔案

以下範例自訂政策允許您為與任何分佈關聯的任何檔案建立簽章的 URL，如 Resource 參數中的 (*) 萬用字元所示。已簽署的 URL 必須使用 https:// 通訊協定，而不是 http://。使用者必須使用 IP 位址 192.0.2.10/32。(CIDR 表示法中的 192.0.2.10/32 值是指單個 IP 位址 192.0.2.10。) 這些檔案只能在 2023 年 1 月 31 日上午 10:00 UTC 至 2023 年 2 月 2 日上午 10:00 UTC 之間使用：

```

{
  "Statement": [
    {

```

```
    "Resource": "https://*",
    "Condition": {
      "IpAddress": {
        "AWS:SourceIp": "192.0.2.10/32"
      },
      "DateGreaterThan": {
        "AWS:EpochTime": 1675159200
      },
      "DateLessThan": {
        "AWS:EpochTime": 1675332000
      }
    }
  }
]
```

您使用此政策的每個已簽署的 URL 都包括一個基本 URL，用於識別特定 CloudFront 分佈中的特定檔案，例如：

`https://d1111111abcdef8.cloudfront.net/training/orientation.pdf`

該已簽署 URL 還包括金鑰對 ID，必須與在基本 URL 中指定的分佈 (d1111111abcdef8.cloudfront.net) 中的可信金鑰群組相關聯。

為使用自訂政策的已簽署 URL 建立簽章

使用自訂政策的已簽章的 URL 的簽章是政策聲明的雜湊、簽章和 base64-encoded 版本。若要建立自訂政策的簽章，請完成以下步驟。

如需有關如何對政策聲明進行雜湊、簽章和編碼的詳細資訊和範例，請參閱：

- [用於 Base64 編碼和加密的 Linux 命令和 OpenSSL](#)
- [為已簽署 URL 建立簽章的程式碼範例](#)

選項 1：使用自訂政策建立簽章

1. 使用 SHA-1 雜湊函數和產生的 RSA 或 ECDSA 私有金鑰對您在 [為使用自訂政策的已簽章的 URL 建立政策聲明](#) 程序中建立的 JSON 政策陳述式進行雜湊和簽署。使用不再包含空格但尚未進行 base64 編碼的政策陳述式版本。

對於雜湊函數所需的私有金鑰，使用其公有金鑰在活動的信任金鑰組中的私有金鑰進行分佈。

Note

用於雜湊和簽名政策聲明的方法取決於您的程式設計語言和平台。如需程式碼範例，請參閱 [為已簽署 URL 建立簽章的程式碼範例](#)。

2. 從雜湊和已簽署字串中移除所有空格 (包括索引標籤和新行字元)。
3. 使用 MIME base64 編碼的 Base64-encode 字串。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的 [第 6.8 節：Base64 Content-Transfer-Encoding](#)。
4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

5. 在 &Signature= 之後，將結果值附加到已簽章的 URL，然後返回到 [使用自訂政策建立簽章的 URL](#) 以完成已簽章的 URL 的各個部分的連接。

使用已簽署 Cookie

當您不想變更目前 URL 或希望提供存取多個限制檔案 (例如，網站的訂閱者區域中的所有檔案) 時，CloudFront 簽署 Cookie 允許您控制那些可以存取您的內容的使用者。本主題說明使用已簽章的 Cookie 時的注意事項，並介紹如何使用標準和自訂政策設定已簽章的 Cookie。

主題

- [決定對已簽署 Cookie 使用標準或自訂政策](#)
- [已簽署 Cookie 的工作方式](#)
- [防止濫用已簽署 Cookie](#)
- [CloudFront 何時檢查已簽署 Cookie 中的到期日期和時間](#)
- [範例程式碼和第三方工具](#)
- [使用標準政策設定已簽署 Cookie](#)

- [使用自訂政策設定已簽署 Cookie](#)
- [使用 PHP 建立已簽署 Cookie](#)

決定對已簽署 Cookie 使用標準或自訂政策

當您建立已簽章的 Cookie 時，您將編寫一個 JSON 格式的政策聲明來指定對已簽章的 Cookie 的限制，例如 Cookie 的有效時間。您可以使用標準政策或自訂政策。下表比較了標準和自訂政策：

描述	標準政策	自訂政策
您可以重複使用多個檔案的政策聲明。要重複使用政策聲明，您必須在 Resource 物件中使用萬用字元。如需詳細資訊，請參閱 您在政策陳述式中為已簽章 Cookie 的自訂政策指定的值 。	否	是
您可以指定使用者可以開始存取您的內容的日期和時間	否	是 (選用)
您可以指定使用者無法再存取您的內容的日期和時間	是	是
您可以指定可以存取您的內容的使用者的 IP 位址或 IP 位址範圍	否	是 (選用)

如需有關使用標準政策建立已簽章的 Cookie 的詳細資訊，請參閱 [使用標準政策設定已簽署 Cookie](#)。

如需有關使用自訂政策建立已簽章的 Cookie 的詳細資訊，請參閱 [使用自訂政策設定已簽署 Cookie](#)。

已簽署 Cookie 的工作方式

以下概觀了如何為已簽署的 Cookie 設定 CloudFront，以及當使用者提交包含已簽署的 Cookie 的請求時，CloudFront 如何回應。

1. 在您的 CloudFront 分佈中，指定一或多個信任的金鑰群組，其中包含 CloudFront 可用來驗證 URL 簽章的公有金鑰。您可以使用對應的私有金鑰來簽署 URL。

如需詳細資訊，請參閱 [指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

2. 您開發應用程式，以判斷使用者是否應該存取您的內容，如果是，則向檢視器傳送三個 Set-Cookie 標頭。(每個 Set-Cookie 標頭只包含一個名稱值組，而 CloudFront 已簽署的 Cookie

需要三個名稱值組。) 在瀏覽者請求您的私有內容之前，您必須將 Set-Cookie 標頭傳送到檢視器。如果您在 Cookie 上設定了較短的到期時間，則可能還需要傳送三個 Set-Cookie 標頭以回應後續請求，以便使用者持續存取。

一般而言，您的 CloudFront 分佈至少有兩個快取行為，一種不需要身分驗證，另一種需要驗證。網站安全部分的錯誤頁面包含重定向器或指向登入頁面的連結。

如果您將分佈設定為根據 Cookie 的快取檔案，CloudFront 不會根據已簽署的 Cookie 中的屬性快取個別檔案。

3. 使用者登入您的網站，以及付費內容或滿足其他存取需求。
4. 您的應用程式傳回回應中的 Set-Cookie 標頭，並且檢視器會儲存名稱值組。
5. 使用者請求了檔案。

使用者的瀏覽器或其他檢視器從步驟 4 取得名稱值組，並將它們新增至 Cookie 標頭中的請求中。這是已簽章的 Cookie。

6. CloudFront 使用公有金鑰來驗證已簽署的 Cookie 中的簽章，並確認該 Cookie 尚未遭到篡改。如果簽章無效，請求會遭到拒絕。

如果 Cookie 中的簽章有效，CloudFront 將查看 Cookie 中的政策聲明 (或者如果使用的是標準政策，則建構一個聲明) 以確認請求仍然有效。例如，如果您為 Cookie 指定了開始和結束日期和時間，CloudFront 會確認使用者在您希望允許存取的時間段內嘗試存取您的內容。

如果該請求符合政策聲明中的請求，則 CloudFront 將為您的內容提供服務，就像是不受限制的內容：它決定檔案是否已經在邊緣上快取，若需要，將請求轉傳到原始伺服器，並將該檔案傳回給使用者。

防止濫用已簽署 Cookie

如果您在 Domain 標頭中指定 Set-Cookie 參數，請指定可能的最精確值，以降低具有相同根網域名稱之人員的存取可能性。例如，app.example.com 優於 example.com，尤其是當您無法控制 example.com 時。這有助於防止他人從 www.example.com 存取您的內容。

要協助避免發生這種類型的攻擊，請執行下列動作：

- 排除 Expires 和 Max-Age 的屬性，以便 Set-Cookie 標頭建立工作階段 Cookie。工作階段 Cookie 會在使用者關閉瀏覽器時自動刪除，以降低有人未經授權存取您的內容的可能性。
- 包含 Secure 屬性，以便檢視器在請求中包含該 Cookie 時，將對其進行加密。
- 如果可能，請使用自訂政策，並包含檢視器的 IP 位址。

- 在 CloudFront-Expires 屬性中，根據您希望使用者訪問您的內容的時間長度，指定最短的合理到期時間。

CloudFront 何時檢查已簽署 Cookie 中的到期日期和時間

若要判斷已簽署的 Cookie 是否仍然有效，CloudFront 在 HTTP 請求時會檢查 Cookie 中的到期日期和時間。如果用戶端在到期前一刻才開始下載大型檔案，則即使在下載期間過期了，下載也應該要完成。如果 TCP 連線中斷並且用戶端在到期時間過後嘗試重新啟動下載，則下載將失敗。

如果用戶端使用範圍 GET 以取得較小型的檔案，則到期時間過後發生的任何 GET 請求都將失敗。如需範圍 GET 的詳細資訊，請參閱 [CloudFront 如何處理物件的部分請求 \(範圍 GET\)](#)。

範例程式碼和第三方工具

私有內容的範本程式碼只說明如何為已簽章的 URL 建立簽章。但是，針對已簽章的 Cookie 建立簽章的程序非常類似，因此大部分範本程式碼仍然相關。如需詳細資訊，請參閱下列主題：

- [使用 Perl 建立 URL 簽章](#)
- [使用 PHP 建立 URL 簽章](#)
- [使用 C# 和 .NET 架構建立 URL 簽章](#)
- [使用 Java 建立 URL 簽章](#)

使用標準政策設定已簽署 Cookie

若要使用標準政策設定已簽章的 Cookie，請完成以下步驟。若要建立簽章，請參閱 [為使用標準政策的已簽署 Cookie 建立簽章](#)。

使用標準政策設定已簽章的 Cookie

1. 如果您使用 .NET 或 Java 建立已簽章的 Cookie，而且尚未將金鑰對的私有金鑰從預設的 .pem 格式重新格式化為與 .NET 或 Java 相容的格式，請現在執行此操作。如需詳細資訊，請參閱 [重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)。
2. 請編寫您的應用程式以發送三個 Set-Cookie 標頭給已核准的瀏覽者。您需要三個 Set-Cookie 標頭，因為每個 Set-Cookie 標頭只能包含一個名稱值組，而 CloudFront 已簽署的 Cookie 需要三個名稱值組。名稱值組是：CloudFront-Expires、CloudFront-Signature 和 CloudFront-Key-Pair-Id。在使用者對要控制存取的檔案發出第一次請求之前，這些值必須出現在檢視器。

 Note

一般而言，建議您排除 Expires 和 Max-Age 屬性。排除這些屬性會導致瀏覽器在使用者關閉瀏覽器時刪除 Cookie，從而降低有人未經授權存取您的內容的可能性。如需詳細資訊，請參閱[防止濫用已簽署 Cookie](#)。

Cookie 屬性的名稱區分大小寫。

只包含分行符號，以使屬性更易讀。

```
Set-Cookie:
CloudFront-Expires=date and time in Unix time format (in seconds) and Coordinated
Universal Time (UTC);
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Signature=hashed and signed version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Key-Pair-Id=public key ID for the CloudFront public key whose
corresponding private key you're using to generate the signature;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly
```

(選用) Domain

請求檔案的網域名稱。如果不指定 Domain 屬性，則預設值是 URL 中的網域名稱，並且它僅適用於指定的網域名稱，而不適用於子網域名稱。如果指定 Domain 屬性，它也適用於子網域名稱。網域名稱中的前導點 (例如 Domain=.example.com) 是可選的。此外，如果指定 Domain 屬性，則 URL 中的網域名稱和 Domain 屬性的值必須相符。

您可以指定 CloudFront 指派給您的分佈的網域名稱，例如，d1111111abcdef8.cloudfront.net，但不能為網域名稱指定 *.cloudfront.net。

如果要在 URL 中使用備用網域名稱 (如 example.com)，則無論您是否指定了 Domain 屬性，都必須將備用網域名稱新增到分佈中。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [備用網域名稱 \(CNAME\)](#)。

(選用) Path

請求檔案的路徑。如果未指定 Path 屬性，則預設值為 URL 中的路徑。

Secure

在檢視器傳送請求之前，需要對 Cookie 進行加密。我們建議您透過 HTTPS 連線發送 Set-Cookie 標頭，以確保 Cookie 屬性免受中間人攻擊。

HttpOnly

定義瀏覽器 (在支援的情況下) 如何與 Cookie 值互動。使用 HttpOnly 時，JavaScript 無法存取 Cookie 值。此預防措施有助於緩解跨網站指令碼 (XSS) 攻擊。如需詳細資訊，請參閱 [使用 HTTP cookie](#)。

CloudFront-Expires

以 Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 指定過期日期和時間。例如，2026 年 1 月 1 日上午 10:00 UTC 以 Unix 時間格式轉換為 1767290400。

若要使用 epoch 時間，請為不晚於 9223372036854775807 的日期指定 64 位元整數 (2262 年 4 月 11 日星期五 23:47:16.854 UTC)。

如需世界協調時間的詳細資訊，請參閱 RFC 3339、網際網路上的日期和時間：時間戳記，<https://tools.ietf.org/html/rfc3339>。

CloudFront-Signature

JSON 政策聲明的雜湊、簽章和 base64-encoded 版本。如需詳細資訊，請參閱 [為使用標準政策的已簽署 Cookie 建立簽章](#)。

CloudFront-Key-Pair-Id

CloudFront 公有金鑰的 ID，例如 K2JCJMDEHXQW5F。公有金鑰 ID 會告訴 CloudFront 要使用哪個公有金鑰來驗證已簽署的 URL。CloudFront 將簽章中的資訊與政策聲明中的資訊進行比較，以驗證該 URL 尚未遭到篡改。

此公有金鑰必須屬於分佈中信任的簽署者金鑰群組。如需詳細資訊，請參閱 [指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

以下範例顯示當您使用與檔案的 URL 中的分佈關聯的網域名稱時，一個已簽章的 Cookie 的 Set-Cookie 標頭：

```
Set-Cookie: CloudFront-Expires=1426500000; Domain=d111111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_;
Domain=d111111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;
Domain=d111111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly
```

以下範例顯示當您在檔案的 URL 中使用備用網域名稱 example.org 時，一個已簽章的 Cookie 的 Set-Cookie 標頭：

```
Set-Cookie: CloudFront-Expires=1426500000; Domain=example.org; Path=/images/*; Secure;
HttpOnly
Set-Cookie: CloudFront-Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_; Domain=example.org;
Path=/images/*; Secure; HttpOnly
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/images/*;
Secure; HttpOnly
```

如果要在 URL 中使用備用網域名稱 (如 example.com)，則無論您是否指定了 Domain 屬性，都必須將備用網域名稱新增到分佈中。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [備用網域名稱 \(CNAME\)](#)。

為使用標準政策的已簽署 Cookie 建立簽章

若要為使用標準政策的已簽署 Cookie 建立簽章，請完成下列程序。

主題

- [為使用標準政策的已簽署 Cookie 建立政策陳述式](#)
- [簽署政策陳述式以為使用標準政策的已簽署 Cookie 建立簽章](#)

為使用標準政策的已簽署 Cookie 建立政策陳述式

當您設定使用標準政策的簽章 Cookie 時，該 CloudFront-Signature 屬性是政策聲明的雜湊及已簽章的版本。對於使用標準政策的已簽章的 Cookie，您不會將政策聲明包括在 Set-Cookie 標頭中，就像使用自訂政策的已簽章的 Cookie 一樣。若要建立政策陳述式，請完成下列步驟。

為使用標準政策的已簽章的 Cookie 建立政策聲明

1. 建構政策聲明，使用下列 JSON 格式，並使用 UTF-8 字元編碼。完全按照規定包含所有標點符號和其他常值。如需有關 Resource 和 DateLessThan 參數的詳細資訊，請參閱 [您在政策陳述式中為已簽署 Cookie 標準政策所指定的值](#)。

```
{
  "Statement": [
    {
      "Resource": "base URL or stream name",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": ending date and time in Unix time format and
          UTC
        }
      }
    }
  ]
}
```

2. 從政策陳述式中刪除所有空格 (包括標籤和新行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。

您在政策陳述式中為已簽署 Cookie 標準政策所指定的值

當您為標準政策建立政策聲明時，您可以指定以下值：

資源

包含查詢字串的基本 URL (如果有)，例如：

```
https://d1111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

您只能為 Resource 指定一個值。

注意下列事項：

- 通訊協定 – 此值必須以 `http://` 或 `https://` 開頭。
- 查詢字串參數 – 如果沒有查詢字串參數，請省略問號。
- 替代網域名稱 – 如果在 URL 中指定了替代網域名稱 (CNAME)，則在引用網頁或應用程式中的檔案時，必須指定替代網域名稱。請勿為檔案指定 Amazon S3 URL。

DateLessThan

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的到期日期和時間。不要將值括在引號中。

例如，2015 年 3 月 16 日上午 10:00 UTC 以 Unix 時間格式轉換為 1426500000。

這個值必須符合 CloudFront-Expires 標頭中 Set-Cookie 屬性的值。不要將值括在引號中。

如需詳細資訊，請參閱 [CloudFront 何時檢查已簽署 Cookie 中的到期日期和時間](#)。

標準政策的範例政策陳述式

當您在已簽章的 Cookie 中使用以下範例政策聲明時，使用者在到 UTC 2015 年 3 月 16 日上午 10:00 之前可以存取檔案 `https://d111111abcdef8.cloudfront.net/horizon.jpg`：

```
{
  "Statement": [
    {
      "Resource": "https://d111111abcdef8.cloudfront.net/horizon.jpg?size=large&license=yes",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": 1426500000
        }
      }
    }
  ]
}
```

簽署政策陳述式以為使用標準政策的已簽署 Cookie 建立簽章

要為 CloudFront-Signature 標頭中的 Set-Cookie 屬性建立值，請對您在 [為使用標準政策的已簽章的 Cookie 建立政策聲明](#) 中建立的政策聲明進行雜湊和簽署。

如需有關如何對政策聲明進行雜湊、簽章和編碼的詳細資訊和範例，請參閱以下主題：

- [用於 Base64 編碼和加密的 Linux 命令和 OpenSSL](#)
- [為已簽署 URL 建立簽章的程式碼範例](#)

使用標準政策為已簽章的 Cookie 建立簽章

1. 使用 SHA-1 雜湊函數和 RSA 對您在[為使用標準政策的已簽章的 Cookie 建立政策聲明](#)程序中建立的政策聲明進行雜湊和簽署。使用不再包含空格之政策陳述式的版本。

對於雜湊函數所需的私有金鑰，使用其公有金鑰在活動的信任金鑰組中的私有金鑰進行分佈。

 Note

用於雜湊和簽名政策聲明的方法取決於您的程式設計語言和平台。如需程式碼範例，請參閱 [為已簽署 URL 建立簽章的程式碼範例](#)。

2. 從雜湊和已簽署字串中移除所有空格 (包括索引標籤和換行字元)。
3. 使用 MIME base64 編碼的 Base64-encode 字串。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的[第 6.8 節：Base64 Content-Transfer-Encoding](#)。
4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

5. 將結果值包含在 Set-Cookie 名稱值組的 CloudFront-Signature 標頭中。然後返回到[使用標準政策設定已簽章的 Cookie](#) 為 Set-Cookie 新增 CloudFront-Key-Pair-Id 標頭。

使用自訂政策設定已簽署 Cookie

若要設定使用標準政策的已簽署 Cookie，請完成以下步驟。

使用自訂政策設定已簽章的 Cookie

1. 如果您使用 .NET 或 Java 建立簽章的 URL，並且您還沒有將金鑰對的私有金鑰從預設的 .pem 格式重新格式化為與 .NET 或 Java 相容的格式，則現在執行此操作。如需詳細資訊，請參閱 [重新格式化私有金鑰 \(僅限 .NET 和 Java\)](#)。
2. 請編寫您的應用程式以發送三個 Set-Cookie 標頭給已核准的瀏覽者。您需要三個 Set-Cookie 標頭，因為每個 Set-Cookie 標頭只能包含一個名稱值組，而 CloudFront 已簽署的 Cookie 需要三個名稱值組。名稱值組是：CloudFront-Policy、CloudFront-Signature 和 CloudFront-Key-Pair-Id。在使用者對要控制存取的檔案發出第一次請求之前，這些值必須出現在檢視器。

Note

一般而言，建議您排除 Expires 和 Max-Age 屬性。這會導致瀏覽器在使用者關閉瀏覽器時刪除 Cookie，從而降低有人未經授權存取您的內容的可能性。如需詳細資訊，請參閱 [防止濫用已簽署 Cookie](#)。

Cookie 屬性的名稱區分大小寫。

只包含分行符號，以使屬性更易讀。

```
Set-Cookie:
CloudFront-Policy=base64 encoded version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Signature=hashed and signed version of the policy statement;
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly

Set-Cookie:
CloudFront-Key-Pair-Id=public key ID for the CloudFront public key whose
corresponding private key you're using to generate the signature;
```

```
Domain=optional domain name;  
Path=/optional directory path;  
Secure;  
HttpOnly
```

(選用) Domain

請求檔案的網域名稱。如果不指定 Domain 屬性，則預設值是 URL 中的網域名稱，並且它僅適用於指定的網域名稱，而不適用於子網域名稱。如果指定 Domain 屬性，它也適用於子網域名稱。網域名稱中的前導點 (例如 Domain=.example.com) 是可選的。此外，如果指定 Domain 屬性，則 URL 中的網域名稱和 Domain 屬性的值必須相符。

您可以指定 CloudFront 指派給您的分佈的網域名稱，例如，d111111abcdef8.cloudfront.net，但不能為網域名稱指定 *.cloudfront.net。

如果要在 URL 中使用備用網域名稱 (如 example.com)，則無論您是否指定了 Domain 屬性，都必須將備用網域名稱新增到分佈中。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [備用網域名稱 \(CNAME\)](#)。

(選用) Path

請求檔案的路徑。如果未指定 Path 屬性，則預設值為 URL 中的路徑。

Secure

在檢視器傳送請求之前，需要對 Cookie 進行加密。我們建議您透過 HTTPS 連線發送 Set-Cookie 標頭，以確保 Cookie 屬性免受中間人攻擊。

HttpOnly

要求檢視器僅在 HTTP 或 HTTPS 請求中傳送 Cookie。

CloudFront-Policy

您的政策陳述式採用 JSON 格式，刪除了空格，然後使用 base64 編碼。如需詳細資訊，請參閱 [為使用自訂政策的已簽署 Cookie 建立簽章](#)。

政策陳述式可控制已簽署 Cookie 授予使用者的存取權限。它包括使用者可以存取的檔案、到期日期和時間、URL 變成有效的選擇性日期和時間，以及允許存取檔案的選用 IP 位址或 IP 位址範圍。

CloudFront-Signature

JSON 政策聲明的雜湊、簽章和 base64-encoded 版本。如需詳細資訊，請參閱 [為使用自訂政策的已簽署 Cookie 建立簽章](#)。

CloudFront-Key-Pair-Id

CloudFront 公有金鑰的 ID，例如 K2JCJMDEHXQW5F。公有金鑰 ID 會告訴 CloudFront 要使用哪個公有金鑰來驗證已簽署的 URL。CloudFront 將簽章中的資訊與政策聲明中的資訊進行比較，以驗證該 URL 尚未遭到篡改。

此公有金鑰必須屬於分佈中信任的簽署者金鑰群組。如需詳細資訊，請參閱 [指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

自訂政策的 Set-Cookie 標頭範例

請參閱下列 Set-Cookie 標頭對的範例。

如果要在 URL 中使用備用網域名稱 (如 example.org)，則無論您是否指定了 Domain 屬性，都必須將備用網域名稱新增到分佈中。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [備用網域名稱 \(CNAME\)](#)。

Example 範例 1

當您使用與檔案的 URL 中的分佈關聯的網域名稱時，可以使用一個已簽署 Cookie 的 Set-Cookie 標頭。

```
Set-Cookie: CloudFront-  
Policy=eyJTdGF0ZWl1bnQiOi0t7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh  
Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_  
Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;  
Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
```

Example 範例 2

當您在檔案的 URL 中使用備用網域名稱 (example.org) 時，可以使用一個已簽署 Cookie 的 Set-Cookie 標頭。

```
Set-Cookie: CloudFront-  
Policy=eyJTdGF0ZW1lbnQiOi0lt7IlJlY291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh  
Domain=example.org; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_; Domain=example.org;  
Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/; Secure;  
HttpOnly
```

Example 範例 3

當您使用與檔案的 URL 中的分佈關聯的網域名稱時，可以使用一個已簽署請求的 Set-Cookie 標頭對。

```
Set-Cookie: CloudFront-  
Policy=eyJTdGF0ZW1lbnQiOi0lt7IlJlY291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh  
Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_;  
Domain=d111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F;  
Domain=dd111111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly
```

Example 範例 4

當您使用與檔案的 URL 中的分佈關聯的替代網域名稱 (example.org) 時，可以使用一個已簽署請求的 Set-Cookie 標頭對。

```
Set-Cookie: CloudFront-  
Policy=eyJTdGF0ZW1lbnQiOi0lt7IlJlY291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dh  
Domain=example.org; Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_; Domain=example.org;  
Path=/; Secure; HttpOnly  
Set-Cookie: CloudFront-Key-Pair-Id=K2JCJMDEHXQW5F; Domain=example.org; Path=/; Secure;  
HttpOnly
```

為使用自訂政策的已簽署 Cookie 建立政策陳述式

若要為自訂政策建立政策陳述式，請執行以下步驟。如需以各種方式控制對檔案存取的幾個政策聲明範例，請參閱 [使用自訂政策的已簽署 Cookie 範例政策陳述式](#)。

為使用自訂政策的簽章的 Cookie 建立政策聲明

1. 使用下列 JSON 格式建構政策聲明。

```
{
  "Statement": [
    {
      "Resource": "URL of the file",
      "Condition": {
        "DateLessThan": {
          "AWS:EpochTime": required ending date and time in Unix time
format and UTC
        },
        "DateGreaterThan": {
          "AWS:EpochTime": optional beginning date and time in Unix time
format and UTC
        },
        "IpAddress": {
          "AWS:SourceIp": "optional IP address"
        }
      }
    }
  ]
}
```

注意下列事項：

- 您只能包含一個聲明。
 - 使用 UTF-8 字元編碼。
 - 完全按照規定包含所有標點符號和參數名稱。不接受參數名稱的縮寫。
 - Condition 部分的參數順序不重要。
 - 如需有關 Resource、DateLessThan、DateGreaterThan 和 IpAddress 值的詳細資訊，請參閱 [您在政策陳述式中為已簽章 Cookie 的自訂政策指定的值](#)。
2. 從政策陳述式中刪除所有空格 (包括標籤和新行字元)。您可能必須在應用程式的程式碼的字串中包含逸出字元。
 3. 使用 MIME base64 編碼 Base64-encode 政策聲明。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的 [第 6.8 節：Base64 Content-Transfer-Encoding](#)。
 4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

- 將結果值包含在您的 Set-Cookie 標頭後的 CloudFront-Policy=。
- 透過進行雜湊、簽章和 base64 編碼政策聲明來為 Set-Cookie 建立 CloudFront-Signature 標頭的簽章。如需詳細資訊，請參閱 [為使用自訂政策的已簽署 Cookie 建立簽章](#)。

您在政策陳述式中為已簽章 Cookie 的自訂政策指定的值

當您為自訂政策建立政策聲明時，您可以指定以下值。

資源

包含查詢字串的基本 URL (如果有)：

```
https://d1111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

Important

如果省略 Resource 參數，則使用者可以存取與用於建立已簽章的 URL 的金鑰對相關聯的任何分佈的所有檔案。

您只能為 Resource 指定一個值。

注意下列事項：

- 通訊協定 – 此值必須以 http:// 或 https:// 開頭。
- 查詢字串參數 – 如果沒有查詢字串參數，請省略問號。
- 萬用字元 – 可以使用比對零或多個字元 (*) 的萬用字元，或比對字串中任意位置單一字元 (?) 的萬用字元。例如，值為：

`https://d111111abcdef8.cloudfront.net/*game_download.zip*`

將包括 (例如) 下列檔案：

- `https://d111111abcdef8.cloudfront.net/game_download.zip`
- `https://d111111abcdef8.cloudfront.net/example_game_download.zip?license=yes`
- `https://d111111abcdef8.cloudfront.net/test_game_download.zip?license=temp`
- 替代網域名稱 – 如果在 URL 中指定了替代網域名稱 (CNAME)，則在引用網頁或應用程式中的檔案時，必須指定替代網域名稱。請勿為檔案指定 Amazon S3 URL。

DateLessThan

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的到期日期和時間。不要將值括在引號中。

例如，2015 年 3 月 16 日上午 10:00 UTC 以 Unix 時間格式轉換為 1426500000。

如需詳細資訊，請參閱 [CloudFront 何時檢查已簽署 Cookie 中的到期日期和時間](#)。

DateGreaterThan (選用)

Unix 時間格式 (以秒為單位) 和國際標準時間 (UTC) 的 URL 的選用日期和時間。不允許使用者在指定的日期和時間之前存取檔案。不要將值括在引號中。

IpAddress (選用)

提出 GET 請求的用戶端 IP 位址。注意下列事項：

- 要允許任何 IP 位址存取該檔案，請省略 `IpAddress` 參數。
- 您可以指定一個 IP 位址或一個 IP 位址範圍。例如，如果用戶端的 IP 位址位於兩個不同的範圍之一，則無法設定政策以允許存取。
- 若要允許從單一 IP 位址存取，您需要指定：

`"IPv4 IP ##/32"`

- 必須以標準 IPv4 CIDR 格式指定 IP 位址範圍 (例如 192.0.2.0/24)。如需詳細資訊，請前往 RFC 4632，無類別網域間路由選擇 (CIDR)：網際網路地址指派和彙總計劃，<https://tools.ietf.org/html/rfc4632>。

⚠ Important

不支援 IPv6 格式的 IP 位址，例如 2001:0db8:85a3::8a2e:0370:7334。

如果您使用的是包括 `IpAddress` 的自訂政策，請不要為分佈啟用 IPv6。如果您希望透過 IP 位址限制對某些內容的存取，並支援對其他內容的 IPv6 請求，則可以建立兩個分佈。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [啟用 IPv6 \(檢視器請求\)](#)。

使用自訂政策的已簽署 Cookie 範例政策陳述式

以下範例說明政策聲明說明了如何控制對特定檔案的存取、目錄中的所有檔案，或與金鑰對 ID 相關聯的所有檔案。此範例還會說明如何控制來自個別的 IP 位址或各種 IP 位址的存取，以及如何防止使用者在指定的日期和時間之後使用已簽章的 Cookie。

如果複製並貼上這些範例中的任何一項，請刪除所有空格 (包括索引標籤和換行符號字元)、用您自我的值取代該值，並在大括弧 () 的後面加上換行字元。

如需詳細資訊，請參閱[您在政策陳述式中為已簽章 Cookie 的自訂政策指定的值](#)。

主題

- [範例政策陳述式：從一個 IP 位址範圍存取一個檔案](#)
- [範例政策陳述式：從一個 IP 位址範圍存取目錄中的所有檔案](#)
- [範例政策陳述式：從一個 IP 位址存取與金鑰對 ID 相關的所有檔案](#)

範例政策陳述式：從一個 IP 位址範圍存取一個檔案

以下範例自訂政策在已簽署的 Cookie 中，其指定使用者在 2023 年 1 月 1 日上午 10:00 (UTC) 之前，可以從 192.0.2.0/24 範圍內的 IP 位址存取檔案 `https://d1111111abcdef8.cloudfront.net/game_download.zip`：

```
{
  "Statement": [
    {
      "Resource": "https://d1111111abcdef8.cloudfront.net/game_download.zip",
      "Condition": {
        "IpAddress": {
```

```

        "AWS:SourceIp": "192.0.2.0/24"
      },
      "DateLessThan": {
        "AWS:EpochTime": 1767290400
      }
    }
  ]
}

```

範例政策陳述式：從一個 IP 位址範圍存取目錄中的所有檔案

以下範例自訂政策允許您為 training 目錄中的任何檔案建立已簽章的 Cookie，如 Resource 參數中的 * 萬用字元所示。使用者在到 UTC 2013 年 1 月 1 日上午 10:00 之前可以從 192.0.2.0/24 範圍內的 IP 位址存取該檔案：

```

{
  "Statement": [
    {
      "Resource": "https://d111111abcdef8.cloudfront.net/training/*",
      "Condition": {
        "IpAddress": {
          "AWS:SourceIp": "192.0.2.0/24"
        },
        "DateLessThan": {
          "AWS:EpochTime": 1767290400
        }
      }
    }
  ]
}

```

您使用此政策的每個已簽章的 Cookie 都包括一個識別特定的檔案的基本 URL，例如：

<https://d111111abcdef8.cloudfront.net/training/orientation.pdf>

範例政策陳述式：從一個 IP 位址存取與金鑰對 ID 相關的所有檔案

以下範例自訂政策允許您為與任何分佈關聯的任何檔案設定已簽章的 Cookie，如 Resource 參數中的 * 萬用字元所示。使用者必須使用 IP 位址 192.0.2.10/32。(CIDR 表示法中的 192.0.2.10/32 值是指單個 IP 位址 192.0.2.10。) 這些檔案只能在 2013 年 1 月 1 日上午 10:00 UTC 至 2013 年 1 月 2 日上午 10:00 UTC 之間可用：

```
{
  "Statement": [
    {
      "Resource": "https://*",
      "Condition": {
        "IpAddress": {
          "AWS:SourceIp": "192.0.2.10/32"
        },
        "DateGreaterThan": {
          "AWS:EpochTime": 1767290400
        },
        "DateLessThan": {
          "AWS:EpochTime": 1767376800
        }
      }
    }
  ]
}
```

您使用此政策的每個已簽署的 Cookie 都包括一個基本 URL，用於識別特定 CloudFront 分佈中的特定檔案，例如：

`https://d111111abcdef8.cloudfront.net/training/orientation.pdf`

該已簽署 Cookie 還包括金鑰對 ID，必須與在基本 URL 中指定的分佈 (d111111abcdef8.cloudfront.net) 內信任的簽署者相關聯。

為使用自訂政策的已簽署 Cookie 建立簽章

使用自訂政策的已簽章的 Cookie 的簽章是政策聲明的雜湊、簽章和 base64-encoded 版本。

如需有關如何對政策聲明進行雜湊、簽章和編碼的詳細資訊和範例，請參閱：

- [用於 Base64 編碼和加密的 Linux 命令和 OpenSSL](#)
- [為已簽署 URL 建立簽章的程式碼範例](#)

使用自訂政策為已簽章的 Cookie 建立簽章

1. 使用 SHA-1 雜湊函數和 RSA 對您在[為使用自訂政策的已簽章的 URL 建立政策聲明](#)程序中建立的 JSON 政策聲明進行雜湊和簽署。使用不再包含空格但尚未進行 base64 編碼的政策陳述式版本。

對於雜湊函數所需的私有金鑰，使用其公有金鑰在活動的信任金鑰組中的私有金鑰進行分佈。

Note

用於雜湊和簽名政策聲明的方法取決於您的程式設計語言和平台。如需程式碼範例，請參閱 [為已簽署 URL 建立簽章的程式碼範例](#)。

2. 從雜湊和已簽署字串中移除所有空格 (包括索引標籤和新行字元)。
3. 使用 MIME base64 編碼的 Base64-encode 字串。如需詳細資訊，請參閱 RFC 2045，MIME (多用途網際網路郵件延伸) 第一部分：網際網路訊息內文的格式中的 [第 6.8 節：Base64 Content-Transfer-Encoding](#)。
4. 將 URL 查詢字串中無效的字元替換為有效的字元。下表列出無效和有效的字元。

取代這些無效的字元	有了這些有效的字元
+	- (連字號)
=	_ (底線)
/	~ (波狀符號)

5. 將結果值包含在 Set-Cookie 名稱值組的 CloudFront-Signature= 標頭中，並返回到 [使用自訂政策設定已簽章的 Cookie](#) 為 Set-Cookie 新增 CloudFront-Key-Pair-Id 標頭。

使用 PHP 建立已簽署 Cookie

下列程式碼範例類似於 [使用 PHP 建立 URL 簽章](#) 中建立影片連結的範例。不過，此範例不會在程式碼中簽署 URL，而是使用 `create_signed_cookies()` 函數簽署 Cookie。用戶端播放器使用 Cookie 對 CloudFront 分佈的每個請求進行身分驗證。

此方法有助於串流內容，例如 HTTP Live Streaming (HLS) 或透過 HTTP 的動態自適應串流 (DASH)，其中用戶端需要提出多個請求來擷取資訊清單、區段和相關播放資產。透過使用簽署的 Cookie，用戶端可以驗證每個請求，而不需要為每個區段產生新的已簽署 URL。

 Note

- 建立 URL 簽章只是私有內容提供服務的程式的一部分，以使用已簽署 Cookie。如需詳細資訊，請參閱[使用已簽署 Cookie](#)。

主題

- [建立 RSA SHA-1 簽章](#)
- [建立已簽署 Cookie](#)
- [完整程式碼](#)

下列各節將程式碼範例細分為個別部分。您可以在下面找到完整的[程式碼範例](#)。

建立 RSA SHA-1 簽章

此程式碼會執行下列操作：

1. `rsa_sha1_sign` 函數會雜湊並簽署政策陳述式。所需的引數為政策陳述式，以及與您分佈之信任金鑰群組中公有金鑰對應的私有金鑰。
2. 接著，該 `url_safe_base64_encode` 函數會建立已簽章的 URL 安全版本。

```
function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);
    openssl_sign($policy, $signature, $pkeyid);
    openssl_free_key($pkeyid);
    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}
```

```
}
```

建立已簽署 Cookie

下列程式碼會使用下列 Cookie 屬性來構建簽署的 Cookie：CloudFront-Expires、CloudFront-Signature 和 CloudFront-Key-Pair-Id。程式碼使用自訂政策。

```
function create_signed_cookies($resource, $private_key_filename, $key_pair_id,
    $expires, $client_ip = null) {
    $policy = array(
        'Statement' => array(
            array(
                'Resource' => $resource,
                'Condition' => array(
                    'DateLessThan' => array('AWS:EpochTime' => $expires)
                )
            )
        )
    );

    if ($client_ip) {
        $policy['Statement'][0]['Condition']['IpAddress'] = array('AWS:SourceIp' =>
$client_ip . '/32');
    }

    $policy = json_encode($policy);
    $encoded_policy = url_safe_base64_encode($policy);
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    $encoded_signature = url_safe_base64_encode($signature);

    return array(
        'CloudFront-Policy' => $encoded_policy,
        'CloudFront-Signature' => $encoded_signature,
        'CloudFront-Key-Pair-Id' => $key_pair_id
    );
}
```

如需詳細資訊，請參閱[使用自訂政策設定已簽署 Cookie](#)。

完整程式碼

下面的範例程式碼提供了使用 PHP 建立 CloudFront 已簽署 Cookie 的完整示範。您可以從 [demo-php.zip](#) 檔案下載完整範例。

您可以在下列範例中修改 `$policy` Condition 元素，以允許 IPv4 與 IPv6 位址範圍。如需範例，請參閱《Amazon Simple Storage Service 使用者指南》中的 [在 IAM 政策中使用 IPv6 位址](#)。

```
<?php

function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);
    openssl_sign($policy, $signature, $pkeyid);
    openssl_free_key($pkeyid);
    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}

function create_signed_cookies($resource, $private_key_filename, $key_pair_id,
    $expires, $client_ip = null) {
    $policy = array(
        'Statement' => array(
            array(
                'Resource' => $resource,
                'Condition' => array(
                    'DateLessThan' => array('AWS:EpochTime' => $expires)
                )
            )
        )
    );

    if ($client_ip) {
```

```

        $policy['Statement'][0]['Condition']['IpAddress'] = array('AWS:SourceIp' =>
$client_ip . '/32');
    }

    $policy = json_encode($policy);
    $encoded_policy = url_safe_base64_encode($policy);
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    $encoded_signature = url_safe_base64_encode($signature);

    return array(
        'CloudFront-Policy' => $encoded_policy,
        'CloudFront-Signature' => $encoded_signature,
        'CloudFront-Key-Pair-Id' => $key_pair_id
    );
}

$private_key_filename = '/home/test/secure/example-priv-key.pem';
$key_pair_id = 'K2JCJMDEHXQW5F';
$base_url = 'https://d1234.cloudfront.net';

$expires = time() + 3600; // 1 hour from now

// Get the viewer real IP from the x-forward-for header as $_SERVER['REMOTE_ADDR']
// will return viewer facing IP. An alternative option is to use CloudFront-Viewer-
// Address header. Note that this header is a trusted CloudFront immutable header. Example
// format: IP:PORT ("CloudFront-Viewer-Address": "1.2.3.4:12345")
$client_ip = $_SERVER['HTTP_X_FORWARDED_FOR'];

// For HLS manifest and segments (using wildcard)
$hls_resource = $base_url . '/sign/*';
$signed_cookies = create_signed_cookies($hls_resource, $private_key_filename,
    $key_pair_id, $expires, $client_ip);

// Set the cookies
$cookie_domain = parse_url($base_url, PHP_URL_HOST);
foreach ($signed_cookies as $name => $value) {
    setcookie($name, $value, $expires, '/', $cookie_domain, true, true);
}

?>

```



```
<!DOCTYPE html>
<html>
<head>
  <title>CloudFront Signed HLS Stream with Cookies</title>
</head>
<body>
  <h1>Amazon CloudFront Signed HLS Stream with Cookies</h1>
  <h2>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?> only viewable by IP
  <?php echo $client_ip; ?></h2>

  <div id='hls-video'>
    <video id="video" width="640" height="360" controls></video>
  </div>

  <script src="https://cdn.jsdelivr.net/npm/hls.js@latest"></script>
  <script>
    var video = document.getElementById('video');
    var manifestUrl = '<?php echo $base_url; ?>/sign/manifest.m3u8';

    if (Hls.isSupported()) {
      var hls = new Hls();
      hls.loadSource(manifestUrl);
      hls.attachMedia(video);
    }
    else if (video.canPlayType('application/vnd.apple.mpegurl')) {
      video.src = manifestUrl;
    }
  </script>
</body>
</html>
```

您可以使用已簽署 URL，而不是使用已簽署 Cookie。如需詳細資訊，請參閱[使用 PHP 建立 URL 簽章](#)。

用於 Base64 編碼和加密的 Linux 命令和 OpenSSL

您可以使用以下 Linux 命令列命令和 OpenSSL 來雜湊和簽署政策聲明、base64 編碼簽章，並將 URL 查詢字串參數中無效的字元替換為有效的字元。

如需 OpenSSL 的詳細資訊，請前往 <https://www.openssl.org>。

```
cat policy | tr -d "\n" | tr -d " \t\n\r" | openssl sha1 -sign private_key.pem |
openssl base64 -A | tr -- '+=' '-_~'
```

在上述命令中：

- `cat` 會讀取 `policy` 檔案。
- `tr -d "\n" | tr -d " \t\n\r"` 會移除 `cat` 所新增的空格和新行字元。
- OpenSSL 會使用 SHA-1 來進行檔案的雜湊演算，並使用私有金鑰檔案 `private_key.pem` 來對該檔案進行簽署。私有金鑰簽章可以是 RSA 2048 或 ECDSA 256。
- 使用 OpenSSL `base64` 來將經過雜湊演算和簽署的政策陳述式編碼。
- `tr` 會以有效字元取代 URL 查詢字串參數中無效的字元。

如需示範建立簽章的更多程式碼範例，請參閱 [為已簽署 URL 建立簽章的程式碼範例](#)。

為已簽署 URL 建立簽章的程式碼範例

本節包含可下載的應用程式範例，示範如何為已簽章的 URL 建立簽章。範例可在 Perl、PHP、C# 和 Java 中找到。您可以使用任何範例來建立簽章的 URL。Perl 指令碼在 Linux 和 macOS 平台上執行。PHP 範例可以在任何執行 PHP 的伺服器上執行。C# 範例使用 .NET 架構。

如需 JavaScript (Node.js) 中的範例程式碼，請參閱 AWS 開發人員部落格上的在 [Node.js 中建立 Amazon CloudFront 簽章 URLs](#)。

例如 Python 中的程式碼，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的 [為 Amazon CloudFront 產生簽署的 URL](#)，以及 Boto3 GitHub 儲存庫中的 [此範例程式碼](#)。

主題

- [使用 Perl 建立 URL 簽章](#)
- [使用 PHP 建立 URL 簽章](#)
- [使用 C# 和 .NET 架構建立 URL 簽章](#)
- [使用 Java 建立 URL 簽章](#)

使用 Perl 建立 URL 簽章

本節包含適用於 Linux/Mac 平台的 Perl 指令碼，可用來建立私有內容的簽章。若要建立簽章，請使用命令列參數指定 CloudFront URL、簽署者的私有金鑰路徑、金鑰 ID 和 URL 的過期日期，來執行指令碼。此工具也可以解碼簽章的 URL。

 Note

建立 URL 簽章只是私有內容提供服務的程式的一部分，以使用簽章 URL。如需端對端程序的詳細資訊，請參閱 [使用已簽署 URL](#)。

主題

- [用於建立已簽署 URL 的 Perl 指令碼來源](#)

用於建立已簽署 URL 的 Perl 指令碼來源

以下 Perl 原始程式碼可用來建立適用於 CloudFront 簽章的 URL。程式碼中的註解包含有關命令列參數和工具功能的詳細資訊。

```
#!/usr/bin/perl -w

# Copyright 2008 Amazon Technologies, Inc. Licensed under the Apache License, Version
# 2.0 (the "License");
# you may not use this file except in compliance with the License. You may obtain a
# copy of the License at:
#
# https://aws.amazon.com/apache2.0
#
# This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
# KIND, either express or implied.
# See the License for the specific language governing permissions and limitations under
# the License.

=head1 cfsign.pl

cfsign.pl - A tool to generate and verify Amazon CloudFront signed URLs

=head1 SYNOPSIS

This script uses an existing RSA key pair to sign and verify Amazon CloudFront signed
URLs

View the script source for details as to which CPAN packages are required beforehand.

For help, try:
```

```
cfsign.pl --help
```

URL signing examples:

```
cfsign.pl --action encode --url https://images.my-website.com/gallery1.zip --policy
sample_policy.json --private-key privkey.pem --key-pair-id mykey
```

```
cfsign.pl --action encode --url https://images.my-website.com/gallery1.zip --expires
1257439868 --private-key privkey.pem --key-pair-id mykey
```

URL decode example:

```
cfsign.pl --action decode --url "http://mydist.cloudfront.net/?Signature=AG0-
PgXkYo99MkJFHvjfGXjG1QDEXeaDb4Qtzmy85wqyJjK7eKojQWa4BCRcow__&Policy=eyJTdGF0ZW1lbnQiOi0t7I1Jlc29
Pair-Id=mykey"
```

To generate an RSA key pair, you can use openssl and the following commands:

```
# Generate a 2048 bit key pair
openssl genrsa -out private-key.pem 2048
openssl rsa -in private-key.pem -pubout -out public-key.pem
```

```
=head1 OPTIONS
```

```
=over 8
```

```
=item B<--help>
```

Print a help message and exits.

```
=item B<--action> [action]
```

The action to execute. action can be one of:

```
    encode - Generate a signed URL (using a canned policy or a user policy)
    decode - Decode a signed URL
```

```
=item B<--url>
```

The URL to en/decode

```
=item B<--stream>
```

The stream to en/decode

```
=item B<--private-key>
```

The path to your private key.

```
=item B<--key-pair-id>
```

The key pair identifier.

```
=item B<--policy>
```

The CloudFront policy document.

```
=item B<--expires>
```

The Unix epoch time when the URL is to expire. If both this option and the `--policy` option are specified, `--policy` will be used. Otherwise, this option alone will use a canned policy.

```
=back
```

```
=cut
```

```
use strict;  
use warnings;
```

```
# you might need to use CPAN to get these modules.  
# run perl -MCPAN -e "install <module>" to get them.  
# The openssl command line will also need to be in your $PATH.  
use File::Temp qw/tempfile/;  
use File::Slurp;  
use Getopt::Long;  
use IPC::Open2;  
use MIME::Base64 qw(encode_base64 decode_base64);  
use Pod::Usage;  
use URI;
```

```
my $CANNED_POLICY  
    = '{"Statement":[{"Resource":"<RESOURCE>","Condition":{"DateLessThan":  
{"AWS:EpochTime":<EXPIRES>}}}]}';
```

```
my $POLICY_PARAM      = "Policy";
```

```
my $EXPIRES_PARAM      = "Expires";
my $SIGNATURE_PARAM    = "Signature";
my $KEY_PAIR_ID_PARAM  = "Key-Pair-Id";

my $verbose = 0;
my $policy_filename = "";
my $expires_epoch = 0;
my $action = "";
my $help = 0;
my $key_pair_id = "";
my $url = "";
my $stream = "";
my $private_key_filename = "";

my $result = GetOptions("action=s"      => \$action,
                        "policy=s"      => \$policy_filename,
                        "expires=i"     => \$expires_epoch,
                        "private-key=s" => \$private_key_filename,
                        "key-pair-id=s" => \$key_pair_id,
                        "verbose"       => \$verbose,
                        "help"          => \$help,
                        "url=s"         => \$url,
                        "stream=s"      => \$stream,
                        );

if ($help or !$result) {
    pod2usage(1);
    exit;
}

if ($url eq "" and $stream eq "") {
    print STDERR "Must include a stream or a URL to encode or decode with the --stream
or --url option\n";
    exit;
}

if ($url ne "" and $stream ne "") {
    print STDERR "Only one of --url and --stream may be specified\n";
    exit;
}

if ($url ne "" and !is_url_valid($url)) {
    exit;
}
```

```
if ($stream ne "") {
    exit unless is_stream_valid($stream);

    # The signing mechanism is identical, so from here on just pretend we're
    # dealing with a URL
    $url = $stream;
}

if ($action eq "encode") {
    # The encode action will generate a private content URL given a base URL,
    # a policy file (or an expires timestamp) and a key pair id parameter
    my $private_key;
    my $public_key;
    my $public_key_file;

    my $policy;
    if ($policy_filename eq "") {
        if ($expires_epoch == 0) {
            print STDERR "Must include policy filename with --policy argument or an
expires" .
                "time using --expires\n";
        }

        $policy = $CANNED_POLICY;
        $policy =~ s/<EXPIRES>/$expires_epoch/g;
        $policy =~ s/<RESOURCE>/$url/g;
    } else {
        if (! -e $policy_filename) {
            print STDERR "Policy file $policy_filename does not exist\n";
            exit;
        }
        $expires_epoch = 0; # ignore if set
        $policy = read_file($policy_filename);
    }

    if ($private_key_filename eq "") {
        print STDERR "You must specific the path to your private key file with --
private-key\n";
        exit;
    }

    if (! -e $private_key_filename) {
        print STDERR "Private key file $private_key_filename does not exist\n";
    }
}
```

```

        exit;
    }

    if ($key_pair_id eq "") {
        print STDERR "You must specify a key pair id with --key-pair-id\n";
        exit;
    }

    my $encoded_policy = url_safe_base64_encode($policy);
    my $signature = rsa_sha1_sign($policy, $private_key_filename);
    my $encoded_signature = url_safe_base64_encode($signature);

    my $generated_url = create_url($url, $encoded_policy, $encoded_signature,
    $key_pair_id, $expires_epoch);

    if ($stream ne "") {
        print "Encoded stream (for use within a swf):\n" . $generated_url . "\n";
        print "Encoded and escaped stream (for use on a webpage):\n" .
escape_url_for_webpage($generated_url) . "\n";
    } else {
        print "Encoded URL:\n" . $generated_url . "\n";
    }
} elsif ($action eq "decode") {
    my $decoded = decode_url($url);
    if (!$decoded) {
        print STDERR "Improperly formed URL\n";
        exit;
    }

    print_decoded_url($decoded);
} else {
    # No action specified, print help. But only if this is run as a program (caller
    will be empty)
    pod2usage(1) unless caller();
}

# Decode a private content URL into its component parts
sub decode_url {
    my $url = shift;

    if ($url =~ /(.*?)\?(.*)/) {
        my $base_url = $1;
        my $params = $2;
    }
}

```



```

my @unparsed_params = split(/&/, $params);
my %params = ();
foreach my $param (@unparsed_params) {
    my ($key, $val) = split(/=/, $param);
    $params{$key} = $val;
}

my $encoded_signature = "";
if (exists $params{$SIGNATURE_PARAM}) {
    $encoded_signature = $params{"Signature"};
} else {
    print STDERR "Missing Signature URL parameter\n";
    return 0;
}

my $encoded_policy = "";
if (exists $params{$POLICY_PARAM}) {
    $encoded_policy = $params{$POLICY_PARAM};
} else {
    if (!exists $params{$EXPIRES_PARAM}) {
        print STDERR "Either the Policy or Expires URL parameter needs to be
specified\n";
        return 0;
    }

    my $expires = $params{$EXPIRES_PARAM};

    my $policy = $CANNED_POLICY;
    $policy =~ s/<EXPIRES>/$expires/g;

    my $url_without_cf_params = $url;
    $url_without_cf_params =~ s/$SIGNATURE_PARAM=[^&]*&?//g;
    $url_without_cf_params =~ s/$POLICY_PARAM=[^&]*&?//g;
    $url_without_cf_params =~ s/$EXPIRES_PARAM=[^&]*&?//g;
    $url_without_cf_params =~ s/$KEY_PAIR_ID_PARAM=[^&]*&?//g;

    if ($url_without_cf_params =~ /(.*?)\?$/) {
        $url_without_cf_params = $1;
    }

    $policy =~ s/<RESOURCE>/$url_without_cf_params/g;

    $encoded_policy = url_safe_base64_encode($policy);

```

```

    }

    my $key = "";
    if (exists $params{$KEY_PAIR_ID_PARAM}) {
        $key = $params{$KEY_PAIR_ID_PARAM};
    } else {
        print STDERR "Missing $KEY_PAIR_ID_PARAM parameter\n";
        return 0;
    }

    my $policy = url_safe_base64_decode($encoded_policy);

    my %ret = ();
    $ret{"base_url"} = $base_url;
    $ret{"policy"} = $policy;
    $ret{"key"} = $key;

    return \%ret;
} else {
    return 0;
}
}

# Print a decoded URL out
sub print_decoded_url {
    my $decoded = shift;

    print "Base URL: \n" . $decoded->{"base_url"} . "\n";
    print "Policy: \n" . $decoded->{"policy"} . "\n";
    print "Key: \n" . $decoded->{"key"} . "\n";
}

# Encode a string with base 64 encoding and replace some invalid URL characters
sub url_safe_base64_encode {
    my ($value) = @_;

    my $result = encode_base64($value);
    $result =~ tr|+|=|/_~/;

    return $result;
}

# Decode a string with base 64 encoding.  URL-decode the string first
# followed by reversing any special character ("+=/") translation.

```

```
sub url_safe_base64_decode {
    my ($value) = @_ ;

    $value =~ s/%([0-9A-Fa-f]{2})/chr(hex($1))/eg;
    $value =~ tr|_|~|+|=|/|;

    my $result = decode_base64($value);

    return $result;
}

# Create a private content URL
sub create_url {
    my ($path, $policy, $signature, $key_pair_id, $expires) = @_ ;

    my $result;
    my $separator = $path =~ /\?/ ? '&' : '?';
    if ($expires) {
        $result = "$path$separator$EXPIRES_PARAM=$expires&$SIGNATURE_PARAM=$signature&
$KEY_PAIR_ID_PARAM=$key_pair_id";
    } else {
        $result = "$path$separator$POLICY_PARAM=$policy&$SIGNATURE_PARAM=$signature&
$KEY_PAIR_ID_PARAM=$key_pair_id";
    }
    $result =~ s/\n//g;

    return $result;
}

# Sign a document with given private key file.
# The first argument is the document to sign
# The second argument is the name of the private key file
sub rsa_sha1_sign {
    my ($to_sign, $pvkFile) = @_ ;
    print "openssl sha1 -sign $pvkFile $to_sign\n";

    return write_to_program($pvkFile, $to_sign);
}

# Helper function to write data to a program
sub write_to_program {
    my ($keyfile, $data) = @_ ;
    unlink "temp_policy.dat" if (-e "temp_policy.dat");
    unlink "temp_sign.dat" if (-e "temp_sign.dat");
```

```
write_file("temp_policy.dat", $data);

system("openssl dgst -sha1 -sign \"\$keyfile\" -out temp_sign.dat temp_policy.dat");

my $output = read_file("temp_sign.dat");

    return $output;
}

# Read a file into a string and return the string
sub read_file {
    my ($file) = @_;

    open(INFILE, "<$file") or die("Failed to open $file: $!");
    my $str = join('', <INFILE>);
    close INFILE;

    return $str;
}

sub is_url_valid {
    my ($url) = @_;

    # HTTP distributions start with http[s]:// and are the correct thing to sign
    if ($url =~ /^https?:\\\/\\\/) {
        return 1;
    } else {
        print STDERR "CloudFront requires absolute URLs for HTTP distributions\\n";
        return 0;
    }
}

sub is_stream_valid {
    my ($stream) = @_;

    if ($stream =~ /^rtmp:\\\/\\\/ or $stream =~ /^\\\/?cfx\\\/st/) {
        print STDERR "Streaming distributions require that only the stream name is
signed.\\n";
        print STDERR "The stream name is everything after, but not including, cfx/st/
\\n";
        return 0;
    } else {
        return 1;
    }
}
```

```
    }  
}  
  
# flash requires that the query parameters in the stream name are url  
# encoded when passed in through javascript, etc. This sub handles the minimal  
# required url encoding.  
sub escape_url_for_webpage {  
    my ($url) = @_;  
  
    $url =~ s/\?/%3F/g;  
    $url =~ s/=/%3D/g;  
    $url =~ s/&/%26/g;  
  
    return $url;  
}  
  
1;
```

使用 PHP 建立 URL 簽章

任何執行 PHP 的 Web 伺服器都可以使用 PHP 範例程式碼為私有 CloudFront 分佈建立政策聲明和簽章。該完整範例以已簽署 URL 連結建立一個正常運作網頁，該連結使用 CloudFront 串流播放視訊串流。您可以從 [demo-php.zip](#) 檔案下載完整範例。

備註

- 建立 URL 簽章只是私有內容提供服務的程式的一部分，以使用簽章 URL。如需有關整個程序的詳細資訊，請參閱 [使用已簽署 URL](#)。
- 您還可以使用適用於 PHP 的 AWS SDK 中的 `UrlSigner` 類別建立簽章的 URL。如需詳細資訊，請參閱適用於 PHP 的 AWS SDK API 參考中的 [Class UrlSigner](#)。

主題

- [建立 RSA SHA-1 簽章](#)
- [建立標準政策](#)
- [建立自訂政策](#)
- [完整程式碼範例](#)

下列各節將程式碼範例細分為個別部分。您可以在下面找到 [完整程式碼範例](#)。

建立 RSA SHA-1 簽章

此程式碼會執行下列操作：

- `rsa_sha1_sign` 函數會雜湊並簽署政策陳述式。所需的引數為政策陳述式，以及與您分佈之信任金鑰群組中公有金鑰對應的私有金鑰。
- 接著，該 `url_safe_base64_encode` 函數會建立已簽章的 URL 安全版本。

```
function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);

    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with
    // the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}
```

下列程式碼片段使用 `get_canned_policy_stream_name()` 和 `get_custom_policy_stream_name()` 函數來建立標準和自訂政策。CloudFront 使用政策來建立串流影片的 URL，包括指定過期時間。

然後，您可以使用標準政策或自訂政策來決定如何管理對內容的存取。如需選擇哪個項目的詳細資訊，請參閱 [決定對已簽署 URL 使用標準或自訂政策](#) 一節。

建立標準政策

下列範例程式碼會建構簽章的標準政策陳述式。

Note

`$expires` 變數是日期/時間戳記，必須整數，而不是字串。

```
function get_canned_policy_stream_name($video_path, $private_key_filename,
    $key_pair_id, $expires) {
    // this policy is well known by CloudFront, but you still need to sign it, since it
    contains your parameters
    $canned_policy = '{"Statement":[{"Resource":"' . $video_path . '","Condition":
{"DateLessThan":{"AWS:EpochTime":' . $expires . '}}]}';
    // the policy contains characters that cannot be part of a URL, so we base64 encode
    it
    $encoded_policy = url_safe_base64_encode($canned_policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature,
    $key_pair_id, $expires);
    // URL-encode the query string characters
    return $stream_name;
}
```

如需有關標準政策的詳細資訊，請參閱 [使用標準政策建立已簽署 URL](#)。

建立自訂政策

下列範例程式碼會建構簽章的自訂政策陳述式。

```
function get_custom_policy_stream_name($video_path, $private_key_filename,
    $key_pair_id, $policy) {
    // the policy contains characters that cannot be part of a URL, so we base64 encode
    it
```

```
$encoded_policy = url_safe_base64_encode($policy);
// sign the original policy, not the encoded version
$signature = rsa_shal_sign($policy, $private_key_filename);
// make the signature safe to be included in a URL
$encoded_signature = url_safe_base64_encode($signature);

// combine the above into a stream name
$stream_name = create_stream_name($video_path, $encoded_policy, $encoded_signature,
$key_pair_id, null);
// URL-encode the query string characters
return $stream_name;
}
```

如需有關自訂政策的詳細資訊，請參閱 [使用自訂政策建立已簽署 URL](#)。

完整程式碼範例

下面的範例程式碼提供了使用 PHP 建立 CloudFront 簽章 URL 的完整示範。您可以從 [demo-php.zip](#) 檔案下載完整範例。

您可以在下列範例中修改 `$policy` Condition 元素，以允許 IPv4 與 IPv6 位址範圍。如需範例，請參閱《Amazon Simple Storage Service 使用者指南》中的 [在 IAM 政策中使用 IPv6 位址](#)。

```
<?php

function rsa_shal_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);

    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}
```



```

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}

function create_stream_name($stream, $policy, $signature, $key_pair_id, $expires) {
    $result = $stream;
    // if the stream already contains query parameters, attach the new query parameters
    to the end
    // otherwise, add the query parameters
    $separator = strpos($stream, '?') == FALSE ? '?' : '&';
    // the presence of an expires time means we're using a canned policy
    if($expires) {
        $result .= $separator . "Expires=" . $expires . "&Signature=" . $signature .
        "&Key-Pair-Id=" . $key_pair_id;
    }
    // not using a canned policy, include the policy itself in the stream name
    else {
        $result .= $separator . "Policy=" . $policy . "&Signature=" . $signature .
        "&Key-Pair-Id=" . $key_pair_id;
    }

    // new lines would break us, so remove them
    return str_replace('\n', '', $result);
}

function get_canned_policy_stream_name($video_path, $private_key_filename,
    $key_pair_id, $expires) {
    // this policy is well known by CloudFront, but you still need to sign it, since it
    contains your parameters
    $canned_policy = '{"Statement":[{"Resource":"' . $video_path . '","Condition":
{"DateLessThan":{"AWS:EpochTime":' . $expires . '}}]}';
    // the policy contains characters that cannot be part of a URL, so we base64 encode
    it
    $encoded_policy = url_safe_base64_encode($canned_policy);
    // sign the original policy, not the encoded version
    $signature = rsa_sha1_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

```

```

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature,
$key_pair_id, $expires);
    // URL-encode the query string characters
    return $stream_name;
}

function get_custom_policy_stream_name($video_path, $private_key_filename,
$key_pair_id, $policy) {
    // the policy contains characters that cannot be part of a URL, so we base64 encode
    it
    $encoded_policy = url_safe_base64_encode($policy);
    // sign the original policy, not the encoded version
    $signature = rsa_shal_sign($policy, $private_key_filename);
    // make the signature safe to be included in a URL
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, $encoded_policy, $encoded_signature,
$key_pair_id, null);
    // URL-encode the query string characters
    return $stream_name;
}

// Path to your private key. Be very careful that this file is not accessible
// from the web!

$private_key_filename = '/home/test/secure/example-priv-key.pem';
$key_pair_id = 'K2JCJMDEHXQW5F';

// Make sure you have "Restrict viewer access" enabled on this path behaviour and using
the above Trusted key groups (recommended).
$video_path = 'https://example.com/secure/example.mp4';

$expires = time() + 300; // 5 min from now
$canned_policy_stream_name = get_canned_policy_stream_name($video_path,
$private_key_filename, $key_pair_id, $expires);

// Get the viewer real IP from the x-forward-for header as $_SERVER['REMOTE_ADDR']
will return viewer facing IP. An alternative option is to use CloudFront-Viewer-
Address header. Note that this header is a trusted CloudFront immutable header. Example
format: IP:PORT ("CloudFront-Viewer-Address": "1.2.3.4:12345")

```

```

$client_ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
$policy =
'{'.
    '"Statement":['.
        '{'.
            '"Resource":' . $video_path . ','.
            '"Condition":{' .
                '"IpAddress":{"AWS:SourceIp":"' . $client_ip . '/32"},'.
                '"DateLessThan":{"AWS:EpochTime":"' . $expires . '}'.
            '}'.
        '}'.
    ']' .
    '}'';

$custom_policy_stream_name = get_custom_policy_stream_name($video_path,
    $private_key_filename, $key_pair_id, $policy);

?>

<html>

<head>
    <title>CloudFront</title>
</head>

<body>
    <h1>Amazon CloudFront</h1>
    <h2>Canned Policy</h2>
    <h3>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?></h3>
    <br />

    <div id='canned'>The canned policy video will be here: <br>

        <video width="640" height="360" autoplay muted controls>
            <source src="<?php echo $canned_policy_stream_name; ?>" type="video/mp4">
            Your browser does not support the video tag.
        </video>
    </div>

    <h2>Custom Policy</h2>
    <h3>Expires at <?php echo gmdate('Y-m-d H:i:s T', $expires); ?> only viewable by IP
    <?php echo $client_ip; ?></h3>
    <div id='custom'>The custom policy video will be here: <br>

        <video width="640" height="360" autoplay muted controls>

```

```
<source src="<?php echo $custom_policy_stream_name; ?>" type="video/mp4">
  Your browser does not support the video tag.
</video>
</div>

</body>

</html>
```

如需其他 URL 簽章範例，請參閱下列主題：

- [使用 Perl 建立 URL 簽章](#)
- [使用 C# 和 .NET 架構建立 URL 簽章](#)
- [使用 Java 建立 URL 簽章](#)

您可以使用已簽署 Cookie，而不是使用已簽署 URL 來建立簽章。如需詳細資訊，請參閱[使用 PHP 建立已簽署 Cookie](#)。

使用 C# 和 .NET 架構建立 URL 簽章

本節中的 C# 範例實作應用程式範例，示範如何使用標準和自訂政策聲明為 CloudFront 私有分佈建立簽章。此範例包含使用 [適用於 .NET 的 AWS SDK](#) 的公用程式函數，這些函數在 .NET 應用程式中非常實用。

您也可以使用適用於 .NET 的 SDK 建立已簽章的 URL 和已簽章的 Cookie。在 [適用於 .NET 的 SDK API 參考](#)中，請參閱下列主題：

- 已簽署 URL – [AmazonCloudFrontUrlSigner](#)
- 已簽署 Cookie – [AmazonCloudFrontCookieSigner](#)

若要下載程式碼，請前往[使用 C# 的簽章程式碼](#)。

備註

- AmazonCloudFrontUrlSigner 和 AmazonCloudFrontCookieSigner 類別已移至單獨的套件。如需有關如何使用的詳細資訊，請參閱《適用於 .NET 的 AWS SDK (V4) 開發人員指南》中的 [CookieSigner](#) 和 [UrlSigner](#)。

- 建立 URL 簽章只是私有內容提供服務的程式的一部分，以使用簽章 URL。如需詳細資訊，請參閱[使用已簽署 URL](#)。若需有關使用已簽署 Cookie 的詳細資訊，請參閱[使用已簽署 Cookie](#)。

在 .NET Framework 中使用 RSA 金鑰

若要在 .NET Framework 中使用 RSA 金鑰，您必須將 AWS 提供的 .pem 檔案轉換為 .NET Framework 使用的 XML 格式。

轉換後，RSA 私有金鑰檔案的格式如下：

Example：XML .NET 架構格式的 RSA 私有金鑰

```
<RSAKeyValue>
  <Modulus>
    w05IvYCP5UcoCKDo1dcspoMehWBZcyfs9QEzGi60e5y+ewGr1oW+vB2GPB
    ANBiVPcUHTFWhwaIBd3oglmF0lGQlJP/j0fmXHUK2kUUnLnJp+o0BL2NiuFtqcW6h/L5lIpD8Yq+NRHg
    Ty4zDsyr2880MvXv88yEFURckqEXAMPLE=
  </Modulus>
  <Exponent>AQAB</Exponent>
  <P>
    5bmKDaTz
    npENGvqz4Cea8XPH+sxt+2VaAwYnsarVUoSBeVt8WLl0VuZGG9IZYmH5KteXEu7fZveYd9UEXAMPLE==
  </P>
  <Q>
    1v9l/WN1a1N3r0K4VGoCokx7kR2SyTMSbZgF9IWJN0ugR/WZw7HTnjip03c9dy1Ms9pUKwUF4
    6d7049EXAMPLE==
  </Q>
  <DP>
    RgrSKuLWXMyBH+/l1Dx/I4tXuAJIrlPyo+Vmi0c7b5NzHptkSHEPFR9s1
    0K0VqjknclqCJ3Ig860MEtEXAMPLE==
  </DP>
  <DQ>
    pjPjvSFw+RoaTu0pgCA/jwW/FGyfN6iim1RFbkT4
    z49DZb2IM885f3vf35eLTaEYRYUHqGZtChNEV0TEXAMPLE==
  </DQ>
  <InverseQ>
    nkV0JTg5QtGNgWb9i
    cVtzrL/1pFE0HbJXwEjdU99N+7sMK+1066DL/HSBUCD63qD4USpnf0myc24in0EXAMPLE==</InverseQ>
  <D>
    Bc7mp7XYHynuPZxChjWNJZIq+A73gm0ASDv6At7F8Vi9r0xU1Qe/v0AQS3ycN8Q1yR4XMbzMLYk
    3yJxFDXo4ZKQt0GzLGteCU2srANiLv26/imXA8FVidZftTatLviWQZBVPTeYIA69ATUYPEq0a5u5wjGy
```

```
U0ij90WyuEXAMPLE=  
</D>  
</RSAKeyValue>
```

C# 的標準政策簽章方式

以下 C# 程式碼會透過執行以下程序建立使用標準政策簽章的 URL：

- 建立政策聲明。
- 使用 SHA1 對政策陳述式進行雜湊處理，並使用 RSA 和其對應的公有金鑰在信任金鑰組中的私有金鑰對結果進行簽署。
- Base64 編碼雜湊和簽章的政策聲明，並取代特殊字元，以使字串安全地用作 URL 請求參數。
- 串連值。

如需完整的實作，請參閱[使用 C# 的簽章程式碼範例](#)。

Note

當您將公有金鑰上傳至 CloudFront 時，會傳回 keyId。如需詳細資訊，請參閱



[&Key-Pair-Id](#)。

Example：C# 的標準政策簽章方式

```
public static string ToUrlSafeBase64String(byte[] bytes)  
{  
    return System.Convert.ToBase64String(bytes)  
        .Replace('+', '-')  
        .Replace('=', '~')  
        .Replace('/', '~');  
}  
  
public static string CreateCannedPrivateURL(string urlString,  
    string durationUnits, string durationNumber, string pathToPolicyStmnt,  
    string pathToPrivateKey, string keyId)  
{  
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days  
    // to expiration, 3-numberOfPreviousUnits, 4-pathToPolicyStmnt,  
    // 5-pathToPrivateKey, 6-keyId
```

```
TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);

// Create the policy statement.
string strPolicy = CreatePolicyStatement(pathToPolicyStmnt,
    urlString,
    DateTime.Now,
    DateTime.Now.Add(timeSpanInterval),
    "0.0.0.0/0");
if ("Error!" == strPolicy) return "Invalid time frame." +
    "Start time cannot be greater than end time.";

// Copy the expiration time defined by policy statement.
string strExpiration = CopyExpirationTimeFromPolicy(strPolicy);

// Read the policy into a byte buffer.
byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

// Initialize the SHA1CryptoServiceProvider object and hash the policy data.
using (SHA1CryptoServiceProvider
    cryptoSHA1 = new SHA1CryptoServiceProvider())
{
    bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);

    // Initialize the RSACryptoServiceProvider object.
    RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
    XmlDocument xmlPrivateKey = new XmlDocument();

    // Load your private key, which you created by converting your
    // .pem file to the XML format that the .NET framework uses.
    // Several tools are available.
    xmlPrivateKey.Load(pathToPrivateKey);

    // Format the RSACryptoServiceProvider providerRSA and
    // create the signature.
    providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
    RSAPKCS1SignatureFormatter rsaFormatter =
        new RSAPKCS1SignatureFormatter(providerRSA);
    rsaFormatter.SetHashAlgorithm("SHA1");
    byte[] signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);

    // Convert the signed policy to URL-safe base64 encoding and
    // replace unsafe characters + = / with the safe characters - _ ~
    string strSignedPolicy = ToUrlSafeBase64String(signedPolicyHash);
```

```
// Concatenate the URL, the timestamp, the signature,  
// and the key pair ID to form the signed URL.  
return urlString +  
    "?Expires=" +  
    strExpiration +  
    "&Signature=" +  
    strSignedPolicy +  
    "&Key-Pair-Id=" +  
    keyId;  
}  
}
```

C# 的自訂政策簽章方式

以下 C# 程式碼透過執行以下步驟建立使用自訂政策簽章的 URL：

1. 建立政策聲明。
2. Base64 編碼政策聲明，並取代特殊字元，以使字串安全地用作 URL 請求參數。
3. 使用 SHA1 對政策陳述式進行雜湊處理，並使用 RSA 和其對應的公有金鑰在信任金鑰組中的私有金鑰對結果進行加密。
4. Base64 編碼雜湊政策聲明，並取代特殊字元，以使字串安全地用作 URL 請求參數。
5. 串連值。

如需完整的實作，請參閱[使用 C# 的簽章程式碼範例](#)。

Note

當您將公有金鑰上傳至 CloudFront 時，會傳回 keyId。如需詳細資訊，請參閱

6

[&Key-Pair-Id](#)。

Example：C# 的自訂政策簽章方式

```
public static string ToUrlSafeBase64String(byte[] bytes)  
{  
    return System.Convert.ToBase64String(bytes)  
        .Replace('+', '-')  
}
```



```
        .Replace('=', '_')
        .Replace('/', '~');
    }

    public static string CreateCustomPrivateURL(string urlString,
        string durationUnits, string durationNumber, string startIntervalFromNow,
        string ipAddress, string pathToPolicyStmnt, string pathToPrivateKey,
        string keyId)
    {
        // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
        // to expiration, 3-numberOfPreviousUnits, 4-starttimeFromNow,
        // 5-ip_address, 6-pathToPolicyStmnt, 7-pathToPrivateKey, 8-keyId

        TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);
        TimeSpan timeSpanToStart = GetDurationByUnits(durationUnits,
            startIntervalFromNow);
        if (null == timeSpanToStart)
            return "Invalid duration units." +
                "Valid options: seconds, minutes, hours, or days";

        string strPolicy = CreatePolicyStatement(
            pathToPolicyStmnt, urlString, DateTime.Now.Add(timeSpanToStart),
            DateTime.Now.Add(timeSpanInterval), ipAddress);

        // Read the policy into a byte buffer.
        byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

        // Convert the policy statement to URL-safe base64 encoding and
        // replace unsafe characters + = / with the safe characters - _ ~

        string urlSafePolicy = ToUrlSafeBase64String(bufferPolicy);

        // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
        byte[] bufferPolicyHash;
        using (SHA1CryptoServiceProvider cryptoSHA1 =
            new SHA1CryptoServiceProvider())
        {
            bufferPolicyHash = cryptoSHA1.ComputeHash(bufferPolicy);

            // Initialize the RSACryptoServiceProvider object.
            RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
            XmlDocument xmlPrivateKey = new XmlDocument();

            // Load your private key, which you created by converting your
```

```

    // .pem file to the XML format that the .NET framework uses.
    // Several tools are available.
    xmlPrivateKey.Load(pathToPrivateKey);

    // Format the RSACryptoServiceProvider providerRSA
    // and create the signature.
    providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
    RSAPKCS1SignatureFormatter RSAFormatter =
        new RSAPKCS1SignatureFormatter(providerRSA);
    RSAFormatter.SetHashAlgorithm("SHA1");
    byte[] signedHash = RSAFormatter.CreateSignature(bufferPolicyHash);

    // Convert the signed policy to URL-safe base64 encoding and
    // replace unsafe characters + = / with the safe characters - _ ~
    string strSignedPolicy = ToUrlSafeBase64String(signedHash);

    return urlString +
        "?Policy=" +
        urlSafePolicy +
        "&Signature=" +
        strSignedPolicy +
        "&Key-Pair-Id=" +
        keyId;
}
}

```

適用於簽章產生的公用方法

以下方法從檔案取得政策聲明和剖析簽章產生的時間間隔。

Example：適用於簽章產生的公用方法

```

public static string CreatePolicyStatement(string policyStmnt,
    string resourceUrl,
    DateTime startTime,
    DateTime endTime,
    string ipAddress)

{
    // Create the policy statement.
    FileStream streamPolicy = new FileStream(policyStmnt, FileMode.Open,
    FileAccess.Read);
    using (StreamReader reader = new StreamReader(streamPolicy))
    {

```

```

    string strPolicy = reader.ReadToEnd();

    TimeSpan startTimeSpanFromNow = (startTime - DateTime.Now);
    TimeSpan endTimeSpanFromNow = (endTime - DateTime.Now);
    TimeSpan intervalStart =
        (DateTime.UtcNow.Add(startTimeSpanFromNow)) -
        new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
    TimeSpan intervalEnd =
        (DateTime.UtcNow.Add(endTimeSpanFromNow)) -
        new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

    int startTimestamp = (int)intervalStart.TotalSeconds; // START_TIME
    int endTimestamp = (int)intervalEnd.TotalSeconds; // END_TIME

    if (startTimestamp > endTimestamp)
        return "Error!";

    // Replace variables in the policy statement.
    strPolicy = strPolicy.Replace("RESOURCE", resourceUrl);
    strPolicy = strPolicy.Replace("START_TIME", startTimestamp.ToString());
    strPolicy = strPolicy.Replace("END_TIME", endTimestamp.ToString());
    strPolicy = strPolicy.Replace("IP_ADDRESS", ipAddress);
    strPolicy = strPolicy.Replace("EXPIRES", endTimestamp.ToString());
    return strPolicy;
}
}

public static TimeSpan GetDuration(string units, string numUnits)
{
    TimeSpan timeSpanInterval = new TimeSpan();
    switch (units)
    {
        case "seconds":
            timeSpanInterval = new TimeSpan(0, 0, 0, int.Parse(numUnits));
            break;
        case "minutes":
            timeSpanInterval = new TimeSpan(0, 0, int.Parse(numUnits), 0);
            break;
        case "hours":
            timeSpanInterval = new TimeSpan(0, int.Parse(numUnits), 0, 0);
            break;
        case "days":
            timeSpanInterval = new TimeSpan(int.Parse(numUnits), 0, 0, 0);
            break;
    }
}

```

```
        default:
            Console.WriteLine("Invalid time units;" +
                "use seconds, minutes, hours, or days");
            break;
    }
    return timeSpanInterval;
}

private static TimeSpan GetDurationByUnits(string durationUnits,
    string startIntervalFromNow)
{
    switch (durationUnits)
    {
        case "seconds":
            return new TimeSpan(0, 0, int.Parse(startIntervalFromNow));
        case "minutes":
            return new TimeSpan(0, int.Parse(startIntervalFromNow), 0);
        case "hours":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0);
        case "days":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0, 0);
        default:
            return new TimeSpan(0, 0, 0, 0);
    }
}

public static string CopyExpirationTimeFromPolicy(string policyStatement)
{
    int startExpiration = policyStatement.IndexOf("EpochTime");
    string strExpirationRough = policyStatement.Substring(startExpiration +
        "EpochTime".Length);
    char[] digits = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

    List<char> listDigits = new List<char>(digits);
    StringBuilder buildExpiration = new StringBuilder(20);

    foreach (char c in strExpirationRough)
    {
        if (listDigits.Contains(c))
            buildExpiration.Append(c);
    }
    return buildExpiration.ToString();
}
```

另請參閱

- [使用 Perl 建立 URL 簽章](#)
- [使用 PHP 建立 URL 簽章](#)
- [使用 Java 建立 URL 簽章](#)

使用 Java 建立 URL 簽章

除了下列程式碼範例之外，您還可以使用 [適用於 Java 的 AWS SDK（第 1 版）中的 CloudFrontUrlSigner 公用程式類別](#) 來建立 [CloudFront 簽章URLs](#)。

如需更多範例，請參閱 AWS SDK 程式碼範例程式碼庫中的 [使用 AWS SDK 建立簽章URLs 和 Cookie](#)。

Note

建立簽署 URL 只是[透過 CloudFront 提供私有內容](#)程序的一個環節。如需有關整個程序的詳細資訊，請參閱 [使用已簽署 URL](#)。

下列範例示範如何建立 CloudFront 簽署的 URL。

Example Java 政策和簽章加密方法

```
package org.example;

import java.time.Instant;
import java.time.temporal.ChronoUnit;
import software.amazon.awssdk.services.cloudfront.CloudFrontUtilities;
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;
import software.amazon.awssdk.services.cloudfront.url.SignedUrl;

public class Main {

    public static void main(String[] args) throws Exception {
        CloudFrontUtilities cloudFrontUtilities = CloudFrontUtilities.create();
        Instant expirationDate = Instant.now().plus(7, ChronoUnit.DAYS);
        String resourceUrl = "https://a1b2c3d4e5f6g7.cloudfront.net";
        String keyPairId = "K1UA3WV15I7JSD";
        CannedSignerRequest cannedRequest = CannedSignerRequest.builder()
```

```
        .resourceUrl(resourceUrl)
        .privateKey(new java.io.File("/path/to/private_key.pem").toPath())
        .keyPairId(keyPairId)
        .expirationDate(expirationDate)
        .build();
    SignedUrl signedUrl =
cloudFrontUtilities.getSignedUrlWithCannedPolicy(cannedRequest);
    String url = signedUrl.url();
    System.out.println(url);
}
}
```

另請參閱：

- [使用 Perl 建立 URL 簽章](#)
- [使用 PHP 建立 URL 簽章](#)
- [使用 C# 和 .NET 架構建立 URL 簽章](#)

限制對 AWS 原始伺服器的存取

您可以透過提供下列優點的方式設定 CloudFront 和一些 AWS 原始伺服器：

- 限制對 AWS 原始伺服器的存取，使其無法公開存取。
- 確定檢視器（使用者）只能透過指定的 CloudFront 分佈存取 AWS 原始伺服器中的內容。這可防止觀眾直接從原始伺服器或透過非預期的 CloudFront 分佈存取內容。

若要這樣做，請將 CloudFront 設定為將已驗證的請求傳送至原始 AWS 伺服器，並將 AWS 原始伺服器設定為僅允許從 CloudFront 存取已驗證的請求。如需詳細資訊，請參閱相容 AWS 原始伺服器類型的下列主題。

主題

- [限制對 AWS Elemental MediaPackage v2 原始伺服器的存取](#)
- [限制對 AWS Elemental MediaStore 原始伺服器的存取](#)
- [限制對 AWS Lambda 函數 URL 原始伺服器的存取](#)
- [限制對 Amazon S3 原始伺服器的存取](#)
- [使用 VPC 原始伺服器限制存取](#)

限制對 AWS Elemental MediaPackage v2 原始伺服器的存取

CloudFront 提供原始存取控制 (OAC)，以限制對 MediaPackage v2 原始伺服器的存取。

Note

CloudFront OAC 僅支援 MediaPackage v2。不支援 MediaPackage v1。

主題

- [建立新的 OAC](#)
- [原始存取控制的進階設定](#)

建立新的 OAC

完成下列主題中說明的步驟，在 CloudFront 中設定新的 OAC。

主題

- [先決條件](#)
- [授予 CloudFront 存取 MediaPackage v2 原始伺服器的許可](#)
- [建立 OAC](#)

先決條件

在建立和設定 OAC 之前，您必須擁有具 MediaPackage v2 原始伺服器的 CloudFront 分佈。如需詳細資訊，請參閱[使用 MediaStore 容器或 MediaPackage 頻道](#)。

授予 CloudFront 存取 MediaPackage v2 原始伺服器的許可

在您建立 OAC 或於 CloudFront 分佈中進行設定之前，請確定 CloudFront 具有存取 MediaPackage v2 原始伺服器的許可。在建立 CloudFront 分佈之後，但在將 OAC 新增至分佈組態中的 MediaPackage v2 原始伺服器之前，請執行此作業。

使用 IAM 政策允許 CloudFront 服務主體 (cloudfront.amazonaws.com) 存取原始伺服器。使用政策中的 Condition 元素，僅當請求代表包含 MediaPackage v2 原始伺服器的 CloudFront 分佈時，才允許 CloudFront 存取 MediaPackage v2 原始伺服器。這是您要新增 OAC 的 MediaPackage v2 原始伺服器分佈。

Example：IAM 政策，允許啟用 OAC 之 CloudFront 分佈的唯讀存取

下列政策允許 CloudFront 分佈 (*E1PDK09ESKHJWT*) 存取 MediaPackage v2 原始伺服器。原始伺服器是為 Resource 元素指定的 ARN。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipal",
      "Effect": "Allow",
      "Principal": {"Service": "cloudfront.amazonaws.com"},
      "Action": "mediapackagev2:GetObject",
      "Resource": "arn:aws:mediapackagev2:us-east-1:123456789012:channelGroup/channel-group-name/channel/channel-name/originEndpoint/origin_endpoint_name",
      "Condition": {
        "StringEquals": {"AWS:SourceArn": "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT"}
      }
    }
  ]
}
```

備註

- 如果您已啟用 MQAR 功能和原始存取控制 (OAC)，請將 `mediapackagev2:GetHeadObject` 動作新增至 IAM 政策。MQAR 需要此許可才能將 HEAD 請求傳送至 MediaPackage v2 原始伺服器。如需有關 MQAR 的詳細資訊，請參閱 [媒體品質感知彈性](#)。
- 如果您建立的分佈沒有 MediaPackage v2 原始伺服器的許可，您可以從 CloudFront 主控台選擇複製政策，然後選擇更新端點許可。然後，您可以將複製的許可連接到端點。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的 [端點政策欄位](#)。

建立 OAC

若要建立 OAC，您可以使用 AWS 管理主控台 CloudFormation、AWS CLI、或 CloudFront API。

Console

建立 OAC

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 於左側導覽窗格中，選擇 Origin access (原始存取)。
3. 選擇 Create control setting (建立控制設定)。
4. 在建立新 OAC 表單上，執行下列操作：
 - a. 輸入 OAC 的名稱，(選用) 輸入描述。
 - b. 於簽署行為中，建議您保留預設設定 (簽署請求 (建議使用))。如需詳細資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。
5. 針對原始伺服器類型，選擇 MediaPackage V2。
6. 選擇建立。

Tip

建立 OAC 之後，請記下名稱。您需要於下列程序中進行使用。

將 OAC 新增至分佈中的 MediaPackage v2 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇一個您想要新增 OAC 之具 MediaPackage V2 原始伺服器的分佈，然後選擇原始伺服器標籤。
3. 選取您想要將 OAC 新增至的 MediaPackage v2 原始伺服器，然後選擇編輯。
4. 在原始伺服器的 Protocol (通訊協定) 選取 HTTPS only (僅限 HTTPS)。
5. 在原始存取控制下拉式功能表中，選擇您想要使用的 OAC 名稱。
6. 選擇儲存變更。

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 MediaPackage v2 原始伺服器的所有請求。

CloudFormation

若要使用 建立 OAC CloudFormation，請使用 `AWS::CloudFront::OriginAccessControl` 資源類型。下列範例顯示用於建立 OAC 的 YAML 格式 CloudFormation 範本語法。

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
    Description: An optional description for the origin access control
    Name: ExampleOAC
    OriginAccessControlOriginType: mediapackagev2
    SigningBehavior: always
    SigningProtocol: sigv4
```

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [AWS::CloudFront::OriginAccessControl](#)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立原始存取控制，請使用 `aws cloudfront create-origin-access-control` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不必分別將每個個別參數指定為命令列輸入。

如要建立原始存取控制 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `origin-access-control.yaml` 的檔案。這個檔案中包含 `create-origin-access-control` 命令的所有輸入參數。

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
origin-access-control.yaml
```

2. 開啟您剛才建立的 `origin-access-control.yaml` 檔案。編輯檔案以新增 OAC 的名稱、說明 (選用)，並將 `SigningBehavior` 變更為 `always`。接著儲存檔案。

如需其他 OAC 設定的相關資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。

3. 使用下列命令，利用 `origin-access-control.yaml` 檔案中的輸入參數建立原始存取控制。

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-access-control.yaml
```

記下命令輸出中的 Id 值，您需要其將 OAC 新增至 CloudFront 分佈中的 MediaPackage v2 原始伺服器。

如要將 OAC 附加至現有分佈 (包含輸入檔案的 CLI) 中的 MediaPackage v2 原始伺服器

1. 使用下列命令來儲存您想要新增之 CloudFront 分佈的分佈組態。分佈必須具有 MediaPackage v2 原始伺服器。

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --output yaml > dist-config.yaml
```

2. 開啟您剛才建立且命名為 dist-config.yaml 的檔案。編輯檔案，進行下列變更：
 - 於 Origins 物件中，將 OAC 的 ID 新增至名為 OriginAccessControlId 的欄位。
 - 從名為 OriginAccessIdentity 的欄位中移除值(如果存在)。
 - 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用原始存取控制。

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-input-yaml file://dist-config.yaml
```

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 MediaPackage v2 原始伺服器的所有請求。

API

如要使用 CloudFront API 建立 OAC，請使用 [CreateOriginAccessControl](#)。如需您在此 API 呼叫中指定欄位的詳細資訊，請參閱 AWS SDK 或其他 API 用戶端的 API 參考文件。

建立 OAC 之後，您可以使用下列其中一個 API 呼叫，將其連接至分佈中的 MediaPackage v2 原始伺服器：

- 如要將其附加至現有分佈中，請使用 [UpdateDistribution](#)。
- 如要將其附加至新分佈中，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請於原始伺服器內部的 `OriginAccessControlId` 欄位中提供 OAC ID。如需您在這些 API 呼叫中指定之其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS SDK 或其他 API 用戶端的 API 參考文件。

原始存取控制的進階設定

CloudFront OAC 功能包含僅適用於特定使用案例的進階設定。除非您對進階設定有特定需求，否則請使用建議的設定。

OAC 包含名為簽署行為（在主控台中）或 `SigningBehavior`（在 API、CLI 和 CloudFormation 中）的設定。此設定提供下列選項：

永遠簽署原始請求 (建議設定)

我們建議使用此設定，於主控台中名為 `Sign requests (recommended)` (簽署請求 (建議使用))，或於 API、CLI 和 CloudFormation 中的 `always`。使用此設定時，CloudFront 一律會簽署傳送至 MediaPackage v2 原始伺服器的所有請求。

絕不簽署原始伺服器請求

此設定於主控台中命名為 `Do not sign requests` (請勿簽署請求)，或 API、CLI 和 CloudFormation 中的 `never`。使用此設定，關閉使用此 OAC 之所有分佈中的所有原始伺服器的 OAC。與從所有使用其原始伺服器和分佈中逐一移除 OAC 相比，此可節省時間和精力。使用此設定時，CloudFront 不會簽署傳送至 MediaPackage v2 原始伺服器的任何請求。

Warning

如要使用此設定，MediaPackage v2 原始伺服器必須可公開存取。如果您對無法公開存取的 MediaPackage v2 原始伺服器使用此設定，CloudFront 將無法存取該原始伺服器。MediaPackage v2 原始伺服器會將錯誤傳回 CloudFront，而 CloudFront 會將這些錯誤傳遞給檢視器。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的 [適用於 MediaPackage 中政策和許可](#) 的 MediaPackage v2 政策範例。

請勿覆寫檢視器 (用戶端) **Authorization** 標題

此設定於主控台中命名為 Do not override authorization header (請勿覆寫授權標頭)，或於 API、CLI 和 CloudFormation 中的 no-override。如果您想要 CloudFront 僅於對應的檢視器請求不包含 Authorization 標題時簽署原始伺服器請求，請使用此設定。利用此設定，當檢視器請求存在時，CloudFront 會傳遞來自檢視器請求的 Authorization 標題，但在檢視器請求不包含 Authorization 標題時對原始伺服器請求進行簽名 (新增其自己的 Authorization 標題)。

Warning

如要從檢視器請求傳遞 Authorization 標題，您必須將 Authorization 標題新增至[快速存取政策](#)中，適用於使用與此原始存取控制相關聯之 MediaPackage v2 原始伺服器的所有快速存取行為。

限制對 AWS Elemental MediaStore 原始伺服器的存取

CloudFront 提供原始存取控制 (OAC)，以限制對 AWS Elemental MediaStore 原始伺服器的存取。

主題

- [建立新的原始存取控制](#)
- [原始存取控制的進階設定](#)

建立新的原始存取控制

完成下列主題中說明的步驟，在 CloudFront 中設定新的原始存取控制。

主題

- [先決條件](#)
- [授予 CloudFront 存取 MediaStore 原始伺服器的許可](#)
- [建立原始存取控制](#)

先決條件

在建立和設定原始存取控制之前，您必須擁有具 MediaStore 原始伺服器的 CloudFront 分佈。

授予 CloudFront 存取 MediaStore 原始伺服器的許可

在您建立原始存取控制或於 CloudFront 分佈中進行設定之前，請確定 CloudFront 具有存取 MediaStore 原始伺服器的許可。在建立 CloudFront 分佈之後，但在將 OAC 新增至分佈組態中的 MediaStore 原始伺服器之前，請執行此作業。

使用 MediaStore 容器政策允許 CloudFront 服務主體 (cloudfront.amazonaws.com) 存取原始伺服器。使用政策中的 Condition 元素，僅當請求代表包含 S3 原始伺服器的 CloudFront 分佈時，才允許 CloudFront 存取 MediaStore 容器。這是您要新增 OAC 的 MediaStore 原始伺服器分佈。

下列是允許 CloudFront 分佈存取 MediaStore 原始伺服器的 MediaStore 容器政策範例。

Example MediaStore 容器政策，允許啟用 OAC 之 CloudFront 分佈的唯讀存取

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadOnly",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": [
        "mediastore:GetObject"
      ],
      "Resource": "arn:aws:mediastore:us-east-1:111122223333:container/<container name>/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
            "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
        },
        "Bool": {
          "aws:SecureTransport": "true"
        }
      }
    }
  ]
}
```

Example MediaStore 容器政策，允許啟用 OAC 之 CloudFront 分佈的讀寫存取

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": [
        "mediastore:GetObject",
        "mediastore:PutObject"
      ],
      "Resource": "arn:aws:mediastore:us-east-1:111122223333:container/container-name/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
            "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
        },
        "Bool": {
          "aws:SecureTransport": "true"
        }
      }
    }
  ]
}
```

 Note

若要允許寫入存取權，您必須設定 Allowed HTTP methods (允許的 HTTP 方法)，以將 PUT 包含在 CloudFront 分發的行為設定中。

建立原始存取控制

若要建立 OAC，您可以使用 AWS 管理主控台 CloudFormation、AWS CLI、或 CloudFront API。

Console

如要建立原始存取控制

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 於左側導覽窗格中，選擇 Origin access (原始存取)。
3. 選擇 Create control setting (建立控制設定)。
4. 在 Create control setting (建立控制設定) 表單上，執行下列動作：
 - a. 於 Details (詳細資訊) 窗格中，輸入 Name (名稱) 和 (選用) Description (描述)，以用於原始存取控制。
 - b. 於 Settings (設定) 窗格中，建議您保留預設設定 (Sign requests (recommended)) (簽署請求 (建議使用))。如需詳細資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。
5. 從 Origin type (原始伺服器類型) 下拉式功能表中選擇 MediaStore。
6. 選擇建立。

建立 OAC 之後，請記下 Name (名稱)。您需要於下列程序中進行使用。

如要將原始存取控制新增至分佈中的 MediaStore 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇一個您想要新增 OAC 之具 MediaStore 原始伺服器的分佈，然後選擇 Origins (原始伺服器) 標籤。
3. 選取您想要將 OAC 新增至的 MediaStore 原始伺服器，然後選擇 Edit (編輯)。
4. 在原始伺服器的 Protocol (通訊協定) 選取 HTTPS only (僅限 HTTPS)。
5. 在 Origin access control (原始存取控制) 下拉式功能表中，選擇您想要使用的 OAC。
6. 選擇儲存變更。

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 MediaStore 儲存貯體原始伺服器的所有請求。

CloudFormation

若要使用 建立原始存取控制 (OAC) CloudFormation，請使用

AWS::CloudFront::OriginAccessControl 資源類型。下列範例顯示用於建立原始存取控制的 YAML 格式 CloudFormation 範本語法。


```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
    Description: An optional description for the origin access control
    Name: ExampleOAC
    OriginAccessControlOriginType: mediastore
    SigningBehavior: always
    SigningProtocol: sigv4
```

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [AWS::CloudFront::OriginAccessControl](#)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立原始存取控制，請使用 `aws cloudfront create-origin-access-control` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不必分別將每個個別參數指定為命令列輸入。

如要建立原始存取控制 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `origin-access-control.yaml` 的檔案。這個檔案中包含 `create-origin-access-control` 命令的所有輸入參數。

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
origin-access-control.yaml
```

2. 開啟您剛才建立的 `origin-access-control.yaml` 檔案。編輯檔案以新增 OAC 的名稱、說明 (選用)，並將 `SigningBehavior` 變更為 `always`。接著儲存檔案。

如需其他 OAC 設定的相關資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。

3. 使用下列命令，利用 `origin-access-control.yaml` 檔案中的輸入參數建立原始存取控制。

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
access-control.yaml
```

記下命令輸出中的 `Id` 值，您需要其將 OAC 新增至 CloudFront 分佈中的 MediaStore 原始伺服器。

如要將 OAC 附加至現有分佈 (包含輸入檔案的 CLI) 中的 MediaStore 原始伺服器

1. 使用下列命令來儲存您想要新增之 CloudFront 分佈的分佈組態。分佈必須具有 MediaStore 原始伺服器。

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --output yaml > dist-config.yaml
```

2. 開啟您剛才建立且命名為 dist-config.yaml 的檔案。編輯檔案，進行下列變更：

- 於 Origins 物件中，將 OAC 的 ID 新增至名為 OriginAccessControlId 的欄位。
- 從名為 OriginAccessIdentity 的欄位中移除值(如果存在)。
- 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用原始存取控制。

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-input-yaml file://dist-config.yaml
```

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 MediaStore 原始伺服器的所有請求。

API

如要使用 CloudFront API 建立原始存取控制，請使用 [CreateOriginAccessControl](#)。如需您在此 API 呼叫中指定欄位的詳細資訊，請參閱 AWS SDK 或其他 API 用戶端的 API 參考文件。

建立原始存取控制之後，您可以使用下列其中一個 API 呼叫，將其連接至分佈中的 MediaStore 原始伺服器：

- 如要將其附加至現有分佈中，請使用 [UpdateDistribution](#)。
- 如要將其附加至新分佈中，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請於原始伺服器內部的 `OriginAccessControlId` 欄位中提供原始存取控制 ID。如需您在這些 API 呼叫中指定之其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS SDK 或其他 API 用戶端的 API 參考文件。

原始存取控制的進階設定

CloudFront 來原始存取控制功能包含僅適用於特定使用案例的進階設定。除非您對進階設定有特定需求，否則請使用建議的設定。

原始存取控制包含名為簽署行為（在主控台中）或 `SigningBehavior`（在 API、CLI 和 中）的設定 CloudFormation。此設定提供下列選項：

永遠簽署原始請求 (建議設定)

我們建議使用此設定，於主控台中名為 `Sign requests (recommended)` (簽署請求 (建議使用))，或於 API、CLI 和 CloudFormation 中的 `always`。使用此設定時，CloudFront 一律會簽署傳送至 MediaStore 原始伺服器的所有請求。

絕不簽署原始伺服器請求

此設定於主控台中命名為 `Do not sign requests` (請勿簽署請求)，或 API、CLI 和 CloudFormation 中的 `never`。使用此設定，關閉使用此原始存取控制之所有分佈中的所有原始伺服器的原始存取控制。與從所有使用其原始伺服器和分佈中逐一移除原始存取控制相比，此可節省時間和精力。使用此設定時，CloudFront 不會簽署傳送至 MediaStore 原始伺服器的任何請求。

Warning

如要使用此設定，MediaStore 原始伺服器必須可公開存取。如果您對無法公開存取的 MediaStore 原始伺服器使用此設定，CloudFront 將無法存取該原始伺服器。MediaStore 原始伺服器會將錯誤傳回 CloudFront，而 CloudFront 會將這些錯誤傳遞給檢視器。如需詳細資訊，請參閱[透過 HTTPS 的公有讀取存取](#)的 MediaStore 容器政策範例。

請勿覆寫檢視器 (用戶端) **Authorization** 標題

此設定於主控台中命名為 `Do not override authorization header` (請勿覆寫授權標頭)，或於 API、CLI 和 CloudFormation 中的 `no-override`。如果您想要 CloudFront 僅於對應的檢視器請求不包含 **Authorization** 標題時簽署原始伺服器請求，請使用此設定。利用此設定，當檢視器請求存在時，CloudFront 會傳遞來自檢視器請求的 **Authorization** 標題，但在檢視器請求不包含 **Authorization** 標題時對原始伺服器請求進行簽名 (新增其自己的 **Authorization** 標題)。

 Warning

如要從檢視器請求傳遞 Authorization 標題，您必須將 Authorization 標題新增至[快速存取政策](#)中，適用於使用與此原始存取控制相關聯之 MediaStore 原始伺服器的所有快取行為。

限制對 AWS Lambda 函數 URL 原始伺服器的存取

CloudFront 提供原始存取控制 (OAC)，以限制對 Lambda 函數 URL 原始伺服器的存取。

主題

- [建立新的 OAC](#)
- [原始存取控制的進階設定](#)
- [範本程式碼範例](#)

建立新的 OAC

完成下列主題中說明的步驟，在 CloudFront 中設定新的 OAC。

 Important

如果您搭配 Lambda 函數 URL 使用 PUT 或 POST 方法，您的使用者必須運算內文的 SHA256，並在將請求傳送至 CloudFront 時，在 x-amz-content-sha256 標頭中包含請求內文的承載雜湊值。Lambda 不支援未簽署的承載。

主題

- [先決條件](#)
- [授予 CloudFront 存取 Lambda 函數 URL 的許可](#)
- [建立 OAC](#)

先決條件

在建立和設定 OAC 之前，您必須擁有具有 Lambda 函數 URL 做為原始伺服器的 CloudFront 分佈。若要使用 OAC，您必須指定 `AWS_IAM` 作為 `AuthType` 參數的值。如需詳細資訊，請參閱[使用 Lambda 函數 URL](#)。

授予 CloudFront 存取 Lambda 函數 URL 的許可

在您建立 OAC 或在 CloudFront 分佈中設定它之前，請確定 CloudFront 具有存取 Lambda 函數 URL 的許可。在建立 CloudFront 分佈之後，但在將 OAC 新增至分佈組態中的 Lambda 函數 URL 之前，請執行此作業。

Note

若要更新 Lambda 函數 URL 的 IAM 政策，您必須使用 AWS Command Line Interface (AWS CLI)。目前不支援在 Lambda 主控台中編輯 IAM 政策。

下列 AWS CLI 命令會授予 CloudFront 服務主體 (`cloudfront.amazonaws.com`) 存取 Lambda 函數 URL 的權限。使用政策中的 `Condition` 元素，僅當請求代表包含 Lambda 函數 URL 的 CloudFront 分佈時，才允許 CloudFront 存取 Lambda。這是您要新增 OAC 的 Lambda 函數 URL 原始伺服器分佈。

Example：AWS CLI command 更新政策，以允許啟用 OAC 之 CloudFront 分佈的唯讀存取

下列 AWS CLI 命令允許 CloudFront 分佈 (*E1PDK09ESKHJWT*) 存取您的 Lambda *FUNCTION_URL_NAME*。

```
aws lambda add-permission \  
--statement-id "AllowCloudFrontServicePrincipal" \  
--action "lambda:InvokeFunctionUrl" \  
--principal "cloudfront.amazonaws.com" \  
--source-arn "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT" \  
--function-name FUNCTION_URL_NAME
```

```
aws lambda add-permission \  
--statement-id "AllowCloudFrontServicePrincipalInvokeFunction" \  
--action "lambda:InvokeFunction" \  
--principal "cloudfront.amazonaws.com" \  

```

```
--source-arn "arn:aws:cloudfront::123456789012:distribution/E1PDK09ESKHJWT" \  
--function-name FUNCTION_URL_NAME
```

Note

如果您建立分佈且沒有 Lambda 函數 URL 的許可，您可以從 CloudFront 主控台選擇複製 CLI 命令，然後從命令列終端機輸入此命令。如需詳細資訊，請參閱 AWS Lambda 開發人員指南中的[授予函數對 AWS 服務的存取](#)。

建立 OAC

若要建立 OAC，您可以使用 AWS 管理主控台 CloudFormation、AWS CLI、或 CloudFront API。

Console

建立 OAC

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 於左側導覽窗格中，選擇 Origin access (原始存取)。
3. 選擇 Create control setting (建立控制設定)。
4. 在建立新 OAC 表單上，執行下列操作：
 - a. 輸入 OAC 的名稱，(選用) 輸入描述。
 - b. 於簽署行為中，建議您保留預設設定 (簽署請求 (建議使用))。如需詳細資訊，請參閱[the section called “原始存取控制的進階設定”](#)。
5. 在原始伺服器類型中，選擇 Lambda。
6. 選擇建立。

Tip

建立 OAC 之後，請記下名稱。您需要於下列程序中進行使用。

將原始存取控制新增至分佈中的 Lambda 函數 URL

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。

2. 選擇一個您想要新增 OAC 之具 Lambda 函數 URL 的分佈，然後選擇原始伺服器標籤。
3. 選取您想要將 OAC 新增至的 Lambda 函數 URL，然後選擇編輯。
4. 在原始伺服器的 Protocol (通訊協定) 選取 HTTPS only (僅限 HTTPS)。
5. 在原始存取控制下拉式功能表中，選擇您想要使用的 OAC 名稱。
6. 選擇儲存變更。

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 Lambda 函數 URL 的所有請求。

CloudFormation

若要使用 建立 OAC CloudFormation，請使用 `AWS::CloudFront::OriginAccessControl` 資源類型。下列範例顯示用於建立 OAC 的 YAML 格式 CloudFormation 範本語法。

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
    Description: An optional description for the origin access control
    Name: ExampleOAC
    OriginAccessControlOriginType: lambda
    SigningBehavior: always
    SigningProtocol: sigv4
```

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [AWS::CloudFront::OriginAccessControl](#)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立原始存取控制，請使用 `aws cloudfront create-origin-access-control` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不必分別將每個個別參數指定為命令列輸入。

如要建立原始存取控制 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `origin-access-control.yaml` 的檔案。這個檔案中包含 `create-origin-access-control` 命令的所有輸入參數。

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
origin-access-control.yaml
```

2. 開啟您剛才建立的 `origin-access-control.yaml` 檔案。編輯檔案以新增 OAC 的名稱、說明 (選用)，並將 `SigningBehavior` 變更為 `always`。接著儲存檔案。

如需其他 OAC 設定的相關資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。

3. 使用下列命令，利用 `origin-access-control.yaml` 檔案中的輸入參數建立原始存取控制。

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-access-control.yaml
```

記下命令輸出中的 `Id` 值，您需要其將 OAC 新增至 CloudFront 分佈中的 Lambda 函數 URL。

如要將 OAC 附加至現有分佈 (包含輸入檔案的 CLI) 中的 Lambda 函數 URL

1. 使用下列命令來儲存您想要新增之 CloudFront 分佈的分佈組態。分佈必須有 Lambda 函數 URL 做為原始伺服器。

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --output yaml > dist-config.yaml
```

2. 開啟您剛才建立且命名為 `dist-config.yaml` 的檔案。編輯檔案，進行下列變更：
 - 於 `Origins` 物件中，將 OAC 的 ID 新增至名為 `OriginAccessControlId` 的欄位。
 - 從名為 `OriginAccessIdentity` 的欄位中移除值 (如果存在)。
 - 將 `ETag` 欄位重新命名為 `IfMatch`，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用原始存取控制。

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-input-yaml file://dist-config.yaml
```

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 Lambda 函數 URL 的所有請求。

API

如要使用 CloudFront API 建立 OAC，請使用 [CreateOriginAccessControl](#)。如需您在此 API 呼叫中指定欄位的詳細資訊，請參閱 AWS SDK 或其他 API 用戶端的 API 參考文件。

建立原始存取控制之後，您可以使用下列其中一個 API 呼叫，將其連接至分佈中的 Lambda 函數 URL：

- 如要將其附加至現有分佈中，請使用 [UpdateDistribution](#)。
- 如要將其附加至新分佈中，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請於原始伺服器內部的 OriginAccessControlId 欄位中提供 OAC ID。如需您在這些 API 呼叫中指定之其他欄位的詳細資訊，請參閱 [AWS SDK](#) 或其他 API 用戶端的 API 參考文件。

原始存取控制的進階設定

CloudFront OAC 功能包含僅適用於特定使用案例的進階設定。除非您對進階設定有特定需求，否則請使用建議的設定。

OAC 包含名為簽署行為（在主控台中）或 SigningBehavior（在 API、CLI 和 CloudFormation 中）的設定。此設定提供下列選項：

永遠簽署原始請求 (建議設定)

我們建議使用此設定，於主控台中名為 Sign requests (recommended) (簽署請求 (建議使用))，或於 API、CLI 和 CloudFormation 中的 always。使用此設定時，CloudFront 一律會簽署傳送至 Lambda 函數 URL 的所有請求。

絕不簽署原始伺服器請求

此設定於主控台中命名為 Do not sign requests (請勿簽署請求)，或 API、CLI 和 CloudFormation 中的 never。使用此設定，關閉使用此 OAC 之所有分佈中的所有原始伺服器的 OAC。與從所有使用其原始伺服器和分佈中逐一移除 OAC 相比，此可節省時間和精力。使用此設定時，CloudFront 不會簽署傳送至 Lambda 函數 URL 的任何請求。

Warning

若要使用此設定，Lambda 函數 URL 必須可公開存取。如果您對無法公開存取的 Lambda 函數 URL 使用此設定，CloudFront 將無法存取該原始伺服器。Lambda 函數 URL 會將錯

誤傳回 CloudFront，而 CloudFront 會將這些錯誤傳遞給檢視器。如需詳細資訊，請參閱 AWS Lambda 使用者指南中的 [Lambda 函數 URL 的安全性和驗證模型](#)。

請勿覆寫檢視器 (用戶端) **Authorization** 標題

此設定於主控台中命名為 Do not override authorization header (請勿覆寫授權標頭)，或於 API、CLI 和 CloudFormation 中的 no-override。如果您想要 CloudFront 僅於對應的檢視器請求不包含 Authorization 標題時簽署原始伺服器請求，請使用此設定。利用此設定，當檢視器請求存在時，CloudFront 會傳遞來自檢視器請求的 Authorization 標題，但在檢視器請求不包含 Authorization 標題時對原始伺服器請求進行簽名 (新增其自己的 Authorization 標題)。

Warning

- 如果您使用此設定，則必須為 Lambda 函數 URL 指定簽章第 4 版簽署，而不是 CloudFront 分佈的名稱或 CNAME。當 CloudFront 將 Authorization 標頭從檢視器請求轉送至 Lambda 函數 URL 時，Lambda 會根據 Lambda URL 網域的主機驗證簽署。如果簽署不是以 Lambda URL 網域為基礎，則簽署中的主機與 Lambda URL 原始伺服器所使用的主機不相符。這表示請求將會失敗，導致簽署驗證錯誤。
- 如要從檢視器請求傳遞 Authorization 標題，您必須將 Authorization 標題新增至 [快取政策](#) 中，適用於使用與此原始存取控制相關聯之 Lambda 函數 URL 的所有快取行為。

範本程式碼範例

如果您的 CloudFront 原始伺服器是與 OAC 相關聯的 Lambda 函數 URL，您可以使用下列 Python 指令碼，透過 POST 方法將檔案上傳至 Lambda 函數。

此程式碼假設您已將預設簽署行為設定為永遠簽署原始請求的 OAC，且未選取請勿覆寫授權標頭設定。

此組態允許 OAC 使用 Lambda 主機名稱，以 Lambda 正確管理 SigV4 授權。承載是從 Lambda 函數 URL 授權的 IAM 身分使用 SigV4 簽署，該身分指定為 IAM_AUTH 類型。

範本示範如何處理來自用戶端 POST 請求的 x-amz-content-sha256 標頭中簽署的承載雜湊值。具體而言，此範本旨在管理表單資料承載。範本可透過 CloudFront 將安全檔案上傳至 Lambda 函數 URL，並使用 AWS 身分驗證機制來確保只有授權的請求才能存取 Lambda 函數。

 程式碼包含下列功能：

- 符合在 x-amz-content-sha256 標頭中包含承載雜湊的需求
- 使用 SigV4 身分驗證進行安全 AWS 服務 存取
- 支援使用分段表單資料上傳檔案
- 包含請求例外狀況的錯誤處理

```
import boto3
from botocore.auth import SigV4Auth
from botocore.awsrequest import AWSRequest
import requests
import hashlib
import os

def calculate_body_hash(body):
    return hashlib.sha256(body).hexdigest()

def sign_request(request, credentials, region, service):
    sigv4 = SigV4Auth(credentials, service, region)
    sigv4.add_auth(request)

def upload_file_to_lambda(cloudfront_url, file_path, region):
    # AWS credentials
    session = boto3.Session()
    credentials = session.get_credentials()

    # Prepare the multipart form-data
    boundary = "-----boundary"

    # Read file content
    with open(file_path, 'rb') as file:
        file_content = file.read()

    # Get the filename from the path
    filename = os.path.basename(file_path)

    # Prepare the multipart body
```

```
body = (
    f'--{boundary}\r\n'
    f'Content-Disposition: form-data; name="file"; filename="{filename}"\r\n'
    f'Content-Type: application/octet-stream\r\n\r\n'
).encode('utf-8')
body += file_content
body += f'\r\n--{boundary}--\r\n'.encode('utf-8')

# Calculate SHA256 hash of the entire body
body_hash = calculate_body_hash(body)

# Prepare headers
headers = {
    'Content-Type': f'multipart/form-data; boundary={boundary}',
    'x-amz-content-sha256': body_hash
}

# Create the request
request = AWSRequest(
    method='POST',
    url=cloudfront_url,
    data=body,
    headers=headers
)

# Sign the request
sign_request(request, credentials, region, 'lambda')

# Get the signed headers
signed_headers = dict(request.headers)

# Print request headers before sending
print("Request Headers:")
for header, value in signed_headers.items():
    print(f"{header}: {value}")

try:
    # Send POST request with signed headers
    response = requests.post(
        cloudfront_url,
        data=body,
        headers=signed_headers
    )
```

```
# Print response status and content
print(f"\nStatus code: {response.status_code}")
print("Response:", response.text)

# Print response headers
print("\nResponse Headers:")
for header, value in response.headers.items():
    print(f"{header}: {value}")

except requests.exceptions.RequestException as e:
    print(f"An error occurred: {e}")

# Usage
cloudfront_url = "https://d1111111abcdef8.cloudfront.net"
file_path = r"filepath"
region = "us-east-1" # example: "us-west-2"

upload_file_to_lambda(cloudfront_url, file_path, region)
```

限制對 Amazon S3 原始伺服器的存取

CloudFront 提供兩種方式，將驗證請求傳送至 Amazon S3 原始伺服器：原始存取控制(OAC) 和原始存取身分 (OAI)。OAC 可協助您保護原始伺服器，例如 Amazon S3。

建議您改用 OAC，因為它支援下列功能：

- 所有 中的所有 Amazon S3 儲存貯體 AWS 區域，包括 2022 年 12 月之後啟動的選擇加入區域
- [使用 AWS KMS 的 Amazon S3 伺服器端加密 \(SSE-KMS\)](#)
- 對 Amazon S3 的動態請求 (PUT 和 DELETE)

OAI 不支援這些功能，或在這些案例中需要額外的解決方法。如果您已經在使用 OAI 且想要移轉，請參閱 [the section called “從原始存取身分 \(OAI\) 遷移至原始存取控制 \(OAC\)”](#)。

備註

- 當您將 CloudFront OAC 與 Amazon S3 儲存貯體原始伺服器搭配使用時，必須將 Amazon S3 物件擁有權設定為強制執行儲存貯體擁有者，這是新 Amazon S3 儲存貯體的預設值。如果您需要 ACL，請使用儲存貯體擁有者偏好設定來維持透過 CloudFront 上傳之物件的控制權。

- 如果您使用設定為[網站端點](#)的 Amazon S3 儲存貯體，則必須使用 CloudFront 將其設定為自訂原始伺服器。這表示您無法使用 OAC (或 OAI)。OAC 不支援使用 Lambda@Edge 的原始伺服器重新導向。

下列主題說明如何將 OAC 與 Amazon S3 原始伺服器搭配使用。

主題

- [the section called “建立新的原始存取控制”](#)
- [the section called “刪除含附加至 S3 儲存貯體之 OAC 的分佈”](#)
- [the section called “從原始存取身分 \(OAI\) 遷移至原始存取控制 \(OAC\)”](#)
- [the section called “原始存取控制的進階設定”](#)

建立新的原始存取控制

完成下列主題中說明的步驟，在 CloudFront 中設定新的原始存取控制。

主題

- [先決條件](#)
- [授予 CloudFront 存取 S3 儲存貯體的許可](#)
- [建立原始存取控制](#)

先決條件

在建立和設定原始存取控制 (OAC) 之前，您必須擁有具 Amazon S3 儲存貯體原始伺服器的 CloudFront 分佈。此原始伺服器必須是一般 S3 儲存貯體，而非設定為[網站端點](#)的儲存貯體。如需使用 S3 儲存貯體原始伺服器設定 CloudFront 分佈的相關資訊，請參閱 [the section called “開始使用標準分佈”](#)。

Important

當您使用 OAC 來保護 Amazon S3 原始伺服器時，CloudFront 和 Amazon S3 之間的通訊一律透過 HTTPS 進行，但前提是您選擇一律簽署請求。您必須在主控台中選擇簽署請求（建議），或在 CloudFront API、AWS CLI 或 CloudFormation `always` 中指定。

如果您選擇不簽署請求或不覆寫授權標頭選項，CloudFront 會使用您在下列政策中指定的連線通訊協定：

- [檢視器通訊協定政策](#)
- [原始伺服器通訊協定政策](#) (僅限自訂原始伺服器)

例如，如果您選擇不要覆寫授權標頭，並想要在 CloudFront 和 Amazon S3 原始伺服器之間使用 HTTPS，請僅針對[檢視器通訊協定政策](#)使用重新導向 HTTP 至 HTTPS 或僅限 HTTPS。

授予 CloudFront 存取 S3 儲存貯體的許可

在您建立原始存取控制 (OAC) 或於 CloudFront 分佈中進行設定之前，請確定 CloudFront 具有存取 S3 儲存貯體原始伺服器的許可。在建立 CloudFront 分佈之後，但在將 OAC 新增至分佈組態中的 S3 原始伺服器之前，請執行此作業。

使用 [S3 儲存貯體政策](#) 允許 CloudFront 服務主體 (cloudfront.amazonaws.com) 存取儲存貯體。使用政策中的 Condition 元素，僅當請求代表包含 S3 原始伺服器的 CloudFront 分佈時，才允許 CloudFront 存取儲存貯體。這是您要新增 OAC 的 S3 原始伺服器分佈。

如需新增或修改儲存貯體政策的相關資訊，請參閱 [Amazon S3 使用者指南](#) 中的使用 Amazon S3 主控台新增儲存貯體政策。

下列是允許啟用 OAC 之 CloudFront 分佈存取 S3 原始伺服器的 S3 儲存貯體政策範例。

Example S3 儲存貯體政策，允許啟用 OAC 之 CloudFront 分佈的唯讀存取

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadOnly",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
```

```

    "AWS:SourceArn":
    "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
  }
}
]
}

```

Example S3 儲存貯體政策，允許啟用 OAC 之 CloudFront 分佈的讀寫存取

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
            "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
        }
      }
    }
  ]
}

```

SSE-KMS

如果使用[伺服器端加密搭配 AWS Key Management Service \(SSE-KMS\) 來加密](#) S3 儲存貯體原始伺服器中的物件，您必須確保 CloudFront 分佈具有使用 AWS KMS 金鑰的許可。如要授予 CloudFront 分

佈使用 KMS 金鑰的許可，請將陳述式新增至 [KMS 金鑰政策](#)。如需如何修改金鑰政策的相關資訊，請參閱《AWS Key Management Service 開發人員指南》中的 [變更金鑰政策](#)。

Example KMS 金鑰政策陳述式

下列範例顯示 AWS KMS 政策陳述式，允許具有 OAC 的 CloudFront 分佈存取 SSE-KMS 的 KMS 金鑰。

```
{
  "Sid": "AllowCloudFrontServicePrincipalSSE-KMS",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "cloudfront.amazonaws.com"
    ]
  },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "AWS:SourceArn":
        "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
    }
  }
}
```

建立原始存取控制

若要建立原始存取控制 (OAC) CloudFormation，您可以使用 AWS 管理主控台、AWS CLI、或 CloudFront API。

Console

如要建立原始存取控制

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 於左側導覽窗格中，選擇 Origin access (原始存取)。

3. 選擇 Create control setting (建立控制設定)。
4. 在 Create control setting (建立控制設定) 表單上，執行下列動作：
 - a. 於 Details (詳細資訊) 窗格中，輸入 Name (名稱) 和 (選用) Description (描述)，以用於原始存取控制。
 - b. 於 Settings (設定) 窗格中，建議您保留預設設定 (Sign requests (recommended)) (簽署請求 (建議使用))。如需詳細資訊，請參閱[the section called “原始存取控制的進階設定”](#)。
5. 從 Origin type (原始伺服器類型) 下拉式功能表中選擇 S3。
6. 選擇建立。

建立 OAC 之後，請記下 Name (名稱)。您需要於下列程序中進行使用。

如要將原始存取控制新增至分佈中的 S3 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇一個您想要新增 OAC 之具 S3 原始伺服器的分佈，然後選擇 Origins (原始伺服器) 標籤。
3. 選取您想要將 OAC 新增至的 S3 原始伺服器，然後選擇 Edit (編輯)。
4. 針對原始存取，選擇原始存取控制設定 (建議使用)。
5. 在 Origin access control (原始存取控制) 下拉式功能表中，選擇您想要使用的 OAC。
6. 選擇儲存變更。

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 S3 儲存貯體原始伺服器的所有請求。

CloudFormation

若要使用 建立原始存取控制 (OAC) CloudFormation，請使用 `AWS::CloudFront::OriginAccessControl` 資源類型。下列範例顯示用於建立原始存取控制的 YAML 格式 CloudFormation 範本語法。

```
Type: AWS::CloudFront::OriginAccessControl
Properties:
  OriginAccessControlConfig:
    Description: An optional description for the origin access control
    Name: ExampleOAC
    OriginAccessControlOriginType: s3
    SigningBehavior: always
```

SigningProtocol: sigv4

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [AWS::CloudFront::OriginAccessControl](#)。

CLI

若要使用 AWS Command Line Interface (AWS CLI) 建立原始存取控制，請使用 `aws cloudfront create-origin-access-control` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不必分別將每個個別參數指定為命令列輸入。

如要建立原始存取控制 (包含輸入檔案的 CLI)

1. 使用下列命令建立名為 `origin-access-control.yaml` 的檔案。這個檔案中包含 `create-origin-access-control` 命令的所有輸入參數。

```
aws cloudfront create-origin-access-control --generate-cli-skeleton yaml-input >
origin-access-control.yaml
```

2. 開啟您剛才建立的 `origin-access-control.yaml` 檔案。編輯檔案以新增 OAC 的名稱、說明 (選用)，並將 `SigningBehavior` 變更為 `always`。接著儲存檔案。

如需其他 OAC 設定的相關資訊，請參閱 [the section called “原始存取控制的進階設定”](#)。

3. 使用下列命令，利用 `origin-access-control.yaml` 檔案中的輸入參數建立原始存取控制。

```
aws cloudfront create-origin-access-control --cli-input-yaml file://origin-
access-control.yaml
```

記下命令輸出中的 `Id` 值，您需要其將 OAC 新增至 CloudFront 分佈中的 S3 儲存貯體原始伺服器。

如要將 OAC 附加至現有分佈 (包含輸入檔案的 CLI) 中的 S3 儲存貯體原始伺服器

1. 使用下列命令來儲存您想要新增之 CloudFront 分佈的分佈組態。分佈必須具有 S3 儲存貯體原始伺服器。

```
aws cloudfront get-distribution-config --id <CloudFront distribution ID> --  
output yaml > dist-config.yaml
```

2. 開啟您剛才建立且命名為 `dist-config.yaml` 的檔案。編輯檔案，進行下列變更：

- 於 `Origins` 物件中，將 `OAC` 的 ID 新增至名為 `OriginAccessControlId` 的欄位。
- 從名為 `OriginAccessIdentity` 的欄位中移除值(如果存在)。
- 將 `ETag` 欄位重新命名為 `IfMatch`，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用原始存取控制。

```
aws cloudfront update-distribution --id <CloudFront distribution ID> --cli-  
input-yaml file://dist-config.yaml
```

分佈開始部署至所有 CloudFront 邊緣節點。當邊緣節點接收到新組態時，其會簽署傳送至 S3 儲存貯體原始伺服器的所有請求。

API

如要使用 CloudFront API 建立原始存取控制，請使用 [CreateOriginAccessControl](#)。如需您在此 API 呼叫中指定欄位的詳細資訊，請參閱 AWS SDK 或其他 API 用戶端的 API 參考文件。

建立原始存取控制之後，您可以使用下列其中一個 API 呼叫，將其連接至分佈中的 S3 儲存貯體原始伺服器：

- 如要將其附加至現有分佈中，請使用 [UpdateDistribution](#)。
- 如要將其附加至新分佈中，請使用 [CreateDistribution](#)。

對於這兩個 API 呼叫，請於原始伺服器內部的 `OriginAccessControlId` 欄位中提供原始存取控制 ID。如需您在這些 API 呼叫中指定之其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS SDK 或其他 API 用戶端的 API 參考文件。

刪除含附加至 S3 儲存貯體之 OAC 的分佈

如果您需要刪除含附加至 S3 儲存貯體之 OAC 的分佈，您應該先刪除分佈，再刪除 S3 儲存貯體原始伺服器。或者，在原始伺服器網域名稱中包含區域。如果無法這麼做，您可以在刪除之前切換到公有，從分佈中移除 OAC。如需詳細資訊，請參閱[刪除分發](#)。

從原始存取身分 (OAI) 遷移至原始存取控制 (OAC)

如要從舊式原始存取身分 (OAI) 移轉至原始存取控制 (OAC)，請先更新 S3 儲存貯體原始伺服器，以允許 OAI 和啟用 OAC 之分佈存取儲存貯體的內容。此可確保 CloudFront 在轉換期間永遠不會失去儲存貯體的存取權限。如要允許 OAI 和啟用 OAC 之分佈存取 S3 儲存貯體，請更新[儲存貯體政策](#)，以包括兩個陳述式，每種主體各一個。

下列範例 S3 儲存貯體政策允許 OAI 和啟用 OAC 之分佈存取 S3 原始伺服器。

Example 允許對 OAI 和啟用 OAC 之 CloudFront 分佈進行唯讀存取的 S3 儲存貯體政策

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudFrontServicePrincipalReadOnly",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::<S3 bucket name>/*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn":
            "arn:aws:cloudfront::111122223333:distribution/<CloudFront distribution ID>"
        }
      }
    },
    {
      "Sid": "AllowLegacyOAIReadOnly",
      "Effect": "Allow",
      "Principal": {
```

```

        "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
Identity <origin access identity ID>"
    },
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::<S3 bucket name>/*"
  }
]
}

```

更新 S3 原始伺服器的儲存貯體政策以允許存取 OAI 和 OAC 之後，您可更新發佈組態以使用 OAC 而非 OAI。如需詳細資訊，請參閱[the section called “建立新的原始存取控制”](#)。

完整部署分佈之後，您可移除儲存貯體政策中允許存取 OAI 的陳述式。如需詳細資訊，請參閱[the section called “授予 CloudFront 存取 S3 儲存貯體的許可”](#)。

原始存取控制的進階設定

CloudFront 來原始存取控制功能包含僅適用於特定使用案例的進階設定。除非您對進階設定有特定需求，否則請使用建議的設定。

原始存取控制包含名為簽署行為（在主控台中）或 `SigningBehavior`（在 API、CLI 和 中）的設定 CloudFormation。此設定提供下列選項：

永遠簽署原始請求 (建議設定)

我們建議使用此設定，於主控台中名為 `Sign requests (recommended)` (簽署請求 (建議使用))，或於 API、CLI 和 CloudFormation 中的 `always`。使用此設定時，CloudFront 一律會簽署傳送至 S3 儲存貯體原始伺服器的所有請求。

絕不簽署原始伺服器請求

此設定於主控台中命名為 `Do not sign requests` (請勿簽署請求)，或 API、CLI 和 CloudFormation 中的 `never`。使用此設定，關閉使用此原始存取控制之所有分佈中的所有原始伺服器的原始存取控制。與從所有使用其原始伺服器和分佈中逐一移除原始存取控制相比，此可節省時間和精力。使用此設定時，CloudFront 不會簽署傳送至 S3 儲存貯體原始伺服器的任何請求。

Warning

如要使用此設定，S3 儲存貯體原始伺服器必須可公開存取。如果您對無法公開存取的 S3 儲存貯體原始伺服器使用此設定，CloudFront 將無法存取該原始伺服器。S3 儲存貯體原始伺服器會將錯誤傳回 CloudFront，而 CloudFront 會將這些錯誤傳遞給檢視器。

請勿覆寫檢視器 (用戶端) **Authorization** 標題

此設定於主控台中命名為 Do not override authorization header (請勿覆寫授權標頭)，或於 API、CLI 和 CloudFormation 中的 no-override。如果您想要 CloudFront 僅於對應的檢視器請求不包含 Authorization 標題時簽署原始伺服器請求，請使用此設定。利用此設定，當檢視器請求存在時，CloudFront 會傳遞來自檢視器請求的 Authorization 標題，但在檢視器請求不包含 Authorization 標題時對原始伺服器請求進行簽名 (新增其自己的 Authorization 標題)。

Warning

如要從檢視器請求傳遞 Authorization 標題，您必須將 Authorization 標題新增至[快速存取政策](#)中，適用於使用與此原始存取控制相關聯之 S3 儲存貯體原始伺服器的所有快取行為。

使用原始存取身分 (舊版，不建議使用)

原始存取身分概觀

CloudFront 原始存取身分 (OAI) 提供與原始存取控制 (OAC) 類似的功能，但其不適用於所有案例。具體而言，OAI 不支援：

- 所有 中的 Amazon S3 儲存貯體 AWS 區域，包括選擇加入區域
- [使用 AWS KMS 的 Amazon S3 伺服器端加密](#) (SSE-KMS)
- 對 Amazon S3 的動態請求 (PUT、POST 或 DELETE)
- 2023 年 1 月之後 AWS 區域 推出的新功能

Tip

我們建議您改用 OAC。若要設定 OAC，請參閱 [建立新的原始存取控制](#)。如需如何從 OAI 遷移至 OAC 的相關資訊，請參閱 [the section called “從原始存取身分 \(OAI\) 遷移至原始存取控制 \(OAC\)”](#)。

授予原始存取身分的許可，以讀取 Amazon S3 儲存貯體中的檔案

當您使用 CloudFront 主控台建立 OAI 或新增 OAI 至分佈時，您可以自動更新 Amazon S3 儲存貯體政策，讓 OAI 有權存取您的儲存貯體。或者，您可以選擇手動建立或更新儲存貯體政策。無論您使用哪種方法，您仍應檢閱許可以確認下列事項：

- CloudFront OAI 可以代表透過 CloudFront 請求檔案的檢視器，來存取儲存貯體中的檔案。
- 檢視器無法使用 Amazon S3 URL 在 CloudFront 之外存取您的檔案。

Important

如果您將 CloudFront 設定為接受和轉送 CloudFront 支援的所有 HTTP 方法，請務必為您的 CloudFront OAI 提供所需的許可。例如，如果您將 CloudFront 設定為接受並轉送使用 DELETE 方法的請求，請將儲存貯體政策設定為適當地處理 DELETE 請求，讓檢視器只能刪除您希望其刪除的檔案。

使用 Amazon S3 儲存貯體政策

您可以透過下列方法建立或更新儲存貯體政策，藉此授予 CloudFront OAI 對 Amazon S3 儲存貯體中檔案的存取權限：

- 使用 [Amazon S3 主控台](#) 中的 Amazon S3 儲存貯體的 Permissions (許可) 標籤。
- 在 Amazon S3 API 中使用 [PutBucket 政策](#)。
- 使用 [CloudFront 主控台](#)。當您將 OAI 新增到 CloudFront 主控台下的原始伺服器設定時，您可以選擇 Yes, update the bucket 政策 (是，更新儲存貯體政策)，請 CloudFront 代表您更新儲存貯體政策。

如果您手動更新儲存貯體政策，請務必：

- 在政策中指定正確的 OAI 做為 Principal。
- 授予 OAI 代表檢視器存取物件所需的許可。

如需詳細資訊，請參閱下列區段。

在儲存貯體政策中指定 OAI 做為 **Principal**

若要於 Amazon S3 儲存貯體政策中指定 OAI 做為 Principal，請使用包含 OAI ID 的 OAI Amazon Resource Name (ARN)。例如：

```
"Principal": {
  "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity <origin
access identity ID>"
}
```

在安全性、原始存取、身分 (舊版) 下的 CloudFront 主控台中尋找 OAI ID。或者，在 CloudFront API 中使用 [ListCloudFrontOriginAccessIdentities](#)。

授予權限給一個 OAI

如要授予 OAI 存取 Amazon S3 儲存貯體中物件的許可，請使用與特定 Amazon S3 API 作業相關政策中的動作。例如，s3:GetObject 動作可讓 OAI 讀取儲存貯體中的物件。如需詳細資訊，請參閱以下章節中的範例，或參閱 Amazon Simple Storage Service 使用者指南中的 [Amazon S3 動作](#)。

Amazon S3 儲存貯體政策範例

下列範例顯示允許 CloudFront OAI 存取 S3 儲存貯體的 Amazon S3 儲存貯體政策。

在安全性、原始存取、身分 (舊版) 下的 CloudFront 主控台中尋找 OAI ID。或者，在 CloudFront API 中使用 [ListCloudFrontOriginAccessIdentities](#)。

Example 授予 OAI 讀取存取權限的 Amazon S3 儲存貯體政策

下列範例可讓 OAI 讀取指定儲存貯體中的物件 (s3:GetObject)。

JSON

```
{
  "Version": "2012-10-17",
  "Id": "PolicyForCloudFrontPrivateContent",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
Identity <origin access identity ID>"
      }
    }
  ]
}
```

```

    },
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::<S3 bucket name>/*"
  }
]
}

```

Example 授予 OAI 讀取和寫入存取權限的 Amazon S3 儲存貯體政策

下列範例可讓 OAI 讀取和寫入指定儲存貯體中的物件 (s3:GetObject 和 s3:PutObject)。這可讓檢視器透過 CloudFront 將檔案上傳到您的 Amazon S3 儲存貯體。

JSON

```

{
  "Version": "2012-10-17",
  "Id": "PolicyForCloudFrontPrivateContent",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
Identity <origin access identity ID>"
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::<S3 bucket name>/*"
    }
  ]
}

```

使用 Amazon S3 物件 ACL (不建議)

Important

我們建議[使用 Amazon S3 儲存貯體政策](#)向 OAI 提供 S3 儲存貯體的存取權。您可以使用存取控制清單 (ACL)，如本節中所述，但我們不建議此方法。

Amazon S3 建議設定 [S3 物件擁有權](#) 為已強制執行儲存貯體擁有者，這意味著儲存貯體及其中物件的 ACL 被停用。當您將此設定套用至「物件擁有權」時，您必須使用儲存貯體政策來授與 OAI 的存取權 (請參閱上一節)。

本節僅適用於需要 ACL 的舊式使用案例。

您可透過下列方法建立或更新檔案的 ACL，藉此授予 CloudFront OAI 對 Amazon S3 儲存貯體中檔案的存取權限：

- 使用 [Amazon S3 主控台](#) 中 Amazon S3 物件的 Permissions (許可) 標籤。
- 使用 Amazon S3 API 中的 [PutObjectAcl](#)。

當您使用 ACL 授予對 OAI 的存取權限時，您必須使用 OAI 的 Amazon S3 正式使用者 ID 來指定 OAI。在 CloudFront 主控台中，您可以在安全性、原始存取、身分 (舊版) 下找到此 ID。如果您使用的是 CloudFront API，請使用 S3CanonicalUserId 元素的值，此元素是在您建立 OAI 或呼叫 CloudFront API 中的 [ListCloudFrontOriginAccessIdentities](#) 時傳回。

在僅支援簽署版本 4 驗證的 Amazon S3 區域中使用原始存取身分

較新 Amazon S3 區域請求您使用簽章版本 4 進行驗證請求 (如需每個 Amazon S3 區域支援的簽署版本，請參閱《AWS 一般參考》中的 [Amazon Simple Storage Service 端點和配額](#)。) 如果您使用的是原始存取身分，且您的儲存貯體位於需要簽章版本 4 的其中一個區域，請注意下列事項：

- DELETE、GET、HEAD、OPTIONS 和 PATCH 請求可以在沒有授權的情況下得到支援。
- 不支援 POST 請求。

使用 VPC 原始伺服器限制存取

您可以使用 CloudFront 從虛擬私有雲端 (VPC) 私有子網路中託管的應用程式交付內容。您可以使用 Application Load Balancer (ALB)、Network Load Balancer (NLB) 和私有子網路中的 EC2 執行個體做為 VPC 原始伺服器。

以下是您可能想要使用 VPC 原始伺服器的一些原因：

- 安全 – VPC 原始伺服器旨在透過將負載平衡器和 EC2 執行個體放置在私有子網路中，使 CloudFront 成為單一進入點，來增強應用程式的安全狀態。使用者請求會透過私有、安全的連線從 CloudFront 轉移到 VPC 原始伺服器，為您的應用程式提供額外的安全性。

- 管理 – VPC 原始伺服器可減少 CloudFront 與原始伺服器之間安全連線所需的營運開銷。您可以將原始伺服器移至沒有公有存取權的私有子網路，而且您不需要實作存取控制清單 (ACL) 或其他機制來限制對原始伺服器的存取。如此一來，您就不必投資未差異化的開發工作，即可使用 CloudFront 保護您的 Web 應用程式。
- 可擴展性和效能 – VPC 原始伺服器可協助您保護 Web 應用程式，騰出時間專注於擴展關鍵業務應用程式，同時透過 CloudFront 提高安全性和維護高效能和全球可擴展性。VPC 原始伺服器可簡化安全管理並降低操作複雜性，讓您可以使用 CloudFront 做為應用程式的單一進入點。

Tip

CloudFront 支援跨 共用 VPC 原始伺服器 AWS 帳戶，無論它們是否在您的組織中。您可以從 CloudFront 主控台共用 VPC 原始伺服器或使用 AWS Resource Access Manager (AWS RAM)。如需詳細資訊，請參閱[在 CloudFront 中使用共用資源](#)。

先決條件

為 CloudFront 分佈建立 VPC 原始伺服器之前，您必須完成下列操作：

- 在 Amazon VPC 上建立虛擬私有雲端 (VPC)。
 - 您的 VPC 必須位於 VPC 原始伺服器 AWS 區域 支援的其中一個 中。如需詳細資訊，請參閱[AWS 區域 支援 VPC 原始伺服器](#)。
 - 當您的 VPC 原始伺服器啟用用戶端 IP 位址保留時，與您的 VPC 子網路相關聯的網路 ACL 會套用至輸出 (傳出) 流量。不過，若要讓流量透過 VPC 原始伺服器結束，您必須將 ACL 設定為傳入和傳出規則。

例如，若要允許使用暫時性原始伺服器連接埠的 TCP 和 UDP 用戶端透過 VPC 原始伺服器連線至您的端點，請將端點子網路與網路 ACL 建立關聯，以允許目的地為暫時性 TCP 或 UDP 連接埠的傳出流量 (連接埠範圍 1024-65535，目的地 0.0.0.0/0)。此外，建立相符的傳入規則 (連接埠範圍 1024-65535，原始伺服器 0.0.0.0/0)。

如需有關建立 VPC 的資訊，請參閱《Amazon VPC 使用者指南》中的[建立 VPC 和其他 VPC 資源](#)。

- 在您的 VPC 中包含下列項目：
 - 網際網路閘道 – 您需要將網際網路閘道新增至具有 VPC 原始資源的 VPC。需要網際網路閘道才能表示 VPC 可以從網際網路接收流量。網際網路閘道不會用於將流量路由到子網路內的原始伺服器，而且您不需要更新路由政策。

- 具有至少一個可用 IPv4 位址的私有子網路 – CloudFront 會使用 CloudFront 在您使用 CloudFront 定義 VPC 原始伺服器資源後建立的服務受管彈性網路介面 (ENI)，路由到您的子網路。您的私有子網路中必須至少有一個可用的 IPv4 位址，以成功完成 ENI 建立程序。IPv4 位址可以是私有位址，而且無需額外費用。

 Note

不支援 IPv6-only 子網路。

- 在私有子網路中，啟動 Application Load Balancer、Network Load Balancer 或 EC2 執行個體以做為原始伺服器。
- 您啟動的資源必須完全部署且處於作用中狀態，才能將其用於 VPC 原始伺服器。
- Gateway Load Balancer、雙堆疊 Network Load Balancer 和具有 TLS 接聽程式的 Network Load Balancer 無法新增為原始伺服器。
- 若要用作 VPC 原始伺服器，Network Load Balancer 必須連接安全群組。
- 更新 VPC 私有原始伺服器的安全群組，以明確允許 CloudFront 受管字首清單。如需詳細資訊，請參閱[使用 CloudFront 受管字首清單](#)。

 Note

CloudFront-VPCOrigins-Service-SG 是用於 VPC 原始伺服器之安全群組的 AWS 預留名稱。您必須為安全群組指定不同的名稱。如需詳細資訊，請參閱[建立安全群組](#)。

- 建立 VPC 原始伺服器之後，安全群組可以進一步限制為僅允許來自 VPC 原始伺服器的流量。若要這樣做，請將允許流量原始伺服器從受管字首清單更新為 CloudFront 安全群組。

 Note

VPC 原始伺服器不支援在 CloudFront 中使用 Lambda@Edge 的 WebSockets、gRPC 流量、原始伺服器請求和原始伺服器回應觸發條件。如需詳細資訊，請參閱 Lambda@Edge 文件中的 [使用請求和回應](#)。

建立 VPC 原始伺服器 (新分佈)

下列程序說明如何在 CloudFront 主控台中為新的 CloudFront 分佈建立 VPC 原始伺服器。或者，您可以使用 [CreateVpcOrigin](#) 和 [CreateDistribution](#) API 操作搭配 AWS CLI 或 AWS SDK。

為新的 CloudFront 分佈建立 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇 VPC 原始伺服器、建立 VPC 原始伺服器。
3. 填妥必要的欄位。針對原始伺服器 ARN，選取 Application Load Balancer、Network Load Balancer 或 EC2 執行個體的 ARN。如果您沒有看到 ARN，您可以複製特定資源 ARN 並將其貼到這裡。
4. 選擇建立 VPC 原始伺服器。
5. 等待您的 VPC 原始伺服器狀態變更為已部署。此程序最多需要 15 分鐘的時間。
6. 選擇分佈、建立分佈。
7. 對於原始伺服器網域，請從下拉式清單中選取您的 VPC 原始伺服器資源。

如果您的 VPC 原始伺服器是 EC2 執行個體，請將執行個體的私有 IP DNS 名稱複製並貼到原始伺服器網域欄位中。

8. 完成建立分佈。如需詳細資訊，請參閱[在主控台中建立 CloudFront 分佈](#)。

建立 VPC 原始伺服器 (現有分佈)

下列程序說明如何在 CloudFront 主控台中為現有的 CloudFront 分佈建立 VPC 原始伺服器，這有助於確保應用程式的持續可用性。或者，您可以使用 [CreateVpcOrigin](#) 和 [UpdateDistributionWithStagingConfig](#) API 操作搭配 AWS CLI 或 AWS SDK。

或者，您可以選擇將 VPC 原始伺服器新增至現有的分佈，而無需建立臨時分佈。

為現有的 CloudFront 分佈建立 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇 VPC 原始伺服器、建立 VPC 原始伺服器。
3. 填妥必要的欄位。針對原始伺服器 ARN，選取 Application Load Balancer、Network Load Balancer 或 EC2 執行個體的 ARN。如果您沒有看到 ARN，您可以複製特定資源 ARN 並將其貼到這裡。
4. 選擇建立 VPC 原始伺服器。

5. 等待您的 VPC 原始伺服器狀態變更為已部署。此程序最多需要 15 分鐘的時間。
6. 在導覽窗格中，選擇 Distributions (分佈)。
7. 選擇您分佈的 ID。
8. 在一般索引標籤的持續部署下，選擇建立臨時分佈。如需詳細資訊，請參閱[使用 CloudFront 持續部署，安全地測試 CDN 組態變更](#)。
9. 依照建立臨時分佈精靈中的步驟來建立臨時分佈。包括下列步驟：
 - 對於原始伺服器，選擇建立原始伺服器。
 - 對於原始伺服器網域，請從下拉式功能表中選取您的 VPC 原始伺服器資源。

如果您的 VPC 原始伺服器是 EC2 執行個體，請將執行個體的私有 IP DNS 名稱複製並貼到原始伺服器網域欄位中。
 - 選擇 Create Origin (建立原始伺服器)。
10. 在您的臨時分佈中，測試 VPC 原始伺服器。
11. 將臨時分佈組態提升為主要分佈。如需詳細資訊，請參閱[提升臨時分佈組態](#)。
12. 將子網路設為私有，以移除對 VPC 原始伺服器的公開存取。執行此操作後，將無法透過網際網路探索 VPC 原始伺服器，但 CloudFront 仍然可以進行私有存取。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的[將子網路與路由表建立關聯或取消關聯](#)。

更新 VPC 原始伺服器

下列程序說明如何在 CloudFront 主控台中更新 CloudFront 分佈的 VPC 原始伺服器。或者，您可以使用 [UpdateDistribution](#) 和 [UpdateVpcOrigin](#) API 操作搭配 AWS CLI 或 AWS SDK。

更新 CloudFront 分佈的現有 VPC 原始伺服器

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 在導覽窗格中，選擇 Distributions (分佈)。
3. 選擇您分佈的 ID。
4. 選擇 Behaviors (動作) 索引標籤。
5. 請確定 VPC 原始伺服器不是快取行為的預設原始伺服器。
6. 選擇 Origins (原始伺服器) 索引標籤。
7. 選取您要更新的 VPC 原始伺服器，然後選擇刪除。這會取消 VPC 原始伺服器與分佈的關聯。重複步驟 2 至 7，以取消 VPC 原始伺服器與任何其他分佈的關聯。
8. 選擇 VPC 原始伺服器。

9. 選取 VPC 原始伺服器，然後選擇編輯。
10. 進行更新，然後選擇更新 VPC 原始伺服器。
11. 等待您的 VPC 原始伺服器狀態變更為已部署。此程序最多需要 15 分鐘的時間。
12. 在導覽窗格中，選擇 Distributions (分佈)。
13. 選擇您分佈的 ID。
14. 選擇 Origins (原始伺服器) 索引標籤。
15. 選擇 Create Origin (建立原始伺服器)。
16. 對於原始伺服器網域，請從下拉式功能表中選取您的 VPC 原始伺服器資源。

如果您的 VPC 原始伺服器是 EC2 執行個體，請將執行個體的私有 IP DNS 名稱複製並貼到原始伺服器網域欄位中。

17. 選擇 Create Origin (建立原始伺服器)。這會再次將 VPC 原始伺服器與您的分佈建立關聯。重複步驟 12 至 17，將更新的 VPC 原始伺服器與任何其他分佈建立關聯。

AWS 區域 支援 VPC 原始伺服器

下列商業商品目前支援 VPC 原始伺服器 AWS 區域。會記下可用區域 (AZ) 例外狀況。

區域名稱	區域
美國東部 (俄亥俄)	us-east-2
美國東部 (維吉尼亞北部)	us-east-1 (except AZ use1-az3)
美國西部 (加利佛尼亞北部)	us-west-1 (except AZ usw1-az2)
美國西部 (奧勒岡)	us-west-2
非洲 (開普敦)	af-south-1
亞太區域 (香港)	ap-east-1
亞太地區 (孟買)	ap-south-1
亞太地區 (海德拉巴)	ap-south-2
亞太地區 (雅加達)	ap-southeast-3

區域名稱	區域
亞太地區 (墨爾本)	ap-southeast-4
亞太地區 (大阪)	ap-northeast-3
亞太地區 (新加坡)	ap-southeast-1
亞太地區 (雪梨)	ap-southeast-2
亞太地區 (東京)	ap-northeast-1 (except AZ apne1-az3)
亞太地區 (首爾)	ap-northeast-2 (except AZ apne2-az1)
加拿大 (中部)	ca-central-1 (except AZ cac1-az3)
加拿大西部 (卡加利)	ca-west-1
歐洲 (法蘭克福)	eu-central-1
歐洲 (愛爾蘭)	eu-west-1
歐洲 (倫敦)	eu-west-2
歐洲 (米蘭)	eu-south-1
Europe (Paris)	eu-west-3
歐洲 (西班牙)	eu-south-2
歐洲 (斯德哥爾摩)	eu-north-1
歐洲 (蘇黎世)	eu-central-2
以色列 (特拉維夫)	il-central-1
Middle East (Bahrain)	me-south-1
中東 (阿拉伯聯合大公國)	me-central-1

區域名稱	區域
南美洲 (聖保羅)	sa-east-1

限制對 Application Load Balancers 的存取

您可以將內部和面向網際網路的 Application Load Balancer 與 Amazon CloudFront 搭配使用。您可以使用 VPC 來源，將私有子網路之中的內部 Application Load Balancer 與 CloudFront 搭配使用。CloudFront VPC 來源可讓您從私有 VPC 子網路中託管的應用程式提供內容，而無需將其暴露至公共網際網路。如需詳細資訊，請參閱[使用 VPC 原始伺服器限制存取](#)。

如果您使用面向網際網路的 Application Load Balancer 搭配 CloudFront，您可以使用下列安全性緩解措施來防止使用者直接存取 Application Load Balancer，並僅允許透過 CloudFront 存取。

1. 設定 CloudFront 將自訂 HTTP 標頭新增至其傳送至 Application Load Balancer 的要求。
2. 將 Application Load Balancer 設定為僅轉寄包含自訂 HTTP 標頭的請求。
3. 需要 HTTPS 來改善此解決方案的安全性。

CloudFront 還可以幫助減少延遲，甚至吸收一些分散式阻斷服務 (DDoS) 攻擊。

如果您的使用案例需要直接透過網際網路從 CloudFront 和 Application Load Balancer 雙重存取 Web 應用程式，請考慮分割您的 Web 應用程式 API (如下所示)：

- API 必須通過 CloudFront。在此情況下，請考慮使用個別的私有 Application Load Balancer 做為來源。
- 需要透過 Application Load Balancer 存取的 API。在此情況下，您會略過 CloudFront。

或者，對於 ELB 中面向網際網路的 Application Load Balancer 提供的 Web 應用程式或其他內容，CloudFront 可以快取物件並將其直接提供給使用者（檢視者），從而減少 Application Load Balancer 上的負載。面對網際網路的負載平衡器具有可公開解析的 DNS 名稱，並會透過網際網路將用戶端的請求路由至目標。

如需詳細資訊，請參閱下列主題。完成這些步驟後，使用者只能透過 CloudFront 存取您的 Application Load Balancer。

主題

- [設定 CloudFront 將自訂 HTTP 標頭新增至請求](#)
- [將 Application Load Balancer 設定為僅轉寄包含特定標頭的請求](#)
- [\(選用\) 改善此解決方案的安全性](#)
- [\(選用\) 使用 CloudFront AWS 的受管字首清單限制對原始伺服器的存取](#)

設定 CloudFront 將自訂 HTTP 標頭新增至請求

您可以將 CloudFront 設定為將自訂 HTTP 標頭新增至其傳送至來源的請求 (在本例中為 Application Load Balancer)。

Important

此用例依賴於保留自訂標頭名稱和值密碼。如果標頭名稱和值不是密碼，其他 HTTP 用戶端可能會將它們包含在直接傳送至 Application Load Balancer 的請求中。這可能會導致 Application Load Balancer 的運作方式如同來自 CloudFront 的請求一樣，即便當它們並非來自於此。若要防止這種情況，請保留自訂標頭名稱和值密碼。

您可以使用 CloudFront 主控台 CloudFormation 或 CloudFront API，設定 CloudFront 將自訂 HTTP 標頭新增至原始伺服器請求。

新增自訂 HTTP 標頭 (CloudFront 主控台)

在 CloudFront 主控台中，使用原始伺服器設定中的原始伺服器自訂標頭設定。輸入標頭名稱及其值。

Note

在生產中，使用隨機產生的標頭名稱和值。將標頭名稱和值視為安全憑證，如使用者名稱和密碼。

當您建立或編輯現有 CloudFront 發佈的來源，以及建立新發佈時，您可以編輯來源自訂標頭設定。如需詳細資訊，請參閱[更新分佈](#)及[建立分佈](#)。

新增自訂 HTTP 標頭 (CloudFormation)

在 CloudFormation 範本中，使用 `OriginCustomHeaders` 屬性，如下列範例所示。

 Note

此範例中的標頭名稱和值僅為演示。在生產中，使用隨機產生的值。將標頭名稱和值視為安全登入資料，如使用者名稱和密碼。

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  TestDistribution:
    Type: 'AWS::CloudFront::Distribution'
    Properties:
      DistributionConfig:
        Origins:
          - DomainName: app-load-balancer.example.com
            Id: Example-ALB
            CustomOriginConfig:
              OriginProtocolPolicy: https-only
              OriginSSLProtocols:
                - TLSv1.2
            OriginCustomHeaders:
              - HeaderName: X-Custom-Header
                HeaderValue: random-value-1234567890
        Enabled: 'true'
        DefaultCacheBehavior:
          TargetOriginId: Example-ALB
          ViewerProtocolPolicy: allow-all
          CachePolicyId: 658327ea-f89d-4fab-a63d-7e88639e58f6
        PriceClass: PriceClass_All
        ViewerCertificate:
          CloudFrontDefaultCertificate: 'true'
```

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [Origin](#) 和 [OriginCustomHeader](#) 屬性。

新增自訂 HTTP 標頭 (CloudFront API)

在 CloudFront API 中，使用 Origin 中的 CustomHeaders 物件。如需詳細資訊，請參閱《Amazon CloudFront API 參考》中的 [CreateDistribution](#) 和 [UpdateDistribution](#)，以及 SDK 或其他 API 用戶端的文件。

有一些標頭名稱不能指定為來源自訂標頭。如需詳細資訊，請參閱[CloudFront 無法新增到原始伺服器請求的自訂標頭](#)。

將 Application Load Balancer 設定為僅轉寄包含特定標頭的請求

將 CloudFront 設定為將自訂 HTTP 標頭新增至其傳送至 Application Load Balancer 的請求之後 (請參閱[上一章節](#))，您可以將負載平衡器設定為僅轉寄包含此自訂標頭的請求。您可以透過在負載平衡器的接聽程式中新增規則並修改預設規則來執行此動作。

先決條件

若要使用下列程序，您必須具有至少一個接聽程式的 Application Load Balancer。如果您尚未建立，請參閱《[Application Load Balancer 使用者指南](#)》中的建立 Application Load Balancer。

下列程序會修改 HTTPS 接聽程式。您可以使用相同的程序來修改 HTTP 接聽程式。

更新 Application Load Balancer 接聽程式中的規則

1. 新增新規則。使用[新增規則](#)中的指示，並進行下列修改：
 - 將規則新增至 CloudFront 分佈來源的負載平衡器。
 - 針對新增條件選擇 Http 標頭。指定您在 CloudFront 中新增為來源自訂標頭的 HTTP 標頭名稱和值。
 - 針對新增動作選擇轉送至。選擇要轉寄請求的目標群組。
2. 編輯負載平衡器接聽程式中的預設規則。使用[編輯規則](#)中的指示，並進行下列修改：
 - 編輯 CloudFront 分佈來源的負載平衡器預設規則。
 - 刪除預設動作，然後針對新增動作選擇傳回固定回應。
 - 在回應代碼中，輸入 **403**。
 - 在回應主體中，輸入 **Access denied**。

完成這些步驟後，負載平衡器接聽程式會有兩個規則。一個規則會轉寄包含 HTTP 標頭的請求 (來自 CloudFront 的請求)。另一個規則會傳送固定回應給所有其他請求 (不是來自 CloudFront 的請求)。

您可以將請求傳送至您的 CloudFront 發佈，並傳送一個請求至您的 Application Load Balancer，以驗證解決方案是否正常運作。對 CloudFront 的請求會傳回您的 Web 應用程式或內容，而直接傳送給 Application Load Balancer 的請求會傳回純文字訊息 Access denied 的 403 回應。

(選用) 改善此解決方案的安全性

為了改善此解決方案的安全性，您可以將 CloudFront 發佈設定為在向 Application Load Balancer 傳送請求時始終使用 HTTPS。請記住，此解決方案僅適用於保留自訂標頭名稱和值密碼的情況下。使用 HTTPS 有助於防止竊聽者發現標頭名稱和值。我們也建議定期輪換標頭名稱和值。

針對來源請求使用 HTTPS

若要將 CloudFront 設定為針對來源請求使用 HTTPS，請將來源協定政策設定設定為僅限 HTTPS。此設定可在 CloudFront 主控台 CloudFormation 和 CloudFront API 中使用。如需詳細資訊，請參閱[通訊協定 \(僅限自訂原始伺服器\)](#)。

您將 CloudFront 設定為使用 HTTPS 處理原始請求時，也適用下列條件：

- 您必須設定 CloudFront，以使用原始伺服器請求政策將 Host 標頭轉送至原始伺服器。您可以使用[AllViewer 受管原始伺服器請求政策](#)。
- 請確定您的 Application Load Balancer 具有 HTTPS 接聽程式 (如[上節](#)所示)。如需詳細資訊，請參閱《Application Load Balancer 使用者指南》中的[建立 HTTPS 接聽程式](#)。如欲使用 HTTPS 接聽程式，需要您擁有與路由至 Application Load Balancer 網域名稱相符的 SSL/TLS 憑證。
- CloudFront 的 SSL/TLS 憑證只能在 us-east-1 AWS 區域 in(ACM) 中請求 AWS Certificate Manager (或匯入)。由於 CloudFront 是全域服務，因此會自動將憑證從 us-east-1 區域分佈到與您的 CloudFront 分佈相關聯的所有區域。
 - 例如，如果您在 ap-southeast-2 區域中有 Application Load Balancer (ALB)，則必須在 ap-southeast-2 區域 (適用於在 CloudFront 和 ALB 原始伺服器之間使用 HTTPS) 和 us-east-1 區域 (適用於在檢視器和 CloudFront 之間使用 HTTPS) 中設定 SSL/TLS 憑證。這兩個憑證都應該符合路由至 Application Load Balancer 的網域名稱。如需詳細資訊，請參閱[AWS 區域 適用於的 AWS Certificate Manager](#)。
- 如果 Web 應用程式的最終使用者 (也稱為檢視器或用戶端) 可以使用 HTTPS，您也可以將 CloudFront 設定為偏好 (甚至需要) 來自最終使用者的 HTTPS 連線。若要這樣做，請使用檢視器協定政策設定。您可以將它設定為將最終使用者從 HTTP 重新導向至 HTTPS，或拒絕使用 HTTP 的請求。此設定可在 CloudFront 主控台 CloudFormation 和 CloudFront API 中使用。如需詳細資訊，請參閱[檢視器通訊協定政策](#)。

輪換標頭名稱和值

除了使用 HTTPS 之外，我們還建議定期輪換標頭名稱和值。執行這項操作的高階步驟如下：

1. 設定 CloudFront 將其他自訂 HTTP 標頭新增至其傳送至 Application Load Balancer 的請求。

2. 更新 Application Load Balancer 接聽程式規則，以轉寄包含此其他自訂 HTTP 標頭的請求。
3. 設定 CloudFront 停止將來源自訂 HTTP 標頭新增至其傳送至 Application Load Balancer 的請求。
4. 更新 Application Load Balancer 接聽程式規則，以停止轉寄包含來源自訂 HTTP 標頭的請求。

如需完成這些步驟的詳細資訊，請參閱前一章節。

(選用) 使用 CloudFront AWS 的受管字首清單限制對原始伺服器的存取

若要進一步限制對 Application Load Balancer 的存取，您可以設定與 Application Load Balancer 相關聯的安全群組，以便在服務使用受管字首清單時只接受來自 CloudFront AWS 的流量。這可防止不是來自 CloudFront 的流量在網路層 (第 3 層) 或傳輸層 (第 4 層) 到達 Application Load Balancer。

如需詳細資訊，請參閱[使用 AWS Amazon CloudFront 部落格文章的受管字首清單限制對原始伺服器的存取](#)。

限制您內容的地理分佈

您可以使用地理限制功能 (有時也稱為地理封鎖)，來防止特定地理位置的使用者存取您透過 Amazon CloudFront 分佈所散發的內容。若要使用地理限制，您有兩個選擇：

- 使用 CloudFront 地理位置限制功能。使用此選項可限制對與分佈相關聯的所有檔案的存取，並限制在國家/地區層級的存取。
- 使用第三方地理位置服務。使用此選項來針對與分佈相關聯檔案的子集作限制存取，或在比國家/地區層級更細的層級作限制存取。

主題

- [使用 CloudFront 地理限制](#)
- [使用第三方地理位置服務。](#)

使用 CloudFront 地理限制

當使用者請求內容時，無論使用者身在何處，CloudFront 通常會為請求的內容提供服務。如果您需要防止特定國家/區域的使用者存取您的內容，則可以使用 CloudFront 地理限制功能執行以下其中一項內容：

- 只有當使用者位於您允許清單上其中一個核准國家/地區時，才能存取您的內容。

- 如果使用者位於您封鎖清單上的被禁國家/地區，則阻止其存取您的內容。

例如，若請求來自您未授權分配內容的國家/區域，則可以使用 CloudFront 地理限制來封鎖該請求。

Note

CloudFront 會使用第三方資料庫來判斷使用者的位置。IP 位址與國家/地區之間的映射的準確性因區域而異。根據最近的測試，整體準確性為 99.8%。如果 CloudFront 無法判斷使用者的位置，CloudFront 會提供使用者所請求的內容。

以下是地理限制的運作方式：

1. 假設您有權僅在列支敦斯登分配您的內容。您更新 CloudFront 分佈，來新增僅包含列支敦斯登的允許清單。(或者，您可以新增包含除了列支敦斯登以外的每個國家/地區的封鎖清單。)
2. 摩納哥的使用者請求您的內容，而 DNS 會將請求路由到位於義大利米蘭的 CloudFront 節點。
3. 米蘭的邊緣節點會查詢您的分佈，並確定摩納哥的使用者沒有下載您的內容的許可。
4. CloudFront 會將 HTTP 狀態碼 403 (Forbidden) 傳回使用者。

您可以選擇性地設定 CloudFront，向使用者傳回自訂錯誤訊息，而且也可指定您希望 CloudFront 針對請求檔案的錯誤回應，為其建立快取的留存時間。預設值為 10 秒。如需詳細資訊，請參閱[針對特定的 HTTP 狀態碼建立自訂錯誤頁面](#)。

地理限制適用於整個分佈。如果您需要對部分內容套用一個限制，並對內容的其他部分套用不同的限制(或不限制)，則必須建立單獨 CloudFront 分佈或[使用第三方地理位置服務](#)。

如果您啟用 CloudFront [標準日誌](#) (即存取日誌)，則可以透過搜尋 `sc-status` 值 (即 HTTP 狀態代碼) 為 403 的日誌項目，來識別 CloudFront 已拒絕的請求。不過，如果只使用標準日誌，您無法區分 CloudFront 是根據使用者所在位置而拒絕的請求，還是 CloudFront 因為使用者不具有存取檔案的許可 (有其他的理由) 而拒絕的請求。如果您有第三方地理位置服務，例如 Digital Element 或 MaxMind，則可以根據存取日誌中 `c-ip` (用戶端 IP) 列中的 IP 位址來識別請求的位置。如需有關 CloudFront 標準日誌的詳細資訊，請參閱 [存取日誌 \(標準日誌\)](#)。

下列程序說明如何使用 CloudFront 主控台將地理限制新增到現有的分佈。如需有關如何使用主控台建立分佈的詳細資訊，請參閱[建立分發](#)。

將地理限制功能新增至您的 CloudFront Web 分佈 (主控台)

1. 登入 AWS 管理主控台，並在 開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇分佈，然後選擇您要更新的分佈。
3. 選擇安全性分頁，然後選擇地理限制。
4. 選擇編輯。
5. 選擇 Allow list (允許清單) 建立允許的國家/地區清單，或 Block list (封鎖清單) 建立封鎖的國家/地區清單。
6. 將所需的國家/地區新增至清單中，然後選擇 Save changes (儲存變更)。

使用第三方地理位置服務。

CloudFront 地理限制功能可讓您控制內容在國家/地區層級的分佈，以便您在特定的 Web 分佈中散發所有檔案。如果您有適用地理限制的使用案例，但此限制並非遵循國界，或是想透過指定分佈，來限制僅可存取提供的某些檔案，您可將 CloudFront 與第三方地理位置服務組合使用。這讓您不僅可以根據國家/地區控制對內容的存取，還可以根據城市、郵遞區號，甚至是緯度和經度來控制對內容的存取。

當您使用第三方地理位置服務時，我們建議您使用 CloudFront 的已簽署 URL，這可讓您指定到期日期和時間，在此日期和時間之後 URL 不再有效。此外，我們建議您使用 Amazon S3 儲存貯體做為原始伺服器，因為您可使用 CloudFront [原始存取控制](#)，以防止使用者直接從原始伺服器存取您的內容。如需有關已簽署的 URL 和原始存取控制的詳細資訊，請參閱 [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。

以下步驟解釋如何使用第三方地理位置服務來控制對檔案的存取。

使用第三方地理位置服務來限制對 CloudFront 分佈中檔案的存取

1. 使用地理位置服務取得帳戶。
2. 將內容上傳至 Amazon S3 儲存貯體。
3. 設定 Amazon CloudFront 和 Amazon S3 以提供私有內容。如需詳細資訊，請參閱 [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。
4. 寫入您的 Web 應用程式，以執行下列動作：
 - 將每個使用者請求的 IP 位址傳送到地理位置服務。
 - 評估來自地理位置服務傳回的值，以判斷使用者是否位於您希望 CloudFront 分配內容的位置。

- 如果您想要將內容分配至使用者的位置，請為您的 CloudFront 內容產生已簽署的 URL。如果您不想將內容分配到該位置，請將 HTTP 狀態碼 403 (Forbidden) 傳回給使用者。或者，您可以設定 CloudFront 以返回自訂錯誤訊息。如需詳細資訊，請參閱[the section called “針對特定的 HTTP 狀態碼建立自訂錯誤頁面”](#)。

如需詳細資訊，請參閱您所使用的地理位置服務的文件。

您可以使用 Web 伺服器變動變數來獲得瀏覽您的網站的使用者的 IP 位址。請注意以下警告：

- 如果您的 Web 伺服器透過負載平衡器無法連線至網際網路，則可以使用 Web 伺服器變數來獲得遠端 IP 位址。不過，此 IP 位址不一定是使用者的 IP 位址。它也可以是代理伺服器的 IP 位址，取決於使用者如何連接到網際網路。
- 如果您的 Web 伺服器透過負載平衡器連線至網際網路，則 Web 伺服器變數可能包含負載平衡器的 IP 位址，而非使用者的 IP 位址。在這個組態中，我們建議您使用 X-Forwarded-For HTTP 標頭中的最後一個 IP 位址。此標頭通常包含多個 IP 位址，其中大部分用於代理或負載平衡器。清單中的最後一個 IP 位址最有可能與使用者的地理位置相關聯。

如果您的 Web 伺服器無法連接到負載平衡器，我們建議您使用 Web 伺服器變數而非 X-Forwarded-For 標頭，以避免 IP 位址詐騙。

使用欄位層級加密來協助保護敏感資料

透過 Amazon CloudFront，您可以使用 HTTPS 強制執行原始伺服器的安全端對端連線。欄位層級加密可新增額外的安全層，可讓您在整個系統處理過程中保護特定的資料，以便只有特定應用程式才能看到它。

欄位層級加密可讓您的使用者安全地將敏感資訊上傳到您的 Web 伺服器。使用者提供的敏感資訊會在邊緣、靠近使用者處加密，並在整個應用程式堆疊中保持加密。此加密可確保只有需要資料的應用程式(並具有可解密的登入資料)才能執行這項操作。

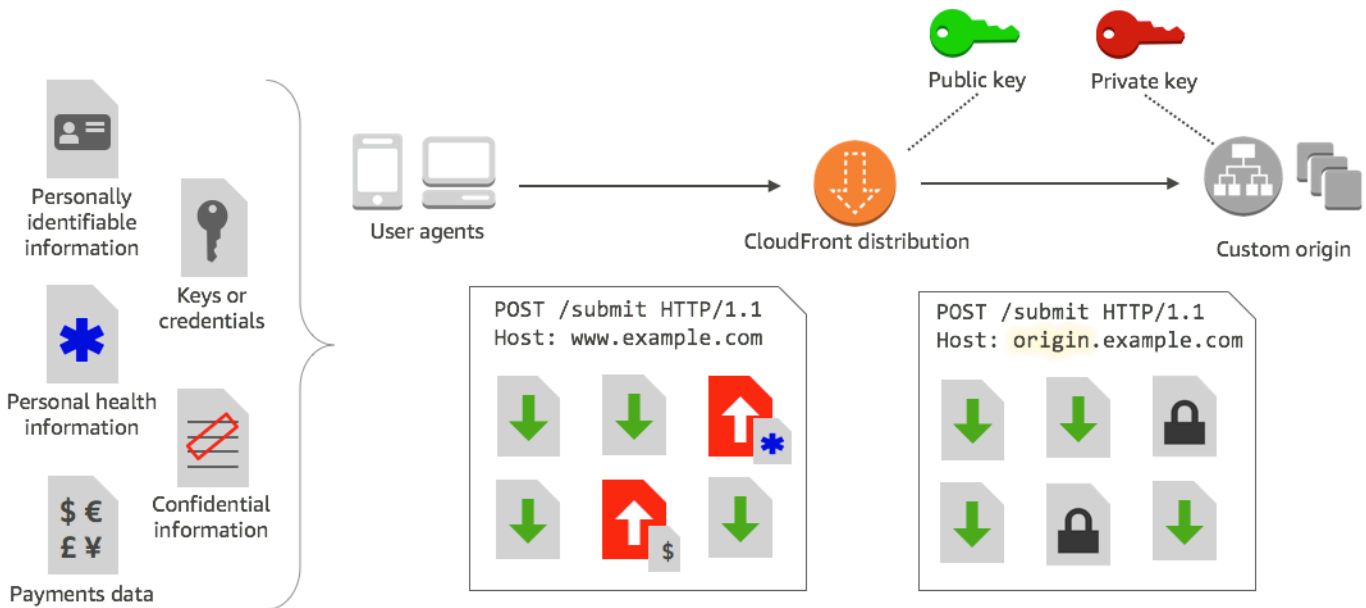
若要使用欄位層級加密，則可以設定 CloudFront 分佈以指定要加密的 POST 請求中的一組欄位，以及對其加密的公有金鑰。您可以在請求中加密多達 10 個資料欄位。(您無法使用欄位層級加密對請求中的所有資料進行加密；您必須指定要加密的各個欄位。)

當將帶有欄位層級加密的 HTTPS 請求轉發到原始伺服器，並且該請求被路由到原始應用程式或子系統中時，敏感資料仍然被加密，從而降低了敏感資料遭受洩露或意外遺失資料的風險。出於業務考量而需

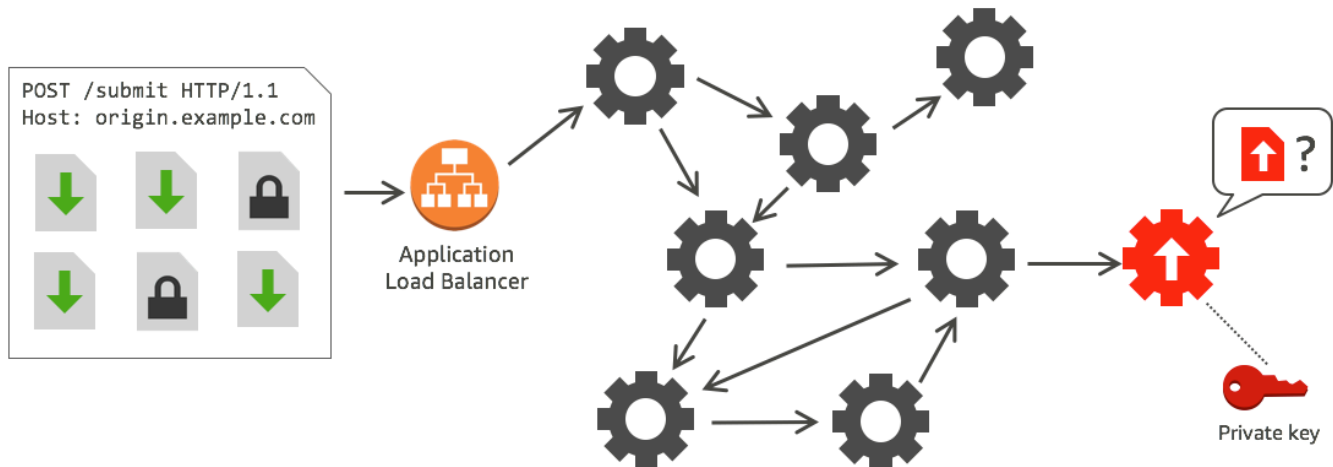
對敏感資料存取的元件，例如需要存取信用卡號碼的付款系統，可以使用適當的私有金鑰來解密和存取該資料。

Note

為了使用欄位層級加密，您的原始伺服器必須支援區塊編碼。



CloudFront 欄位層級加密使用非對稱加密，也稱為公有金鑰加密。您提供了 CloudFront 的公有金鑰，所指定的所有敏感資料都會進行自動加密。您提供給 CloudFront 的金鑰不能用於解密加密的值；只有您的私有金鑰可以做到這一點。



主題

- [欄位層級加密概觀](#)
- [設定欄位層級加密](#)
- [在您的原始伺服器解密資料欄位](#)

欄位層級加密概觀

以下步驟概述了設定欄位層級加密。對於特定的步驟，請參閱[設定欄位層級加密](#)。

1. 取得公有金鑰/私有金鑰對。CloudFront 開始在 CloudFront 中設定欄位層級加密前，必須取得並新增公有金鑰。
2. 建立欄位層級加密的設定檔。CloudFront 您在 CloudFront 中建立的欄位層級加密設定檔定義了要加密的欄位。
3. 建立欄位層級加密的組態。組態會指定要使用的設定檔 (根據請求的內容類型或查詢引數)，用來將特定資料欄位加密。您也可以針對不同案例選擇所需的請求轉發行為選項。例如，您可以設定要求 URL 中查詢引數所指定的設定檔名稱不存在於 CloudFront 的行為。
4. 至快取行為的連結。將組態連結到分佈的快取行為，以指定 CloudFront 應該何時加密資料。

設定欄位層級加密

依照以下步驟開始使用欄位層級加密。若要了解欄位層級加密的配額 (先前稱為限制)，請參閱[配額](#)。

- [步驟 1：建立 RSA 金鑰對](#)
- [步驟 2：將您的公有金鑰新增到 CloudFront](#)
- [步驟 3：建立欄位層級加密的設定檔。](#)
- [步驟 4：建立組態](#)
- [步驟 5：將組態新增到快取行為](#)

步驟 1：建立 RSA 金鑰對

若要開始使用，您必須建立包含公有金鑰和私有金鑰的 RSA 金鑰對。公有金鑰可讓 CloudFront 加密資料，私有金鑰使您原始伺服器的元件能夠解密已加密的欄位。您可以使用 OpenSSL 或其他工具來建立金鑰對。金鑰大小必須為 2048 個位元。

例如，如果您使用的是 OpenSSL，您可以使用以下命令來產生長度為 2048 個位元組的金鑰對，並將其儲存在檔案 `private_key.pem` 中：

```
openssl genrsa -out private_key.pem 2048
```

產生的檔案同時包含公有和私有金鑰。要從該檔案中擷取公有金鑰，請執行以下命令：

```
openssl rsa -pubout -in private_key.pem -out public_key.pem
```

公有金鑰檔案 (`public_key.pem`) 包含您在下列步驟中貼上的編碼鍵值。

步驟 2：將您的公有金鑰新增到 CloudFront

取得您的 RSA 金鑰對之後，將公有金鑰新增到 CloudFront。

將您的公有金鑰新增至 CloudFront (主控台)

1. 登入 AWS 管理主控台，並在開啟 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 在導覽窗格中，選擇 Public key (公有金鑰)。
3. 選擇 Add public key (新增公有金鑰)。
4. 在 Key name (金鑰名稱) 中，輸入金鑰的獨特名稱。名稱不能有空格，並且只能包含英數字元、底線 (_) 和 連字號 (-)。最大字元數為 128。

5. 針對 Key value (鍵值)，貼上公有金鑰的編碼鍵值 (包括 -----BEGIN PUBLIC KEY----- 和 -----END PUBLIC KEY----- 行)。
6. 針對 Comment (註解)，加入選擇性的註解。例如，您可以包含公有金鑰的到期日期。
7. 選擇 Add (新增)。

透過重複程序中的步驟，您可以新增更多用於 CloudFront 的金鑰。

步驟 3：建立欄位層級加密的設定檔。

將至少一個公有金鑰新增到 CloudFront 後，建立設定檔，告知 CloudFront 要加密哪些欄位。

建立欄位層級加密的設定檔 (主控台)

1. 在導覽窗格中，選擇 Field-level encryption (欄位層級加密)。
2. 選擇 Create profile (建立設定檔)。
3. 填寫下列欄位：

設定檔名稱

請輸入設定檔的專屬名稱。名稱不能有空格，並且只能包含英數字元、底線 (_) 和 連字號 (-)。最大字元數為 128。

公有金鑰名稱

在下拉式清單中，選擇您在步驟 2 中新增到 CloudFront 公有金鑰的名稱。CloudFront 使用金鑰來加密您在此設定檔中指定欄位。

供應商名稱

輸入片語以協助識別您的金鑰，例如您取得到金鑰對的供應商。當應用程式解密資料欄位時，將需要此資訊以及私有金鑰。供應商名稱不能有空格，並且只能包含英數字元、冒號 (:)、底線 (_) 和 連字號 (-)。最大字元數為 128。

欄位名稱模式符合

輸入您希望 CloudFront 加密的資料欄位的名稱，或識別請求中資料欄位名稱的模式。選擇 + 選項來新增您想要使用此金鑰加密的所有欄位。

對於欄位名稱模式，您可以輸入資料欄位的整個名稱，例如 DateOfBirth，或僅使用萬用字元 (*) 的名稱的第一部分，例如 CreditCard*。除了可選的萬用字元 (*) 外，欄位名稱模式必須只包含英數字元、方括號 ([和])、句號 (.)、底線 (_) 和連字號 (-)。

請確定不要為不同的欄位名稱模式使用重疊的字元。例如，如果您有 ABC* 的欄位名稱模式，則不能新增另一個 AB* 的欄位名稱模式。此外，欄位名稱區分大小寫，可以使用的字元數上限為 128。

註解

(選用) 輸入有關此設定檔的評論。最多可使用 128 個字元。

4. 填寫完欄位後，選擇 Create profile (建立設定檔)。
5. 如果您想新增更多設定檔，請選擇 Add profile (新增設定檔)。

步驟 4：建立組態

在您建立一或多個欄位層級加密設定檔後，請建立一個組態，指定包含要加密的資料的請求的內容類型、用於加密的設定檔，以及指定 CloudFront 如何處理加密的其他選項。

例如，當 CloudFront 無法加密資料時，可以指定在以下情況下，CloudFront 是否應該封鎖或轉送請求到您的原始伺服器：

- 當請求的內容類型不在組態中 – 如果您尚未在組態中新增內容類型，則可以指定 CloudFront 是否應將具有該內容類型的請求轉傳到原始伺服器，而不加密資料欄位，或封鎖請求並傳回一個錯誤。

Note

如果將內容類型新增到組態中，但未指定與該類型一起使用的設定檔，則具有該內容類型的 CloudFront 將一律把請求轉傳到原始伺服器。

- 當查詢引數中提供的設定檔名稱未知時 – 當您使用分佈中不存在的設定檔名稱指定 fle-profile 查詢引數時，您可以指定 CloudFront 是否應將請求發送到原始伺服器，而不加密資料欄位，或封鎖請求並傳回錯誤。

在組態中，還可以指定一個在 URL 中做為查詢參數所提供的設定檔是否要覆寫了一個已對應到該查詢內容類型的設定檔。在預設情況下，如果指定了一個已對應到內容類型的設定檔，則 CloudFront 將使用該設定檔。這可讓您擁有一個預設情況下使用的設定檔，但也會決定您所希望強制執行不同設定檔的特定請求。

因此，例如，您可以指定 (在您的組態中) **SampleProfile** 做為要使用的查詢參引數設定檔。然後，您可以使用 URL `https://d1234.cloudfront.net?fle-profile=SampleProfile` 而非

`https://d1234.cloudfront.net`，來讓 CloudFront 針對此請求使用 **SampleProfile**，而非使用您為請求的內容類型所建立的設定檔。

您最多可以為單一帳戶建立 10 個組態，然後將其中一個組態與的該帳戶的任何分佈的快取行為相關聯。

建立欄位層級加密的組態 (主控台)

1. 在 Field-level encryption (欄位層級加密) 頁面，選擇 Create configuration (建立組態)。

注意：如果您尚未建立至少一個設定檔，則不會看到用於建立組態的選項。

2. 請填寫以下欄位指定要使用的設定檔。(有些欄位無法變更。)

內容類型 (無法變更)

內容類型設定為 `application/x-www-form-urlencoded`，無法變更。

預設設定檔 ID (選用)

在下拉式清單中選擇設定檔，此設定檔會對應到 Content type (內容類型) 欄位中的內容類型。

內容格式 (無法變更)

內容格式設定為 `URLencoded`，無法變更。

3. 如果您想要變更以下選項的 CloudFront 預設行為，請選取合適的核取方塊。

當請求的內容類型未作設定時，請轉發請求到原始來源

如果您尚未指定用於請求內容類型的設定檔，而要允許請求轉傳到您的原始伺服器，請勾選此核取方塊。

使用提供的查詢參數覆寫內容類型的設定檔

如果您要允許查詢引數中所提供的設定檔，覆寫您針對內容類型指定的設定檔，請勾選此核取方塊。

4. 如果您選取核取方塊，以允許查詢參數來覆寫預設的設定檔，則必須完成組態的下列其他欄位。在這些查詢參數對應中，您最多可以建立五個，以便與查詢一起使用。

查詢參數

輸入要包含在 URL 中以用於 `file-profile` 查詢參數的值。這個值通知 CloudFront 使用與此查詢參數相關聯的設定檔 ID (在下一個欄位中指定的)，以用於此查詢的欄位層級加密。

最多可使用 128 個字元。該值不能包含空格，而且必須使用英數字元或以下字元：破折號 (-)、句點 (.)、底線 (_)、星號 (*)、加號 (+)、百分比 (%)。

設定檔 ID

在下拉式清單中選擇設定檔，您要將此設定檔與您針對 Query argument (查詢引數) 輸入的值建立關聯。

當查詢參數中指定的設定檔不存在時，將請求轉發到原始來源

如果在 CloudFront 中未定義查詢參數中指定的設定檔，且如果您想要允許請求轉到原始伺服器，請選取核取方塊。

步驟 5：將組態新增到快取行為

若要使用欄位層級加密，請透過將組態 ID 新增為分佈的值，將組態連結到分佈的快取行為。

Important

若要將欄位層級的加密設定連結至快取行為，必須將分佈設定為永遠使用 HTTPS，並接受來自瀏覽者的 HTTP POST 和 PUT 請求。也就是說，下列條件必須為真：

- 快取行為的 Viewer Protocol 政策 (檢視器通訊協定政策) 必須設定為將 Redirect HTTP to HTTPS (HTTP 重新引導至 HTTPS) 或 HTTPS Only (僅 HTTPS)。(在 CloudFormation 或 CloudFront API 中，ViewerProtocolPolicy 必須設定為 `redirect-to-https` 或 `https-only`。)
- 快取行為的允許的 HTTP 方法必須設為 GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE。(在 CloudFormation 或 CloudFront API 中，AllowedMethods 必須設定為 GET、HEAD、OPTIONS、PUT、POST、PATCH、DELETE。這些可以依任何順序指定。)
- 原始伺服器設定的 Origin Protocol 政策 (原始伺服器通訊協定政策) 必須設定為 Match Viewer (符合檢視器) 或 HTTPS Only (僅 HTTPS)。(在 CloudFormation 或 CloudFront API 中，OriginProtocolPolicy 必須設定為 `match-viewer` 或 `https-only`。)

如需詳細資訊，請參閱[所有分佈設定參考](#)。

在您的原始伺服器解密資料欄位

CloudFront 透過使用 [AWS Encryption SDK](#) 加密資料欄位。資料在整個應用程式堆疊中保持加密狀態，只能由具有解密憑證的應用程式存取。

加密後，加密文字是 base64 編碼的。當您的應用程式在原始伺服器解密文字時，必須先對加密文字解碼，然後使用 AWS 加密開發套件來解密資料。

以下程式碼範例說明應用程式如何在原始伺服器中解密資料。注意下列事項：

- 為了簡化範例，本範例從工作目錄中的檔案載入公有金鑰和私有金鑰 (以 DER 格式)。在實務上，您可以將私有金鑰存放在安全的離線位置，例如離線硬體安全模組中，並將公有金鑰分佈到您的開發團隊。
- CloudFront 在加密資料時使用特定的資訊，並且應該在原始伺服器上使用相同的一組參數對其進行解密。在初始化 MasterKey 時使用的參數 CloudFront 包括下列項目：
 - PROVIDER_NAME：當您建立欄位層級加密設定檔時指定了這個值。在這裡使用相同的值。
 - KEY_NAME：當您將公有金鑰上傳至 CloudFront 時，您為其建立一個名稱，然後在設定檔中指定了金鑰名稱。在這裡使用相同的值。
 - ALGORITHM：CloudFront 使用 RSA/ECB/OAEPWithSHA-256AndMGF1Padding 做為加密演算法，因此您必須使用相同的演算法來解密資料。
- 如果您以加密文字做為輸入來執行以下範例程式時，則解密的資料將輸出到您的主控台。如需詳細資訊，請參閱 AWS 加密 SDK 中的 [Java 範例程式碼](#)。

範本程式碼

```
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.KeyFactory;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

import org.apache.commons.codec.binary.Base64;

import com.amazonaws.encryptionsdk.AwsCrypto;
import com.amazonaws.encryptionsdk.CryptoResult;
import com.amazonaws.encryptionsdk.jce.JceMasterKey;
```

```
/**
 * Sample example of decrypting data that has been encrypted by CloudFront field-level
 * encryption.
 */
public class DecryptExample {

    private static final String PRIVATE_KEY_FILENAME = "private_key.der";
    private static final String PUBLIC_KEY_FILENAME = "public_key.der";
    private static PublicKey publicKey;
    private static PrivateKey privateKey;

    // CloudFront uses the following values to encrypt data, and your origin must use
    // same values to decrypt it.
    // In your own code, for PROVIDER_NAME, use the provider name that you specified
    // when you created your field-level
    // encryption profile. This sample uses 'DEMO' for the value.
    private static final String PROVIDER_NAME = "DEMO";
    // In your own code, use the key name that you specified when you added your public
    // key to CloudFront. This sample
    // uses 'DEMOKEY' for the key name.
    private static final String KEY_NAME = "DEMOKEY";
    // CloudFront uses this algorithm when encrypting data.
    private static final String ALGORITHM = "RSA/ECB/OAEPWithSHA-256AndMGF1Padding";

    public static void main(final String[] args) throws Exception {

        final String dataToDecrypt = args[0];

        // This sample uses files to get public and private keys.
        // In practice, you should distribute the public key and save the private key
        // in secure storage.
        populateKeyPair();

        System.out.println(decrypt(debase64(dataToDecrypt)));
    }

    private static String decrypt(final byte[] bytesToDecrypt) throws Exception {
        // You can decrypt the stream only by using the private key.

        // 1. Instantiate the SDK
        final AwsCrypto crypto = new AwsCrypto();

        // 2. Instantiate a JCE master key
```

```
        final JceMasterKey masterKey = JceMasterKey.getInstance(
            publicKey,
            privateKey,
            PROVIDER_NAME,
            KEY_NAME,
            ALGORITHM);

        // 3. Decrypt the data
        final CryptoResult <byte[], ? > result = crypto.decryptData(masterKey,
bytesToDecrypt);
        return new String(result.getResult());
    }

    // Function to decode base64 cipher text.
    private static byte[] debase64(final String value) {
        return Base64.decodeBase64(value.getBytes());
    }

    private static void populateKeyPair() throws Exception {
        final byte[] PublicKeyBytes =
Files.readAllBytes(Paths.get(PUBLIC_KEY_FILENAME));
        final byte[] privateKeyBytes =
Files.readAllBytes(Paths.get(PRIVATE_KEY_FILENAME));
        publicKey = KeyFactory.getInstance("RSA").generatePublic(new
X509EncodedKeySpec(PublicKeyBytes));
        privateKey = KeyFactory.getInstance("RSA").generatePrivate(new
PKCS8EncodedKeySpec(privateKeyBytes));
    }
}
```

使用 CloudFront 提供隨需視訊和即時串流視訊

您可以透過 CloudFront 傳遞隨需視訊 (VOD) 或使用任何 HTTP 原始伺服器的即時串流視訊。在雲端設定視訊工作流程的一種方式就是將 CloudFront 搭配 [AWS Media Services](#) 一起使用。

主題

- [關於串流視訊](#)
- [使用 CloudFront 傳遞隨需視訊](#)
- [使用 CloudFront 和 AWS Media Services 交付影片串流](#)
- [媒體品質感知彈性](#)

關於串流視訊

您必須先使用編碼器封裝視訊內容，然後 CloudFront 才可以分配該內容。封裝程序會建立區段，其包含您的音訊、視訊和字幕內容。它也會產生資訊清單檔案，其以特定順序描述播放哪些區段以及何時播放。常見的封裝格式包括 MPEG DASH、Apple HLS、Microsoft Smooth Streaming 和 CMAF。

VOD 串流

如果是 VOD 串流，您的視訊內容會存放在伺服器上，檢視器隨時都可以觀看。若要製作瀏覽者可進行串流的資產，請使用編碼器，例如 [AWS Elemental MediaConvert](#)，以格式化和封裝您的媒體檔案。

當您的視訊封裝成正確的格式後，您可以將其存放在伺服器或 Amazon S3 儲存貯體中，然後在檢視器請求時透過 CloudFront 傳遞該視訊。

即時視訊串流

對於即時視訊串流，會在發生即時事件時，即時串流您的視訊內容，或設為全天候的即時頻道。若要建立廣播和串流交付的即時輸出 AWS Elemental MediaLive，請使用等編碼器來壓縮影片，並將其格式化以供檢視裝置。

對影片進行編碼後，您可以將其存放在中，AWS Elemental MediaStore 或使用將其轉換為不同的交付格式 AWS Elemental MediaPackage。使用以下其中一個原始伺服器，設定 CloudFront 分佈來傳遞內容。如需建立分佈來搭配這些服務一起使用的特定步驟和指導方針，請參閱[使用 AWS Elemental MediaStore 做為原始伺服器來提供影片](#)和[提供以 AWS Elemental MediaPackage 格式化的即時視訊](#)。

Wowza 和 Unified Streaming 也提供可用於透過 CloudFront 進行視訊串流的工具。如需搭配 CloudFront 使用 Wowza 的詳細資訊，請參閱 Wowza 文件網站上的 [Bring your Wowza Streaming Engine license to CloudFront live HTTP streaming \(將您的 Wowza 串流引擎授權給 CloudFront 即時 HTTP 串流\)](#)。如需使用 Unified Streaming 和 CloudFront 進行 VOD 串流的相關資訊，請參閱 Unified Streaming 文件網站上的 [CloudFront](#)。

使用 CloudFront 傳遞隨需視訊

若要使用 CloudFront 傳遞隨需視訊 (VOD) 串流，請使用以下服務：

- Amazon S3，將內容以其原始格式存放，並存放已轉碼的視訊。
- 將影片轉碼為串流格式的編碼器（例如 AWS Elemental MediaConvert）。
- CloudFront，將已轉碼的視訊傳遞給檢視器。對於 Microsoft Smooth Streaming，請參閱 [為 Microsoft Smooth Streaming 設定隨需視訊](#)。

使用 CloudFront 建立 VOD 解決方案

1. 將內容上傳至 Amazon S3 儲存貯體。若要進一步了解如何使用 Amazon S3，請參閱 [Amazon Simple Storage Service 使用者指南](#)。
2. 使用 MediaConvert 任務將內容轉碼。此任務會將視訊轉換為您的檢視器使用之播放器所需的格式。您也可以使用任務來建立具有不同解析度和位元速率的資產。這些資產用於調整式位元速率 (ABR) 串流，這種串流會根據檢視器的可用頻寬，調整觀看品質。MediaConvert 會將經過轉碼的視訊存放在 S3 儲存貯體。
3. 使用 CloudFront 分佈傳遞轉換後的內容。檢視器可以隨時在任何裝置上觀看內容。

為 Microsoft Smooth Streaming 設定隨需視訊

您可利用下列選項，使用 CloudFront 來分配已轉碼為 Microsoft Smooth Streaming 格式的隨需視訊 (VOD) 內容：

- 指定執行 Microsoft IIS 並支援 Smooth Streaming 的 Web 伺服器做為分佈的來源。
- 在 CloudFront 分佈的快取行為中啟用 Smooth Streaming。由於您可以在分佈中使用多個快取行為，因此您可以將一個分佈用於 Smooth Streaming 媒體檔案及其他內容。

⚠ Important

如果您指定執行 Microsoft IIS 的 Web 伺服器做為原始伺服器，請「勿」在 CloudFront 分佈的快取行為中啟用 Smooth Streaming。如果啟用 Smooth Streaming 做為快取行為，CloudFront 就無法使用 Microsoft IIS 伺服器做為原始伺服器。

如果您在快取行為中啟用 Smooth Streaming (也就是說，您沒有執行 Microsoft IIS 的伺服器)，請注意以下事項：

- 如果內容與該快取行為 Path Pattern (路徑模式) 的值相符，您仍然可以使用相同的快取行為來分配其他內容。
- CloudFront 可以使用 Amazon S3 儲存貯體或適用於 Smooth Streaming 媒體檔案的自訂原始伺服器。如果啟用 Smooth Streaming 用於快取行為，CloudFront 就無法使用 Microsoft IIS 伺服器做為原始伺服器。
- 您不能使 Smooth Streaming 格式的媒體檔案無效。如果想要在過期之前更新檔案，您必須重新命名。如需詳細資訊，請參閱[新增、移除或取代 CloudFront 分配的內容](#)。

如需關於 Smooth Streaming 用戶端的詳細資訊，請參閱 Microsoft 文件網站上的 [Smooth Streaming](#)。

在 Microsoft IIS Web 伺服器不是原始伺服器時，使用 CloudFront 來分配 Smooth Streaming 檔案

1. 將您的媒體檔案轉碼成 Smooth Streaming 片段的 MP4 格式。
2. 請執行下列其中一項：
 - 如果您使用的是 CloudFront 主控台：建立或更新分佈時，在一個或多個分佈的快取行為中，啟用 Smooth Streaming。
 - 如果您使用的是 CloudFront API：針對一個或多個分佈的快取行為，將 SmoothStreaming 元素新增至其 DistributionConfig 複雜類型中。
3. 將 Smooth Streaming 檔案上傳到原始伺服器。
4. 建立一個 `clientaccesspolicy.xml` 或 `crossdomainpolicy.xml` 檔案，並將它加入到可在分佈的根目錄中存取的位置，例如，`https://d1111111abcdef8.cloudfront.net/clientaccesspolicy.xml`。以下為政策的範例：

```
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
<cross-domain-access>
```

```
<policy>
<allow-from http-request-headers="*">
<domain uri="*" />
</allow-from>
<grant-to>
<resource path="/" include-subpaths="true" />
</grant-to>
</policy>
</cross-domain-access>
</access-policy>
```

如需詳細資訊，請參閱 Microsoft Developer Network 網站上的 [Making a Service Available Across Domain Boundaries \(讓服務可跨網域邊界使用\)](#)。

5. 對於您的應用程式 (例如，媒體播放器) 中的連結，請以下列格式為媒體檔案指定 URL：

```
https://d1111111abcdef8.cloudfront.net/video/presentation.ism/Manifest
```

使用 CloudFront 和 AWS Media Services 交付影片串流

若要搭配 CloudFront 使用 AWS Media Services 將即時內容交付給全球受眾，請參閱下列指引。

使用 [AWS Elemental MediaLive](#) 即時編碼即時視訊串流。為了編碼大型視訊串流，MediaLive 會將其壓縮至可分配給檢視器的較小版本 (「編碼」)。

壓縮即時視訊串流之後，您可以使用以下兩個主要選項的其中之一來準備和提供內容：

- 將您的內容轉換成所需格式，然後提供它 – 如果您需要多種格式的內容，請使用 [AWS Elemental MediaPackage](#) 來封裝不同裝置類型的內容。封裝內容時，您也可以實作其他功能，並新增數位版權管理 (DRM)，以防止未經授權使用您的內容。如需使用 CloudFront 提供 MediaPackage 格式化內容的逐步說明，請參閱[提供以 AWS Elemental MediaPackage 格式化的即時視訊](#)。
- 使用可擴展的原始伺服器儲存及提供您的內容 – 如果 MediaLive 以瀏覽者使用之所有裝置所需的格式對內容進行編碼，請使用 [AWS Elemental MediaStore](#) 等具有高度可擴展性的原始伺服器來提供內容。如需使用 CloudFront 提供存放於 MediaStore 容器之內容的逐步說明，請參閱[使用 AWS Elemental MediaStore 做為原始伺服器來提供影片](#)。

當您使用這其中一個選項設定您的原始伺服器後，就可以使用 CloudFront 將即時串流視訊分配給檢視器。

i Tip

您可以了解可自動部署服務以建置高度可用即時檢視體驗 AWS 的解決方案。若要檢視自動部署此解決方案的步驟，請參閱[即時串流自動化部署](#)。

主題

- [使用 AWS Elemental MediaStore 做為原始伺服器來提供影片](#)
- [提供以 AWS Elemental MediaPackage 格式化的即時視訊](#)
- [使用 AWS Elemental MediaPackage 提供隨選視訊內容](#)

使用 AWS Elemental MediaStore 做為原始伺服器來提供影片

如果您有儲存於 [AWS Elemental MediaStore](#) 容器中的視訊，可以建立 CloudFront 分佈來提供內容。

若要開始使用，您可以授與 CloudFront 存取您的 MediaStore 容器的權限。然後，建立一個 CloudFront 分佈，並將其設定為可搭配 MediaStore 使用。

從 AWS Elemental MediaStore 容器提供內容

1. 依照[允許 Amazon CloudFront 存取您的 AWS Elemental MediaStore 容器](#)中所述的程序進行，然後再回來執行這些步驟，以建立分佈。
2. 透過下列設定建立分佈：
 - a. 原始伺服器網域 – 指派至 MediaStore 容器的資料端點。從下拉式清單中，選擇您的即時視訊適用的 MediaStore 容器。
 - b. 原始伺服器路徑 – 在存放物件的 MediaStore 容器中的資料夾結構。如需詳細資訊，請參閱[the section called “原始伺服器路徑”](#)。
 - c. 新增自訂標頭 – 如果您希望 CloudFront 在將請求轉送到原始伺服器時新增自訂標頭，請新增標頭名稱與值。
 - d. 檢視器通訊協定政策 – 選擇將 HTTP 重新導向至 HTTPS。如需詳細資訊，請參閱[the section called “檢視器通訊協定政策”](#)。
 - e. 快取政策和原始伺服器請求政策
 - 針對 Cache policy (快取政策)，選擇 Create policy (建立政策)，然後建立適合您快取需求和區段持續時間的快取政策。建立政策後，重新整理快取政策的清單，然後選擇您剛建立的政策。

- 針對 Origin request policy (原始伺服器請求政策)，從下拉式清單中選擇 CORS-CustomOrigin。

對於其他設定，您可以根據您的其他技術需求或業務需求來設定特定的值。如需適用於分佈的所有選項清單以及設定相關資訊，請參閱[the section called “所有分佈設定”](#)。

3. 對於應用程式中的連結 (例如，媒體播放器)，請採用與使用 CloudFront 進行分佈的其他物件相同格式來指定媒體檔案的名稱。

提供以 AWS Elemental MediaPackage 格式化的即時視訊

如果您使用 格式化即時串流 AWS Elemental MediaPackage，您可以建立 CloudFront 分佈並設定快取行為來提供即時串流。以下程序假設您已經[建立頻道](#)，並已使用 MediaPackage 為您的即時視訊[新增端點](#)。

若要手動為 MediaPackage 建立 CloudFront 分佈，請依照下列步驟進行：

步驟 1：建立及設定 CloudFront 分佈

完成以下程序，針對您使用 MediaPackage 建立的即時視訊頻道設定 CloudFront 分佈。

為您的即時視訊頻道建立分佈

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇 Create Distribution (建立分佈)。
3. 選擇分佈的設定，包括下列項目：

原始網域

您的 MediaPackage 即時視訊頻道和端點所在的原始伺服器。選擇文字欄位，然後從下拉式清單中，選擇您的即時視訊適用的 MediaPackage 原始網域。您可以將一個網域對應到多個原始端點。

如果您已使用另一個 AWS 帳戶建立原始網域，請在此欄位中輸入原始 URL 值。原始伺服器必須為 HTTPS URL。

例如，對於像是 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8` 的 HLS 端點，原始網域會是 `3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com`。

如需詳細資訊，請參閱[the section called “原始網域”](#)。

原始伺服器路徑

提供內容之 MediaPackage 端點的路徑。

如需有關原始伺服器路徑如何運作的詳細資訊，請參閱 [the section called “原始伺服器路徑”](#)。

Important

須有萬用字元路徑 * 才能路由至 CloudFront 分佈中的某處。為了避免不符合明確路徑的請求路由至真實來源的情況，請為該萬用字元路徑建立一個「虛設」來源。

Example：建立「虛設」來源

在下列範例中，端點 abc123 和 def456 路由至「真實」來源，但對任何其他端點的視訊內容的請求會路由至沒有適當子網域的 `mediapackage.us-west-2.amazonaws.com`，這樣會導致 HTTP 404 錯誤。

MediaPackage 端點：

```
https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8
https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/def456/index.m3u8
```

CloudFront 來源 A：

```
Domain: 3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com
Path: None
```

CloudFront 來源 B：

```
Domain: mediapackage.us-west-2.amazonaws.com
Path: None
```

CloudFront 快取行為：

1. Path: /out/v1/abc123/* forward to Origin A
2. Path: /out/v1/def456/* forward to Origin A
3. Path: * forward to Origin B

對於其他分佈設定，請根據您的其他技術需求或業務需求來設定特定的值。如需適用於分佈的所有選項清單以及設定相關資訊，請參閱[the section called “所有分佈設定”](#)。

當您完成選擇其他分佈設定時，請選擇 Create Distribution (建立分佈)。

4. 選擇您剛才建立的分佈，然後選擇 Behaviors (行為)。
5. 選擇預設的快取行為，然後選擇 Edit (編輯)。針對您為原始伺服器選擇的頻道指定正確的快取行為設定。您之後將會新增額外的一或多部原始伺服器，並編輯其快取行為設定。
6. 前往 [CloudFront 分佈頁面](#)。
7. 等到分佈的 Last modified (上次修改) 欄中的值從 Deploying (部署) 變成日期和時間，則表示 CloudFront 已建立您的分佈。

步驟 2：為 MediaPackage 端點的網域新增來源

重複這裡的步驟，將每個 MediaPackage 頻道端點新增至您的分佈，並且務必記住要建立「虛設」來源。

新增其他端點當做原始伺服器

1. 在 CloudFront 主控台中，選擇您為頻道建立的分佈。
2. 選擇 Origins (原始伺服器)，然後選擇 Create origin (建立原始伺服器)。
3. 針對 Origin domain (原始網域)，從下拉式清單中選擇您頻道的 MediaPackage 端點。
4. 對於其他設定，請根據您的其他技術需求或業務需求來設定值。如需詳細資訊，請參閱[the section called “原始設定”](#)。
5. 選擇 Create Origin (建立原始伺服器)。

步驟 3：為所有端點設定快取行為

對於每個端點，您都必須設定快取行為，以新增路徑模式來正確地路由傳遞請求。您指定的路徑模式取決於您提供的視訊格式。以下程序包含用於 Apple HLS、CMAF、DASH 和 Microsoft Smooth Streaming 格式的路徑模式資訊。

您通常會針對每個端點設定兩個快取行為：

- 父系資訊清單，也就是檔案的索引。
- 區段，也就是影片內容的檔案。

為端點建立快取行為

1. 在 CloudFront 主控台中，選擇您為頻道建立的分佈。
2. 選擇 Behaviors (行為)，然後選擇 Create behavior (建立行為)。
3. 對於路徑模式，使用特定 MediaPackage OriginEndpoint GUID 作為路徑字首。

路徑模式

如果是 HLS 端點 (如 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8`)，請建立以下兩個快取行為：

- 對於父系和子系資訊清單，請使用 `/out/v1/abc123/*.m3u8`。
- 對於內容區段，請使用 `/out/v1/abc123/*.ts`。

如果是 CMAF 端點 (如 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.m3u8`)，請建立以下兩個快取行為：

- 對於父系和子系資訊清單，請使用 `/out/v1/abc123/*.m3u8`。
- 對於內容區段，請使用 `/out/v1/abc123/*.mp4`。

如果是 DASH 端點 (如 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.mpd`)，請建立以下兩個快取行為：

- 對於父系資訊清單，請使用 `/out/v1/abc123/*.mpd`。
- 對於內容區段，請使用 `/out/v1/abc123/*.mp4`。

如果是 Microsoft Smooth Streaming 端點

(如 `https://3ae97e9482b0d011.mediapackage.us-west-2.amazonaws.com/out/v1/abc123/index.ism`)，則只會提供清單檔案，所以您只會建立一個快取行為：`out/v1/abc123/index.ism/*`。

4. 對於每個快取行為，請指定下列設定的值：

檢視器通訊協定政策

選擇 Redirect HTTP to HTTPS (將 HTTP 重新引導至 HTTPS)。

快取政策和原始伺服器請求政策

針對 Cache policy (快取政策)，選擇 Create policy (建立政策)。針對新的快取政策，請指定下列設定：

最短 TTL

設定為 5 秒或更少，以協助防止提供過時內容。

查詢字串

針對 (Cache key settings (快取金鑰設定) 中的) Query strings (查詢字串)，選擇 Include specified query strings (包含指定查詢字串)。針對 Allow (允許)，輸入下列值來進行新增，並選擇 Add item (新增項目)：

- 新增 `m` 作為您希望 CloudFront 用於快取基礎的查詢字串參數。MediaPackage 回應一律包含 `?m=###` 標籤，以擷取端點的修改時間。如果已使用此標籤的另一個值來快取內容，CloudFront 會請求新的資訊清單，而不是提供快取的版本。
- 如果使用 MediaPackage 中的時間轉換檢視行為功能，請針對資訊清單請求 (`start`、`end` 和 `*.m3u8`)，將 `*.mpd` 和 `index.ism/*` 指定為快取行為的額外查詢字串參數。如此一來，就會提供資訊清單請求內所要求之時段的專屬內容。如需有關時間轉移檢視和格式化內容 `start` 與 `end` 請求參數的詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的[時間轉移檢視](#)。
- 如果您在 MediaPackage 中使用資訊清單篩選功能，請針對資訊清單請求 (`*.m3u8`、`*.mpd` 和 `index.ism/*`)，將 `aws.manifestfilter` 指定為搭配快取行為使用快取政策中額外的查詢字串參數。這會將您的分佈設定為將 `aws.manifestfilter` 查詢字串轉發到您的 MediaPackage 原始伺服器，這是資訊清單篩選功能運作所必需的。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的[資訊清單檔案篩選](#)。
- 如果您使用低延遲 HLS (LL-HLS)，請指定 `_HLS_msn` 和 `_HLS_part` 作為您搭配清單檔案請求的快取行為所使用快取政策 (`*.m3u8`) 的其他查詢字串參數。這會將您的分佈設定為將 `_HLS_msn` 和 `_HLS_part` 查詢字串轉送到您的 MediaPackage 來源，如此 LL-HLS 封鎖播放清單請求功能才能運作。

5. 選擇建立。
6. 建立快取政策後，返回快取行為建立工作流程。重新整理快取政策的清單，然後選擇您剛建立的政策。
7. 選擇 Create behavior (建立行為)。

8. 如果您的端點並非 Microsoft Smooth Streaming 端點，請重複前述的步驟來建立第二個快取行為。

步驟 4：啟用標頭為主的 MediaPackage CDN 授權

我們建議您在 MediaPackage 端點和 CloudFront 分佈之間啟用標頭為主的 MediaPackage CDN 授權。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的[在 MediaPackage 中啟用 CDN 授權](#)。

步驟 5：使用 CloudFront 提供即時串流頻道

在您建立分佈、新增來源、建立快取行為，以及啟用標頭為主的 CDN 授權之後，就可以使用 CloudFront 提供即時串流頻道。CloudFront 會根據您為快取行為設定的設定，將檢視器的請求路由傳送至正確的 MediaPackage 端點。

對於您的應用程式 (例如，媒體播放器) 中的連結，請以 CloudFront URL 的標準格式為媒體檔案指定 URL。如需詳細資訊，請參閱[the section called “自訂檔案 URL”](#)。

使用 AWS Elemental MediaPackage 提供隨選視訊內容

如果您從 AWS Elemental MediaPackage 原始伺服器產生 video-on-demand (VOD) 內容，您可以建立 CloudFront 分佈並設定最佳化快取行為，將 VOD 內容提供給檢視器。下列程序假設您已建立具有[封裝組態](#)的[封裝群組](#)，並使用 MediaPackage [擷取資產](#)。

若要手動為 MediaPackage 建立 CloudFront 分佈，請依照下列步驟進行：

步驟 1：建立及設定 CloudFront 分佈

完成以下程序，針對您使用 MediaPackage 建立的封裝群組設定 CloudFront 分佈。

為您的 VOD 內容建立分佈

1. 登入 AWS 管理主控台 並開啟位於的 CloudFront 主控台<https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇 Create Distribution (建立分佈)。
3. 選擇分佈的設定，包括下列項目：

原始網域

MediaPackage 封裝群組的原始伺服器。在文字欄位中輸入原始伺服器 URL 值。原始伺服器必須為 HTTPS URL。

例如，對於像是 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8` 的 HLS 端點，原始網域會是 `3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com`。

如需詳細資訊，請參閱 [the section called “原始網域”](#)。

原始伺服器路徑

提供內容的路徑。

如需有關原始伺服器路徑如何運作的詳細資訊，請參閱 [the section called “原始伺服器路徑”](#)。

Important

須有萬用字元路徑 * 才能路由至 CloudFront 分佈中的某處。為了避免不符合明確路徑的請求路由至真實來源的情況，請為該萬用字元路徑建立一個「虛設」來源。

Example：建立「虛設」來源

在下列範例中，封裝組態 `def456` 和 `321xyz` 路由至「真實」原始伺服器，但對任何其他端點的視訊內容的請求會路由至沒有適當子網域的 `mediapackage-vod.us-west-2.amazonaws.com`，這樣會導致 HTTP 404 錯誤。

具有兩種封裝組態的封裝群組之單一資產的 MediaPackage 內容 URL：

```
https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8
https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/321xyz/654uvw/index.m3u8
```

CloudFront 來源 A：

```
Domain: 3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com
Path: None
```

CloudFront 來源 B：


```
Domain: mediapackage-vod.us-west-2.amazonaws.com
Path: None
```

CloudFront 快取行為：

1. Path: /out/v1/*/def456/* forward to Origin A
2. Path: /out/v1/*/321xyz/* forward to Origin A
3. Path: * forward to Origin B

對於其他分佈設定，請根據您的其他技術需求或業務需求來設定特定的值。如需適用於分佈的所有選項清單以及設定相關資訊，請參閱[the section called “所有分佈設定”](#)。

當您完成選擇其他分佈設定時，請選擇 Create Distribution (建立分佈)。

4. 選擇您剛才建立的分佈，然後選擇 Behaviors (行為)。
5. 選擇預設的快取行為，然後選擇 Edit (編輯)。針對您為原始伺服器選擇的封裝組態指定正確的快取行為設定。您之後將會新增額外的一或多部原始伺服器，並編輯其快取行為設定。
6. 前往 [CloudFront 分佈頁面](#)。
7. 等到分佈的 Last modified (上次修改) 欄中的值從 Deploying (部署) 變成日期和時間，則表示 CloudFront 已建立您的分佈。

步驟 2：為 MediaPackage 封裝群組的網域新增原始伺服器

重複這裡的步驟，將每個 MediaPackage 封裝群組新增至您的分佈，並且務必記住要建立「虛設」原始伺服器。

新增其他封裝群組做為原始伺服器

1. 在 CloudFront 主控台中，選擇您為頻道建立的分佈。
2. 選擇 Origins (原始伺服器)，然後選擇 Create origin (建立原始伺服器)。
3. 對於原始伺服器網域，輸入 MediaPackage 封裝群組的 URL。
4. 對於其他設定，請根據您的其他技術需求或業務需求來設定值。如需詳細資訊，請參閱[the section called “原始設定”](#)。
5. 選擇 Create Origin (建立原始伺服器)。

步驟 3：設定所有封裝組態的快取行為

對於每個封裝組態，您都必須設定快取行為，以新增路徑模式來正確地路由傳遞請求。您指定的路徑模式取決於您提供的視訊格式。以下程序包含用於 Apple HLS、CMAF、DASH 和 Microsoft Smooth Streaming 格式的路徑模式資訊。

您通常會為每個封裝組態設定多個快取行為：

- 父系資訊清單，也就是檔案的索引。
- 區段，也就是影片內容的檔案。格式可能會針對內容使用多個副檔名，視您的組態而定。每個延伸模組都需要快取行為。

為封裝組態建立快取行為

1. 在 CloudFront 主控台中，選擇您為頻道建立的分佈。
2. 選擇 Behaviors (行為)，然後選擇 Create behavior (建立行為)。
3. 對於路徑模式，使用特定 MediaPackage VOD 封裝組態 GUID 作為路徑字首。這是 MediaPackage VOD 路徑中的第二個 GUID。

路徑模式

如果是 HLS 內容 (如 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8`)，請建立以下兩個快取行為：

- 對於父系和子系資訊清單，請使用 `/out/v1/*/def456/*.m3u8`。
- 對於內容區段，使用 `/out/v1/*/def456/*.ts` 並重複執行所有必要的區段延伸。

如果是 CMAF 內容 (如 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.m3u8`)，請建立以下快取行為：

- 對於父系和子系資訊清單，請使用 `/out/v1/*/def456/*.m3u8`。
- 對於內容區段，使用 `/out/v1/*/def456/*.mp4` 並重複執行所有必要的區段延伸。

如果是 DASH 內容 (如 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.mpd`)，請建立以下快取行為：

- 對於父系資訊清單，請使用 `/out/v1/*/def456/*.mpd`。

- 對於內容區段，請使用 `/out/v1/*/def456/*.mp4`。

如果是 Microsoft Smooth Streaming 端點

(如 `https://3ae97e9482b0d011.egress.mediapackage-vod.us-west-2.amazonaws.com/out/v1/abc123/def456/ghi789/index.ism/Manifest`)，則只會提供清單檔案，所以您只會建立一個快取行為：`out/v1/*/def456/*/index.ism/*`。

4. 對於每個快取行為，請指定下列設定的值：

檢視器通訊協定政策

選擇 Redirect HTTP to HTTPS (將 HTTP 重新引導至 HTTPS)。

快取政策和原始伺服器請求政策

針對 Cache policy (快取政策)，選擇 Create policy (建立政策)。針對新的快取政策，請指定下列設定：

最短 TTL

設定為 5 秒或更少，以協助防止提供過時內容。

查詢字串

針對 (Cache key settings (快取金鑰設定) 中的) Query strings (查詢字串)，選擇 Include specified query strings (包含指定查詢字串)。針對 Allow (允許)，輸入下列值來進行新增，並選擇 Add item (新增項目)：

- 如果您在 MediaPackage 中使用資訊清單篩選功能，請針對資訊清單請求 (`*.m3u8`、`*.mpd` 和 `index.ism/*`)，將 `aws.manifestfilter` 指定為搭配快取行為使用快取政策中額外的查詢字串參數。這會將您的分佈設定為將 `aws.manifestfilter` 查詢字串轉發到您的 MediaPackage 原始伺服器，這是資訊清單篩選功能運作所必需的。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的[資訊清單檔案篩選](#)。

5. 選擇建立。
6. 建立快取政策後，返回快取行為建立工作流程。重新整理快取政策的清單，然後選擇您剛建立的政策。
7. 選擇 Create behavior (建立行為)。
8. 如果您的端點並非 Microsoft Smooth Streaming 端點，請重複前述的步驟來建立第二個快取行為。

步驟 4：啟用標頭為主的 MediaPackage CDN 授權

我們建議您在 MediaPackage VOD 內容和 CloudFront 分佈之間啟用標頭為主的 MediaPackage CDN 授權。如需詳細資訊，請參閱《AWS Elemental MediaPackage 使用者指南》中的[在 MediaPackage 中啟用 CDN 授權](#)。

步驟 5：使用 CloudFront 提供 VOD 內容

在您建立分佈、新增原始伺服器、建立快取行為，以及啟用標頭為主的 CDN 授權之後，就可以使用 CloudFront 提供 VOD 內容。CloudFront 會根據您為快取行為設定的設定，將檢視器的請求路由傳送至正確的 MediaPackage VOD 內容。

對於您的應用程式 (例如，媒體播放器) 中的連結，請以 CloudFront URL 的標準格式為媒體檔案指定 URL。如需詳細資訊，請參閱[the section called “自訂檔案 URL”](#)。

媒體品質感知彈性

媒體品質感知彈性 (MQAR) 是 [Amazon CloudFront](#) 和 [AWS Media Services](#) 之間的整合功能。MQAR 根據媒體品質可信度分數 (MQCS) 提供自動跨區域原始伺服器選擇。MQCS 由 AWS Elemental MediaLive 根據會影響瀏覽者感知媒體品質體驗的參數合成。您可以使用可在 CloudFront 原始群組容錯移轉條件中指定的多個選項，設定 CloudFront 和 AWS Media Services 以高彈性提供即時事件串流。

當您為分佈啟用 MQAR 功能時，您可以授權 CloudFront 自動選取被視為具有最高品質分數的原始伺服器。

品質分數代表您原始伺服器的感知媒體串流品質問題，例如黑色影格、凍結或捨棄影格，或重複影格。例如，如果您的 AWS Elemental MediaPackage v2 原始伺服器部署在兩個不同的 AWS 區域，而一個報告比另一個更高的媒體品質分數，CloudFront 會自動切換到報告較高分數的原始伺服器。

為實現此目標，CloudFront 將執行下列操作：

1. CloudFront 會將 GET 請求轉送至主要 MediaPackage 原始伺服器，同時向次要 MediaPackage 原始伺服器啟動 HEAD 請求。CloudFront 會在每個原始伺服器的回應標頭中收到媒體品質分數。
2. 接下來，CloudFront 會追蹤每個原始伺服器的分數，並在新請求送達時，使用此資訊來判斷分數較高的原始伺服器。

原始伺服器的媒體品質分數可以即時變更。CloudFront 會透過耗用 MQCS 變更來決定這一點，並在原始伺服器之間切換，以確保瀏覽者看到更高的媒體品質內容。如需詳細資訊，請參閱《AWS Elemental MediaPackage V2 使用者指南》中的[透過 MediaPackage 運用媒體品質分數](#)。

MQAR 可協助 CloudFront 盡早判斷是否有可能影響客戶的問題。例如，網路連線、視訊處理、音訊遺失或下降等問題，編碼器速度問題都可能影響檢視器的媒體品質分數。

MQAR 可在原始伺服器之間無縫切換，讓您可以在 上部署彈性的跨區域end-to-end媒體交付工作流程 AWS，並為瀏覽者提供高品質的內容。

Note

目前，此功能僅支援 MediaPackage v2 原始伺服器。

若要為您的分佈啟用此功能，請完成下列步驟：

1. 建立 MediaPackage v2 原始伺服器並在端點組態中啟用此功能 (如果您尚未這樣做)。對於跨區域部署，AWS 區域 請使用相同的設定在不同的 中建立次要頻道。如需詳細資訊，請參閱《AWS Elemental MediaPackage V2 使用者指南》中的以下主題：
 - [建立頻道和端點](#)
 - [啟用媒體品質分數](#)
2. 若要使用 CloudFront 的 MediaPackage v2 原始伺服器，請建立或更新 CloudFront 分佈。請參閱 [建立分發](#) 和 [更新分佈](#)。
3. 建立原始伺服器群組，然後選取兩個原始伺服器做為主要和次要伺服器。在您的原始伺服器群組中，啟用媒體品質分數選項。如需詳細資訊，請參閱[建立原始伺服器群組](#)。
4. 在分佈的快取行為中，選取您建立的[原始伺服器群組](#)。建議快取行為應符合頻道路徑模式。

如果 CloudFront 判斷兩個 MediaPackage v2 原始伺服器都有相同的分數，則會將請求轉送至原始伺服器群組中列出的主要原始伺服器。如果最初選取的原始伺服器回應的錯誤碼符合您在原始伺服器群組中指定的容錯移轉準則，則 CloudFront 會重試原始伺服器群組中替代原始伺服器的請求，無論其媒體品質分數為何。

備註

- CloudFront 會追蹤每個快取行為的品質分數，這些行為利用針對媒體品質分數啟用的原始伺服器群組。如果相同的原始伺服器群組用於發出媒體品質分數的多個頻道，請為每個頻道的路徑模式建立個別的快取行為，以避免混合其分數。如需原始伺服器群組的詳細資訊，請參閱 [分佈的一般配額](#)。

- 目前，當您在與分佈快取行為相關聯的原始伺服器面向觸發 (原始伺服器請求和原始伺服器回應) 中使用 [Lambda@Edge](#) 函數時，無法使用 MQAR。如需詳細資訊，請參閱[快取行為設定](#)。
- 如果您已啟用 MQAR 功能和原始存取控制 (OAC)，請將 `mediapackagev2:GetHeadObject` 動作新增至 IAM 政策。MQAR 需要此許可才能將 HEAD 請求傳送至 MediaPackage v2 原始伺服器。如需 OAC 的詳細資訊，請參閱[限制對 AWS Elemental MediaPackage v2 原始伺服器的存取](#)。

MQAR 日誌欄位

CloudFront 會在即時存取日誌中提供下列欄位，以反映品質分數和選取的原始伺服器。您可以在 CloudFront 即時存取日誌日誌中啟用這些欄位：

- `r-host`
- `sr-reason`
- `x-edge-mqcs`

如需詳細資訊，請參閱 [欄位](#) 65-67。

使用函數在邊緣自訂

使用 Amazon CloudFront，您可以撰寫自己的程式碼，以自訂 CloudFront 分發處理 HTTP 請求和回應的方式。程式碼會靠近檢視器 (使用者) 執行，以將延遲降至最低，而且您不必管理伺服器或其他基礎設施。您可以撰寫程式碼可以操作流經 CloudFront 的請求和回應，執行基本身分驗證和授權，在邊緣產生 HTTP 回應等。

您撰寫並附加至 CloudFront 分發的程式碼就稱為邊緣函數。CloudFront 提供兩種撰寫和管理邊緣函數的方式：

CloudFront Functions

您可以在 JavaScript 中撰寫輕量型函數，以進行高度擴展及延遲敏感的 CDN 自訂。CloudFront Functions 執行時間環境提供低於一毫秒的啟動時間，可立即擴展以每秒處理數百萬個要求，並且非常安全。CloudFront Functions 是 CloudFront 的原生功能，這表示您可以在 CloudFront 中完全建置、測試和部署程式碼。

Lambda@Edge

Lambda@Edge 是 [AWS Lambda](#) 的延伸，為複雜函數提供強大且靈活的運算，同時提供更靠近檢視器的完整應用程式邏輯，並且具有高度安全性。Lambda@Edge 函數在 Node.js 或 Python 執行時期環境中執行。您可以將它們發佈到單一 AWS 區域，但當您將函數與 CloudFront 分佈建立關聯時，Lambda@Edge 會自動將程式碼複製到世界各地。

如果您在 CloudFront AWS WAF 上執行，則可以對 CloudFront Functions 和 Lambda@Edge 使用 AWS WAF 插入的標頭。這適用於檢視器和原始伺服器請求和回應。

主題

- [CloudFront Functions 與 Lambda@Edge 之間的差異](#)
- [使用 CloudFront Functions 在邊緣進行自訂](#)
- [使用 CloudFront Connection Functions 自訂](#)
- [使用 Lambda@Edge 在邊緣自訂](#)
- [對邊緣函數的限制](#)

CloudFront Functions 與 Lambda@Edge 之間的差異

CloudFront Functions 和 Lambda@Edge 都提供了一種方法來執行回應 CloudFront 事件的程式碼。

CloudFront Functions 適合在以下使用案例用於輕量型且執行時間較短的函數：

- 快取金鑰標準化：轉換 HTTP 請求屬性 (標頭、查詢字串、Cookie，甚至 URL 路徑)，以建立最佳[快取金鑰](#)，這可提升您的快取命中率。
- 標頭操作：在請求或回應中插入、修改或刪除 HTTP 標頭。例如，您可以為每個請求新增一個 True-Client-IP 標頭。
- URL 重新導向或重寫：根據請求中的資訊將檢視器重新導向至其他頁面，或將全部請求從一個路徑重寫至另一個路徑。
- 請求授權：透過檢查授權標頭或其他請求中繼資料驗證雜湊的授權記號，例如 JSON Web 記號 (JWT)。

若要開始使用 CloudFront Functions，請參閱[使用 CloudFront Functions 在邊緣進行自訂](#)。

Lambda@Edge 非常適合下列使用案例：

- 需要數毫秒或更長時間才能完成的函數
- 需要可調整 CPU 或記憶體體的函數
- 依賴第三方程式庫的函數（包括 AWS SDK，用於與其他程式庫整合 AWS 服務）
- 需要網路存取才能使用外部服務進行處理的函數
- 需要檔案系統存取或存取 HTTP 請求內文的函數

若要開始使用 Lambda@Edge，請參閱[使用 Lambda@Edge 在邊緣自訂](#)。

為了協助您選擇適合使用案例的選項，請使用下表瞭解 CloudFront Functions 和 Lambda@Edge 之間的差異。如需更多資訊瞭解適用於原始修改協助程式方法的差異，請參閱[在 CloudFront Functions 和 Lambda@Edge 之間進行選擇](#)。

	CloudFront Functions	Lambda@Edge
程式設計語言	JavaScript (與 ECMAScript 5.1 相容)	Node.js 和 Python
事件來源	• 檢視器請求 • 檢視器回應	• 檢視器請求 • 檢視器回應 • 原始伺服器請求 • 原始伺服器回應

	CloudFront Functions	Lambda@Edge
支援 Amazon CloudFront KeyValuesStore	是 CloudFront KeyValuesStore 僅支援 JavaScript 執行時期 2.0	否
擴展	每秒最多數百萬個請求	每秒每個區域最多 10,000 個請求
函數持續時間	低於一毫秒	最長 30 秒 (檢視器請求和檢視器回應) 最多 30 秒 (原始伺服器請求和原始伺服器回應)
最大函數記憶體大小	2 MB	128 MB (檢視器請求和檢視器回應) 10,240 MB (10 GB) (原始伺服器請求和原始伺服器回應) 如需詳細資訊，請參閱 Lambda@Edge 的配額 。
函數程式碼和包含程式庫的最大規模	10 KB	50 MB (檢視器請求和檢視器回應) 50 MB (原始伺服器請求和原始伺服器回應)
網路存取	否	是
檔案系統存取	否	是
請求內文存取	否	是

	CloudFront Functions	Lambda@Edge
存取地理位置和裝置資料	是	否 (檢視器請求和檢視器回應) 是 (原始伺服器請求和原始伺服器回應)
可以完全在 CloudFront 內建置和測試	是	否
函數日誌記錄和指標	是	是

使用 CloudFront Functions 在邊緣進行自訂

使用 CloudFront Functions，您可以在 JavaScript 中撰寫輕量型函數，以進行高擴展性、延遲敏感的 CDN 自訂。您的函數可以操作流經 CloudFront 的請求和回應，執行基本身分驗證和授權，在邊緣產生 HTTP 回應等。CloudFront Functions 執行時間環境提供低於一毫秒的啟動時間，可立即擴展以每秒處理數百萬個要求，並且非常安全。CloudFront Functions 是 CloudFront 的原生功能，這表示您可以在 CloudFront 中完全建置、測試和部署程式碼。

當您將 CloudFront 函式與 CloudFront 分佈建立關聯時，CloudFront 會在 CloudFront 邊緣位置攔截請求和回應，並將它們傳遞給您的函數。發生下列事件時，您可以調用 CloudFront 函數：

- 當 CloudFront 接收到來自檢視器的請求 (檢視器請求) 時
- 在 CloudFront 傳回回應給檢視器 (檢視器回應) 之前
- 在建立 TLS 連線 (連線請求) 期間 - 目前可用於交互 TLS (mTLS) 連線

如需 CloudFront Functions 的詳細資訊，請參閱下列主題：

主題

- [教學課程：使用 CloudFront Functions 建立簡單的函數](#)
- [教學課程：建立包含鍵值的 CloudFront 函數](#)
- [撰寫函數程式碼](#)
- [建立函數](#)
- [測試函數](#)
- [更新函數](#)

- [發佈函數](#)
- [將函數與分佈相關聯](#)
- [Amazon CloudFront KeyValueStore](#)

教學課程：使用 CloudFront Functions 建立簡單的函數

本教學課程將協助您開始使用 CloudFront Functions。您可建立簡單函數將檢視器重新導向至不同的 URL，此外也會傳回自訂回應標頭。

內容

- [必要條件](#)
- [建立函數](#)
- [驗證函數](#)

必要條件

要使用 CloudFront Functions，您需要一個 CloudFront 分佈。如果您沒有帳戶，請參閱 [開始使用 CloudFront 標準分佈](#)。

建立函數

您可使用 CloudFront 主控台建立簡單函數，該函數將檢視器重新導向至不同的 URL，以及傳回自訂回應標頭。

建立 CloudFront 函數

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽面板上選擇函數，然後選擇建立函數。
3. 在建立函數頁面對名稱輸入函數名稱，例如 *MyFunctionName*。
4. (選用) 對於說明，請輸入函數的說明內容，例如 **Simple test function**。
5. 對於執行時期，請保留預設選取的 JavaScript 版本。
6. 選擇 Create function (建立函數)。
7. 複製下列函數程式碼。此函數程式碼會將檢視者重新導向至不同的 URL，並傳回自訂回應標頭。

```
function handler(event) {  
    // NOTE: This example function is for a viewer request event trigger.  
    // Choose viewer request for event trigger when you associate this function  
    with a distribution.  
    var response = {  
        statusCode: 302,  
        statusDescription: 'Found',  
        headers: {  
            'cloudfront-functions': { value: 'generated-by-CloudFront-Functions' },  
            'location': { value: 'https://aws.amazon.com/cloudfront/' }  
        }  
    };  
    return response;  
}
```

8. 對於函數程式碼，請將程式碼貼到程式碼編輯器中，以取代預設程式碼。
9. 選擇儲存變更。
10. (選用) 您可以在發佈之前測試函數。本教學課程不會介紹如何測試函數。如需更多詳細資訊，請參閱 [測試函數](#)。
11. 選擇發佈索引標籤，然後選擇發佈函數。您必須先發佈該函數，才能將其與 CloudFront 分佈建立關聯。
12. 接下來您可以將函數與分佈或快取行為建立關聯。在 *MyFunctionName* 頁面選擇發佈索引標籤。

 Warning

在下列步驟中，選擇用於測試的分佈或快取行為。請勿將此測試函數與正式作業環境中使用的分佈或快取行為建立關聯。

13. 選擇 Add association (建立關聯)。
14. 在關聯對話方塊中，選擇分佈和/或快取行為。對於事件類型，請保留預設值。
15. 選擇 Add association (建立關聯)。

關聯的分佈資料表中會顯示關聯的分佈。

16. 等待幾分鐘，讓關聯的分佈完成部署。若要檢查分佈狀態，請在關聯的分佈資料表中選取分佈，然後選擇檢視分佈。

當分佈的狀態為已部署時，您就可以確認該函數正常運作。

驗證函數

您部署函數之後，可以驗證該函數是否為您的分佈發揮作用。

驗證函數

1. 在 Web 瀏覽器中，導覽至分佈的網域名稱 (例如 `https://d111111abcdef8.cloudfront.net`)。

該函數傳回重定向到瀏覽器，因此瀏覽器會自動轉到 `https://aws.amazon.com/cloudfront/`。

2. 您可在命令列視窗中使用 `curl` 之類的工具，將請求傳送至分佈的網域名稱。

```
curl -v https://d111111abcdef8.cloudfront.net/
```

您在回應中會看到重新導向回應 (302 Found) 和函數新增的自訂回應標頭。以下範例為可能的回應形式。

Example

```
curl -v https://d111111abcdef8.cloudfront.net/
> GET / HTTP/1.1
> Host: d111111abcdef8.cloudfront.net
> User-Agent: curl/7.64.1
> Accept: */*
>
< HTTP/1.1 302 Found
< Server: CloudFront
< Date: Tue, 16 Mar 2021 18:50:48 GMT
< Content-Length: 0
< Connection: keep-alive
< Location: https://aws.amazon.com/cloudfront/
< Cloudfront-Functions: generated-by-CloudFront-Functions
< X-Cache: FunctionGeneratedResponse from cloudfront
< Via: 1.1 3035b31bddaf14eded329f8d22cf188c.cloudfront.net (CloudFront)
< X-Amz-Cf-Pop: PHX50-C2
< X-Amz-Cf-Id: ULZdIz6j43uGB1Xyob_JctF9x7CCbwpNniiMlmNbmwzH1YWP9FsEHg==
```

教學課程：建立包含鍵值的 CloudFront 函數

本教學課程說明如何使用 CloudFront 函數納入鍵值。鍵值是鍵值對的一部分。您可以在函數程式碼中包含名稱 (來自鍵值對)。函數執行時，CloudFront 會以值取代該名稱。

鍵值對是存放在鍵值存放區中的變數。當您在函數中使用鍵 (而不是硬式編碼值) 時，您的函數會更靈活。您可以變更鍵的值，而不需要部署程式碼變更。鍵值對也可以減少函數的大小。如需更多詳細資訊，請參閱 [???。](#)

內容

- [必要條件](#)
- [建立鍵值存放區](#)
- [將鍵值對新增到存放區](#)
- [建立鍵值存放區與函數的關聯](#)
- [測試並發佈函數程式碼](#)

必要條件

如果您是初次使用 CloudFront Functions 函數和鍵值存放區，建議您遵循 [the section called “教學課程：建立簡單的 CloudFront 函數”](#) 中的教學課程。

完成該教學課程後，您可以遵循本教學課程來延伸您建立的函數。在本教學課程中，建議您先建立鍵值存放區。

建立鍵值存放區

首先請建立用於函數的鍵值存放區。

建立鍵值存放區

1. 規劃您要包含在函數中的鍵值對。請記下這些鍵的名稱。您要在函數中使用的所有鍵值對都必須位於單一鍵值存放區中。
2. 決定工作的順序。有兩種方式可以繼續：
 - 建立一個鍵值存放區，並將鍵值對新增到存放區中。然後建立 (或修改) 函數並加入鍵的名稱。
 - 或者，建立 (或修改) 函數，並加入您要使用的鍵的名稱。然後建立一個鍵值存放區，並新增鍵值對。

3. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
4. 在導覽視窗中選擇函數，然後選擇 KeyValueStores 索引標籤。
5. 選擇建立 KeyValueStore 並輸入下列欄位：
 - 輸入存放區的名稱及 (選用) 說明。
 - 將 S3 URI 保留空白。在本教學課程中，您將手動輸入鍵/值對。
6. 選擇建立。此時會顯示新鍵值存放區的詳細資訊頁面。此頁面包含目前空白的鍵值對區段。

將鍵值對新增到存放區

接著手動將鍵值對清單新增至您先前建立的鍵值存放區。

將鍵值對新增到鍵值存放區

1. 在鍵值對區段中選擇新增鍵值對。
2. 選擇新增配對，然後輸入鍵和值。選擇核取記號以確認您的變更，然後重複此步驟以新增更多鍵值對。
3. 完成後請選擇儲存變更，以儲存在鍵值存放區中的所有鍵值對。在確認對話上選擇完成。

您現在有一個包含一組鍵值對的鍵值存放區。

建立鍵值存放區與函數的關聯

您現在已建立鍵值存放區。並且您已經建立或修改了一個函數，其中包含鍵值存放區中的鍵名稱。您現在可以建立鍵值存放區與函數的關聯。您從函數內建立該關聯。

建立函數與鍵值存放區的關聯

1. 在導覽視窗中，選擇函數。依據預設，函數索引標籤會顯示在頂端。
2. 選擇函數名稱，並在關聯的 KeyValueStore 區段中選擇關聯現有 KeyValueStore。
3. 選取鍵值存放區，然後選擇關聯 KeyValueStore。

 Note

每個函數只能與一個鍵值存放區建立關聯。

測試並發佈函數程式碼

將鍵值存放區與函數建立關聯後，您可以測試和發佈函數程式碼。每次修改函數程式碼時，應一律對其進行測試，包括執行以下操作時：

- 將鍵值存放區與函數建立關聯。
- 修改函數及其鍵值存放區，以包含新的鍵值對。
- 變更鍵值對的值。

測試並發佈函數程式碼

1. 如需有關如何測試函數的詳細資訊，請參閱 [the section called “測試函數”](#)。確定您選擇在 DEVELOPMENT 階段測試函數。
2. 您準備好在 LIVE 環境中使用函數 (搭配新的或修訂的鍵值對) 時，請發佈該函數。

當您發佈時，CloudFront 會將函數的版本從 DEVELOPMENT 階段複製到即時階段。該函數具有新程式碼，並與鍵值存放區相關聯。(在即時階段無需再次執行關聯。)

如需有關如何發佈函數的詳細資訊，請參閱 [the section called “發佈函數”](#)。

撰寫函數程式碼

您可以使用 CloudFront Functions 撰寫 JavaScript 的輕量型函數，以進行高擴展性、對延遲敏感的 CDN 自訂。您的函數程式碼可以操作流經 CloudFront 的請求和回應，執行基本身分驗證和授權，在邊緣產生 HTTP 回應等。

為協助您為 CloudFront Functions 撰寫函數程式碼，請參閱下列主題。如需程式碼範例，請參閱 [CloudFront 的 CloudFront Functions 範例](#) 和 GitHub 上的 [amazon-cloudfront-functions 儲存庫](#)。

主題

- [確定函數的用途](#)
- [CloudFront Functions 事件結構](#)

- [適用於 CloudFront Functions 的 JavaScript 執行時期功能](#)
- [鍵值存放區的協助程式方法](#)
- [原始伺服器修改的 Helper 方法](#)
- [CloudFront SaaS Manager 屬性的 helper 方法](#)
- [使用 async 和 await](#)
- [CloudFront Functions 的 CWT 支援](#)
- [一般協助程式方法](#)

確定函數的用途

在撰寫函數程式碼之前，請先確定函數的用途。CloudFront Functions 中的大多數函數都具有以下用途之一。

主題

- [修改檢視器請求事件類型中的 HTTP 請求](#)
- [在檢視器請求事件類型中產生 HTTP 回應](#)
- [在檢視者回應事件類型中修改 HTTP 回應](#)
- [驗證連線請求事件類型的 mTLS 連線](#)
- [相關資訊](#)

無論函數的用途如何，handler 都是任何函數的入口點。它採用一個稱為 event 的參數，該參數由 CloudFront 傳遞給函數。event 是一個 JSON 對象，其中包含 HTTP 請求的表示 (以及回應，前提是您的函數修改了 HTTP 回應)。

修改檢視器請求事件類型中的 HTTP 請求

您的函數可以修改 CloudFront 從檢視者 (用戶端) 接收的 HTTP 請求，並將修改後的請求返回給 CloudFront 以繼續處理。例如，您的函數程式碼可能會標準化[快取金鑰](#)或修改請求標頭。

建立及發布修改 HTTP 請求的函數時，請務必新增檢視器請求事件類型的關聯。如需詳細資訊，請參閱[建立函數](#)。這會使 CloudFront 每次收到來自檢視器的請求時執行該函數，然後再檢視請求的物件是否位於 CloudFront 快取中。

Example 範例

下面的虛擬程式碼顯示了修改 HTTP 請求的函數結構。

```
function handler(event) {  
    var request = event.request;  
  
    // Modify the request object here.  
  
    return request;  
}
```

該函數將修改後的 `request` 物件返回到 CloudFront。CloudFront 會按如下方式繼續處理返回的請求：檢查 CloudFront 快取以取得快取命中率，並在必要時將請求傳送至來源。

在檢視器請求事件類型中產生 HTTP 回應

您的函數可以在邊緣產生 HTTP 回應，並直接將其返回給檢視者 (用戶端)，而無需檢查快取的回應，也無需 CloudFront 進一步處理。例如，您的函數程式碼可能會將請求重新導向至新的 URL，或檢查授權並將 401 或 403 回應返回給未經授權的請求。

建立產生 HTTP 回應的函數時，請務必選擇檢視者請求事件類型。這表示 CloudFront 每次收到來自檢視者的請求時，該函數就會執行，然後 CloudFront 才進一步處理請求。

Example 範例

下面的虛擬程式碼顯示了產生 HTTP 回應的函數結構。

```
function handler(event) {  
    var request = event.request;  
  
    var response = ...; // Create the response object here,  
                        // using the request properties if needed.  
  
    return response;  
}
```

該函數將 `response` 物件返回到 CloudFront，CloudFront 立即將其返回給檢視者，而無需檢查 CloudFront 快取或將請求傳送至來源。

在檢視者回應事件類型中修改 HTTP 回應

您的函數可以在 CloudFront 將其發送給檢視者 (用戶端) 之前修改 HTTP 回應，無論該回應來自 CloudFront 快取還是來源。例如，您的函數代碼可能會新增或修改回應標頭、狀態碼，與本文內容。

建立可修改 HTTP 回應的函數時，請務必選擇檢視者回應事件類型。這表示無論回應是來自 CloudFront 快取還是來源，此函數都會在 CloudFront 將回應返回給檢視者之前執行。

Example 範例

下面的虛擬程式碼顯示了修改 HTTP 回應的函數結構。

```
function handler(event) {
    var request = event.request;
    var response = event.response;

    // Modify the response object here,
    // using the request properties if needed.

    return response;
}
```

該函數將修改後的 response 物件返回到 CloudFront，CloudFront 立即將其返回給檢視者。

驗證連線請求事件類型的 mTLS 連線

連線函數是一種在 TLS 連線期間執行的 CloudFront 函數類型，可提供自訂驗證和身分驗證邏輯。連線函數目前可用於交互 TLS (mTLS) 連線，您可以在其中驗證用戶端憑證，並實作超出標準憑證驗證的自訂身分驗證邏輯。在 TLS 交握程序期間執行的連線函數，可以根據憑證屬性、用戶端 IP 地址或其他條件允許或拒絕連線。

建立和發佈連線函數之後，請務必為啟用 mTLS 的分佈新增連線請求事件類型的關聯。這可讓函數在每次用戶端嘗試與 CloudFront 建立 mTLS 連線時執行。

Example

下列虛擬程式碼顯示連線函數的結構：

```
function connectionHandler(connection) {
    // Validate certificate and connection properties here.

    if (/* validation passes */) {
        connection.allow();
    } else {
        connection.deny();
    }
}
```

```
}
```

函數使用協助程式方法來判斷是否允許或拒絕連線。與檢視器請求和檢視器回應函數不同，連線函數無法修改 HTTP 請求或回應。

相關資訊

如需使用 CloudFront Functions 的詳細資訊，請參閱下列主題：

- [事件結構](#)
- [JavaScript 執行時期功能](#)
- [CloudFront Functions 範例](#)
- [對邊緣函數的限制](#)

CloudFront Functions 事件結構

CloudFront Functions 在運行該函數時將 event 物件作為輸入傳遞給您的函數程式碼。[測試函數](#)時，建立 event 物件並將其傳遞給函數。建立用於測試函數的 event 物件時，您可以省略 distributionDomainName 物件中的 distributionId、requestId 和 context 欄位。此外，請確保標頭名稱為小寫，CloudFront Functions 在生產期間傳遞給函數的 event 物件中也會始終採用此形式。

以下是此事件物件結構的概觀。

```
{
  "version": "1.0",
  "context": {
    <context object>
  },
  "viewer": {
    <viewer object>
  },
  "request": {
    <request object>
  },
  "response": {
    <response object>
  }
}
```

如需詳細資訊，請參閱下列主題：

主題

- [版本欄位](#)
- [內容物件](#)
- [連線事件結構](#)
- [檢視者物件](#)
- [請求物件](#)
- [回應物件](#)
- [狀態碼和本文](#)
- [查詢字串、標頭和 Cookie 的結構](#)
- [範例回應物件](#)
- [範例事件物件](#)

版本欄位

version 欄位包含指定 CloudFront Functions 事件物件版本的字串。目前版本是 1.0。

內容物件

context 物件包含有關事件的關聯式資訊。它包括以下欄位：

distributionDomainName

與事件關聯之標準分佈的 CloudFront 網域名稱 (例如 d111111abcdef8.cloudfront.net)。

只有在您的函數為標準分佈調用時，distributionDomainName 欄位才會出現。

endpoint

與事件關聯之連線群組的 CloudFront 網域名稱 (例如 d111111abcdef8.cloudfront.net)。

只有在您的函數為多租用戶分佈調用時，endpoint 欄位才會出現。

distributionId

與事件相關聯的發佈的 ID (例如 EDFDVBD6EXAMPLE)。

eventType

事件類型，viewer-request 或 viewer-response。

requestId

唯一識別 CloudFront 請求 (及其相關回應) 的字串。

連線事件結構

連線函數會收到與檢視器函數不同的事件結構。如需連線事件結構和回應格式的詳細資訊，請參閱 [關聯 CloudFront 連線函數](#)。

檢視者物件

viewer 物件包含一個 ip 欄位，其值是傳送請求的檢視者 (用戶端) 的 IP 位址。如果檢視者的請求來自 HTTP 代理或負載平衡器，此值為代理或負載平衡器的 IP 位址。

請求物件

request 物件包含檢視者到 CloudFront HTTP 請求的表示。在傳遞給函數的 event 物件中，request 物件代表 CloudFront 從檢視者接收的實際請求。

如果您的函數程式碼將 request 物件返回到 CloudFront，其必須使用相同的結構。

request 物件包含下列欄位：

method

請求的 HTTP 方法。如果您的函數程式碼傳回 request，則無法修改此欄位。這是 request 物件中唯一的唯讀欄位。

uri

請求物件的相對路徑。

Note

如果您的函數修改 uri 值，就適用下列事項：

- 全新的 uri 值必須以正斜線 (/) 作為開頭。
- 當函數變更 uri 值時，它會變更檢視者請求的物件。

- 當函數變更 `uri` 值時，它不會變更請求的快取行為或原始伺服器請求傳送的來源。

queryString

代表請求中的查詢字串的物件。如果請求不包含查詢字串，`request` 物件仍會包含空白的 `queryString` 物件。

`queryString` 物件針對請求中的每個查詢字串參數包含一個欄位。

headers

代表請求中 HTTP 標頭的物件。如果請求包含任何 Cookie 標頭，則這些標頭不是 `headers` 物件的一部分。Cookies 在 `cookies` 物件中單獨表示。

`headers` 物件針對請求中的每個標頭包含一個欄位。在事件物件中，標頭名稱會轉換為 ASCII 小寫，而在函數程式碼新增標頭名稱時，標頭名稱必須是 ASCII 小寫。CloudFront Functions 將事件物件轉換回 HTTP 請求時，標頭名稱中每個單字的第一個字母會大寫 (若為 ASCII 字母)。CloudFront Functions 不會將任何變更套用至標頭名稱中的非 ASCII 符號。例如 `Tèst-header` 將在函數內部變為 `tÈst-header`。非 ASCII 符號 `È` 保持不變。

單字以連字號分隔 (-)。例如，如果函數程式碼新增名為 `example-header-name` 的標頭，CloudFront 會將其轉換為 HTTP 請求中的 `Example-Header-Name`。

cookies

代表請求 (Cookie 標頭) 中 Cookie 的物件。

`cookies` 物件針對請求中的每個 Cookie 包含一個欄位。

如需有關查詢字串、標頭和 Cookie 結構的詳細資訊，請參閱 [查詢字串、標頭和 Cookie 的結構](#)。

如需範例 `event` 物件，請參閱 [範例事件物件](#)。

回應物件

`response` 物件包含 CloudFront 到檢視者的 HTTP 回應的表示。在傳遞給函數的 `event` 物件中，`response` 物件代表 CloudFront 對檢視者請求的實際回應。

如果您的函數程式碼返回 `response` 物件，其必須使用相同的結構。

response 物件包含下列欄位：

statusCode

回應的 HTTP 狀態碼。該值是一個整數，而不是字串。

您的函數可以產生或修改 statusCode。

statusDescription

回應的 HTTP 狀態說明。如果您的函數程式碼產生回應，則此欄位為選用。

headers

代表回應中 HTTP 標頭的物件。如果回應包含任何 Set-Cookie 標頭，則這些標頭不是 headers 物件的一部分。Cookies 在 cookies 物件中單獨表示。

headers 物件針對回應中的每個標頭包含一個欄位。在事件物件中，標頭名稱會轉換為小寫，而在函數程式碼新增標頭名稱時，標頭名稱必須是小寫。當 CloudFront Functions 將事件物件轉換回 HTTP 回應時，標頭名稱中每個單字的第一個字母會大寫。單字以連字號分隔 (-)。例如，如果函數程式碼新增名為 example-header-name 的標頭，CloudFront 會將其轉換為 HTTP 回應中的 Example-Header-Name。

cookies

代表回應 (Set-Cookie 標頭) 中 Cookie 的物件。

cookies 物件針對回應中的每個 Cookie 包含一個欄位。

body

新增 body 欄位是選擇性的，除非您在函數中進行指定，否則它不會出現在 response 物件中。您的函數無法存取 CloudFront 快取或原始伺服器傳回的原始本文。如果您未在檢視器回應函數中指定 body 欄位，則 CloudFront 快取或原始伺服器回傳的原始本文將傳回給檢視器。

如果您希望 CloudFront 將自訂本文傳回給檢視器，請在 data 欄位中指定本文內容，並在 encoding 欄位中指定本文編碼。您可以將編碼指定為純文字 ("encoding": "text") 或 Base64 編碼的內容 ("encoding": "base64")。

作為捷徑，您也可以直接在 body 欄位 ("body": "<specify the body content here>") 中指定本文內容。執行此操作時，請省略 data 和 encoding 欄位。在這種情況下，CloudFront 會將本文視為純文字。

encoding

body 內容 (data 欄位) 的編碼。唯一的有效編碼是 text 和 base64。

如果您將 encoding 指定為 base64 但本文不是有效的 base64，CloudFront 會傳回錯誤。

data

body 內容。

如需有關已修改狀態碼和本文內容的更多資訊，請參閱 [狀態碼和本文](#)。

如需有關標頭和 Cookie 結構的詳細資訊，請參閱 [查詢字串、標頭和 Cookie 的結構](#)。

如需範例 response 物件，請參閱 [範例回應物件](#)。

狀態碼和本文

透過 CloudFront Functions，您可以更新檢視者回應本文或將其移除。在評估來自 CloudFront 快取或原始伺服器回應的各個層面後，更新檢視器回應的一些常見案例包括：

- 變更狀態以設定 HTTP 200 狀態碼並建立靜態本文內容，以傳回給檢視器。
- 變更狀態以設定 HTTP 301 或 302 狀態碼來重新導向使用者到另一個網站。
- 決定是否要提供或捨棄檢視者回應的本文。

Note

如果原始伺服器傳回 400 及以上的 HTTP 錯誤，CloudFront Function 將無法執行。如需更多資訊，請參閱[對所有邊緣函數的限制](#)。

在您使用 HTTP 回應時，CloudFront Functions 將無法存取回應本文。您可以透過將本文內容設定為所需的值來進行替換，或設定為空值來移除本文。如果您不更新函數中的本文欄位，CloudFront 快取或原始伺服器回傳的原始本文會傳回給檢視器。

Tip

使用 CloudFront Functions 取代本文時，請務必將對應的標頭 (例如 content-encoding、content-type 或 content-length) 對齊新的本文內容。

例如，如果 CloudFront 原始伺服器或快取傳回 `content-encoding: gzip`，但檢視器回應函數設定了純文字的本文，則該函數也需要相應變更 `content-encoding` 和 `content-type` 標頭。

如果您的 CloudFront Function 設定為傳回 400 或更高的 HTTP 錯誤，您的檢視者將不會看到您為相同狀態碼指定的[自訂錯誤頁面](#)。

查詢字串、標頭和 Cookie 的結構

查詢字串、標頭和 Cookie 共用相同的結構。查詢字串可能會出現在請求中。標頭會出現在請求和回應中。Cookie 會出現在請求和回應中。

每個查詢字串、標頭或 Cookie 都是父系 `querystring`、`headers` 或 `cookies` 物件中的唯一欄位。欄位名稱是查詢字串、標頭或 Cookie 的名稱。每個欄位都包含具有查詢字串、標頭或 Cookie 值的 `value` 屬性。

內容

- [查詢字串值或查詢字串物件](#)
- [標頭的特殊考量](#)
- [重複的查詢字串、標頭和 Cookie \(multiValue 陣列\)](#)
- [Cookie 屬性](#)

查詢字串值或查詢字串物件

除了查詢字串物件之外，函數還可以傳回查詢字串值。查詢字串值可用來依任意自訂順序排列查詢字串參數。

Example 範例

若要在函數程式碼中修改查詢字串，請使用如下所示的程式碼。

```
var request = event.request;
request.querystring =
  'ID=42&Exp=1619740800&TTL=1440&NoValue=&querymv=val1&querymv=val2,val3';
```

標頭的特殊考量

僅針對標頭，標頭名稱會在事件物件中轉換為小寫，而在函數程式碼新增標頭名稱時，標頭名稱則必須是小寫。當 CloudFront Functions 將事件物件轉換回 HTTP 請求或回應時，標頭名稱中每個單字的第

一個字母會大寫。單字以連字號分隔 (-)。例如，如果函數程式碼新增名為 `example-header-name` 的標題，CloudFront 會將其轉換為 HTTP 請求或回應中的 `Example-Header-Name`。

Example 範例

請考慮 HTTP 請求中的下列 Host 標頭。

```
Host: video.example.com
```

此標頭在 `request` 物件中的表示方式如下：

```
"headers": {  
  "host": {  
    "value": "video.example.com"  
  }  
}
```

若要在函數程式碼中存取 Host 標頭，請使用如下所示的程式碼：

```
var request = event.request;  
var host = request.headers.host.value;
```

若要在函數程式碼中新增或修改標頭，請使用如下所示的程式碼 (此程式碼會新增名為 `X-Custom-Header` 且包含值 `example value` 的標頭)：

```
var request = event.request;  
request.headers['x-custom-header'] = {value: 'example value'};
```

重複的查詢字串、標頭和 Cookie (**multiValue** 陣列)

HTTP 請求或回應可以包含多個具有相同名稱的查詢字串、標頭或 Cookie。在此情況下，重複的查詢字串、標頭或 Cookie 會折疊成 `request` 或 `response` 物件中的一個欄位，但此欄位包含一個名為 `multiValue` 的額外屬性。`multiValue` 屬性包含一個陣列，其中帶有每個重複查詢字串、標頭或 Cookie 的值。

Example 範例

請考慮包含下列 `Accept` 標頭的 HTTP 請求。

```
Accept: application/json  
Accept: application/xml
```

```
Accept: text/html
```

這些標頭在 request 物件中的表示方式如下。

```
"headers": {
  "accept": {
    "value": "application/json",
    "multiValue": [
      {
        "value": "application/json"
      },
      {
        "value": "application/xml"
      },
      {
        "value": "text/html"
      }
    ]
  }
}
```

Note

第一個標頭值 (在此情況為 application/json) 會在 value 和 multiValue 屬性中重複。這可讓您透過在 multiValue 陣列中執行迴圈來存取所有值。

如果您的函數程式碼修改具有 multiValue 陣列的查詢字串、標頭或 Cookie，CloudFront Functions 會使用下列規則來套用變更：

1. 如果 multiValue 陣列存在且有任何修改，則套用此修改。value 屬性中的第一個元素被忽略。
2. 否則，會套用 value 屬性的任何修改，並且後續的值 (如果存在) 保持不變。

只有當 HTTP 請求或回應包含具有相同名稱的重複查詢字串、標頭或 Cookie 時，才會使用 multiValue 屬性，如上述範例所示。但是，如果單一查詢字串、標頭或 Cookie 中有多個值，則不會使用該 multiValue 屬性。

Example 範例

考慮具有一個 Accept 標頭的請求，而標頭中包含三個值。

```
Accept: application/json, application/xml, text/html
```

此標頭在 request 物件中的表示方式如下。

```
"headers": {  
  "accept": {  
    "value": "application/json, application/xml, text/html"  
  }  
}
```

Cookie 屬性

在 HTTP 回應的 Set-Cookie 標頭中，標頭包含 Cookie 的名稱/值對，以及選用的一組屬性 (以分號分隔)。

Example 範例

```
Set-Cookie: cookie1=val1; Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021  
07:28:00 GMT
```

在 response 物件中，這些屬性會在 Cookie 欄位的 attributes 屬性中表示。例如，前面的 Set-Cookie 標頭表示如下：

```
"cookie1": {  
  "value": "val1",  
  "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021  
07:28:00 GMT"  
}
```

範例回應物件

以下範例顯示一個本文已被檢視器回應函數替換的 response 物件 (檢視器回應函數的輸出)。

```
{  
  "response": {  
    "statusCode": 200,  
    "statusDescription": "OK",  
    "headers": {  
      "date": {  
        "value": "Mon, 04 Apr 2021 18:57:56 GMT"  
      }  
    }  
  }  
}
```

```

    },
    "server": {
        "value": "gunicorn/19.9.0"
    },
    "access-control-allow-origin": {
        "value": "*"
    },
    "access-control-allow-credentials": {
        "value": "true"
    },
    "content-type": {
        "value": "text/html"
    },
    "content-length": {
        "value": "86"
    }
},
"cookies": {
    "ID": {
        "value": "id1234",
        "attributes": "Expires=Wed, 05 Apr 2021 07:28:00 GMT"
    },
    "Cookie1": {
        "value": "val1",
        "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
07:28:00 GMT",
        "multiValue": [
            {
                "value": "val1",
                "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021
07:28:00 GMT"
            },
            {
                "value": "val2",
                "attributes": "Path=/cat; Domain=example.com; Expires=Wed, 10 Jan 2021
07:28:00 GMT"
            }
        ]
    }
},

// Adding the body field is optional and it will not be present in the response
object
// unless you specify it in your function.

```

```
// Your function does not have access to the original body returned by the
CloudFront
// cache or origin.
// If you don't specify the body field in your viewer response function, the
original
// body returned by the CloudFront cache or origin is returned to viewer.

"body": {
  "encoding": "text",
  "data": "<!DOCTYPE html><html><body><p>Here is your custom content.</p></body></
html>"
}
}
}
```

範例事件物件

以下範例顯示完整的 event 物件。這是標準分佈的範例調用，而不是多租用戶分佈。對於多租用戶分佈，會使用 endpoint 欄位而不是 distributionDomainName。endpoint 的值是與事件相關聯之連線群組的 CloudFront 網域名稱 (例如 d111111abcdef8.cloudfront.net)。

Note

event 物件是函數的輸入。您的函數僅返回 request 或 response 物件，而不是完整的事件物件。

```
{
  "version": "1.0",
  "context": {
    "distributionDomainName": "d111111abcdef8.cloudfront.net",
    "distributionId": "EDFDVBD6EXAMPLE",
    "eventType": "viewer-response",
    "requestId": "EXAMPLEentjQpEXAMPLE_SG5Z-EXAMPLEPmPfEXAMPLEu3EqEXAMPLE=="
  },
  "viewer": {"ip": "198.51.100.11"},
  "request": {
    "method": "GET",
    "uri": "/media/index.mpd",
    "queryString": {
      "ID": {"value": "42"},
      "Exp": {"value": "1619740800"},

```

```

    "TTL": {"value": "1440"},
    "NoValue": {"value": ""},
    "querymv": {
      "value": "val1",
      "multiValue": [
        {"value": "val1"},
        {"value": "val2,val3"}
      ]
    },
    },
    "headers": {
      "host": {"value": "video.example.com"},
      "user-agent": {"value": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0) Gecko/20100101 Firefox/83.0"},
      "accept": {
        "value": "application/json",
        "multiValue": [
          {"value": "application/json"},
          {"value": "application/xml"},
          {"value": "text/html"}
        ]
      },
      "accept-language": {"value": "en-GB,en;q=0.5"},
      "accept-encoding": {"value": "gzip, deflate, br"},
      "origin": {"value": "https://website.example.com"},
      "referer": {"value": "https://website.example.com/videos/12345678?
action=play"},
      "cloudfront-viewer-country": {"value": "GB"}
    },
    "cookies": {
      "Cookie1": {"value": "value1"},
      "Cookie2": {"value": "value2"},
      "cookie_consent": {"value": "true"},
      "cookiemv": {
        "value": "value3",
        "multiValue": [
          {"value": "value3"},
          {"value": "value4"}
        ]
      }
    },
    },
    "response": {
      "statusCode": 200,

```



```

    "statusDescription": "OK",
    "headers": {
      "date": {"value": "Mon, 04 Apr 2021 18:57:56 GMT"},
      "server": {"value": "unicorn/19.9.0"},
      "access-control-allow-origin": {"value": "*"},
      "access-control-allow-credentials": {"value": "true"},
      "content-type": {"value": "application/json"},
      "content-length": {"value": "701"}
    },
    "cookies": {
      "ID": {
        "value": "id1234",
        "attributes": "Expires=Wed, 05 Apr 2021 07:28:00 GMT"
      },
      "Cookie1": {
        "value": "val1",
        "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021 07:28:00 GMT",
        "multiValue": [
          {
            "value": "val1",
            "attributes": "Secure; Path=/; Domain=example.com; Expires=Wed, 05 Apr 2021 07:28:00 GMT"
          },
          {
            "value": "val2",
            "attributes": "Path=/cat; Domain=example.com; Expires=Wed, 10 Jan 2021 07:28:00 GMT"
          }
        ]
      }
    }
  }
}

```

適用於 CloudFront Functions 的 JavaScript 執行時期功能

CloudFront Functions JavaScript 執行時期環境符合 [ECMAScript \(ES\) 版本 5.1](#)，並且支援 ES 版本 6 到 12 的一些功能。

我們建議您使用 JavaScript 執行時期 2.0，以獲得最新功能。

相較於 1.0，JavaScript 執行時期 2.0 功能有下列變更：

- 可使用緩衝區模組方法
- 以下非標準字串原型方法無法使用：
 - `String.prototype.bytesFrom()`
 - `String.prototype.fromBytes()`
 - `String.prototype.fromUTF8()`
 - `String.prototype.toBytes()`
 - `String.prototype.toUTF8()`
- 密碼編譯模組有以下變更：
 - `hash.digest()`：如果未提供編碼，則傳回類型變更為 `Buffer`
 - `hmac.digest()`：如果未提供編碼，則傳回類型變更為 `Buffer`
- 如需其他新功能的詳細資訊，請參閱 [適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)。

主題

- [適用於 CloudFront Functions 的 JavaScript 執行時期 1.0 功能](#)
- [適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)

適用於 CloudFront Functions 的 JavaScript 執行時期 1.0 功能

CloudFront Functions JavaScript 執行時期環境符合 [ECMAScript \(ES\) 版本 5.1](#)，並且支援 ES 版本 6 到 9 的一些功能。它還提供了一些不屬於 ES 規格的非標準方法。

下列主題列出所有支援的語言功能。

主題

- [核心功能](#)
- [基本物件](#)
- [內建物件](#)
- [錯誤類型](#)
- [全域變數](#)
- [內建模組](#)
- [限制功能](#)

核心功能

支援 ES 的以下核心功能。

類型

支援所有 ES 5.1 類型。這包括布林值、數字、字串、物件、陣列、函數、函數建構子和常規表達式。

運算子

支援所有 ES 5.1 運算子。

支援 ES 7 指數運算子 (**)。

聲明

Note

不支援 `const` 和 `let` 陳述式。

支援下列 ES 5.1 陳述式：

- `break`
- `catch`
- `continue`
- `do-while`
- `else`
- `finally`
- `for`
- `for-in`
- `if`
- `return`
- `switch`
- `throw`
- `try`

- `var`
- `while`
- 標記的陳述式

文字

支援 ES 6 範本文字：多行字串、表達式插補和巢狀範本。

函數

支援所有 ES 5.1 函數功能。

支援 ES 6 箭頭函數，支援 ES 6 剩餘參數語法。

Unicode

來源文字和字串常值可以包含 Unicode 編碼的字元。也支援由六個字元 (例如 `\uXXXX`) 組成的 Unicode 字碼指標逸出序列。

嚴格模式

函數按預設會在嚴格模式下運作，因此您不需要在函數程式碼中新增 `use strict` 陳述式。無法對此進行變更。

基本物件

支援 ES 的以下基本物件。

物件

支援物件上的以下 ES 5.1 方法：

- `create`(不含屬性清單)
- `defineProperties`
- `defineProperty`
- `freeze`
- `getOwnPropertyDescriptor`
- `getOwnPropertyNames`
- `getPrototypeOf`
- `hasOwnProperty`

- `isExtensible`
- `isFrozen`
- `prototype.isPrototypeOf`
- `isSealed`
- `keys`
- `preventExtensions`
- `prototype.propertyIsEnumerable`
- `seal`
- `prototype.toString`
- `prototype.valueOf`

支援物件上的以下 ES 6 方法：

- `assign`
- `is`
- `prototype.setPrototypeOf`

支援物件上的以下 ES 8 方法：

- `entries`
- `values`

字串

支援以下針對字串的 ES 5.1 方法：

- `fromCharCode`
- `prototype.charAt`
- `prototype.concat`
- `prototype.indexOf`
- `prototype.lastIndexOf`
- `prototype.match`
- `prototype.replace`
- `prototype.search`
- `prototype.slice`

- `prototype.split`
- `prototype.substr`
- `prototype.substring`
- `prototype.toLowerCase`
- `prototype.trim`
- `prototype.toUpperCase`

支援字串上的以下 ES 6 方法：

- `fromCodePoint`
- `prototype.codePointAt`
- `prototype.endsWith`
- `prototype.includes`
- `prototype.repeat`
- `prototype.startsWith`

支援字串上的以下 ES 8 方法：

- `prototype.padStart`
- `prototype.padEnd`

支援字串上的以下 ES 9 方法：

- `prototype.trimStart`
- `prototype.trimEnd`

支援字串上的以下非標準方法：

- `prototype.bytesFrom(array | string, encoding)`

從八位元陣列或編碼字串建立一個位元組字串。字串編碼選項為 `hex`、`base64` 和 `base64url`。

- `prototype.fromBytes(start[, end])`

從位元組字串建立 Unicode 字串，其中每個位元組都會以對應的 Unicode 字碼指標取代。

- `prototype.fromUTF8(start[, end])`

從 UTF-8 編碼的位元組字串建立一個 Unicode 字串。如果編碼不正確，則返回 `null`。

- `prototype.toBytes(start[, end])`

從 Unicode 字串建立一個位元組字串。所有字元均必須在 [0,255] 範圍內。如果不在此範圍內，則返回 `null`。

- `prototype.toUTF8(start[, end])`

從一個 Unicode 字串建立一個 UTF-8 編碼的位元組字串。

數字

支援數字上的所有 ES 5.1 方法。

支援數字上的以下 ES 6 方法：

- `isFinite`
- `isInteger`
- `isNaN`
- `isSafeInteger`
- `parseFloat`
- `parseInt`
- `prototype.toExponential`
- `prototype.toFixed`
- `prototype.toPrecision`
- `EPSILON`
- `MAX_SAFE_INTEGER`
- `MAX_VALUE`
- `MIN_SAFE_INTEGER`
- `MIN_VALUE`
- `NEGATIVE_INFINITY`
- `NaN`
- `POSITIVE_INFINITY`

內建物件

支援 ES 的以下內建物件。

數學

支援所有 ES 5.1 數學方法。

Note

在 CloudFront Functions 執行時間環境中，`Math.random()` 實作會使用植入函數執行時間戳記的 OpenBSD `arc4random`。

支援以下 ES 6 數學方法：

- `acosh`
- `asinh`
- `atanh`
- `cbrt`
- `clz32`
- `cosh`
- `expm1`
- `fround`
- `hypot`
- `imul`
- `log10`
- `log1p`
- `log2`
- `sign`
- `sinh`
- `tanh`
- `trunc`
- `E`
- `LN10`
- `LN2`
- `LOG10E`

- LOG2E
- PI
- SQRT1_2
- SQRT2

日期

支援所有 ES 5.1 Date 功能。

Note

基於安全考量，在單一函數執行的生命週期內，Date 始終返回相同的值 (函數的開始時間)。如需詳細資訊，請參閱 [限制功能](#)。

函數

支援 apply、bind 和 call 方法。

不支援函數建構子。

常規表達式

支援所有 ES 5.1 常規表達式功能。常規表達式語言與 Perl 相容。支援 ES 9 命名的擷取群組。

JSON

支援所有 ES 5.1 JSON 功能，包括 parse 和 stringify。

Array

支援陣列上的以下 ES 5.1 方法：

- isArray
- prototype.concat
- prototype.every
- prototype.filter
- prototype.forEach
- prototype.indexOf
- prototype.join

- `prototype.lastIndexOf`
- `prototype.map`
- `prototype.pop`
- `prototype.push`
- `prototype.reduce`
- `prototype.reduceRight`
- `prototype.reverse`
- `prototype.shift`
- `prototype.slice`
- `prototype.some`
- `prototype.sort`
- `prototype.splice`
- `prototype.unshift`

支援陣列上的以下 ES 6 方法：

- `of`
- `prototype.copyWithIn`
- `prototype.fill`
- `prototype.find`
- `prototype.findIndex`

支援陣列上的以下 ES 7 方法：

- `prototype.includes`

類型陣列

支援以下 ES 6 類型陣列：

- `Int8Array`
- `Uint8Array`
- `Uint8ClampedArray`
- `Int16Array`
- `Uint16Array`

- `Int32Array`
- `Uint32Array`
- `Float32Array`
- `Float64Array`
- `prototype.copyWithIn`
- `prototype.fill`
- `prototype.join`
- `prototype.set`
- `prototype.slice`
- `prototype.subarray`
- `prototype.toString`

陣列緩衝區

支援 `ArrayBuffer` 上的以下方法：

- `prototype.isView`
- `prototype.slice`

Promise

支援 `Promise` 上的以下方法：

- `reject`
- `resolve`
- `prototype.catch`
- `prototype.finally`
- `prototype.then`

加密

密碼編譯模組提供標準雜湊和雜湊型訊息身分驗證碼 (HMAC) 協助程式。您可以使用 `require('crypto')` 加載模組。此模組會公開下列方法，其行為與 Node.js 對應方法完全相同：

- `createHash(algorithm)`
- `hash.update(data)`
- `hash.digest([encoding])`

- `createHmac(algorithm, secret key)`
- `hmac.update(data)`
- `hmac.digest([encoding])`

如需詳細資訊，請參閱內建模組一節中的 [加密 \(雜湊和 HMAC\)](#)。

主控台

這是一個用於偵錯的協助程式物件。它僅支援 `log()` 方法以記錄日誌訊息。

Note

CloudFront Functions 不支援逗號語法，例如 `console.log('a', 'b')`。因此請改為使用 `console.log('a' + ' ' + 'b')` 格式。

錯誤類型

支援以下錯誤物件：

- `Error`
- `EvalError`
- `InternalError`
- `MemoryError`
- `RangeError`
- `ReferenceError`
- `SyntaxError`
- `TypeError`
- `URIError`

全域變數

支援 `globalThis` 物件。

支援以下 ES 5.1 全局函數：

- `decodeURI`

- `decodeURIComponent`
- `encodeURI`
- `encodeURIComponent`
- `isFinite`
- `isNaN`
- `parseFloat`
- `parseInt`

支援以下全局常數：

- `NaN`
- `Infinity`
- `undefined`

內建模組

支援以下內建模組：

模組

- [加密 \(雜湊和 HMAC\)](#)
- [查詢字串](#)

加密 (雜湊和 HMAC)

密碼編譯模組 (`crypto`) 提供標準雜湊和雜湊型訊息身分驗證碼 (HMAC) 協助程式。您可以使用 `require('crypto')` 加載模組。此模組提供下列方法，其行為與 Node.js 對應方法完全相同：

雜湊方法

`crypto.createHash(algorithm)`

建立並傳回雜湊物件，藉助此物件，您可以使用給定的演算法產生雜湊摘要：`md5`、`sha1` 或 `sha256`。

`hash.update(data)`

使用給定的 `data` 更新雜湊內容。

```
hash.digest([encoding])
```

計算使用 `hash.update()` 傳遞的所有資料的摘要。編碼可以是 `hex`、`base64` 或 `base64url`。

HMAC 方法

```
crypto.createHmac(algorithm, secret key)
```

建立並返回使用給定 `algorithm` 和 `secret key` 的 HMAC 物件。演算法可以是 `md5`、`sha1` 或 `sha256`。

```
hmac.update(data)
```

使用給定的 `data` 更新 HMAC 內容。

```
hmac.digest([encoding])
```

計算使用 `hmac.update()` 傳遞的所有資料的摘要。編碼可以是 `hex`、`base64` 或 `base64url`。

查詢字串

Note

[CloudFront Functions 事件物件](#) 會自動剖析 URL 查詢字串。這意味著在大多數情況下，您不需要使用此模組。

查詢字串模組 (`querystring`) 提供剖析和格式化 URL 查詢字串的方法。您可以使用 `require('querystring')` 加載模組。此模組提供下列方法。

```
querystring.escape(string)
```

URL 編碼給定的 `string`，傳回逸出的查詢字串。該方法由 `querystring.stringify()` 使用，並且不應直接使用。

```
querystring.parse(string[, separator[, equal[, options]])]
```

剖析查詢字串 (`string`) 並傳回物件。

`separator` 參數是用來分隔查詢字串中的鍵/值對的子字串。其在預設情況下為 `&`。

`equal` 參數是用來分隔查詢字串中的鍵和值的子字串。其在預設情況下為 `=`。

`options` 參數是具有下列鍵的物件：

`decodeURIComponent` *function*

解碼查詢字串中百分比編碼字元的函數。其在預設情況下為 `querystring.unescape()`。

`maxKeys` *number*

要剖析的鍵的最大數量。其在預設情況下為 1000。使用 0 的值移除計數鍵的限制。

根據預設，查詢字串中的百分比編碼字元會假設為使用 UTF-8 編碼。無效的 UTF-8 序列會被取代為 U+FFFD 取代字元。

例如，對於下列查詢字串：

```
'name=value&abc=xyz&abc=123'
```

`querystring.parse()` 的返回值是：

```
{
  name: 'value',
  abc: ['xyz', '123']
}
```

`querystring.decode()` 是 `querystring.parse()` 的別名。

`querystring.stringify(object[, separator[, equal[, options]])`

序列化 `object` 並傳回查詢字串。

`separator` 參數是用來分隔查詢字串中的鍵/值對的子字串。其在預設情況下為 `&`。

`equal` 參數是用來分隔查詢字串中的鍵和值的子字串。其在預設情況下為 `=`。

`options` 參數是具有下列鍵的物件：

`encodeURIComponent` *function*

用於將 URL 不安全字元轉換為查詢字串中的百分比編碼的函數。其在預設情況下為 `querystring.escape()`。

根據預設，在查詢字串中需要百分比編碼的字元會編碼為 UTF-8。若要使用不同的編碼，請指定 `encodeURIComponent` 選項。

例如，對於以下程式碼：

```
querystring.stringify({ name: 'value', abc: ['xyz', '123'], anotherName: '' });
```

返回值是：

```
'name=value&abc=xyz&abc=123&anotherName='
```

`querystring.encode()` 是 `querystring.stringify()` 的別名。

`querystring.unescape(string)`

解碼給定 `string` 中的 URL 百分比編碼字元，傳回未逸出的查詢字串。此方法由 `querystring.parse()` 使用，並且不應直接使用。

限制功能

由於安全考量，下列 JavaScript 語言功能不受支援或受到限制。

動態程式碼評估

不支援動態程式碼評估。如果嘗試此評估，`eval()` 和 `Function` 建構子都會丟出錯誤。例如，`const sum = new Function('a', 'b', 'return a + b')` 丟出錯誤。

計時器

不支援 `setTimeout()`、`setImmediate()` 和 `clearTimeout()` 函數。在函數執行時期間未推遲或產生任何佈建。您的函數必須同步執行方可完成。

日期和時間戳記

基於安全考量，無法存取高解析度計時器。查詢當前時間的所有 `Date` 方法始終在單個函數執行的生命週期內返回相同的值。返回的時間戳記是函數開始執行的時間。因此，您無法測量函數中的經過時間。

檔案系統存取

沒有檔案系統存取權。例如，沒有類似 Node.js 中的檔案系統存取 `fs` 模組。

處理存取

沒有處理存取。例如沒有 `process` 全域物件可像 Node.js 一樣處理資訊存取。

環境變數

無法存取環境變數。

您可以改為使用 CloudFront KeyValueCollection 為您的 CloudFront Functions 建立鍵值對的集中式資料儲存。CloudFront KeyValueCollection 可動態更新組態資料，不需要部署程式碼變更。您必須使用 [JavaScript 執行時期 2.0](#)，才能使用 CloudFront KeyValueCollection。如需更多詳細資訊，請參閱 [Amazon CloudFront KeyValueCollection](#)。

網路存取

不支援網路呼叫。例如，不支援 XHR、HTTP(S) 和通訊端。

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

CloudFront Functions JavaScript 執行時期環境符合 [ECMAScript \(ES\) 版本 5.1](#)，並且支援 ES 版本 6 到 12 的一些功能。它還提供了一些不屬於 ES 規格的非標準方法。以下主題列出此執行時期支援的所有功能。

主題

- [核心功能](#)
- [基本物件](#)
- [內建物件](#)
- [錯誤類型](#)
- [全域變數](#)
- [內建模組](#)
- [限制功能](#)

核心功能

支援 ES 的以下核心功能。

類型

支援所有 ES 5.1 類型。這包括布林值、數字、字串、物件、陣列、函數和常規表達式。

運算子

支援所有 ES 5.1 運算子。

支援 ES 7 指數運算子 (**)。

陳述式

支援下列 ES 5.1 陳述式：

- break
- catch
- continue
- do-while
- else
- finally
- for
- for-in
- if
- label
- return
- switch
- throw
- try
- var
- while

支援以下 ES 6 陳述式：

- const
- let

支援以下 ES 8 陳述式：

- async
- await

Note

async、await、const 和 let 是 JavaScript 執行時期 2.0 支援的功能。

`await` 只能在 `async` 函數內使用，不支援 `async` 引數和關閉。

文字

支援 ES 6 範本文字：多行字串、表達式插補和巢狀範本。

函數

支援所有 ES 5.1 函數功能。

支援 ES 6 箭頭函數，支援 ES 6 剩餘參數語法。

Unicode

來源文字和字串常值可以包含 Unicode 編碼的字元。也支援由六個字元 (例如 `\uXXXX`) 組成的 Unicode 字碼指標逸出序列。

嚴格模式

函數按預設會在嚴格模式下運作，因此您不需要在函數程式碼中新增 `use strict` 陳述式。無法對此進行變更。

基本物件

支援 ES 的以下基本物件。

物件

支援物件上的以下 ES 5.1 方法：

- `Object.create()` (不含屬性清單)
- `Object.defineProperties()`
- `Object.defineProperty()`
- `Object.freeze()`
- `Object.getOwnPropertyDescriptor()`
- `Object.getOwnPropertyDescriptors()`
- `Object.getOwnPropertyNames()`
- `Object.getPrototypeOf()`

- `Object.isExtensible()`
- `Object.isFrozen()`
- `Object.isSealed()`
- `Object.keys()`
- `Object.preventExtensions()`
- `Object.seal()`

支援物件上的以下 ES 6 方法：

- `Object.assign()`

支援物件上的以下 ES 8 方法：

- `Object.entries()`
- `Object.values()`

支援以下針對物件的 ES 5.1 原型方法：

- `Object.prototype.hasOwnProperty()`
- `Object.prototype.isPrototypeOf()`
- `Object.prototype.propertyIsEnumerable()`
- `Object.prototype.toString()`
- `Object.prototype.valueOf()`

支援以下針對物件的 ES 6 原型方法：

- `Object.prototype.is()`
- `Object.prototype.setPrototypeOf()`

String

支援以下針對字串的 ES 5.1 方法：

- `String.fromCharCode()`

支援以下針對字串的 ES 6 方法：

- `String.fromCodePoint()`

支援以下針對字串的 ES 5.1 原型方法：

- `String.prototype.charAt()`
- `String.prototype.concat()`
- `String.prototype.indexOf()`
- `String.prototype.lastIndexOf()`
- `String.prototype.match()`
- `String.prototype.replace()`
- `String.prototype.search()`
- `String.prototype.slice()`
- `String.prototype.split()`
- `String.prototype.substr()`
- `String.prototype.substring()`
- `String.prototype.toLowerCase()`
- `String.prototype.trim()`
- `String.prototype.toUpperCase()`

支援以下針對字串的 ES 6 原型方法：

- `String.prototype.codePointAt()`
- `String.prototype.endsWith()`
- `String.prototype.includes()`
- `String.prototype.repeat()`
- `String.prototype.startsWith()`

支援以下針對字串的 ES 8 原型方法：

- `String.prototype.padStart()`
- `String.prototype.padEnd()`

支援以下針對字串的 ES 9 原型方法：

- `String.prototype.trimStart()`
- `String.prototype.trimEnd()`

支援以下針對字串的 ES 12 原型方法：

- `String.prototype.replaceAll()`

 Note

`String.prototype.replaceAll()` 是 JavaScript 執行時間 2.0 的新功能。

數字

支援所有 ES 5 數字。

支援以下 ES 6 數字屬性：

- `Number.EPSILON`
- `Number.MAX_SAFE_INTEGER`
- `Number.MIN_SAFE_INTEGER`
- `Number.MAX_VALUE`
- `Number.MIN_VALUE`
- `Number.NaN`
- `Number.NEGATIVE_INFINITY`
- `Number.POSITIVE_INFINITY`

支援數字上的以下 ES 6 方法：

- `Number.isFinite()`
- `Number.isInteger()`
- `Number.isNaN()`
- `Number.isSafeInteger()`
- `Number.parseInt()`
- `Number.parseFloat()`

支援以下針對數字的 ES 5.1 原型方法：

- `Number.prototype.toExponential()`
- `Number.prototype.toFixed()`
- `Number.prototype.toPrecision()`

支援 ES 12 數字分隔符號。

 Note

ES 12 數字分隔符號是 JavaScript 執行時間 2.0 的新功能。

內建物件

支援 ES 的以下內建物件。

數學

支援所有 ES 5.1 數學方法。

 Note

在 CloudFront Functions 執行時間環境中，`Math.random()` 實作會使用植入函數執行時間戳記的 OpenBSD `arc4random`。

支援以下 ES 6 數學屬性：

- `Math.E`
- `Math.LN10`
- `Math.LN2`
- `Math.LOG10E`
- `Math.LOG2E`
- `Math.PI`
- `Math.SQRT1_2`
- `Math.SQRT2`

支援以下 ES 6 數學方法：

- `Math.abs()`
- `Math.acos()`
- `Math.acosh()`
- `Math.asin()`
- `Math.asinh()`

- `Math.atan()`
- `Math.atan2()`
- `Math.atanh()`
- `Math.cbrt()`
- `Math.ceil()`
- `Math.clz32()`
- `Math.cos()`
- `Math.cosh()`
- `Math.exp()`
- `Math.expm1()`
- `Math.floor()`
- `Math.fround()`
- `Math.hypot()`
- `Math.imul()`
- `Math.log()`
- `Math.log1p()`
- `Math.log2()`
- `Math.log10()`
- `Math.max()`
- `Math.min()`
- `Math.pow()`
- `Math.random()`
- `Math.round()`
- `Math.sign()`
- `Math.sinh()`
- `Math.sin()`
- `Math.sqrt()`
- `Math.tan()`
- `Math.tanh()`
- `Math.trunc()`

日期

支援所有 ES 5.1 Date 功能。

Note

基於安全考量，在單一函數執行的生命週期內，Date 始終返回相同的值 (函數的開始時間)。如需詳細資訊，請參閱 [限制功能](#)。

函數

支援以下 ES 5.1 原型方法：

- `Function.prototype.apply()`
- `Function.prototype.bind()`
- `Function.prototype.call()`

不支援函數建構子。

常規表達式

支援所有 ES 5.1 常規表達式功能。常規表達式語言與 Perl 相容。

支援以下 ES 5.1 原型存取子屬性：

- `RegExp.prototype.global`
- `RegExp.prototype.ignoreCase`
- `RegExp.prototype.multiline`
- `RegExp.prototype.source`
- `RegExp.prototype.sticky`
- `RegExp.prototype.flags`

Note

`RegExp.prototype.sticky` 和 `RegExp.prototype.flags` 是 JavaScript 執行時間 2.0 的新功能。

支援以下 ES 5.1 原型方法：

- `RegExp.prototype.exec()`
- `RegExp.prototype.test()`
- `RegExp.prototype.toString()`
- `RegExp.prototype[@@replace]()`
- `RegExp.prototype[@@split]()`

 Note

`RegExp.prototype[@@split]()` 是 JavaScript 執行時間 2.0 的新功能。

支援以下 ES 5.1 執行個體屬性：

- `lastIndex`

支援 ES 9 命名的擷取群組。

JSON

支援以下 ES 5.1 方法：

- `JSON.parse()`
- `JSON.stringify()`

Array

支援陣列上的以下 ES 5.1 方法：

- `Array.isArray()`

支援陣列上的以下 ES 6 方法：

- `Array.of()`

支援以下 ES 5.1 原型方法：

- `Array.prototype.concat()`
- `Array.prototype.every()`
- `Array.prototype.filter()`
- `Array.prototype.forEach()`
- `Array.prototype.indexOf()`
- `Array.prototype.join()`

- `Array.prototype.lastIndexOf()`
- `Array.prototype.map()`
- `Array.prototype.pop()`
- `Array.prototype.push()`
- `Array.prototype.reduce()`
- `Array.prototype.reduceRight()`
- `Array.prototype.reverse()`
- `Array.prototype.shift()`
- `Array.prototype.slice()`
- `Array.prototype.some()`
- `Array.prototype.sort()`
- `Array.prototype.splice()`
- `Array.prototype.unshift()`

支援以下 ES 6 原型方法

- `Array.prototype.copyWithIn()`
- `Array.prototype.fill()`
- `Array.prototype.find()`
- `Array.prototype.findIndex()`

支援以下 ES 7 原型方法：

- `Array.prototype.includes()`

類型陣列

支援以下 ES 6 類型陣列建構子：

- `Float32Array`
- `Float64Array`
- `Int8Array`
- `Int16Array`
- `Int32Array`
- `Uint8Array`
- `Uint8ClampedArray`

- Uint16Array
- Uint32Array

支援以下 ES 6 方法：

- `TypedArray.from()`
- `TypedArray.of()`

 Note

`TypedArray.from()` 和 `TypedArray.of()` 是 JavaScript 執行時間 2.0 的新功能。

支援以下 ES 6 原型方法：

- `TypedArray.prototype.copyWithin()`
- `TypedArray.prototype.every()`
- `TypedArray.prototype.fill()`
- `TypedArray.prototype.filter()`
- `TypedArray.prototype.find()`
- `TypedArray.prototype.findIndex()`
- `TypedArray.prototype.forEach()`
- `TypedArray.prototype.includes()`
- `TypedArray.prototype.indexOf()`
- `TypedArray.prototype.join()`
- `TypedArray.prototype.lastIndexOf()`
- `TypedArray.prototype.map()`
- `TypedArray.prototype.reduce()`
- `TypedArray.prototype.reduceRight()`
- `TypedArray.prototype.reverse()`
- `TypedArray.prototype.some()`
- `TypedArray.prototype.set()`
- `TypedArray.prototype.slice()`
- `TypedArray.prototype.sort()`
- `TypedArray.prototype.subarray()`

- `TypedArray.prototype.toString()`

 Note

`TypedArray.prototype.every()`、`TypedArray.prototype.fill()`、`TypedArray.prototype.forEach()` 和 `TypedArray.prototype.some()` 是 JavaScript 執行時間 2.0 的新功能。

ArrayBuffer

支援以下針對 `ArrayBuffer` 的 ES 6 方法：

- `isView()`

支援以下針對 `ArrayBuffer` 的 ES 6 原型方法：

- `ArrayBuffer.prototype.slice()`

Promise

支援以下針對 `Promise` 的 ES 6 方法：

- `Promise.all()`
- `Promise.allSettled()`
- `Promise.any()`
- `Promise.reject()`
- `Promise.resolve()`
- `Promise.race()`

 Note

`Promise.all()`、`Promise.allSettled()`、`Promise.any()` 和 `Promise.race()` 是 JavaScript 執行時間 2.0 的新功能。

支援以下針對 `Promise` 的 ES 6 原型方法：

- `Promise.prototype.catch()`
- `Promise.prototype.finally()`
- `Promise.prototype.then()`

DataView

支援以下 ES 6 原型方法：

- `DataView.prototype.getFloat32()`
- `DataView.prototype.getFloat64()`
- `DataView.prototype.getInt16()`
- `DataView.prototype.getInt32()`
- `DataView.prototype.getInt8()`
- `DataView.prototype.getUint16()`
- `DataView.prototype.getUint32()`
- `DataView.prototype.getUint8()`
- `DataView.prototype.setFloat32()`
- `DataView.prototype.setFloat64()`
- `DataView.prototype.setInt16()`
- `DataView.prototype.setInt32()`
- `DataView.prototype.setInt8()`
- `DataView.prototype.setUint16()`
- `DataView.prototype.setUint32()`
- `DataView.prototype.setUint8()`

 Note

所有 DataView ES 6 原型方法都是 JavaScript 執行時間 2.0 的新功能。

符號

支援以下 ES 6 方法：

- `Symbol.for()`
- `Symbol.keyfor()`

 Note

所有符號 ES 6 方法都是 JavaScript 執行時間 2.0 的新功能。

文字解碼器

支援以下原型方法：

- `TextDecoder.prototype.decode()`

支援以下原型存取子屬性：

- `TextDecoder.prototype.encoding`
- `TextDecoder.prototype.fatal`
- `TextDecoder.prototype.ignoreBOM`

文字編碼器

支援以下原型方法：

- `TextEncoder.prototype.encode()`
- `TextEncoder.prototype.encodeInto()`

錯誤類型

支援以下錯誤物件：

- `Error`
- `EvalError`
- `InternalError`
- `RangeError`
- `ReferenceError`
- `SyntaxError`
- `TypeError`
- `URIError`

全域變數

支援 `globalThis` 物件。

支援以下 ES 5.1 全局函數：

- `decodeURI()`
- `decodeURIComponent()`
- `encodeURI()`
- `encodeURIComponent()`

- `isFinite()`
- `isNaN()`
- `parseFloat()`
- `parseInt()`

支援以下 ES 6 全域函數：

- `atob()`
- `btoa()`

 Note

`atob()` 和 `btoa()` 是 JavaScript 執行時間 2.0 的新功能。

支援以下全局常數：

- `NaN`
- `Infinity`
- `undefined`
- `arguments`

內建模組

支援以下內建模組：

模組

- [緩衝區](#)
- [查詢字串](#)
- [加密](#)

緩衝區

此模組提供以下方法：

- `Buffer.alloc(size[, fill[, encoding]])`

配置 Buffer。

- `size`：緩衝區大小。輸入整數。
 - `fill`：選用。輸入字串、Buffer、Uint8Array 或整數。預設值為 0。
 - `encoding`：選用。當 `fill` 為字串，請輸入以下其中一項：utf8、hex、base64、base64url。預設值為 utf8。
- `Buffer.allocUnsafe(size)`

配置一個未初始化的 Buffer。

- `size`：輸入整數。
- `Buffer.byteLength(value[, encoding])`

返回值的長度，以位元組為單位。

- `value`：字串、Buffer、TypedArray、DataView 或 ArrayBuffer。
 - `encoding`：選用。當 `value` 為字串，請輸入以下其中一項：utf8、hex、base64、base64url。預設值為 utf8。
- `Buffer.compare(buffer1, buffer2)`

比較兩個 Buffer 以協助對陣列進行排序。如果兩者相同則傳回 0，如果 `buffer1` 在前面則傳回 -1，或如果 `buffer2` 在前面則傳回 1。

- `buffer1`：輸入 Buffer。
 - `buffer2`：輸入不同的 Buffer。
- `Buffer.concat(list[, totalLength])`

連接多個 Buffer。如果沒有則傳回 0。最多傳回 `totalLength`。

- `list`：輸入 Buffer 的清單。請注意，這將被截斷為 `totalLength`。
 - `totalLength`：選用。輸入不帶正負號的整數。如果清單為空白，則使用清單中 Buffer 執行個體的總和。
- `Buffer.from(array)`

從陣列建立 Buffer。

- `array`：輸入從 0 到 255 的位元組陣列。
- `Buffer.from(arrayBuffer, byteOffset[, length])`

從 `arrayBuffer` 建立檢視，從偏移值 `byteOffset` 開始，長度為 `length`。

- `arrayBuffer` : 輸入 Buffer 陣列。
- `byteOffset` : 輸入整數。
- `length` : 選用。輸入整數。
- `Buffer.from(buffer)`

建立 Buffer 的複本。

- `buffer` : 輸入 Buffer。
- `Buffer.from(object[, offsetOrEncoding[, length]])`

從物件建立 Buffer。如果 `valueOf()` 不等於物件，則傳回 `Buffer.from(object.valueOf(), offsetOrEncoding, length)`。

- `object` : 輸入物件。
- `offsetOrEncoding` : 選用。輸入整數或編碼字串。
- `length` : 選用。輸入整數。
- `Buffer.from(string[, encoding])`

從字串建立一個 Buffer。

- `string` : 輸入字串。
- `encoding` : 選用。輸入以下其中之一：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。
- `Buffer.isBuffer(object)`

檢查 `object` 是否為緩衝區。傳回 `true` 或 `false`。

- `object` : 輸入物件。
- `Buffer.isEncoding(encoding)`

檢查是否支援 `encoding`。傳回 `true` 或 `false`。

- `encoding` : 選用。輸入以下其中之一：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。

此模組提供以下緩衝區原型方法：

- `Buffer.prototype.compare(target[, targetStart[, targetEnd[, sourceStart[, sourceEnd]]]])`

將 Buffer 與目標比較。如果兩者相同則傳回 0，如果 `buffer` 在前面則傳回 1，或如果 `target` 在前面則傳回 -1。

- `target` : 輸入 Buffer。
- `targetStart` : 選用。輸入整數。預設值為 0。
- `targetEnd` : 選用。輸入整數。預設值為 `target` 長度。
- `sourceStart` : 選用。輸入整數。預設值為 0。
- `sourceEnd` : 選用。輸入整數。預設值為 Buffer 長度。
- `Buffer.prototype.copy(target[, targetStart[, sourceStart[, sourceEnd]])`

將緩衝區複製到 `target`。

- `target` : 輸入 Buffer 或 `Uint8Array`。
- `targetStart` : 選用。輸入整數。預設值為 0。
- `sourceStart` : 選用。輸入整數。預設值為 0。
- `sourceEnd` : 選用。輸入整數。預設值為 Buffer 長度。
- `Buffer.prototype.equals(otherBuffer)`

將 Buffer 與 `otherBuffer` 比較。傳回 `true` 或 `false`。

- `otherBuffer` : 輸入字串。
- `Buffer.prototype.fill(value[, offset[, end][, encoding])`

以 `value` 填入 Buffer。

- `value` : 輸入字串、Buffer 或整數。
- `offset` : 選用。輸入整數。
- `end` : 選用。輸入整數。
- `encoding` : 選用。輸入以下其中之一：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。
- `Buffer.prototype.includes(value[, byteOffset][, encoding])`

搜尋 Buffer 中的 `value`。傳回 `true` 或 `false`。

- `value` : 輸入字串、Buffer、`Uint8Array`、或整數。
- `byteOffset` : 選用。輸入整數。
- `encoding` : 選用。輸入以下其中之一：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。
- `Buffer.prototype.indexOf(value[, byteOffset][, encoding])`

首先搜尋 Buffer 中的第一個 `value`。如果找到了，則傳回 `index`；如果找不到，則傳回 `-1`。

撰寫函數程式碼

- `value`: 輸入字串、Buffer、`Unit8Array` 或 0 到 255 之間的整數。

- `byteOffset` : 選用。輸入整數。
- `encoding` : 選用。如果 `value` 是字串，請輸入以下其中一項：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。
- `Buffer.prototype.lastIndexOf(value[, byteOffset][, encoding])`

搜尋 `Buffer` 中的最後一個 `value`。如果找到了，則傳回 `index`；如果找不到，則傳回 `-1`。

- `value`: 輸入字串、`Buffer`、`Unit8Array` 或 0 到 255 之間的整數。
- `byteOffset` : 選用。輸入整數。
- `encoding` : 選用。如果 `value` 是字串，請輸入以下其中一項：`utf8`、`hex`、`base64`、`base64url`。預設值為 `utf8`。
- `Buffer.prototype.readInt8(offset)`

從 `Buffer` 的偏移值 `offset` 讀取 `Int8`。

- `offset` : 輸入整數。
- `Buffer.prototype.readIntBE(offset, byteLength)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `Int`。

- `offset` : 輸入整數。
- `byteLength` : 選用。輸入從 1 到 6 的整數。
- `Buffer.prototype.readInt16BE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `Int16`。

- `offset` : 輸入整數。
- `Buffer.prototype.readInt32BE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `Int32`。

- `offset` : 輸入整數。
- `Buffer.prototype.readIntLE(offset, byteLength)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `Int`。

- `offset` : 輸入整數。
- `byteLength` : 輸入從 1 到 6 的整數。
- `Buffer.prototype.readInt16LE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `Int16`。

- `offset` : 輸入整數。

- `Buffer.prototype.readInt32LE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `Int32`。

- `offset` : 輸入整數。

- `Buffer.prototype.readUInt8(offset)`

從 `Buffer` 的偏移值 `offset` 讀取 `UInt8`。

- `offset` : 輸入整數。

- `Buffer.prototype.readUIntBE(offset, byteLength)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `UInt`。

- `offset` : 輸入整數。

- `byteLength` : 輸入從 1 到 6 的整數。

- `Buffer.prototype.readUInt16BE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `UInt16`。

- `offset` : 輸入整數。

- `Buffer.prototype.readUInt32BE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取大端序 `UInt32`。

- `offset` : 輸入整數。

- `Buffer.prototype.readUIntLE(offset, byteLength)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `UInt`。

- `offset` : 輸入整數。

- `byteLength` : 輸入從 1 到 6 的整數。

- `Buffer.prototype.readUInt16LE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `UInt16`。

- `offset` : 輸入整數。

- `Buffer.prototype.readUInt32LE(offset)`

從 `Buffer` 的偏移值 `offset` 讀取小端序 `UInt32`。

- `offset` : 輸入整數。

- `Buffer.prototype.readDoubleBE([offset])`

從 Buffer 的偏移值 `offset` 讀取 64 位元大端序雙精度浮點數。

- `offset` : 選用。輸入整數。

- `Buffer.prototype.readDoubleLE([offset])`

從 Buffer 的偏移值 `offset` 讀取 64 位元小端序雙精度浮點數。

- `offset` : 選用。輸入整數。

- `Buffer.prototype.readFloatBE([offset])`

從 Buffer 的偏移值 `offset` 讀取 32 位元大端序浮點數。

- `offset` : 選用。輸入整數。

- `Buffer.prototype.readFloatLE([offset])`

從 Buffer 的偏移值 `offset` 讀取 32 位元小端序浮點數。

- `offset` : 選用。輸入整數。

- `Buffer.prototype.subarray([start[, end]])`

傳回 Buffer 的副本，並使用新的 `start` 和 `end` 偏移和裁剪。

- `start` : 選用。輸入整數。預設值為 0。
- `end` : 選用。輸入整數。預設值為緩衝區長度。

- `Buffer.prototype.swap16()`

交換 Buffer 陣列的位元組順序，將其視為 16 位元數字的陣列。Buffer 的長度必須是 2 的倍數，否則您將收到錯誤訊息。

- `Buffer.prototype.swap32()`

交換 Buffer 陣列的位元組順序，將其視為 32 位元數字的陣列。Buffer 的長度必須是 4 的倍數，否則您將收到錯誤訊息。

- `Buffer.prototype.swap64()`

交換 Buffer 陣列的位元組順序，將其視為 64 位元數字的陣列。Buffer 的長度必須是 8 的倍數，否則您將收到錯誤訊息。

- `Buffer.prototype.toJSON()`

以 JSON 格式傳回 Buffer。

- `Buffer.prototype.toString([encoding[, start[, end]]])`

將 Buffer 從 start 到 end 轉換為編碼字串。

- `encoding` : 選用。輸入以下其中之一 : `utf8`、`hex`、`base64` 或 `base64url`。預設值為 `utf8`。
 - `start` : 選用。輸入整數。預設值為 0。
 - `end` : 選用。輸入整數。預設值為緩衝區長度。
- `Buffer.prototype.write(string[, offset[, length]][, encoding])`

如果空間足夠，則將編碼 string 寫入 Buffer，如果空間不足，則寫入被截斷的 string。

- `string` : 輸入字串。
 - `offset` : 選用。輸入整數。預設值為 0。
 - `length` : 選用。輸入整數。預設值是字串的長度。
 - `encoding` : 選用。選擇性地輸入以下其中一項 : `utf8`、`hex`、`base64` 或 `base64url`。預設值為 `utf8`。
- `Buffer.prototype.writeInt8(value, offset, byteLength)`

將 Int8 value (長度為 byteLength) 寫入 Buffer 的偏移值 offset。

- `value` : 輸入整數。
 - `offset` : 輸入整數
 - `byteLength` : 輸入從 1 到 6 的整數。
- `Buffer.prototype.writeIntBE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- `value` : 輸入整數。
 - `offset` : 輸入整數
 - `byteLength` : 輸入從 1 到 6 的整數。
- `Buffer.prototype.writeInt16BE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- `value` : 輸入整數。
 - `offset` : 輸入整數
 - `byteLength` : 輸入從 1 到 6 的整數。
- `Buffer.prototype.writeInt32BE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- value：輸入整數。
- offset：輸入整數
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeIntLE(offset, byteLength)

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- offset：輸入整數。
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeInt16LE(offset, byteLength)

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- offset：輸入整數。
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeInt32LE(offset, byteLength)

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- offset：輸入整數。
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeUInt8(value, offset, byteLength)

將 UInt8 value (長度為 byteLength) 寫入 Buffer 的偏移值 offset。

- value：輸入整數。
- offset：輸入整數
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeUIntBE(value, offset, byteLength)

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- value：輸入整數。
- offset：輸入整數
- byteLength：輸入從 1 到 6 的整數。
- Buffer.prototype.writeUInt16BE(value, offset, byteLength)

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- value：輸入整數。
 - offset：輸入整數
 - byteLength：輸入從 1 到 6 的整數。
- `Buffer.prototype.writeUInt32BE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- value：輸入整數。
 - offset：輸入整數
 - byteLength：輸入從 1 到 6 的整數。
- `Buffer.prototype.writeUIntLE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- value：輸入整數。
 - offset：輸入整數
 - byteLength：輸入從 1 到 6 的整數。
- `Buffer.prototype.writeUInt16LE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- value：輸入整數。
 - offset：輸入整數
 - byteLength：輸入從 1 到 6 的整數。
- `Buffer.prototype.writeUInt32LE(value, offset, byteLength)`

將 value 寫入 Buffer 的偏移值 offset，使用小端序。

- value：輸入整數。
 - offset：輸入整數
 - byteLength：輸入從 1 到 6 的整數。
- `Buffer.prototype.writeDoubleBE(value, [offset])`

將 value 寫入 Buffer 的偏移值 offset，使用大端序。

- value：輸入整數。
- offset：選用。輸入整數。預設值為 0。

- `Buffer.prototype.writeDoubleLE(value, [offset])`

將 `value` 寫入 `Buffer` 的偏移值 `offset`，使用小端序。

- `value`：輸入整數。
- `offset`：選用。輸入整數。預設值為 0。

- `Buffer.prototype.writeFloatBE(value, [offset])`

將 `value` 寫入 `Buffer` 的偏移值 `offset`，使用大端序。

- `value`：輸入整數。
- `offset`：選用。輸入整數。預設值為 0。

- `Buffer.prototype.writeFloatLE(value, [offset])`

將 `value` 寫入 `Buffer` 的偏移值 `offset`，使用小端序。

- `value`：輸入整數。
- `offset`：選用。輸入整數。預設值為 0。

支援以下執行個體方法：

- `buffer[index]`

取得和設定 `Buffer` 中偏移值 `index` 的八位元組 (位元組)。

- 取得從 0 到 255 的數字。或設定一個從 0 到 255 的數字。

支援以下執行個體屬性：

- `buffer`

取得緩衝區的 `ArrayBuffer` 物件。

- `byteOffset`

取得緩衝區 `Arraybuffer` 物件的 `byteOffset`。

- `length`

取得緩衝區位元組計數。

Note

所有緩衝區模組方法都是 JavaScript 執行時間 2.0 的新功能。

查詢字串

Note

[CloudFront Functions 事件物件](#)會自動剖析 URL 查詢字串。這意味著在大多數情況下，您不需要使用此模組。

查詢字串模組 (`querystring`) 提供剖析和格式化 URL 查詢字串的方法。您可以使用 `require('querystring')` 加載模組。此模組提供下列方法。

`querystring.escape(string)`

URL 編碼給定的 `string`，傳回逸出的查詢字串。該方法由 `querystring.stringify()` 使用，並且不應直接使用。

`querystring.parse(string[, separator[, equal[, options]])`

剖析查詢字串 (`string`) 並傳回物件。

`separator` 參數是用來分隔查詢字串中的鍵/值對的子字串。其在預設情況下為 `&`。

`equal` 參數是用來分隔查詢字串中的鍵和值的子字串。其在預設情況下為 `=`。

`options` 參數是具有下列鍵的物件：

`decodeURIComponent function`

解碼查詢字串中百分比編碼字元的函數。其在預設情況下為 `querystring.unescape()`。

`maxKeys number`

要剖析的鍵的最大數量。其在預設情況下為 1000。使用 0 的值移除計數鍵的限制。

根據預設，查詢字串中的百分比編碼字元會假設為使用 UTF-8 編碼。無效的 UTF-8 序列會被取代為 U+FFFD 取代字元。

例如，對於下列查詢字串：

```
'name=value&abc=xyz&abc=123'
```

`querystring.parse()` 的返回值是：

```
{
  name: 'value',
  abc: ['xyz', '123']
}
```

`querystring.decode()` 是 `querystring.parse()` 的別名。

`querystring.stringify(object[, separator[, equal[, options]])`

序列化 `object` 並傳回查詢字串。

`separator` 參數是用來分隔查詢字串中的鍵/值對的子字串。其在預設情況下為 `&`。

`equal` 參數是用來分隔查詢字串中的鍵和值的子字串。其在預設情況下為 `=`。

`options` 參數是具有下列鍵的物件：

`encodeURIComponent` *function*

用於將 URL 不安全字元轉換為查詢字串中的百分比編碼的函數。其在預設情況下為 `querystring.escape()`。

根據預設，在查詢字串中需要百分比編碼的字元會編碼為 UTF-8。若要使用不同的編碼，請指定 `encodeURIComponent` 選項。

例如，對於以下程式碼：

```
querystring.stringify({ name: 'value', abc: ['xyz', '123'], anotherName: '' });
```

返回值是：

```
'name=value&abc=xyz&abc=123&anotherName='
```

`querystring.encode()` 是 `querystring.stringify()` 的別名。

`querystring.unescape(string)`

解碼給定 `string` 中的 URL 百分比編碼字元，傳回未逸出的查詢字串。此方法由 `querystring.parse()` 使用，並且不應直接使用。

加密

密碼編譯模組 (crypto) 提供標準雜湊和雜湊型訊息身分驗證碼 (HMAC) 協助程式。您可以使用 `require('crypto')` 加載模組。

雜湊方法

```
crypto.createHash(algorithm)
```

建立並傳回雜湊物件，藉助此物件，您可以使用給定的演算法產生雜湊摘要：md5、sha1 或 sha256。

```
hash.update(data)
```

使用給定的 *data* 更新雜湊內容。

```
hash.digest([encoding])
```

計算使用 `hash.update()` 傳遞的所有資料的摘要。編碼可以是 hex、base64 或 base64url。

HMAC 方法

```
crypto.createHmac(algorithm, secret key)
```

建立並返回使用給定 *algorithm* 和 *secret key* 的 HMAC 物件。演算法可以是 md5、sha1 或 sha256。

```
hmac.update(data)
```

使用給定的 *data* 更新 HMAC 內容。

```
hmac.digest([encoding])
```

計算使用 `hmac.update()` 傳遞的所有資料的摘要。編碼可以是 hex、base64 或 base64url。

限制功能

由於安全考量，下列 JavaScript 語言功能不受支援或受到限制。

動態程式碼評估

不支援動態程式碼評估。如果嘗試此評估，`eval()` 和 `Function` 建構子都會丟出錯誤。例如，`const sum = new Function('a', 'b', 'return a + b')` 丟出錯誤。

計時器

不支援 `setTimeout()`、`setImmediate()` 和 `clearTimeout()` 函數。在函數執行時期間未推遲或產生任何佈建。您的函數必須同步執行方可完成。

日期和時間戳記

基於安全考量，無法存取高解析度計時器。查詢當前時間的所有 `Date` 方法始終在單個函數執行的生命週期內返回相同的值。返回的時間戳記是函數開始執行的時間。因此，您無法測量函數中的經過時間。

檔案系統存取

沒有檔案系統存取權。例如，沒有類似 Node.js 中的檔案系統存取 `fs` 模組。

處理存取

沒有處理存取。例如沒有 `process` 全域物件可像 Node.js 一樣處理資訊存取。

環境變數

無法存取環境變數。您可以改為使用 CloudFront KeyValueCollection 為您的 CloudFront Functions 建立鍵值對的集中式資料儲存。CloudFront KeyValueCollection 可動態更新組態資料，不需要部署程式碼變更。如需更多詳細資訊，請參閱 [Amazon CloudFront KeyValueCollection](#)。

網路存取

不支援網路呼叫。例如，不支援 XHR、HTTP(S) 和通訊端。

鍵值存放區的協助程式方法

Note

CloudFront Functions 的鍵值存放區 helper 方法呼叫不會觸發 AWS CloudTrail 資料事件。這些事件不會記錄在 CloudTrail 事件歷史記錄中。如需詳細資訊，請參閱[使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫](#)。

如果您使用 [CloudFront 鍵值存放區](#) 在您建立的函數中包含鍵值，則本節內容適用。CloudFront Functions 具有的模組可提供三種協助程式方法來讀取鍵值存放區中的值。

若要在函數程式碼中使用此模組，請確定您已[將索引鍵值存放區與函數建立關聯](#)。

接著，在函數程式碼的第一行中包含下列陳述式：

```
import cf from 'cloudfront';  
const kvsHandle = cf.kvs();
```

get() 方法

使用此方法傳回您指定的鍵名稱的鍵值。

請求

```
get("key", options);
```

- key：需要擷取其值的鍵名稱
- options：有一個選項，format。它可以確保函數正確解析資料。可能的值如下：
 - string：(預設值) 以 UTF8 編碼
 - json
 - bytes：原始二進位資料緩衝區

請求範例

```
const value = await kvsHandle.get("myFunctionKey", { format: "string"});
```

回應

回應是 promise，會以使用 options 請求的格式解析為值。根據預設，值會以字串傳回。

錯誤處理

當您請求的鍵不存在於關聯的鍵值存放區時，get() 方法會傳回錯誤。若要管理此使用案例，您可以將 try 和 catch 區塊新增至程式碼。

Warning

使用 promise 組合器 (例如 Promise.all、Promise.any) 和 promise 鏈接方法 (例如 then 和 catch) 可能需要高函數記憶體用量。如果您的函數超過[函數記憶體配額上限](#)，則無法執行。為了避免此錯誤，建議您依序或在迴圈中使用 await 語法來請求多重值。

範例

```
var value1 = await kvs.get('key1');
```

```
var value2 = await kvs.get('key2');
```

目前，使用 promise 組合器取得多重值並不會改善效能，範例如下。

```
var values = await Promise.all([kvs.get('key1'), kvs.get('key2'),]);
```

exists() 方法

使用此方法來識別鍵是否存在於鍵值存放區中。

請求

```
exists("key");
```

請求範例

```
const exist = await kvsHandle.exists("myFunctionkey");
```

回應

回應是傳回布林值 (true 或 false) 的 promise。此值指定鍵是否存在於鍵值存放區中。

meta() 方法

此方法會傳回有關鍵值存放區的中繼資料。

請求

```
meta();
```

請求範例

```
const meta = await kvsHandle.meta();
```

回應

回應是一個 promise，可解析為具有以下屬性的物件：

- **creationDateTime**：鍵值存放區建立的日期和時間，以 ISO 8601 格式表示。

- `lastUpdatedDateTime`：上次從來源同步的鍵值存放區的日期和時間，以 ISO 8601 格式表示。該值不包括到邊緣的傳播時間。
- `keyCount`：上次從來源同步後 KVS 中的總鍵數。

回應範例

```
{keyCount:3,creationDateTime:2023-11-30T23:07:55.765Z,lastUpdatedDateTime:2023-12-15T03:57:52.4
```

原始伺服器修改的 Helper 方法

如果您動態更新或變更 CloudFront Functions 程式碼內請求所使用的原始伺服器，則本節適用。您只能在檢視器請求 CloudFront Functions 時更新原始伺服器。CloudFront Functions 有一個模組，提供 helper 方法來動態更新或變更原始伺服器。

若要使用此模組，請使用 JavaScript 執行時期 2.0 建立 CloudFront 函數，並在函數程式碼的第一行中包含下列陳述式：

```
import cf from 'cloudfront';
```

如需詳細資訊，請參閱[適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)。

Note

測試 API 和測試主控台頁面不會測試是否已發生原始伺服器修改。不過，測試可確保函數程式碼執行時不會發生錯誤。

在 CloudFront Functions 和 Lambda@Edge 之間進行選擇

您可以使用 CloudFront Functions 或 Lambda@Edge 來更新原始伺服器。

使用 CloudFront Functions 更新原始伺服器時，您可以使用檢視器請求事件觸發程序，這表示使用此函數時，每個請求都會執行此邏輯。使用 Lambda@Edge 時，原始伺服器更新功能位於原始伺服器請求事件觸發程序上，這表示此邏輯僅在快取遺漏時執行。

您的選擇主要取決於您的工作負載，以及分佈上 CloudFront Functions 和 Lambda@Edge 的任何現有用量。下列考量可協助您決定是否使用 CloudFront Functions 或 Lambda@Edge 來更新您的原始伺服器。

CloudFront Functions 在下列情況下最有用：

- 當您的請求是動態的 (表示無法快取) 且一律會前往原始伺服器時。CloudFront Functions 可提供更好的效能並降低整體成本。
- 當您已有現有的檢視器請求 CloudFront 函數會在每個請求上執行時，您可以將原始伺服器更新邏輯新增至現有的函數。

若要使用 CloudFront Functions 更新原始伺服器，請參閱下列主題中的 helper 方法。

Lambda@Edge 在下列情況下最有用：

- 當您有高度可快取的內容時，Lambda@Edge 可能更具成本效益，因為它只會在快取遺漏時執行，而 CloudFront Functions 會在每次請求時執行。
- 當您已有現有的原始伺服器請求 Lambda@Edge 函數時，您可以將原始伺服器更新邏輯新增至現有的函數。
- 當您的原始伺服器更新邏輯需要從第三方資料來源擷取資料時，例如 Amazon DynamoDB 或 Amazon S3。

如需 Lambda@Edge 的詳細資訊，請參閱「[使用 Lambda@Edge 在邊緣自訂](#)」。

updateRequestOrigin() 方法

使用 updateRequestOrigin() 方法更新請求的原始伺服器設定。您可以使用此方法更新已在分佈中定義的原始伺服器現有的原始伺服器屬性，或定義請求的新原始伺服器。若要這樣做，請指定您要變更的屬性。

Important

您在 updateRequestOrigin() 中未指定的任何設定都會繼承現有原始伺服器的組態中的相同設定。

updateRequestOrigin() 方法設定的原始伺服器可以是任何 HTTP 端點，而且不需要是 CloudFront 分佈中的現有原始伺服器。

備註

- 如果您要更新屬於原始伺服器群組的原始伺服器，則只會更新原始伺服器群組的主要原始伺服器。次要原始伺服器保持不變。來自修改原始伺服器且符合容錯移轉準則的任何回應代碼都會觸發容錯移轉至次要原始伺服器。

- 如果您要變更原始伺服器類型並啟用 OAC，請確定 `originAccessControlConfig` 中的原始伺服器類型符合新的原始伺服器類型。
- 您無法使用 `updateRequestOrigin()` 方法更新 [VPC 原始伺服器](#)。請求將會失敗。

請求

```
updateRequestOrigin({origin properties})
```

`origin properties` 可以包含以下：

`domainName` (選用)

原始伺服器的網域名稱。如果未提供，則會改用來自指派原始伺服器的網域名稱。

對於自訂原始伺服器

指定 DNS 網域名稱，例如 `www.example.com`。網域名稱不能包含冒號 (:)，也不能是 IP 地址。網域名稱長度上限為 253 個字元。

對於 S3 原始伺服器

指定 Amazon S3 儲存貯體的 DNS 網域名稱，例如 `amzn-s3-demo-bucket.s3.eu-west-1.amazonaws.com`。名稱可以高達 128 個字元，而且必須全部小寫。

`hostHeader` (選用，適用於non-S3 自訂原始伺服器)

向原始伺服器提出請求時要使用的主機標頭。如果未提供，則會使用 `domainName` 參數的值。如果未提供主機標頭或網域名稱參數，則會使用指派原始伺服器的網域名稱，如果轉送至原始伺服器 (FTO) 政策包含主機，則會使用來自傳入請求的主機標頭。主機標頭不能包含冒號 (:)，也不能是 IP 地址。主機標頭最多可達 253 個字元。

`originPath` (選用)

目錄路徑位於需定位內容請求的原始伺服器中。路徑的開頭應為正斜線 (/)，但結尾不應為正斜線。例如，不應以 `example-path/` 結尾。如果未提供，則會使用指派原始伺服器的原始路徑。

對於自訂原始伺服器

此路徑應以 URL 編碼且最多 255 個字元。

`customHeaders` (選用)

您可以透過指定的標頭名稱與每個自訂標頭的值對，於請求中包含自訂標頭。格式與事件結構中請求和回應標頭的格式不同。使用下列索引鍵/值對語法：

```
{"key1": "value1", "key2": "value2", ...}
```

您無法新增已列入不允許的標頭，且具有相同名稱的標頭也無法出現在傳入的請求 headers 中。函數程式碼中的標頭名稱必須為小寫。當 CloudFront Functions 將事件物件轉換回 HTTP 請求時，標頭名稱中每個單字的第一個字母會大寫，且單字之間用連字符分隔。

例如，如果函數程式碼新增名為 example-header-name 的標頭，CloudFront 會將其轉換為 HTTP 請求中的 Example-Header-Name。如需詳細資訊，請參閱[CloudFront 無法新增到原始伺服器請求的自訂標頭](#)及[對邊緣函數的限制](#)。

如果未提供，則會使用來自指派原始伺服器的任何自訂標頭。

connectionAttempts (選用)

CloudFront 會嘗試連線至原始伺服器的次數。最小值為 1，最大值為 3。如果未提供，則會使用來自指派原始伺服器的連線嘗試。

originShield (選用)

這會啟用或更新 CloudFront Origin Shield。使用 Origin Shield 有利於降低原始伺服器的負載。如需詳細資訊，請參閱[使用 Amazon CloudFront Origin Shield](#)。如果未提供，則會使用來自指派原始伺服器的 Origin Shield 設定。

enabled (必要)

啟用或停用 Origin Shield 的布林值表達式。接受值：true 或 false。

region (啟用時需要)

AWS 區域適用於 Origin Shield 的。指定對原始伺服器延遲最低的 AWS 區域。使用區域代碼，而非區域名稱。例如，使用 us-east-2 指定美國東部 (俄亥俄) 區域。

啟用 CloudFront Origin Shield 時，您必須為其指定 AWS 區域。如需可用 AWS 區域的清單並協助您選擇最適合原始伺服器的區域，請參閱[選擇 Origin Shield AWS 的區域](#)。

originAccessControlConfig (選用)

此原始伺服器的原始存取控制 (OAC) 唯一識別碼。這只會在原始伺服器支援 CloudFront OAC 時使用，例如 Amazon S3、Lambda 函數 URL、MediaStore 和 MediaPackage V2。如果未提供，則會使用來自指派原始伺服器的 OAC 設定。

這不支援舊版原始存取身分 (OAI)。如需詳細資訊，請參閱[限制對 AWS 原始伺服器的存取](#)。

enabled (必要)

啟用或停用 OAC 的布林值表達式。接受值：true 或 false。

signingBehavior (啟用時需要)

指定 CloudFront 會簽署哪些請求 (將身分驗證資訊新增至其中)。針對最常見的使用案例指定 `always`。如需詳細資訊，請參閱[原始存取控制的進階設定](#)。

此欄位可以有下列其中一個值：

- `always` – CloudFront 會簽署所有原始伺服器請求，並覆寫來自檢視器請求的 `Authorization` 標頭 (如果存在)。
- `never` – CloudFront 不簽署任何原始伺服器請求。此值會關閉原始伺服器的原始存取控制。
- `no-override` – 如果檢視器請求不包含 `Authorization` 標頭，則 CloudFront 會簽署原始伺服器請求。如果檢視器請求包含 `Authorization` 標頭，則 CloudFront 不會簽署原始伺服器請求，而是傳遞來自檢視器請求的 `Authorization` 標頭。

Warning

如要從檢視器請求傳遞 `Authorization` 標題，您必須將標題新增至原始伺服器請求政策中，適用於使用與此原始存取控制相關聯之原始伺服器的所有快取行為。如需詳細資訊，請參閱[使用政策控制原始伺服器請求](#)。

signingProtocol (啟用時需要)

OAC 的簽署通訊協定，將決定 CloudFront 簽署 (驗證) 請求的方式。唯一有效的值為 `sigv4`。

originType (啟用時需要)

此 OAC 的原始伺服器類型。有效值包括 `s3`、`mediapackagev2`、`mediastore` 與 `lambda`。

timeout (選用)

您可以指定 CloudFront 應嘗試等待原始伺服器回應或傳送資料的逾時時間。如果未提供，則會使用來自指派原始伺服器的逾時設定。

Note

除非另行指定，否則這些逾時支援自訂原始伺服器和 Amazon S3 原始伺服器。

readTimeout (選用)

`readTimeout` 適用於下列兩個值：

- 在將請求轉送到自訂原始伺服器之後，CloudFront 等待回應的時間 (以秒為單位)。
- CloudFront 在收到來自原始伺服器的回應封包後，並在接收下一個封包前，等待的時間長短 (以秒為單位)

最短的逾時時間是 1 秒，最長是 120 秒。如需詳細資訊，請參閱[回應逾時](#)。

`responseCompletionTimeout` (選用)

從 CloudFront 到原始伺服器的請求可以保持開啟並等待回應的時間 (以秒為單位)。如果此時未從原始伺服器收到完整回應，CloudFront 會結束連線。

`responseCompletionTimeout` 值必須大於或等於 `readTimeout` 的值。如需詳細資訊，請參閱[回應完成逾時](#)。

`keepAliveTimeout` (選用)

此逾時僅適用於自訂原始伺服器，不適用於 Amazon S3 原始伺服器。(S3 原始伺服器組態會忽略這些設定。)

`keepAliveTimeout` 指定 CloudFront 在收到上次封包的回應後維持與原始伺服器的連線所需的時間長度。最短的逾時時間是 1 秒，最長是 120 秒。如需詳細資訊，請參閱[保持連線逾時 \(僅限自訂與 VPC 原始伺服器\)](#)。

`connectionTimeout` (選用)

嘗試建立與原始伺服器的連線時，CloudFront 所等待的秒數。最短的逾時時間是 1 秒，最長是 10 秒。如需詳細資訊，請參閱[連線逾時](#)。

`customOriginConfig` (選用)

使用 `customOriginConfig` 指定非 Amazon S3 儲存貯體之原始伺服器的連線設定。有一個例外：如果 S3 儲存貯體設定為靜態網站託管，您可以指定這些設定。(其他類型的 S3 儲存貯體組態會忽略這些設定。) 如果未提供 `customOriginConfig`，則會使用來自指派原始伺服器的設定。
`port` (必要)。

CloudFront 用來連線至來源的 HTTP 連接埠。指定來源接聽使用的 HTTP 連接埠。

`protocol` (必要)

指定 CloudFront 用來連線至來源的通訊協定 (HTTP 或 HTTPS)。有效值如下：

- `http` — CloudFront 一律使用 HTTP 來連線至原始伺服器
- `https` — CloudFront 一律使用 HTTPS 來連線至原始伺服器

sslProtocols (必要)

指定 CloudFront 透過 HTTPS 連線至原始伺服器時所使用的最低 SSL/TLS 通訊協定的清單。有效值包括 SSLv3、TLSv1、TLSv1.1 與 TLSv1.2。如需詳細資訊，請參閱[最低來源 SSL 通訊協定](#)。

ipAddressType (選用)

指定 CloudFront 用來連線至原始伺服器的 IP 位址類型。有效值包括 ipv4、ipv6 與 dualstack。只有在同時變更 domainName 屬性時，才支援變更 ipAddressType。

sni (選用，適用於non-S3 自訂原始伺服器)

伺服器名稱指示 (SNI) 是 Transport Layer Security (TLS) 通訊協定的延伸，其中用戶端會指出其嘗試在 TLS 交握程序開始時連線到哪個主機名稱。此值應與原始伺服器上 TLS 憑證上的通用名稱相符。否則，您的原始伺服器可能會擲回錯誤。

如果未提供，則會使用 hostHeader 參數的值。如果未提供主機標頭，則會使用 domainName 參數的值。

如果未提供主機標頭或網域名稱參數，則會使用指派原始伺服器的網域名稱，如果轉送至原始伺服器 (FTO) 政策包含主機，則會使用來自傳入請求的主機標頭。SNI 不能包含冒號 (:)，也不能是 IP 地址。SNI 最多可達 253 個字元。

allowedCertificateNames (選用，適用於non-S3 自訂原始伺服器)

您可以包含 CloudFront 在 TLS 與您的原始伺服器交握期間，用來驗證原始伺服器 TLS 憑證之網域相符的有效憑證名稱清單。此欄位預期一系列有效的網域名稱，並可包含萬用字元網域，例如 *.example.com。

您最多可以指定 20 個允許的憑證名稱。每個憑證名稱最多可有 64 個字元。

Example – 更新至 Amazon S3 請求原始伺服器

下列範例會將檢視器請求的原始伺服器變更為 S3 儲存貯體、啟用 OAC，以及重設傳送至原始伺服器的自訂標頭。

```
cf.updateRequestOrigin({
  "domainName" : "amzn-s3-demo-bucket-in-us-east-1.s3.us-east-1.amazonaws.com",
  "originAccessControlConfig": {
    "enabled": true,
    "signingBehavior": "always",
```



```
        "signingProtocol": "sigv4",
        "originType": "s3"
    },
    // Empty object resets any header configured on the assigned origin
    "customHeaders": {}
});
```

Example – 更新 Application Load Balancer 請求原始伺服器

下列範例會將檢視器請求的原始伺服器變更為 Application Load Balancer 原始伺服器，並設定自訂標頭和逾時。

```
cf.updateRequestOrigin({
    "domainName" : "example-1234567890.us-east-1.elb.amazonaws.com",
    "timeouts": {
        "readTimeout": 30,
        "connectionTimeout": 5
    },
    "customHeaders": {
        "x-stage": "production",
        "x-region": "us-east-1"
    }
});
```

Example – 在啟用 Origin Shield 的情況下更新原始伺服器

在下列範例中，分佈中的原始伺服器已啟用 Origin Shield。函數程式碼只會更新用於原始伺服器的網域名稱，並省略所有其他選用參數。在此情況下，Origin Shield 仍會與修改後的原始網域名稱搭配使用，因為 Origin Shield 參數未更新。

```
cf.updateRequestOrigin({
    "domainName" : "www.example.com"
});
```

Example – 更新主機標頭、SNI 和允許的憑證名稱

Warning

對於大多數使用案例，您不需要對前往原始伺服器的請求使用此類型的修改。除非您了解變更這些值的影響，否則不應使用這些參數。

下列範例會變更請求原始伺服器的網域名稱、主機標頭、SNI 和允許的憑證。

```
cf.updateRequestOrigin({
  "domainName": "www.example.com",
  "hostHeader": "test.example.com",
  "sni": "test.example.net",
  "allowedCertificateNames": ["*.example.com", "*.example.net"],
});
```

selectRequestOriginById() 方法

使用 `selectRequestOriginById()` 來更新現有的原始伺服器，方法是選取已在分佈中設定的不同原始伺服器。此方法使用更新原始伺服器所定義的所有相同設定。

此方法僅接受已在執行函數時所用的相同分佈中定義的原始伺服器。原始伺服器由原始伺服器 ID 參考，這是您在設定原始伺服器時定義的原始伺服器名稱。

如果您的分佈中已設定 VPC 原始伺服器，您可以使用此方法將原始伺服器更新為 VPC 原始伺服器。如需詳細資訊，請參閱[使用 VPC 原始伺服器限制存取](#)。

請求

```
cf.selectRequestOriginById(origin_id, {origin_overrides})
```

在先前的範例中，`origin_id` 是一個字串，指向執行函數之分佈中原始伺服器的原始伺服器名稱。`origin_overrides` 參數可以包含下列項目：

`hostHeader` (選用，適用於non-S3 自訂原始伺服器)

向原始伺服器提出請求時要使用的主機標頭。如果未提供，則會使用 `domainName` 參數的值。

如果未提供主機標頭或網域名稱參數，則會使用指派原始伺服器的網域名稱，如果轉送至原始伺服器 (FTO) 政策包含主機，則會使用來自傳入請求的主機標頭。主機標頭不能包含冒號 (:)，也不能是 IP 地址。主機標頭最多可達 253 個字元。

`sni` (選用，適用於non-S3 自訂原始伺服器)

伺服器名稱指示 (SNI) 是 Transport Layer Security (TLS) 通訊協定的延伸，其中用戶端會指出其嘗試在 TLS 交握程序開始時連線到哪個主機名稱。此值應與原始伺服器上 TLS 憑證上的通用名稱相符。否則，您的原始伺服器可能會擲回錯誤。

如果未提供，則會使用 `hostHeader` 參數的值。如果未提供主機標頭，則會使用 `domainName` 參數的值。

如果未提供主機標頭或網域名稱參數，則會使用指派原始伺服器的網域名稱，如果轉送至原始伺服器 (FTO) 政策包含主機，則會使用來自傳入請求的主機標頭。SNI 不能包含冒號 (:)，也不能是 IP 地址。SNI 最多可達 253 個字元。

`allowedCertificateNames` (選用，適用於non-S3 自訂原始伺服器)

您可以包含 CloudFront 在 TLS 與您的原始伺服器交握期間，用來驗證原始伺服器 TLS 憑證之網域相符的有效憑證名稱清單。此欄位預期一系列有效的網域名稱，並可包含萬用字元網域，例如 `*.example.com`。

您最多可以指定 20 個允許的憑證名稱。每個憑證名稱最多可有 64 個字元。

請求

```
selectRequestOriginById(origin_id)
```

在上述範例中，`origin_id` 是指向執行函數之分佈中原始伺服器的原始伺服器名稱的字串。

Example – 選取 Amazon S3 請求原始伺服器

下列範例會從與分佈相關聯的原始伺服器清單中選擇名為 `amzn-s3-demo-bucket-in-us-east-1` 的原始伺服器，並將 `amzn-s3-demo-bucket-in-us-east-1` 原始伺服器的組態設定套用至請求。

```
cf.selectRequestOriginById("amzn-s3-demo-bucket-in-us-east-1");
```

Example – 選取 Application Load Balancer 請求原始伺服器

下列範例會從與分佈相關聯的原始伺服器清單中選擇名為 `myALB-prod` 的 Application Load Balancer 原始伺服器，並將 `myALB-prod` 的組態設定套用至請求。

```
cf.selectRequestOriginById("myALB-prod");
```

Example – 選取 Application Load Balancer 請求原始伺服器並覆寫主機標頭

如同上述範例，下列範例 `myALB-prod` 會從與分佈相關聯的原始伺服器清單中選擇名為 `myALB-prod` 的 Application Load Balancer 原始伺服器，並將 `myALB-prod` 的組態設定套用至 `myALB-prod` 請求。不過，此範例會使用覆寫主機標頭值 `origin_overrides`。

```
cf.overrideRequestOrigin("myALB-prod",{
    "hostHeader" : "test.example.com"
});
```

createRequestOriginGroup() 方法

使用 `createRequestOriginGroup()` 定義兩個原始伺服器，以在需要高可用性的情況下用作容錯移轉的[原始伺服器群組](#)。

原始伺服器群組包括兩個原始伺服器 (主要原始伺服器和次要原始伺服器)，以及您指定的容錯移轉準則。您建立原始伺服器群組以支援 CloudFront 中的原始伺服器容錯移轉。當您使用此方法建立或更新原始伺服器群組時，您可以指定原始伺服器群組，而不是單一原始伺服器。CloudFront 將使用容錯移轉準則，從主要原始伺服器容錯移轉至次要原始伺服器。

如果您的分佈中已設定 VPC 原始伺服器，您可以使用此方法來使用 VPC 原始伺服器建立原始伺服器群組。如需詳細資訊，請參閱[使用 VPC 原始伺服器限制存取](#)。

請求

```
createRequestOriginGroup({origin_group_properties})
```

在前面的範例中，`origin_group_properties` 包含以下：

originIds (必要)

`origin_ids` 陣列，其中 `origin_id` 是指向執行函數之分佈中原始伺服器的原始伺服器名稱的字串。您必須提供兩個原始伺服器做為陣列的一部分。清單中的第一個原始伺服器是主要原始伺服器，第二個則做為容錯移轉用途的次要原始伺服器。

originOverrides (選用)

使用 `{origin_overrides}` 參數可覆寫一些進階設定。`origin overrides` 可以包含以下：

`hostHeader` (選用，適用於non-S3 自訂原始伺服器)

向原始伺服器提出請求時要使用的主機標頭。如果未提供，則會使用 `domainName` 參數的值。

如果未提供主機標頭或網域名稱參數，則會使用指派原始伺服器的網域名稱，如果轉送至原始伺服器 (FTO) 政策包含主機，則會使用來自傳入請求的主機標頭。主機標頭不能包含冒號 (:)，也不能是 IP 地址。主機標頭最多可達 253 個字元。

sni (選用, 適用於non-S3 自訂原始伺服器)

伺服器名稱指示 (SNI) 是 Transport Layer Security (TLS) 通訊協定的延伸, 用戶端會在 TLS 交握程序開始時, 指出其嘗試連線到哪個主機名稱。此值應與原始伺服器上 TLS 憑證上的通用名稱相符, 否則原始伺服器可能會擲出錯誤。

如果未提供, 則會使用 hostHeader 參數的值。如果未提供主機標頭, 則會使用 domainName 參數的值。

如果未提供主機標頭或網域名稱參數, 則會使用指派原始伺服器的網域名稱, 如果轉送至原始伺服器 (FTO) 政策包含主機, 則會使用來自傳入請求的主機標頭。SNI 不能包含冒號 (:), 也不能是 IP 地址。SNI 最多可達 253 個字元。

allowedCertificateNames (選用, 適用於non-S3 自訂原始伺服器)

您可以包含 CloudFront 在 TLS 與您的原始伺服器交握期間, 用來驗證原始伺服器 TLS 憑證之網域相符的有效憑證名稱清單。此欄位預期一系列有效的網域名稱, 並可包含萬用字元網域, 例如 *.example.com。

您最多可以指定 20 個允許的憑證名稱。每個憑證名稱最多可有 64 個字元。

selectionCriteria (選用)

選取要使用 default 原始伺服器容錯移轉準則或使用 media-quality-score 型容錯移轉邏輯。有效值如下:

- default 根據 failoverCriteria 中指定的狀態碼使用容錯移轉準則。如果您未在函數中設定 selectionCriteria, 則會使用 default。
- 使用媒體感知路由功能時, 會使用 media-quality-score。

failoverCriteria (必要)

從主要原始伺服器傳回的狀態碼陣列, 將觸發 CloudFront 容錯移轉至次要原始伺服器。如果您覆寫現有的原始伺服器群組, 此陣列會覆寫原始伺服器群組原始組態中設定的所有容錯移轉狀態碼。

當您使用 media-quality-score selectionCriteria 時, CloudFront 會根據媒體品質分數嘗試路由請求。如果選取的原始伺服器傳回此陣列中設定的錯誤碼, CloudFront 會容錯移轉至另一個原始伺服器。

Example – 建立請求原始伺服器群組

下列範例會使用原始伺服器 ID 為請求建立原始伺服器群組。這些原始伺服器 ID 來自用於執行此函數之分佈的原始伺服器群組組態。

或者，您可以使用 `originOverrides` 覆寫 `sni`、`hostHeader` 和 的原始伺服器群組組態 `allowedCertificateNames`。

```
import cf from 'cloudfront';

function handler(event) {
  cf.createRequestOriginGroup({
    "originIds": [
      {
        "originId": "origin-1",
        "originOverrides": {
          "hostHeader": "hostHeader.example.com",
          "sni": "sni.example.com",
          "allowedCertificateNames": ["cert1.example.com",
"cert2.example.com", "cert3.example.com"]
        }
      },
      {
        "originId": "origin-2",
        "originOverrides": {
          "hostHeader": "hostHeader2.example.com",
          "sni": "sni2.example.com",
          "allowedCertificateNames": ["cert4.example.com",
"cert5.example.com"]
        }
      }
    ],
    "failoverCriteria": {
      "statusCodes": [500]
    }
  });

  event.request.headers['x-hookx'] = { value: 'origin-overrides' };
  return event.request;
}
```

CloudFront SaaS Manager 屬性的 helper 方法

使用 CloudFront SaaS Manager 的下列 helper 函數，在您建立的函數中擷取多租用戶分佈的值。若要在此頁面上使用範例，您必須先使用 JavaScript 執行時期 2.0 建立 CloudFront 函數。如需詳細資訊，[適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)。

主題

- [連線群組](#)
- [分佈租用戶](#)

連線群組

與分佈租用戶相關聯的連線群組具有網域名稱。

若要取得此值，請使用事件物件 context 子物件的 endpoint 欄位。

請求

```
const value = event.context.endpoint;
```

回應

回應是包含連線群組網域名稱的 string，例如 d111111abcdef8.cloudfront.net。只有當您的函數被調用具有相關聯連線群組的多租用戶分佈時，endpoint 欄位才會出現。如需詳細資訊，請參閱[內容物件](#)。

分佈租用戶

CloudFront Functions 有一個模組，可讓您存取特定的分佈租用戶值。

若要使用此模組，請在函數程式碼的第一行中包含下列陳述式：

```
import cf from 'cloudfront';
```

您只能在 handler 函數中直接或透過任何巢狀呼叫函數使用下列範例。

distributionTenant.id 欄位

使用此欄位來取得分佈租用戶 ID 的值。

請求

```
const value = cf.distributionTenant.id;
```

回應

回應是包含分佈租用戶 ID 的 string，例如 dt_1a2b3c4d5e6f7。

錯誤處理

如果您的函數是針對標準分佈調用，則指定 `distributionTenant.id` 欄位將傳回 `distributionTenant module is not available` 類型錯誤。若要處理此使用案例，您可以將 `try` 和 `catch` 區塊新增至程式碼。

`distributionTenant.parameters.get()` 方法

使用此方法可傳回您指定之分佈租用戶參數名稱的值。

```
distributionTenant.parameters.get("key");
```

`key`：您要為其擷取值的分佈租用戶參數名稱。

請求

```
const value = distributionTenant.parameters.get("key");
```

回應

回應是 `string`，其中包含分佈租用戶參數的值。例如，如果您的鍵名為 `TenantPath`，則此參數的值可能是 `tenant1`。

錯誤處理

您可能會收到下列錯誤：

- 如果您的函數是針對標準分佈調用，則 `distributionTenant.parameters.get()` 方法將傳回 `distributionTenant module is not available` 類型錯誤。
- 當您指定的分佈租用戶參數不存在時，會傳回 `DistributionTenantParameterKeyNotFound` 錯誤。

若要管理這些使用案例，您可以將 `try` 和 `catch` 區塊新增至程式碼。

使用 `async` 和 `await`

CloudFront Functions JavaScript 執行時期函數 2.0 提供處理 Promise 物件的 `async` 和 `await` 語法。Promises 代表可以透過函數中標記為 `async` 的關鍵字 `await` 存取延遲結果。各種新的 WebCrypto 函數使用 Promises。

如需 Promise 物件的詳細資訊，請參閱 [Promise](#)。

Note

您必須針對下列程式碼範例使用 JavaScript 執行時期 2.0。
await 只能在 async 函數內使用，不支援 async 引數和關閉。

```
async function answer() {
    return 42;
}

// Note: async, await can be used only inside an async function. async arguments and
// closures are not supported.

async function handler(event) {
    // var answer_value = answer(); // returns Promise, not a 42 value
    let answer_value = await answer(); // resolves Promise, 42
    console.log("Answer"+answer_value);
    event.request.headers['answer'] = { value : ""+answer_value };
    return event.request;
}
```

下面的範例 JavaScript 程式碼顯示如何使用 then 鏈接方法查看 Promises。您可以使用 catch 來檢視錯誤。

Warning

使用 promise 組合器 (例如 Promise.all、Promise.any) 和 promise 鏈接方法 (例如 then 和 catch) 可能需要高函數記憶體用量。如果您的函數超過[函數記憶體配額上限](#)，則無法執行。為了避免此錯誤，建議您使用 await 語法，而非 promise 方法。

```
async function answer() {
    return 42;
}

async function squared_answer() {
    return answer().then(value => value * value)
}
```



```
// Note: async, await can be used only inside an async function. async arguments and
// closures are not supported.
async function handler(event) {
    // var answer_value = answer(); // returns Promise, not a 42 value
    let answer_value = await squared_answer(); // resolves Promise, 42
    console.log("Answer"+answer_value);
    event.request.headers['answer'] = { value : ""+answer_value };
    return event.request;
}
```

CloudFront Functions 的 CWT 支援

本節提供 CloudFront Functions 中 CBOR Web Token (CWT) 支援的詳細資訊，可在 CloudFront Edge Locations 啟用安全字串型身分驗證和授權。此支援是以模組形式提供，可在 CloudFront Function 中存取。

若要使用此模組，請使用 JavaScript 執行期 2.0 建立 CloudFront 函數，並在函數程式碼的第一行中包含下列陳述式：

```
import cf from 'cloudfront';
```

與此模組相關聯的方法可透過 存取（其中 * 是萬用字元，代表模組中存在的不同函數）：

```
cf.cwt.*
```

如需詳細資訊，請參閱[適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)。

目前，模組僅支援具有 HS256 (HMAC-SHA256) 演算法的 MAC0 結構，字符大小上限為 1KB。

字符結構

本節涵蓋 CWT 模組預期的字符結構。模組預期字符會正確標記和識別（例如 COSE MAC0）。此外，對於字符的結構，模組遵循 [CBOR 物件簽署和加密 \(COSE\) 【RFC 8152】](#) 設定的標準。

```
( // CWT Tag (Tag value: 61) --- optional
  ( // COSE MAC0 Structure Tag (Tag value: 17) --- required
    [
      protectedHeaders,
      unprotectedHeaders,
      payload,
      tag,
```

```
    ]
  )
)
```

Example：使用 COSE MAC0 結構的 CWT

```
61( // CWT tag
  17( // COSE_MAC0 tag
    [
      { // Protected Headers
        1: 4 // algorithm : HMAC-256-64
      },
      { // Unprotected Headers
        4: h'53796d6d6574726963323536' // kid : Symmetric key id
      },
      { // Payload
        1: "https://iss.example.com", // iss
        2: "exampleUser", // sub
        3: "https://aud.example.com", // aud
        4: 1444064944, // exp
        5: 1443944944, // nbf
        6: 1443944944, // iat
      },
      h'093101ef6d789200' // tag
    ]
  )
)
```

Note

產生字符時，CWT 標籤是選用的。不過，需要 COSE 結構標籤。

validateToken() 方法

函數會使用指定的金鑰解碼和驗證 CWT 字符。如果驗證成功，則會傳回解碼的 CWT 字符。否則，它會擲回錯誤。請注意，此函數不會對宣告集進行驗證。

請求

```
cf.cwt.validateToken(token, handlerContext{key})
```

Parameters

字符（必要）

用於驗證的編碼字符。這必須是 JavaScript 緩衝區。

handlerContext（必要）

存放 validateToken 呼叫內容的 JavaScript 物件。目前僅支援 金鑰屬性。

金鑰（必要）

訊息摘要運算的私密金鑰。可做為字串或 JavaScript 緩衝區提供。

回應

當 validateToken() 方法傳回成功驗證的字符時，函數的回應會以下列格式 CWTObject 顯示。解碼後，所有宣告金鑰都會以字串表示。

```
CWTObject {
  protectedHeaders,
  unprotectedHeaders,
  payload
}
```

範例 - 使用作為字符一部分傳送的 kid 驗證字符

此範例示範 CWT 字符驗證，其中會從 標頭擷取 kid。接著 kid 會傳遞至 CloudFront Functions KeyValueCollection，以擷取用於驗證權杖的私密金鑰。

```
import cf from 'cloudfront'

const CwtClaims = {
  iss: 1,
  aud: 3,
  exp: 4
}

async function handler(event) {
  try {
    let request = event.request;
    let encodedToken = request.headers['x-cwt-token'].value;
```

```
let kid = request.headers['x-cwt-kid'].value;

// Retrieve the secret key from the kvs
let secretKey = await cf.kvs().get(kid);

// Now you can use the secretKey to decode & validate the token.
let tokenBuffer = Buffer.from(encodedToken, 'base64url');

let handlerContext = {
  key: secretKey,
}

try {
  let cwtObj = cf.cwt.validateToken(tokenBuffer, handlerContext);

  // Check if token is expired
  const currentTime = Math.floor(Date.now() / 1000); // Current time in
seconds
  if (cwtObj[CwtClaims.exp] && cwtObj[CwtClaims.exp] < currentTime) {
    return {
      statusCode: 401,
      statusDescription: 'Token expired'
    };
  }
} catch (error) {
  return {
    statusCode: 401,
    statusDescription: 'Invalid token'
  };
} catch (error) {
  return {
    statusCode: 402,
    statusDescription: 'Token processing failed'
  };
}
return request;
}
```

generateToken() 方法

此函數會使用提供的承載和內容設定產生新的 CWT 字符。

請求

```
cf.cwt.generateToken(generatorContext, payload)
```

Parameters

generatorContext (必要)

這是 JavaScript 物件，可做為產生字符的內容，並包含下列索引鍵值對：

cwtTag (選用)

此值是布林值，如果 `true` 指定 `cwtTag` 應新增。

coseTag (必要)

指定 COSE 標籤類型。目前僅支援 `MAC0`。

金鑰 (必要)

用於運算訊息摘要的私密金鑰。此值可以是字串或 JavaScript Buffer。

承載 (必要)

用於編碼的字符承載。承載必須是 `CWTObject` 格式。

回應

傳回包含編碼字符的 JavaScript 緩衝區。

Example：產生 CWT 字符

```
import cf from 'cloudfront';

const CwtClaims = {
  iss: 1,
  sub: 2,
  exp: 4
};

const CatClaims = {
  catu: 401,
  catnip: 402,
  catm: 403,
  catr: 404
};
```

```
const Catu = {
  host: 1,
  path: 2,
  ext: 3
};

const CatuMatchTypes = {
  prefix_match: 1,
  suffix_match: 2,
  exact_match: 3
};

const Catr = {
  renewal_method: 1,
  next_renewal_time: 2,
  max_uses: 3
};

async function handler(event) {
  try {
    const response = {
      statusCode: 200,
      statusDescription: 'OK',
      headers: {}
    };

    const commonAccessToken = {
      protected: {
        1: "5",
      },
      unprotected: {},
      payload: {
        [CwtClaims.iss]: "cloudfront-documentation",
        [CwtClaims.sub]: "cwt-support-on-cloudfront-functions",
        [CwtClaims.exp]: 1740000000,
        [CatClaims.catu]: {
          [Catu.host]: {
            [CatuMatchTypes.suffix_match]: ".cloudfront.net"
          },
          [Catu.path]: {
            [CatuMatchTypes.prefix_match]: "/media/live-stream/cf-4k/"
          },
          [Catu.ext]: {
```

```

        [CatuMatchTypes.exact_match]: [
            ".m3u8",
            ".ts",
            ".mpd"
        ]
    },
    [CatClaims.catnip]: [
        "[IP_ADDRESS]",
        "[IP_ADDRESS]"
    ],
    [CatClaims.catm]: [
        "GET",
        "HEAD"
    ],
    [CatClaims.catr]: {
        [Catr.renewal_method]: "header_renewal",
        [Catr.next_renewal_time]: 1750000000,
        [Catr.max_uses]: 5
    }
}
};

if (!request.headers['x-cwt-kid']) {
    throw new Error('Missing x-cwt-kid header');
}

const kid = request.headers['x-cwt-kid'].value;
const secretKey = await cf.kvs().get(kid);

if (!secretKey) {
    throw new Error('Secret key not found for provided kid');
}

try {
    const genContext = {
        cwtTag: true,
        coseTag: "MAC0",
        key: secretKey
    };

    const tokenBuffer = cf.cwt.generateToken(commonAccessToken, genContext);
    response.headers['x-generated-cwt-token'] = { value:
tokenBuffer.toString('base64url') };

```

```

        return response;
    } catch (tokenError) {
        return {
            statusCode: 401,
            statusDescription: 'Could not generate the token'
        };
    }
} catch (error) {
    return {
        statusCode: 402,
        statusDescription: 'Token processing failed'
    };
}
}

```

Example：根據某些邏輯重新整理權杖

```

import cf from 'cloudfront'

const CwtClaims = {
  iss: 1,
  aud: 3,
  exp: 4
}

async function handler(event) {
  try {
    let request = event.request;
    let encodedToken = request.headers['x-cwt-token'].value;
    let kid = request.headers['x-cwt-kid'].value;
    let secretKey = await cf.kvs().get(kid); // Retrieve the secret key from the
    kvs

    // Now you can use the secretKey to decode & validate the token.
    let tokenBuffer = Buffer.from(encodedToken, 'base64url');

    let handlerContext = {
      key: secretKey,
    }

    try {
      let cwtJSON = cf.cwt.validateToken(tokenBuffer, handlerContext);

```



```
// Check if token is expired
const currentTime = Math.floor(Date.now() / 1000); // Current time in
seconds
if (cwtJSON[CwtClaims.exp] && cwtJSON[CwtClaims.exp] < currentTime) {
    // We can regenerate the token and add 8 hours to the expiry time
    cwtJSON[CwtClaims.exp] = Math.floor(Date.now() / 1000) + (8 * 60 * 60);

    let genContext = {
        coseTag: "MAC0",
        key: secretKey
    }

    let newTokenBuffer = cf.cwt.generateToken(cwtJSON, genContext);
    request.headers['x-cwt-regenerated-token'] =
newTokenBuffer.toString('base64url');
}
} catch (error) {
    return {
        statusCode: 401,
        statusDescription: 'Invalid token'
    };
}
}
catch (error) {
    return {
        statusCode: 402,
        statusDescription: 'Token processing failed'
    };
}
return request;
}
```

一般協助程式方法

此頁面提供 CloudFront Functions 內的其他協助程式方法。若要使用這些方法，請使用 JavaScript 執行期 2.0 建立 CloudFront 函數。

```
import cf from 'cloudfront';
```

如需詳細資訊，請參閱[適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能](#)。

edgeLocation 中繼資料

此方法需要使用 cloudfront 模組。

Note

您只能將此方法用於檢視器請求函數。對於檢視器-回應函數，此方法為空。

使用此 JavaScript 物件來取得節點機場代碼、預期的 [區域節點快取](#) 區域或用於處理請求的 CloudFront 伺服器 IP 地址。此中繼資料僅適用於檢視器請求事件觸發。

```
cf.edgeLocation = {  
  name: SEA  
  serverIp: 1.2.3.4  
  region: us-west-2  
}
```

cf.edgeLocation 物件可以包含下列項目：

name

處理請求之節點的三個字母 [IATA 代碼](#)。

serverIp

處理請求之伺服器的 IPv4 或 IPv6 地址。

region

如果發生快取遺漏，預期請求使用的 CloudFront Regional Edge Cache (REC)。如果預期的 REC 無法使用，且請求使用備份 REC，則不會更新此值。這不包括正在使用的 Origin Shield 位置，除非主要 REC 和 Origin Shield 是相同的位置。

Note

當 CloudFront 設定為使用原始伺服器容錯移轉時，不會再次叫用 CloudFront Functions。如需詳細資訊，請參閱 [透過 CloudFront 原始伺服器容錯移轉最佳化高可用性](#)。

rawQueryString() 方法

此方法不需要 cloudFront 模組。

使用 rawQueryString() 方法將未剖析和未變更的查詢字串擷取為字串。

請求

```
function handler(event) {  
    var request = event.request;  
    const qs = request.rawQueryString();  
}
```

回應

將傳入請求的完整查詢字串傳回為字串值，而不含前置 ?。

- 如果沒有查詢字串，但 ? 存在，則函數會傳回空字串。
- 如果沒有查詢字串且 ? 不存在，則函數會傳回 undefined。

案例 1：傳回完整查詢字串（沒有前置 ?）

傳入請求 URL：https://example.com/page?name=John&age=25&city=Boston

rawQueryString() 傳回："name=John&age=25&city=Boston"

案例 2：傳回空字串（當 ? 存在但沒有參數時）

傳入請求 URL：https://example.com/page?

rawQueryString() 傳回：""

案例 3：undefined 傳回（沒有查詢字串，也沒有 ?）

傳入請求 URL：https://example.com/page

rawQueryString() 傳回：undefined

建立函數

您可以分兩個階段建立函數：

1. 建立函數程式碼做為 JavaScript。您可以從 CloudFront 主控台使用預設範例或自行撰寫。如需詳細資訊，請參閱下列主題：
 - [撰寫函數程式碼](#)
 - [the section called “事件結構”](#)
 - [CloudFront 的 CloudFront Functions 範例](#)
2. 然後，您可以使用 CloudFront 建立函數並包含程式碼。程式碼存在於函數內部 (而不是引用形式)。

Console

建立 函數

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。
2. 選擇 Create function (建立函數)。
3. 輸入在 AWS 帳戶內唯一的函數名稱，然後選擇 JavaScript 版本，接著選擇繼續。新函數的詳細資訊頁面會隨即顯示。

Note

如要在函數中使用[鍵值對](#)，您必須選擇 JavaScript 執行時期 2.0。

4. 在函數程式碼區段中，選擇建置索引標籤，然後輸入您的函數程式碼。建置索引標籤中包含的範例程式碼會說明函數程式碼的基本語法。
5. 選擇儲存變更。
6. 如果函數程式碼使用鍵值對，則必須關聯鍵值存放區。

在首次建立函數時，您可以關聯鍵值存放區。或者，您可以稍後透過[更新函數](#)來關聯它。

若要立即關聯鍵值存放區，請依照下列步驟執行：

- 前往關聯 KeyValueStore 區段，然後選擇關聯現有的 KeyValueStore。
- 選取函數中包含鍵值對的鍵值存放區，然後選擇關聯 KeyValueStore。

CloudFront 會立即將存放區與該函數建立關聯。您無需儲存函數。

CLI

如果您使用 CLI，通常會先在檔案中建立函數程式碼，然後使用 AWS CLI 來建立函數。

建立 函數

1. 在檔案中建立函數程式碼，並將其儲存於電腦可以連線的目錄中。
2. 執行命令，如範例所示。此範例會使用 `fileb://` 標記法來傳入檔案。它還會包括換行符號，讓命令更易於讀取。

```
aws cloudfront create-function \  
  --name MaxAge \  
  --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-  
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":  
[{"KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-store/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}]}' \  
  --function-code fileb://function-max-age-v1.js
```

備註

- Runtime – JavaScript 版本。如要在函數中使用[鍵值對](#)，您必須指定版本 2.0。
- KeyValueStoreAssociations – 如果您的函數使用鍵值對，則可以在初次建立函數時關連鍵值存放區。或者，您可以稍後使用 `update-function` 來關聯它。Quantity 永遠等於 1，因為每個函數只能有一個與其關聯的鍵值存放區。

如果命令成功執行，您會看到如下所示的輸出。

```
ETag: ETVABCEXAMPLE  
FunctionSummary:  
  FunctionConfig:  
    Comment: Max Age 2 years  
    Runtime: cloudfront-js-2.0  
    KeyValueStoreAssociations= \  
      {Quantity=1, \  
        Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-  
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]} \  
  FunctionMetadata:  
    CreatedTime: '2021-04-18T20:38:56.915000+00:00'  
    FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge
```

```
LastModifiedTime: '2023-11-19T20:38:56.915000+00:00'  
Stage: DEVELOPMENT  
Name: MaxAge  
Status: UNPUBLISHED  
Location: https://cloudfront.amazonaws.com/2020-05-31/function/  
arn:aws:cloudfront::function/MaxAge
```

大多數資訊都是從請求中複製的。其他資訊會由 CloudFront 新增。

備註

- ETag – 每次修改鍵值存放區時，這個值都會變更。您可以使用此值和函數名稱，以在未來參考函數。確保始終使用目前的 ETag。
- FunctionARN – CloudFront 函數的 ARN。
- 111122223333 – AWS 帳戶。
- Stage – 函數的階段 (LIVE 或 DEVELOPMENT)。
- Status – 函數的狀態 (PUBLISHED 或 UNPUBLISHED)。

建立函數之後，該函數會新增至 DEVELOPMENT 階段。建議您[在發佈函數](#)之前進行[測試](#)。發佈函數之後，函數會變更為 LIVE 階段。

測試函數

在將函數部署到即時階段 (生產環境) 之前，可以測試該函數以確保其運作正常。若要測試函數，您可以指定一個事件物件，代表您的 CloudFront 分佈可在生產環境中接收的 HTTP 請求或回應。

CloudFront Functions 會執行下列作業：

1. 執行該函數，使用提供的事件物件作為函數的輸入。
2. 返回函數的結果 (修改後的事件物件)，同時返回任何函數日誌或錯誤訊息以及函數的運算利用率。如需運算使用率的詳細資訊，請參閱 [the section called “了解運算利用率”](#)。

 Note

當您測試函數時，CloudFront 只會驗證函數執行錯誤。CloudFront 不會驗證發佈後請求是否成功通過。例如，如果您的函數刪除必要的標頭，測試將會成功，因為程式碼沒有問題。不過，如果您發佈函數並將其與分佈建立關聯，則函數會在透過 CloudFront 提出請求時失敗。

內容

- [設定事件物件](#)
- [測試函數](#)
- [了解運算利用率](#)

設定事件物件

在測試函數之前，您必須建立事件物件以進行測試。有幾種選項。

選項 1：設定事件物件而不儲存

您可以在 CloudFront 主控台的視覺化編輯器中設定事件物件，而不儲存該事件物件。

您可以使用此事件物件從 CloudFront 主控台測試函數，即使該函數尚未儲存也是如此。

選項 2：在視覺化編輯器中建立事件物件

您可以在 CloudFront 主控台的視覺化編輯器中設定事件物件，而不儲存該事件物件。您可以針對每個函數建立 10 個事件物件，例如，可測試不同的可能輸入。

以這種方式建立事件物件時，您可以使用事件物件，在 CloudFront 主控台中測試函數。您不能使用它來測試使用 AWS API 或 SDK 的函數。

選項 3：使用文字編輯器建立事件物件

您可以使用文字編輯器，以 JSON 格式建立事件物件。如需有關事件物件結構的詳細資訊，請參閱 [事件結構](#)。

您可以使用此事件物件來測試使用 CLI 的函數。但是您無法使用它來測試 CloudFront 主控台內的函數。

建立事件物件 (選項 1 或 2)

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。

選擇您要測試的函數。

2. 在函數詳細資訊頁面上，選擇測試索引標籤。
3. 針對事件類型，選擇以下其中一個選項：
 - 如果函數會根據請求修改 HTTP 請求或產生回應，請選擇檢視者請求。請求區段隨即出現。
 - 選擇檢視器回應。請求和回應區段隨即出現。
4. 完成您想要包含在事件中的所有欄位。您可選擇編輯 JSON 來檢視原始 JSON。
5. (選用) 若要儲存事件，請選擇儲存，然後在儲存測試事件中輸入名稱，然後選擇儲存。

您也可以選擇編輯 JSON 並複製原始 JSON，並將其儲存在 CloudFront 之外您自己的檔案中。

建立事件物件 (選項 3)

使用文字編輯器建立事件物件。將檔案儲存於電腦可以連線的目錄中。

請確定您遵循這些準則：

- 省略 distributionDomainName、distributionId 和 requestId 欄位。
- 標頭、Cookie 和查詢字串的名稱必須為小寫。

以這種方式建立事件物件的一個選項是使用視覺化編輯器建立範例。您可以確定範例格式正確。然後您可以複製原始 JSON 並將其貼到文字編輯器中並儲存檔案。

如需有關事件結構的詳細資訊，請參閱 [事件結構](#)。

測試函數

您可以在 CloudFront 主控台中或使用 AWS Command Line Interface (AWS CLI) 測試函數。

Console

若要測試函數

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。
2. 選擇您要測試的函數。
3. 選擇測試標籤。
4. 確定已顯示正確的事件。若要從目前顯示的事件切換，請在選取測試事件欄位中選擇另一個事件。
5. 選擇測試函數。控制台顯示函數的輸出，包括函數日誌和運算使用率。

CLI

您可以使用 `aws cloudfront test-function` 命令來測試函數。

若要測試函數

1. 開啟命令列視窗。
2. 從包含指定檔案的目錄執行下列命令。

此範例會使用 `fileb://` 標記法來傳入事件物件檔案。它還會包括換行符號，讓命令更易於讀取。

```
aws cloudfront test-function \  
  --name MaxAge \  
  --if-match ETVABCEXAMPLE \  
  --event-object fileb://event-maxage-test01.json \  
  --stage DEVELOPMENT
```

備註

- 您可以透過其名稱和 ETag (在 `if-match` 參數中) 引用該函數。您可以依照事件物件在檔案系統中的位置來參照事件物件。
- 此階段可以是 `DEVELOPMENT` 或 `LIVE`。

如果命令成功執行，您會看到如下所示的輸出。

```
TestResult:
  ComputeUtilization: '21'
  FunctionErrorMessage: ''
  FunctionExecutionLogs: []
  FunctionOutput: '{"response":{"headers":{"cloudfront-functions":
{"value":"generated-by-CloudFront-Functions"},"location":{"value":"https://
aws.amazon.com/cloudfront/"}},"statusDescription":"Found","cookies":
{},"statusCode":302}}'
  FunctionSummary:
    FunctionConfig:
      Comment: MaxAge function
      Runtime: cloudfront-js-2.0
      KeyValueStoreAssociations= \
      {Quantity=1, \
      Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]]} \
    FunctionMetadata:
      CreatedTime: '2021-04-18T20:38:56.915000+00:00'
      FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge
      LastModifiedTime: '2023-17-20T10:38:57.057000+00:00'
      Stage: DEVELOPMENT
      Name: MaxAge
      Status: UNPUBLISHED
```

備註

- FunctionExecutionLogs 包含函數在 console.log() 語句中撰寫的日誌行清單 (如果有的話)。
- ComputeUtilization 包含執行函數的相關資訊。請參閱 [the section called “了解運算利用率”](#)。
- FunctionOutput 包含該函數返回的事件物件。

了解運算利用率

運用利用率是指執行函數所花費的時間，以所允許時間上限的百分比表示。例如，35 的值表示該函數以所允許時間上限的 35% 完成。

如果某個函數持續時間，超過允許時間上限，則 CloudFront 會將該函數限流。下列清單說明，根據運算利用率的值，函數限流的可能性。

運算利用率值：

- 1 — 50—函數遠低於允許時間上限，應不會受到限流。
- 51 — 70—函數接近允許時間上限。考慮將函數程式碼最佳化。
- 71 — 100—函數非常接近或超過允許時間上限。如果您將此函數與分佈產生關聯，CloudFront 很有可能會將此函數限流。

更新函數

您隨時都可以更新。這些變更只會對 DEVELOPMENT 階段中的函數版本進行。若要將更新從 DEVELOPMENT 階段複製到 LIVE，您必須[發佈函數](#)。

您可以在 CloudFront 主控台中或使用 AWS Command Line Interface (AWS CLI) 更新函數的程式碼。

Console

更新函數程式碼

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。

選擇要更新的函數。

2. 選擇編輯，然後進行您想要的變更：

- 更新詳細資訊區段中的任何欄位。
- 變更或移除關聯的鍵值存放區。如需鍵值存放區的詳細資訊，請參閱 [the section called “CloudFront KeyValueCollection”](#)。
- 變更函數程式碼。選擇建置索引標籤，進行變更，然後選擇儲存變更，儲存對程式碼的變更。

CLI

若要更新函式程式碼

1. 開啟命令列視窗。
2. 執行下列命令。

此範例會使用 `fileb://` 標記法來傳入檔案。它還會包括換行符號，讓命令更易於讀取。

```
aws cloudfront update-function \  
  --name MaxAge \  
  --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-  
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":  
[{"KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-store/  
a1b2c3d4-5678-90ab-cdef-EXAMPLE1111"}]}' \  
  --function-code fileb://function-max-age-v1.js \  
  --if-match ETVABCEXAMPLE
```

 備註

- 您可以透過名稱和 ETag (在 `if-match` 參數中) 來識別函數。請確定您使用目前的 ETag。您可以從 [DescribeFunction](#) API 作業取得此值。
- 即使您不想要變更，您也必須包含 `function-code`。
- 要小心 `function-config`。您應該傳遞您想要在組態中保留的所有內容。具體而言，請依下列方式處理鍵值存放區：
 - 若要保留現有的鍵值存放區關聯 (如果有的話)，請指定現有存放區的名稱。
 - 若要變更關聯，請指定新鍵值存放區的名稱。
 - 若要移除關聯，請省略 `KeyValueStoreAssociations` 參數。

如果命令成功執行，您會看到如下所示的輸出。

```
ETag: ETVXYZEXAMPLE  
FunctionSummary:  
  FunctionConfig:  
    Comment: Max Age 2 years \  
    Runtime: cloudfront-js-2.0 \  
    KeyValueStoreAssociations= \  
    
```

```
{Quantity=1, \
  Items=[{KeyValueStoreARN='arn:aws:cloudfront::111122223333:key-value-
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'}]]} \
FunctionMetadata: \
  CreatedTime: '2021-04-18T20:38:56.915000+00:00' \
  FunctionARN: arn:aws:cloudfront::111122223333:function/MaxAge \
  LastModifiedTime: '2023-12-19T23:41:15.389000+00:00' \
  Stage: DEVELOPMENT \
  Name: MaxAge \
  Status: UNPUBLISHED
```

大多數資訊都是從請求中複製的。其他資訊會由 CloudFront 新增。

備註

- ETag – 每次修改鍵值存放區時，這個值都會變更。
- FunctionARN – CloudFront 函數的 ARN。
- Stage – 函數的階段 (LIVE 或 DEVELOPMENT)。
- Status – 函數的狀態 (PUBLISHED 或 UNPUBLISHED)。

發佈函數

您發佈函數時，這會將函數從 DEVELOPMENT 階段複製到 LIVE 階段。

如果沒有與函數相關聯的快取行為，發佈該函數可讓您將其與快取行為產生關聯。您只能將快取行為與 LIVE 階段中的函數產生關聯。

Important

- 建議您在發佈之前先[測試函數](#)。
- 發佈函數後，與該函數相關聯的所有快取行為會在分佈完成部署後立即開始使用新發佈的副本。

您可以在 CloudFront 主控台中或使用 AWS CLI 發佈函數。

Console

發佈函數

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。
2. 選擇要更新的函數。
3. 選擇發佈索引標籤，然後選擇發佈。如果您的函數已附加至一或多個快取行為，請選擇發佈並更新。
4. (選用) 若要檢視與函數相關聯的分佈，請選擇關聯的 CloudFront 分佈以展開該區段。

成功後，您會在頁面頂端看到一個橫幅，顯示已成功發佈####。您也可以選擇建置索引標籤，然後選擇即時以檢視函數程式碼的即時版本。

CLI

發佈函數

1. 開啟命令列視窗。
2. 執行下列 `aws cloudfront publish-function` 命令。在此範例中，提供分行符號以使範例更具可讀性。

```
aws cloudfront publish-function \  
  --name MaxAge \  
  --if-match ETVXYZEXAMPLE
```

如果命令成功執行，您會看到如下所示的輸出。

```
FunctionSummary:  
  FunctionConfig:  
    Comment: Max Age 2 years  
    Runtime: cloudfront-js-2.0  
  FunctionMetadata:  
    CreatedTime: '2021-04-18T21:24:21.314000+00:00'  
    FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction  
    LastModifiedTime: '2023-12-19T23:41:15.389000+00:00'  
    Stage: LIVE  
  Name: MaxAge  
  Status: UNASSOCIATED
```

將函數與分佈相關聯

若要將函數與分佈搭配使用，請將該函數與分佈中的一或多個快取行為建立關聯。您可以將函數與多個分佈中的多個快取行為相關聯。

您可以將函數與下列任何行為建立關聯：

- 現有的快取行為
- 現有分佈中新的新快取行為
- 新分佈中的新快取行為

將函數與快取行為建立關聯時，您必須選擇事件類型。事件類型決定 CloudFront 何時執行該函數。

您可以選擇下列事件類型：

- 檢視者請求 – 當 CloudFront 收到來自檢視者的請求時，該函數會執行。
- 檢視者回應 – 此函數會在 CloudFront 傳回檢視者回應之前執行。

您不能將面向原始伺服器的事件類型 (原始伺服器請求和原始伺服器回應) 與 CloudFront Functions 搭配使用。您可以改為使用 Lambda@Edge。如需更多詳細資訊，請參閱 [可以觸發 Lambda@Edge 函數的 CloudFront 事件](#)。

Note

在關聯函數之前，您必須將[其發佈](#)至 LIVE 階段。

您可以將函數與 CloudFront 主控台的分佈建立關聯，或與 AWS Command Line Interface (AWS CLI) 建立關聯。下列程序顯示如何將函數與現有的快取行為建立關聯。

Console

將函數與現有快取行為產生關聯

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。
2. 選擇您要關聯的函數。
3. 在函數頁面上，選擇發佈索引標籤。

4. 選擇發佈函數。
5. 選擇 Add association (建立關聯)。在出現的對話方塊中，選擇分佈、事件類型及/或快取行為。

在事件類型中，選擇您希望此函數執行的時間：

- 檢視器請求 – 在 CloudFront 每次收到請求時執行該函數。
- 檢視器回應 – 在 CloudFront 每次傳回回應時執行該函數。

6. 若要儲存組態，請選擇新增關聯。

CloudFront 會將分佈與函數建立關聯。等待幾分鐘，讓關聯的分佈完成部署。您可以在函數詳細資訊頁面上選擇檢視分佈來檢查進度。

CLI

將函數與現有快取行為產生關聯

1. 開啟命令列視窗。
2. 使用下列命令儲存分佈的組態，該分佈的快取行為將與函數產生關聯。此命令會將分佈組態儲存到名為 dist-config.yaml 的檔案中。若要使用此命令，請執行下列動作：
 - 將 *DistributionID* 取代為分佈的 ID。
 - 在一行上執行命令。在此範例中，提供分行符號以使範例更具可讀性。

```
aws cloudfront get-distribution-config \  
  --id DistributionID \  
  --output yaml > dist-config.yaml
```

命令成功時，AWS CLI 不會返回任何輸出。

3. 開啟您建立且命名為 dist-config.yaml 的檔案。編輯檔案以進行下列變更。
 - a. 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。
 - b. 在快取行為中，尋找名為 FunctionAssociations 的物件。更新此物件以新增函數關聯。如下的範例給出函數關聯的 YAML 語法。
 - 下列範例顯示檢視者請求事件物件 (觸發條件)。若要使用檢視者回應事件類型，請將 viewer-request 取代為 viewer-response。

- 將 `arn:aws:cloudfront::111122223333:function/ExampleFunction` 取代為與此快取行為相關聯之函數的 Amazon Resource Name (ARN)。要獲取函數 ARN，您可以使用 `aws cloudfront list-functions` 命令。

```
FunctionAssociations:
  Items:
    - EventType: viewer-request
      FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction
  Quantity: 1
```

- c. 進行這些變更後，請儲存檔案。
4. 使用以下命令更新分佈，同時新增函數關聯。若要使用此命令，請執行下列動作：
 - 將 `DistributionID` 取代為分佈的 ID。
 - 在一行上執行命令。在此範例中，提供分行符號以使範例更具可讀性。

```
aws cloudfront update-distribution \
  --id DistributionID \
  --cli-input-yaml file://dist-config.yaml
```

如果命令成功執行，您會看到如下所示的輸出，其中描述剛使用函數關聯更新的分佈。為便於閱讀，對如下的範例輸出進行了截斷。

```
Distribution:
  ARN: arn:aws:cloudfront::111122223333:distribution/EBEDLT3BGRBBW
  ... truncated ...
DistributionConfig:
  ... truncated ...
DefaultCacheBehavior:
  ... truncated ...
FunctionAssociations:
  Items:
    - EventType: viewer-request
      FunctionARN: arn:aws:cloudfront::111122223333:function/ExampleFunction
  Quantity: 1
  ... truncated ...
DomainName: d1111111abcdef8.cloudfront.net
Id: EDFDVBD6EXAMPLE
LastModifiedTime: '2021-04-19T22:39:09.158000+00:00'
```

```
Status: InProgress
ETag: E2VJGGQEG1JT8S
```

重新部署分佈時，此分佈的 Status 變更為 InProgress。一旦新的分佈組態到達 CloudFront 邊緣節點，該邊緣節點就會開始使用關聯的函數。當分佈完全部署時，Status 會變更回 Deployed。這表示相關聯的 CloudFront 函數已在全球所有 CloudFront 邊緣節點中上線。通常這需要幾分鐘的時間。

Amazon CloudFront KeyValueStore

CloudFront KeyValueStore 是安全、全域、低延遲的鍵值資料儲存，可從 [CloudFront Functions](#) 內進行讀取存取，從而在 CloudFront 邊緣節點啟用進階可自訂邏輯。

使用 CloudFront KeyValueStore，您可以更新函數程式碼，並且彼此獨立地更新與函數相關聯的資料。這種分離簡化了函數程式碼，並且可以輕鬆更新資料，而無需部署程式碼變更。

Note

若要使用 CloudFront KeyValueStore，您的 CloudFront 函數必須使用 [JavaScript 執行時期 2.0](#)。

使用鍵值對的一般程序如下：

- 建立鍵值存放區，並填入一組鍵值對。您可以將鍵值存放區新增至 Amazon S3 儲存貯體，或手動輸入。
- 將鍵值存放區與您的 CloudFront 函數建立關聯。
- 在函數程式碼中，使用鍵的名稱來擷取與鍵關聯的值或評估鍵是否存在。如需有關在函數程式碼中使用鍵值對的詳細資訊，以及協助程式方法的相關資訊，請參閱 [the section called “鍵值存放區的協助程式方法”](#)。

使用案例

您可以針對下列範例使用鍵值對：

- URL 重寫或重新導向：鍵值對可以保留重寫 URL 或重新導向 URL。
- A/B 測試及功能旗標：您可指派一定百分比的流量至特定版本網站，即可建立執行實驗的函數。
- 存取授權：您可實作存取控制，根據您定義的條件和鍵值存放區中儲存的資料來允許或拒絕請求。

支援的值格式

您可採用下列任何一種格式儲存鍵值對中的值：

- String
- 位元組編碼字串
- JSON

安全

CloudFront 函數及其所有鍵值存放區資料均可安全地處理，如下所示：

- 您呼叫 [CloudFront KeyValueStore](#) API 操作時，CloudFront 會在靜態和傳輸期間 (讀取或寫入鍵值存放區時) 將每個鍵值存放區加密。
- 執行函數時，CloudFront 會解密 CloudFront 邊緣節點記憶體中的每個鍵值對。

若要開始使用 CloudFront KeyValueStore，請參閱下列主題。

主題

- [使用鍵值存放區](#)
- [使用鍵值資料](#)
- 如需開始使用 CloudFront KeyValueStore 的詳細資訊，請參閱[簡介 Amazon CloudFront KeyValueStore](#) AWS 部落格文章。

使用鍵值存放區

您必須建立鍵值存放區，以保留要在 CloudFront Functions 中使用的鍵值對。

建立鍵值存放區並新增鍵值對之後，您可以在 CloudFront 函數程式碼中使用這些鍵值。

若要開始使用，請參閱下列主題：

主題

- [建立鍵值存放區](#)
- [將鍵值存放區與函數建立關聯。](#)

- [更新鍵值存放區](#)
- [獲取鍵值存放區參考資料](#)
- [刪除鍵值存放區](#)
- [鍵值對的檔案格式](#)

Note

JavaScript 執行時期 2.0 包含一些協助程式方法，可用來處理函數程式碼中的鍵值。如需更多詳細資訊，請參閱 [the section called “鍵值存放區的協助程式方法”](#)。

建立鍵值存放區

您可以同時建立一個鍵值存放區及其鍵值對。您也可以建立一個空的鍵值存放區，並於稍後在其中新增鍵值對。

Note

如果您從 Amazon S3 儲存貯體指定資料來源，就必須擁有該儲存貯體的 `s3:GetObject` 和 `s3:GetBucketLocation` 許可權限。如果您沒有這些許可權限，CloudFront 就無法成功建立您的鍵值存放區。

請決定是否要在建立鍵值存放區的同時新增鍵值對。您可以使用 CloudFront 主控台、CloudFront API 或 AWS SDK 匯入鍵值對。不過您只能在最初建立鍵值存放區時匯入鍵值對檔案。

若要建立鍵值對檔案，請參閱 [鍵值對的檔案格式](#)。

Console

建立鍵值存放區

1. 登入 AWS 管理主控台 並在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 的 CloudFront 主控台中開啟 Functions (函數) 頁面。
2. 選擇 KeyValueStores 索引標籤，然後選擇建立 KeyValueStore。
3. 輸入鍵值存放區的名稱，以及選擇性的描述。

4. 完成 S3 URI：

- 如果您有鍵值對檔案，請輸入您存放該檔案的 Amazon S3 儲存貯體路徑。
- 如果您打算手動輸入鍵值對，請將此欄位保留空白。

5. 選擇建立。鍵值存放區現已建立。

此時會顯示新鍵值存放區的詳細資訊頁面。頁面上的資訊包括鍵值存放區的 ID 和 ARN。

- ID 是您 AWS 帳戶中唯一的隨機字元字串。
- ARN 具有以下語法：

AWS **##**:key-value-store/ **##### ID**

6. 請查看鍵值對區段。如果您匯入檔案，此區段會顯示一些鍵值對。您可以執行下列作業：

- 如果您匯入了檔案，也可以手動新增更多值。
- 如果您沒有從 Amazon S3 儲存貯體匯入檔案，而且現在想要新增鍵值對，則可以完成下個步驟。
- 您可以略過此步驟，稍後再新增鍵值對。

7. 立即新增鍵值對：

- a. 選擇新增鍵值對。
- b. 選擇新增配對，然後輸入名稱和值。若要新增更多配對，請重複此步驟。
- c. 完成後，選擇儲存變更以儲存鍵值存放區中的所有鍵值對。在出現的對話方塊中選擇完成。

8. 如果您要立即將鍵值存放區與函數相關聯，請完成關聯函數區段。如需詳細資訊，請參閱 [???](#) 或 [???](#)。

您也可以稍後從此鍵值存放區詳細資訊頁面或從函數詳細資訊頁面與函數建立關聯。

AWS CLI

建立鍵值存放區

- 您可執行下列命令來建立鍵值存放區，並從 Amazon S3 儲存貯體匯入鍵值對。

```
aws cloudfront create-key-value-store \  
  --name=keyvaluestore1 \  
  --comment="This is my key value store file" \  
  --s3-bucket=...
```

```
--import-source=SourceType=S3,SourceARN=arn:aws:s3:::amzn-s3-demo-bucket1/kvs-input.json
```

回應：

```
{
  "ETag": "ETVABCEXAMPLE",
  "Location": "https://cloudfront.amazonaws.com/2020-05-31/key-value-store/arn:aws:cloudfront::123456789012:key-value-store/8aa76c93-3198-462c-aaf6-example",
  "KeyValueStore": {
    "Name": "keyvaluestore1",
    "Id": "8aa76c93-3198-462c-aaf6-example",
    "Comment": "This is my key value store file",
    "ARN": "arn:aws:cloudfront::123456789012:key-value-store/8aa76c93-3198-462c-aaf6-example",
    "Status": "PROVISIONING",
    "LastModifiedTime": "2024-08-06T22:19:10.813000+00:00"
  }
}
```

API

建立鍵值存放區

1. 使用 [CloudFront CreateKeyValueStore](#) 操作。該操作需要幾個參數：
 - 鍵值存放區的 name。
 - 包含註解的 comment 參數。
 - 可讓您從存放在 Amazon S3 儲存貯體的檔案匯入鍵值對的 import-source 參數。您只能在第一次建立鍵值存放區時從檔案匯入。如需檔案結構的相關資訊，請參閱 [the section called “鍵值對的檔案格式”](#)。

操作回應包含下列資訊：

- 請求中傳遞的值，包括您指派的名稱。
- 建立時間等資料。
- ETag (例如 ETVABCEXAMPLE) 為包含鍵值存放區名稱的 ARN (例如 arn:aws:cloudfront::123456789012:key-value-store/keyvaluestore1)。

您將使用 ETag、ARN 和名稱的某些組合，以程式設計方式處理鍵值存放區。

鍵值存放區狀態

您建立鍵值存放區時，資料存放區可能有下列狀態值。

值	說明
佈建中	鍵值存放區已建立，CloudFront 正在處理您指定的資料來源。
備妥	鍵值存放區已建立，CloudFront 已成功處理您指定的資料來源。
匯入失敗	CloudFront 無法處理您指定的資料來源。如果您的檔案格式無效或超過大小限制，就會出現此狀態。如需更多詳細資訊，請參閱 鍵值對的檔案格式 。

將鍵值存放區與函數建立關聯。

您建立鍵值存放區之後，就可以更新函數，將其與鍵值存放區建立關聯。您必須建立此關聯，才能在該函數中使用該存放區中的鍵值對。適用的規定如下：

- 一個函數只能有一個鍵值存放區
- 同一個鍵值存放區可以與多個函數建立關聯

Console

將鍵值存放區與函數建立關聯

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 登入 CloudFront 主控台，並選擇函數頁面。
2. 選擇函數名稱。
3. 前往關聯 KeyValueStore 區段，然後選擇關聯現有的 KeyValueStore。
4. 選取函數中包含鍵值對的鍵值存放區，然後選擇關聯 KeyValueStore。

CloudFront 會立即將存放區與該函數建立關聯。您無需儲存函數。

5. 若要指定不同的鍵值存放區，請選擇更新關聯的 KeyValueStore，選擇另一個鍵值存放區名稱，然後選擇關聯 KeyValueStore。

如需更多詳細資訊，請參閱 [the section called “更新函數”](#)。

AWS CLI

將鍵值存放區與函數建立關聯

- 您可執行下列命令來更新 *MaxAge* 函數，並與鍵值存放區資源建立關聯。

```
aws cloudfront update-function \  
  --name MaxAge \  
  --function-config '{"Comment":"Max Age 2 years","Runtime":"cloudfront-  
js-2.0","KeyValueStoreAssociations":{"Quantity":1,"Items":  
[{"KeyValueStoreARN":"arn:aws:cloudfront::123456789012:key-value-  
store/8aa76c93-3198-462c-aaf6-example"}]}}' \  
  --function-code fileb://function-max-age-v1.js \  
  --if-match ETVABCEXAMPLE
```

- 若要将鍵值存放區與函數建立關聯，請指定 `KeyValueStoreAssociations` 參數和鍵值存放區 ARN。
- 若要變更關聯，請指定另一個鍵值存放區 ARN。
- 若要移除關聯，請移除 `KeyValueStoreAssociations` 參數。

如需更多詳細資訊，請參閱 [the section called “更新函數”](#)。

API

將鍵值存放區與函數建立關聯

- 使用 [UpdateFunction](#) API 操作。如需更多詳細資訊，請參閱 [the section called “更新函數”](#)。

備註

- 如果您修改鍵值存放區而不變更鍵值對，或者只修改鍵值存放區中的鍵值對，則不需要再次關聯鍵值存放區。您也不需要重新發佈函數。

不過我們建議您測試函數，以確認函數是否如預期般運作。如需更多詳細資訊，請參閱 [測試函數](#)。

- 您可以檢視使用特定鍵值存放區的所有函數。在 CloudFront 主控台上，選擇鍵值存放區詳細資訊頁面。

更新鍵值存放區

您更新鍵值存放區時可以變更鍵值對，或變更鍵值存放區和函數之間的關聯。

Console

更新鍵值存放區

1. 登入 AWS 管理主控台 並在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 的 CloudFront 主控台中開啟 Functions (函數) 頁面。
2. 選擇 KeyValueStores 索引標籤。
3. 選擇您想要更新的鍵值存放區。
 - 若要更新鍵值對，請選擇鍵值對區段中的編輯。您可以新增或刪除任何鍵值對。您也可以變更現有鍵值對的值。完成時，請選擇儲存變更。
 - 若要更新此鍵值存放區的關聯，請選擇前往函數。如需更多詳細資訊，請參閱 [the section called “將鍵值存放區與函數建立關聯。”](#)。

AWS CLI

更新鍵值存放區

1. 變更鍵值對：您可以新增更多鍵值對、刪除一或多個鍵值對，也可以變更現有鍵值對的值。如需更多詳細資訊，請參閱 [使用鍵值資料](#)。
2. 變更鍵值存放區的函數關聯：若要更新鍵值存放區的關聯函數，請參閱 [將鍵值存放區與函數建立關聯](#)。

Tip

您將需要鍵值存放區的 ARN。如需更多詳細資訊，請參閱 [the section called “獲取鍵值存放區參考資料”](#)。

API

更新鍵值存放區

1. 變更鍵值對：您可以新增更多鍵值對、刪除一或多個鍵值對，也可以變更現有鍵值對的值。如需更多詳細資訊，請參閱 [使用鍵值資料](#)。
2. 變更鍵值存放區的函數關聯：若要更新鍵值存放區的關聯函數，請使用 [UpdateFunction](#) API 操作。如需更多詳細資訊，請參閱 [the section called “更新函數”](#)。

Tip

您將需要鍵值存放區的 ARN。如需更多詳細資訊，請參閱 [the section called “獲取鍵值存放區參考資料”](#)。

獲取鍵值存放區參考資料

為了以程式設計方式使用鍵值存放區，您需要 ETag 和鍵值存放區的名稱。

若要取得這兩個值，您可以使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API。

AWS CLI

取得鍵值存放區參考資料

1. 若要傳回鍵值存放區清單，請執行下列命令。尋找您要變更的鍵值存放區名稱。

```
aws cloudfront list-key-value-stores
```

2. 從回應中尋找您想要的鍵值存放區名稱。

回應：

```
{
  "KeyValueStoreList": {
    "Items": [
      {
        "Name": "keyvaluestore3",
        "Id": "37435e19-c205-4271-9e5c-example3",
        "ARN": "arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-example3",
        "Status": "READY",
```

```

        "LastModifiedTime": "2024-05-08T14:50:18.876000+00:00"
      },
      {
        "Name": "keyvaluestore2",
        "Id": "47970d59-6408-474d-b850-example2",
        "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/47970d59-6408-474d-b850-example2",
        "Status": "READY",
        "LastModifiedTime": "2024-05-30T21:06:22.113000+00:00"
      },
      {
        "Name": "keyvaluestore1",
        "Id": "8aa76c93-3198-462c-aaf6-example",
        "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example",
        "Status": "READY",
        "LastModifiedTime": "2024-08-06T22:19:30.510000+00:00"
      }
    ]
  }
}

```

3. 執行下列命令，以傳回指定鍵值存放區的 ETag。

```
aws cloudfront describe-key-value-store \
  --name=keyvaluestore1
```

回應：

```

{
  "ETag": "E3UN6WX5RR02AG",
  "KeyValueStore": {
    "Name": "keyvaluestore1",
    "Id": "8aa76c93-3198-462c-aaf6-example",
    "Comment": "This is an example KVS",
    "ARN": "arn:aws:cloudfront::123456789012:key-value-
store/8aa76c93-3198-462c-aaf6-example",
    "Status": "READY",
    "LastModifiedTime": "2024-08-06T22:19:30.510000+00:00"
  }
}

```

API

取得鍵值存放區參考資料

1. 使用 [CloudFront ListKeyValueStores](#) API 操作傳回鍵值存放區清單。尋找您想要變更的鍵值存放區名稱。
2. 使用 [CloudFront DescribeKeyValueStore](#) API 操作，並指定您從上一個步驟傳回的鍵值存放區名稱。

回應包括 UUID、鍵值存放區的 ARN 以及鍵值存放區的 ETag。

- ETag (例如 E3UN6WX5RR02AG)
- UUID 為 128 位元 (例如 8aa76c93-3198-462c-aaf6-example)
- ARN 包含 AWS 帳戶 數字、常數 key-value-store 和 UUID，如下列範例所示：

```
arn:aws:cloudfront::123456789012:key-value-store/8aa76c93-3198-462c-aaf6-example
```

如需 DescribeKeyValueStore 操作的詳細資訊，請參閱 [the section called “關於 CloudFront KeyValueStore”](#)。

刪除鍵值存放區

您可以使用 Amazon CloudFront 主控台或 API 來刪除鍵值存放區。

Console

刪除鍵值存放區

1. 登入 AWS 管理主控台 並在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 的 CloudFront 主控台中開啟 Functions (函數) 頁面。
2. 選擇函數名稱。
3. 在關聯的 KeyValueStore 區段下，驗證鍵值存放區是否與函數相關聯。如果是，請選擇取消關聯 KeyValueStore 以移除關聯，然後選擇移除關聯。
4. 在導覽面板中選擇函數頁面，然後選擇 KeyValueStores 索引標籤。
5. 選取您要刪除的鍵值存放區，然後選擇刪除。

AWS CLI

刪除鍵值存放區

1. 獲取 ETag 和鍵值存放區的名稱。如需更多詳細資訊，請參閱 [the section called “獲取鍵值存放區參考資料”](#)。
2. 驗證鍵值存放區是否與函數相關聯。如果是，將移除關聯。如需這些步驟的詳細資訊，請參閱 [???](#)。
3. 如果您已取得鍵值存放區的名稱和 ETag，且鍵值存放區不再與函數相關聯，就可以將其刪除。

請執行下列命令來刪除指定的鍵值存放區。

```
aws cloudfront delete-key-value-store \  
  --name=keyvaluestore1 \  
  --if-match=E3UN6WX5RR02AG
```

API

刪除鍵值存放區

1. 獲取 ETag 和鍵值存放區的名稱。如需更多詳細資訊，請參閱 [the section called “獲取鍵值存放區參考資料”](#)。
2. 驗證鍵值存放區是否與函數相關聯。如果是，將移除關聯。如需這些步驟的詳細資訊，請參閱 [???](#)。
3. 若要刪除鍵值存放區，請使用 CloudFront [DeleteKeyValueStore](#) API 操作。

鍵值對的檔案格式

建立 UTF-8 編碼的檔案時，請使用下列 JSON 格式：

```
{  
  "data": [  
    {  
      "key": "key1",  
      "value": "value"  
    },  
    {  
      "key": "key2",
```

```
    "value": "value"  
  }  
]  
}
```

您的檔案不能包含重複的鍵。如果您在 Amazon S3 儲存貯體中指定了無效檔案，您可以更新檔案以移除任何重複項目，然後再次嘗試建立您的鍵值存放區。

如需更多詳細資訊，請參閱 [建立鍵值存放區](#)。

Note

資料來源及其鍵值組的檔案具有以下限制：

- 檔案大小 - 5 MB
- 鍵大小 — 512 個字元
- 鍵大小 — 1024 個字元

使用鍵值資料

本節說明如何將鍵值對新增至現有的鍵值存放區。若要在最初建立鍵值存放區時包含鍵值對，請參閱 [the section called “建立鍵值存放區”](#)。

主題

- [使用鍵值對 \(主控台\)](#)
- [關於 CloudFront KeyValueCollection](#)
- [使用鍵值對 \(AWS CLI\)](#)
- [使用鍵值對 \(API\)](#)

使用鍵值對 (主控台)

您可使用 CloudFront 主控台處理鍵值對。

使用鍵值對

1. 登入 AWS 管理主控台 並在位於 <https://console.aws.amazon.com/cloudfront/v4/home#/functions> 的 CloudFront 主控台中開啟 Functions (函數) 頁面。

2. 選擇 KeyValueStores 索引標籤。
3. 選擇您想要變更的鍵值存放區。
4. 在鍵值對區段中選擇編輯。
5. 您可以新增鍵值對、刪除鍵值對，或變更現有鍵值對的值。
6. 完成時，請選擇儲存變更。

關於 CloudFront KeyValueStore

Tip

CloudFront KeyValueStore API 是使用 Signature 第 4A 版 (SigV4A) 進行身分驗證的全域服務。使用臨時憑證搭配 SigV4A 需要第 2 版工作階段記號。如需更多詳細資訊，請參閱 [搭配 CloudFront KeyValueStore API 使用臨時憑證](#)。

如果您使用 AWS Command Line Interface (AWS CLI) 或自己的程式碼來呼叫 CloudFront KeyValueStore API，請參閱下列各節。

您使用鍵值存放區及其鍵值對時，您呼叫的服務取決於您的使用案例：

- 若要使用現有鍵值存放區的鍵值對，請使用 CloudFront KeyValueStore 服務。
- 若要在最初建立鍵值存放區時在其中包含一些鍵值對，請使用 CloudFront 服務。

CloudFront API 和 CloudFront KeyValueStore API 都具有 DescribeKeyValueStore 操作。您會依據不同原因呼叫這兩項服務。若要瞭解其中差異，請參閱下表。

	CloudFront DescribeKeyValueStore API	CloudFront KeyValueStore DescribeKeyValueStore API
關於鍵值存放區的資料	傳回資料，例如狀態和上次修改鍵值存放區的日期。	傳回有關儲存資源內容的資料：存放區中的鍵值對，以及內容的大小。
識別鍵值存放區的資料	傳回鍵值存放區的 ETag、UUID 和 ARN。	傳回鍵值存放區的 ETag 和 ARN。

 備註

- 每個 DescribeKeyValueStore 操作會傳回不同的 ETag。ETags 不可互換。
- 您呼叫 API 操作以完成動作時，必須從適當的 API 指定 ETag。例如在 CloudFront KeyValueStore 的 [DeleteKey](#) 操作中，您要指定從 CloudFront KeyValueStore [DescribeKeyValueStore](#) 操作傳回的 ETag。
- 您使用 CloudFront KeyValueStore 調用 CloudFront Functions 時，鍵值存放區的值在調用函數期間不會更新或變更。更新會在函數調用之間進行處理。

使用鍵值對 (AWS CLI)

您可以針對 CloudFront KeyValueStore 執行下列 AWS Command Line Interface 命令。

內容

- [列出鍵值對](#)
- [取得鍵值對](#)
- [描述鍵值存放區](#)
- [建立鍵值對](#)
- [刪除鍵值對](#)
- [更新鍵值對](#)

列出鍵值對

若要列出鍵值存放區中的鍵值對，請執行下列命令。

```
aws cloudfront-keyvaluestore list-keys \  
    --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-  
example
```

回應：

```
{  
  "Items": [  
    {  
      "Key": "key1",
```



```
        "Value": "value1"
      }
    ]
  }
}
```

取得鍵值對

若要取得鍵值存放區中的鍵值對，請執行下列命令。

```
aws cloudfront-keyvaluestore get-key \
  --key=key1 \
  --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-example
```

回應：

```
{
  "Key": "key1",
  "Value": "value1",
  "ItemCount": 1,
  "TotalSizeInBytes": 11
}
```

描述鍵值存放區

若要描述鍵值存放區，請執行下列命令。

```
aws cloudfront-keyvaluestore describe-key-value-store \
  --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-example
```

回應：

```
{
  "ETag": "KV1F83G8C2AR07P",
  "ItemCount": 1,
  "TotalSizeInBytes": 11,
  "KvsARN": "arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-example",
  "Created": "2024-05-08T07:48:45.381000-07:00",
}
```

```
"LastModified": "2024-08-05T13:50:58.843000-07:00",  
"Status": "READY"  
}
```

建立鍵值對

若要在鍵值存放區建立鍵值對，請執行下列命令。

```
aws cloudfront-keyvaluestore put-key \  
  --if-match=KV1PA6795UKMFR9 \  
  --key=key2 \  
  --value=value2 \  
  --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-  
example
```

回應：

```
{  
  "ETag": "KV13V1IB3VIYZZH",  
  "ItemCount": 3,  
  "TotalSizeInBytes": 31  
}
```

刪除鍵值對

若要刪除鍵值對，請執行下列命令。

```
aws cloudfront-keyvaluestore delete-key \  
  --if-match=KV13V1IB3VIYZZH \  
  --key=key1 \  
  --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-  
example
```

輸出

```
{  
  "ETag": "KV1VC38T7YXB528",  
  "ItemCount": 2,  
  "TotalSizeInBytes": 22  
}
```

更新鍵值對

您可以使用 `update-keys` 命令來更新多個鍵值對。例如若要刪除現有鍵值對並建立另一個，請執行下列命令。

```
aws cloudfront-keyvaluestore update-keys \
  --if-match=KV2EUQ1WTGCTBG2 \
  --kvs-arn=arn:aws:cloudfront::123456789012:key-value-store/37435e19-c205-4271-9e5c-
example \
  --deletes ' [{"Key": "key2"} ]' \
  --puts ' [{"Key": "key3", "Value": "value3"} ]'
```

回應：

```
{
  "ETag": "KV3AEGXETSR30VB",
  "ItemCount": 3,
  "TotalSizeInBytes": 28
}
```

使用鍵值對 (API)

您可遵循本節說明以程式設計方式使用鍵值對。

內容

- [獲取鍵值存放區參考資料](#)
- [變更鍵值存放區中的鍵值對](#)
- [CloudFront KeyValueStore 的範例程式碼](#)

獲取鍵值存放區參考資料

您使用 CloudFront KeyValueStore API 呼叫寫入操作時，需要指定鍵值存放區的 ARN 和 ETag。若要取得此資料，請依下列步驟執行：

取得鍵值存放區參考資料

1. 使用 [CloudFront ListKeyValueStores](#) API 操作取得鍵值存放區清單。尋找您想要變更的鍵值存放區。
2. 使用 [CloudFrontKeyValueStore DescribeKeyValueStore API 操作](#)，並指定上一個步驟的鍵值存放區。

回應包括鍵值存放區的 ARN 和 ETag。

- ARN 包含 AWS 帳戶 數字、常數 key-value-store 和 UUID，如下列範例所示：

```
arn:aws:cloudfront::123456789012:key-value-store/a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111
```

- ETag 如下列範例所示：

```
ETVABCEXAMPLE2
```

變更鍵值存放區中的鍵值對

您可以指定包含所需更新鍵值對的鍵值存放區。

請參閱下列 CloudFront KeyValueStore API 操作：

- [CloudFrontKeyValueStore DeleteKey](#)：刪除鍵值對
- [CloudFrontKeyValueStore GetKey](#)：傳回鍵值對
- [CloudFrontKeyValueStore ListKeys](#)：傳回鍵值對清單
- [CloudFrontKeyValueStore PutKey](#)：您可以執行下列任務：
 - 指定新的鍵名稱及值，以在鍵值存放區中建立新的鍵值對。
 - 指定現有鍵名稱及新的鍵值，以在現有鍵值對中設定不同的值。
- [CloudFrontKeyValueStore UpdateKeys](#)：您可以在一個「全有或全無」操作中執行下列其中一個或多個動作：
 - 刪除一或多個鍵值對
 - 建立一或多個新的鍵值對
 - 在一個或多個現有鍵值對中設定不同的值

CloudFront KeyValueStore 的範例程式碼

Example

下列程式碼示範如何呼叫鍵值存放區的 DescribeKeyValueStore API 操作。

```
const {  
  CloudFrontKeyValueStoreClient,
```

```
DescribeKeyValueStoreCommand,  
} = require("@aws-sdk/client-cloudfront-keyvaluestore");  
  
require("@aws-sdk/signature-v4-crt");  
  
(async () => {  
  try {  
    const client = new CloudFrontKeyValueStoreClient({  
      region: "us-east-1"  
    });  
    const input = {  
      KvsARN: "arn:aws:cloudfront::123456789012:key-value-store/a1b2c3d4-5678-90ab-  
cdef-EXAMPLE11111",  
    };  
    const command = new DescribeKeyValueStoreCommand(input);  
  
    const response = await client.send(command);  
  } catch (e) {  
    console.log(e);  
  }  
})();
```

使用 CloudFront Connection Functions 自訂

CloudFront Connection Functions 可讓您為 mTLS 憑證驗證和自訂身分驗證邏輯撰寫輕量型 JavaScript 函數。您的連線函數會在建立 mTLS 連線期間執行，以驗證用戶端憑證、實作裝置特定的身分驗證規則，以及處理憑證撤銷案例。Connection Functions 執行時間環境提供低於毫秒的啟動時間、立即擴展以處理每秒數百萬個連線，而且非常安全。Connection Functions 是 CloudFront 的原生功能，這表示您可以完全在 CloudFront 內建置、測試和部署程式碼。

當您將連線函數與啟用 mTLS 的 CloudFront 分佈建立關聯時，CloudFront 會在 CloudFront 節點攔截 TLS 連線請求，並將憑證資訊傳遞給您的函數。當發生下列事件時，您可以叫用連線函數：

- 在建立 TLS 連線（連線請求）期間 - 用於交互 TLS (mTLS) 連線

如需 Connection Functions 的詳細資訊，請參閱下列主題。

主題

- [概觀和工作流程](#)
- [組態和限制](#)

- [建立 CloudFront 連線函數以進行交互 TLS（檢視器）驗證](#)
- [寫入 CloudFront 連線函數程式碼以進行交互 TLS（檢視器）驗證](#)
- [在部署之前測試 CloudFront 連線函數](#)
- [將連線函數與分佈建立關聯](#)
- [使用 CloudFront Functions 和 KeyValueStore 實作交互 TLS（檢視器）的憑證撤銷](#)

概觀和工作流程

CloudFront Connection Functions 是一種特殊類型的 CloudFront Functions，當用戶端嘗試建立 mTLS 連線時，會在 TLS 交握期間執行。您的 Connection Function 可以存取用戶端憑證資訊、mTLS 組態參數、憑證撤銷檢查結果，以及用戶端 IP 地址。

在 CloudFront 執行標準憑證驗證（信任鏈、過期、簽章驗證）後叫用連線函數，但即使憑證撤銷檢查失敗，也可以執行。這可讓您實作自訂邏輯，以處理撤銷的憑證或新增其他驗證條件。

建立並發佈 Connection Function 之後，請務必為啟用 mTLS 的分佈新增連線請求事件類型的關聯。這可讓函數在每次用戶端嘗試與 CloudFront 建立 mTLS 連線時執行。

CloudFront Connection Functions 遵循兩階段生命週期，可讓您在將函數部署到生產環境之前開發和測試函數。此工作流程可確保連線函數在影響即時流量之前正常運作。

主題

- [函數階段](#)
- [開發工作流程](#)
- [與其他函數類型的差異](#)

函數階段

連線函數存在於兩個階段之一：

- 開發 – 可修改、測試和更新此階段中的函數。使用此階段來寫入和偵錯函數程式碼。
- LIVE – 此階段中的函數為唯讀並處理生產流量。您無法直接修改 LIVE 階段中的函數。

當您建立新的連線函數時，它會在 DEVELOPMENT 階段啟動。測試和驗證之後，您會發佈函數以將其移至 LIVE 階段。

開發工作流程

遵循此工作流程來開發和部署連線函數：

1. 建立 – 在 DEVELOPMENT 階段使用初始程式碼和組態建立新的連線函數。
2. 測試 – 使用測試功能，在部署之前使用範例連線事件來驗證您的函數。
3. 更新 – 根據測試結果，視需要修改函數程式碼和組態。
4. 發佈 – 準備生產時，發佈函數以將其從 DEVELOPMENT 移至 LIVE 階段。
5. 關聯 – 將已發佈的函數與已啟用 mTLS 的分佈建立關聯，以處理即時連線。

若要變更 LIVE 函數，您必須更新 DEVELOPMENT 版本並再次發佈。這會在 LIVE 階段中建立新的版本。

與其他函數類型的差異

連線函數與檢視器請求和檢視器回應函數有幾個重要方面不同：

- 在進行任何 HTTP 處理之前，連線函數會在 mTLS 交握之後執行
- 連線函數可存取 TLS 憑證資訊，而不是 HTTP 請求/回應資料
- 連線函數只能允許或拒絕連線，不能修改 HTTP 資料
- 僅針對新的 TLS 連線叫用連線函數，而非針對連線重複使用
- mTLS 不支援 TLS 工作階段恢復，以確保憑證驗證在每個連線上發生
- 除了標準檢視器請求和檢視器回應函數之外，連線函數也會執行
- 您可以在分佈層級建立連線函數的關聯，而不是在快取行為層級建立關聯。
- 連線函數僅支援 JavaScript 執行期 2.0。

組態和限制

CloudFront Connection Functions 具有特定的組態需求和服務限制，因為其在 TLS 連線驗證和邊緣運算效能方面的專業角色。

主題

- [函數程式碼需求](#)
- [服務限制](#)
- [函數篩選選項](#)

函數程式碼需求

連線函數需要處理 TLS 連線事件的 JavaScript 程式碼。函數程式碼必須：

- 以 JavaScript 撰寫
- 處理連線事件並做出允許/拒絕決策
- 在時間限制內完成執行
- 處理憑證和連線驗證邏輯

服務限制

連線函數受到下列限制：

- 函數大小 – 函數程式碼和組態的大小有限
- 執行時間 – 函數對於 TLS 連線處理有嚴格的執行時間限制
- 關聯限制 – 每個分佈只能有一個關聯的連線函數
- 階段限制 – 只有 LIVE 階段函數可以與分佈建立關聯

函數篩選選項

列出連線函數時，您可以使用下列篩選條件：

- 階段篩選條件 – 依 DEVELOPMENT 或 LIVE 階段篩選
- 關聯篩選條件 – 依分佈 ID 或鍵值存放區 ID 關聯進行篩選

這些篩選條件可協助您整理和管理不同環境和使用案例的連線函數。

建立 CloudFront 連線函數以進行交互 TLS（檢視器）驗證

您會分兩個階段建立 CloudFront Connection Function：

1. 建立函數程式碼做為 JavaScript。您可以從 CloudFront 主控台使用預設範例或自行撰寫。如需詳細資訊，請參閱下列主題：
 - 寫入 mTLS 驗證的 CloudFront 連線函數程式碼
 - CloudFront Connection Function 事件結構和回應格式
 - 連線函數程式碼範例

2. 使用 CloudFront 建立連線函數並包含您的程式碼。程式碼存在於函數內部 (而不是引用形式)。

主題

- [CloudFront 主控台](#)
- [AWS CLI](#)

CloudFront 主控台

建立連線函數

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇 Create function (建立函數)。
3. 輸入 中唯一的函數名稱 AWS 帳戶，選擇連線函數做為函數類型，然後選擇繼續。
4. 新連線函數的詳細資訊頁面隨即出現。

Note

連線函數僅支援 JavaScript 執行期 2.0。若要在函數中使用 CloudFront Connection Function KeyValueStore 整合，您必須使用此執行時間版本。

5. 在函數程式碼區段中，選擇建置索引標籤，然後輸入您的連線函數程式碼。建置索引標籤中包含的範例程式碼說明 Connection Function 程式碼的基本語法。
6. 選擇儲存變更。
7. 如果連線函數程式碼使用 KeyValueStore 進行憑證撤銷檢查或裝置驗證，您必須建立 KeyValueStore 的關聯。

您可以在第一次建立函數時關聯 KeyValueStore。或者，您可以稍後透過關聯連線函數來建立關聯。

若要立即建立 KeyValueStore 的關聯，請遵循下列步驟：

- 前往關聯 KeyValueStore 區段，然後選擇關聯現有的 KeyValueStore。
- 選取包含連線函數憑證資料的 KeyValueStore，然後選擇關聯 KeyValueStore。

CloudFront 會立即將存放區與該函數建立關聯。您無需儲存函數。

AWS CLI

如果您使用 AWS CLI，通常會先在 檔案中建立連線函數程式碼，然後使用 建立函數 AWS CLI。

建立連線函數

1. 在 檔案中建立連線函數程式碼，並將其存放在電腦可連線的目錄中。
2. 執行命令，如範例所示。此範例會使用 `fileb://` 標記法來傳入檔案。它還會包括換行符號，讓命令更易於讀取。

```
aws cloudfront create-connection-function \  
  --name CertificateValidator \  
  --connection-function-config '{  
    "Comment":"Device certificate validation",  
    "Runtime":"cloudfront-js-2.0",  
    "KeyValueStoreAssociations":{  
      "Quantity":1,  
      "Items":[{  
        "KeyValueStoreARN":"arn:aws:cloudfront::111122223333:key-value-  
store/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"  
      }]  
    }  
  }' \  
  --connection-function-code fileb://certificate-validator.js
```

Note

- 執行期 – 連線函數僅支援 JavaScript 執行期 2.0 (cloudfront-js-2.0)。
- KeyValueStoreAssociations – 如果您的連線函數使用 KeyValueStore 進行憑證驗證，您可以在第一次建立函數時關聯 KeyValueStore。或者，您可以稍後使用 `update-connection-function` 建立關聯。數量一律為 1，因為每個連線函數只能有一個與其相關聯的 KeyValueStore。

3. 如果命令成功執行，您會看到如下所示的輸出。

```
ETag: ETVABCEXAMPLE  
ConnectionFunctionSummary:  
  ConnectionFunctionConfig:  
    Comment: Device certificate validation  
    Runtime: cloudfront-js-2.0
```

```
KeyValueStoreAssociations:
  Quantity: 1
  Items:
    - KeyValueStoreARN: arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
ConnectionFunctionMetadata:
  CreatedTime: '2024-09-04T16:32:54.292000+00:00'
  ConnectionFunctionARN: arn:aws:cloudfront::111122223333:connection-function/
CertificateValidator
  LastModifiedTime: '2024-09-04T16:32:54.292000+00:00'
  Stage: DEVELOPMENT
  Name: CertificateValidator
  Status: UNPUBLISHED
Location: https://cloudfront.amazonaws.com/2020-05-31/connection-function/
arn:aws:cloudfront::connection-function/CertificateValidator
```

大多數資訊都是從請求中複製的。其他資訊會由 CloudFront 新增。

Note

- ETag – 每次修改連線函數時，此值都會變更。您需要此值才能更新或發佈函數。
- 階段 – 新的連線函數會在 DEVELOPMENT 階段啟動。您必須先發佈函數，才能將其移至 LIVE 階段，再將其與分佈建立關聯。
- 狀態 – 函數狀態為未發佈，直到您將其發佈至 LIVE 階段為止。

寫入 CloudFront 連線函數程式碼以進行交互 TLS（檢視器）驗證

CloudFront Connection Functions 可讓您撰寫輕量型 JavaScript 函數，以進行 mTLS 憑證驗證和自訂身分驗證邏輯。您的 Connection Function 程式碼可以驗證用戶端憑證、實作裝置特定的身分驗證規則、處理憑證撤銷案例，以及針對全球 CloudFront 節點的 TLS 連線做出允許/拒絕決策。

連線函數提供一種強大的方法，使用您自己的商業邏輯來擴展 CloudFront 的內建憑證驗證。與處理 HTTP 資料的檢視器請求和檢視器回應函數不同，Connection Functions 會在 TLS 層運作，並可存取憑證資訊、用戶端 IP 地址和 TLS 連線詳細資訊。這使得它們非常適合用於實作零信任安全模型、裝置身分驗證系統和超出標準 PKI 驗證的自訂憑證驗證政策。

您的 Connection Function 程式碼會在安全、隔離且啟動時間低於毫秒的環境中執行，並且可以擴展以處理每秒數百萬個連線。執行時間已針對憑證驗證工作負載進行最佳化，並提供與 CloudFront

KeyValueStore 的內建整合，以進行即時資料查詢操作，實現複雜的身分驗證案例，例如憑證撤銷清單檢查和裝置允許清單驗證。

為了協助您撰寫有效的連線函數程式碼，請參閱下列主題。如需完整的程式碼範例和step-by-step教學課程，請參閱本指南中的教學課程章節，並探索 CloudFront 主控台中提供的連線函數範例。

主題

- [CloudFront Connection Function 使用案例和用途](#)
- [CloudFront Connection Function 事件結構和回應格式](#)
- [CloudFront Connection Functions JavaScript 執行期功能](#)
- [CloudFront Connection Function 協助程式方法和 APIs](#)
- [CloudFront Connection Function KeyValueStore 整合](#)
- [使用非同步並等待](#)
- [連線函數程式碼範例](#)

CloudFront Connection Function 使用案例和用途

撰寫 CloudFront Connection Function 之前，請仔細判斷您需要實作的憑證驗證或身分驗證邏輯類型。連線函數是專為需要超出標準 PKI 憑證檢查自訂驗證的特定使用案例而設計。了解您的使用案例可協助您設計符合您安全需求的高效程式碼，同時維持最佳效能。

常見的連線函數使用案例包括：

- 憑證撤銷處理 – 實作自訂政策來處理撤銷的憑證，包括憑證輪換的寬限期、內部裝置的信任網路例外狀況，或撤銷的憑證可能需要暫時存取的緊急存取案例。
- 選用的 mTLS 支援 – 使用不同的身分驗證政策處理 mTLS 和非 mTLS 連線，可讓您為支援憑證的用戶端提供增強的安全性，同時保持與舊版用戶端的相容性。
- 以 IP 為基礎的身分驗證 – 結合憑證驗證與用戶端 IP 地址檢查，以提高安全性，例如限制來自特定地理區域、公司網路或已知惡意 IP 範圍的存取。
- 多租戶憑證驗證 – 實作租戶特定的驗證規則，其中根據用戶端憑證發行者或主體屬性，套用不同的憑證授權單位或驗證條件。
- 以時間為基礎的存取控制 – 強制執行以時間為基礎的限制，其中憑證僅在特定時段、維護時段或商業期間內有效，即使憑證本身尚未過期。

在 CloudFront 執行標準憑證驗證（信任鏈驗證、過期檢查和簽章驗證）之後，但在建立 TLS 連線之前執行連線函數。此時間可讓您靈活地新增自訂驗證條件，同時受益於 CloudFront 的內建憑證驗證。您的函數會收到標準驗證的結果，並可以根據標準和自訂條件，做出是否允許或拒絕連線的明智決策。

設計連線函數時，請考慮驗證邏輯的效能影響。函數具有 5 毫秒的執行限制，因此應針對速度最佳化複雜的操作。使用 KeyValueCollection 進行快速資料查詢，而非複雜的計算，並建構驗證邏輯以針對無效的憑證快速失敗。

CloudFront Connection Function 事件結構和回應格式

CloudFront Connection Functions 會收到與檢視器請求和檢視器回應函數不同的事件結構。連線函數會收到憑證和連線資訊，可用來做出身分驗證決策，而不是 HTTP 請求/回應資料。

主題

- [Connection Functions 的事件結構](#)
- [Connection Functions 回應格式](#)

Connection Functions 的事件結構

Connection Functions 會收到事件物件，其中包含憑證和連線資訊。函數的事件結構如下所示：

```
{
  "clientCertificate": {
    "certificates": {
      "leaf": {
        "serialNumber": "string",
        "issuer": "string",
        "subject": "string",
        "validity": {
          "notBefore": "string",
          "notAfter": "string",
        },
        "sha256Fingerprint": "string"
      }
    }
  },
  "clientIp": "string",
  "endpoint": "string",
  "distributionId": "string",
  "connectionId": "string"
```

```
}
```

以下是事件物件結構的範例：

```
{
  "clientCertificate": {
    "certificates": {
      "leaf": {
        "serialNumber": "00:9e:2a:af:16:56:e5:47:25:7d:2e:38:c3:f9:9d:57:fa",
        "issuer": "C=US, O=Ram, OU=Edge, ST=WA, CN=mTLS-CA, L=Snoqualmie",
        "subject": "C=US, O=Ram, OU=Edge, ST=WA, CN=mTLS-CA, L=Snoqualmie",
        "validity": {
          "notBefore": "2025-09-10T23:43:10Z",
          "notAfter": "2055-09-11T00:43:02Z"
        },
        "sha256Fingerprint": "_w6bJ7a0AlG0j7NUhJxTfsfee-0Ng_xop3_PTgTJpqqs="
      }
    }
  },
  "clientIp": "127.0.0.1",
  "endpoint": "d3lch071jze0cb.cloudfront.net",
  "distributionId": "E1NXS4MQZH501R",
  "connectionId": "NpvTe1925xfj24a67sPQr7ae42BIq03FGhJJKfrQYWZcWZFp96SIIg=="
}
```

Connection Functions 回應格式

您的 Connection Function 必須傳回回應物件，指出是否允許或拒絕連線。使用協助程式方法來做出連線決策：

```
function connectionHandler(connection) {
  // Helper methods to allow or deny connections
  if (/* some logic to determine if function should allow connection */) {
    connection.allow();
  } else {
    connection.deny();
  }
}
```

與檢視器請求和檢視器回應函數不同，連線函數無法修改 HTTP 請求或回應。他們只能允許或拒絕 TLS 連線。

CloudFront Connection Functions JavaScript 執行期功能

CloudFront Connection Functions 使用 CloudFront Functions JavaScript 執行期 2.0，可提供專為憑證驗證工作負載最佳化的安全且高效能環境。執行時間的設計是為了在幾毫秒內開始，並處理 CloudFront 全球邊緣網路中數百萬個並行執行。

執行期環境包含完整的 JavaScript 語言支援：

- ECMAScript 2020 (ES11) 支援 – 現代 JavaScript 功能，包括用於處理大型憑證序號的選用鏈結 (?.)、nullish coalescing (??) 和 BigInt
- 內建物件 – 物件、陣列、JSON、數學和日期等標準 JavaScript 物件
- 主控台記錄 – 使用 `console.log()` 進行偵錯和監控憑證驗證決策。日誌可在測試期間即時提供，有助於疑難排解開發中的驗證邏輯
- KeyValueStore 整合 – 原生存取 CloudFront KeyValueStore 以進行超快速的資料查詢操作，啟用即時憑證撤銷檢查、裝置允許清單驗證和租戶特定組態擷取

連線函數已針對憑證驗證案例的高效能進行最佳化。執行時間會自動處理記憶體管理、垃圾收集和資源清理，以確保數百萬個並行連線的效能一致。所有操作的設計都具有決定性和快速性，KeyValueStore 查詢通常在微秒內完成。

執行時間環境在函數執行之間完全隔離，確保不同用戶端連線之間的資料不會洩漏。每個函數執行一開始都是清除狀態，而且無法存取先前的執行結果或來自其他連線的用戶端資料。

CloudFront Connection Function 協助程式方法和 APIs

CloudFront Connection Functions 提供專門的協助程式方法，旨在簡化憑證驗證決策並增強可觀測性。這些方法針對連線驗證工作流程進行最佳化，並與 CloudFront 的連線記錄和監控系統無縫整合。

- `connection.allow()` – 允許 TLS 連線繼續。此方法會通知 CloudFront 完成 TLS 交握，並允許用戶端建立連線。當憑證驗證通過且滿足任何自訂身分驗證邏輯時，請使用此選項
- `connection.deny()` – 拒絕 TLS 連線並終止交握。此方法會立即關閉連線，並防止任何 HTTP 流量流動。用戶端將收到 TLS 連線錯誤。將此用於無效的憑證、失敗的身分驗證或政策違規
- `connection.logCustomData()` – 將自訂資料新增至連線日誌（最多 800 個位元組的 UTF-8 文字）。此方法可讓您在 CloudFront 連線日誌中包含驗證結果、憑證詳細資訊或決策理由，以進行安全性監控、合規稽核和故障診斷

這些方法提供簡潔的宣告式界面，用於進行連線決策，並記錄監控和偵錯的相關資訊。允許/拒絕模式可確保函數的意圖明確，且 CloudFront 可以根據您的決定最佳化連線處理。CloudFront 連線日誌會立即提供自訂記錄資料，並可搭配日誌分析工具使用，以進行安全性監控和操作洞察。

在函數完成之前，請務必呼叫 `connection.allow()` 或 `connection.deny()`。如果不呼叫任何方法，CloudFront 預設會拒絕連線做為安全預防措施。

CloudFront Connection Function KeyValueType 整合

CloudFront Connection Functions 可以使用 CloudFront KeyValueType 對憑證驗證案例執行超快速的資料查詢。KeyValueType 對 Connection Functions 特別強大，因為它可在所有 CloudFront 節點提供全域、最終一致的資料存取，以及微秒的查詢時間。這使得它非常適合用於維護 TLS 交握期間需要存取的憑證撤銷清單、裝置允許清單、租戶組態和其他驗證資料。

KeyValueType 整合專為高效能連線驗證工作流程而設計：

- `kvsHandle.exists(key)` – 檢查 KeyValueType 中是否存在金鑰，而不擷取值。這是二進位驗證案例最有效的方法，例如憑證撤銷檢查，您只需知道憑證序號是否在撤銷清單中
- `kvsHandle.get(key)` – 針對更複雜的驗證案例，從 KeyValueType 擷取值。當您需要存取與憑證或裝置識別符相關聯的組態資料、驗證規則或中繼資料時，請使用此選項

KeyValueType 操作是非同步的，必須搭配非同步/等待語法使用。KeyValueType 的總大小限制為 10MB，並支援最多 1,000 萬個鍵值對。KeyValueType 資料最終在所有節點保持一致，更新通常會在幾秒鐘內傳播。

為了獲得最佳效能，請建構您的 KeyValueType 金鑰，將查詢操作降至最低。使用憑證序號做為簡易撤銷檢查的金鑰，或建立合併多 CA 環境發行者雜湊和序號的複合金鑰。設計資料結構時，請考慮金鑰複雜性與 KeyValueType 容量之間的權衡。

使用非同步並等待

連線函數支援使用非同步/等待語法的非同步操作，這對於使用 KeyValueType 操作或其他非同步任務至關重要。非同步/等待模式可確保函數在做出連線決策之前等待 KeyValueType 查詢完成，同時維持 TLS 交握處理所需的高效能特性。

適當的非同步/等待用量對於連線函數至關重要，因為 KeyValueType 操作雖然速度非常快，但仍是需要跨 CloudFront 分散式基礎設施協調的網路操作。執行時間會自動處理承諾解析度，並確保您的函數在 5 毫秒的執行限制內完成。

Example：使用 KeyValueStore 的非同步連線函數

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
  const kvsHandle = cf.kvs();

  // Async operation to check KeyValueStore for certificate revocation
  const isRevoked = await
    kvsHandle.exists(connection.clientCertificate.certificates.leaf.serialNumber);

  if (isRevoked) {
    // Log the revocation decision with certificate details
    connection.logCustomData(`REVOKED_CERT:
${connection.clientCertificate.certificates.leaf.serialNumber}:
${connection.clientCertificate.certificates.leaf.issuer}`);
    console.log(`Denying connection for revoked certificate:
${connection.clientCertificate.certificates.leaf.serialNumber}`);
    return connection.deny();
  }

  // Log successful validation for monitoring
  connection.logCustomData(`VALID_CERT:
${connection.clientCertificate.certificates.leaf.serialNumber}`);
  console.log(`Allowing connection for valid certificate:
${connection.clientCertificate.certificates.leaf.serialNumber}`);
  return connection.allow();
}
```

呼叫 KeyValueStore 方法或其他非同步操作時，一律使用非同步/等待。Connection Function 執行時間會自動處理 promise 解析，並確保在 TLS 交握處理的嚴格時間限制內執行流程正確。避免使用 .then() 或回呼模式，因為非同步/等待可在連線函數環境中提供更明確的錯誤處理和更好的效能。

設計非同步連線函數時，請建構程式碼以盡可能減少 KeyValueStore 操作的數量，並儘早在驗證邏輯中執行這些操作。這可確保最大效能，並降低高流量期間逾時問題的風險。請考慮批次處理相關的驗證檢查，並針對您的使用案例使用最有效率的 KeyValueStore 方法 (exists() vs get())。

連線函數程式碼範例

下列範例示範適用於不同驗證案例的常見連線函數模式。使用這些範例做為您自己的連線函數實作的起點。

Example：裝置憑證驗證

此範例會驗證 IoT 裝置、遊戲主控台和其他用戶端身分驗證案例的裝置序號和憑證主旨欄位：

```
async function connectionHandler(connection) {
  // Custom validation: check device serial number format
  const serialNumber = connection.clientCertificate.certificates.leaf.serialNumber;
  if (!serialNumber.startsWith("DEV")) {
    connection.logCustomData(`INVALID_SERIAL:${serialNumber}`);
    return connection.deny();
  }

  // Validate certificate subject contains required organizational unit
  const subject = connection.clientCertificate.certificates.leaf.subject;
  if (!subject.includes("OU=AuthorizedDevices")) {
    connection.logCustomData(`INVALID_OU:${subject}`);
    return connection.deny();
  }

  // Allow connection for valid devices
  connection.logCustomData(`VALID_DEVICE:${serialNumber}`);
  return connection.allow();
}
```

此函數會在標準憑證驗證之外執行多個驗證檢查，包括裝置序號格式和組織單位驗證。

Example：選用具有混合身分驗證的 mTLS

此範例使用不同的身分驗證政策處理 mTLS 和非 mTLS 連線：

```
async function connectionHandler(connection) {
  if (connection.clientCertificate) {
    // mTLS connection - enhanced validation for certificate holders
    const subject = connection.clientCertificate.certificates.leaf.subject;
    connection.logCustomData(`MTLS_SUCCESS:${subject}:${connection.clientIp}`);
    console.log(`mTLS connection from: ${subject}`);
    return connection.allow();
  } else {
    // Non-mTLS connection - apply IP-based restrictions
    const clientIp = connection.clientIp;

    // Only allow non-mTLS from specific IP ranges
    if (clientIp.startsWith("203.0.113.") || clientIp.startsWith("198.51.100.")) {
```

```
        connection.logCustomData(`NON_MTLS_ALLOWED:${clientIp}`);
        console.log(`Non-mTLS connection allowed from: ${clientIp}`);
        return connection.allow();
    }

    connection.logCustomData(`NON_MTLS_DENIED:${clientIp}`);
    return connection.deny();
}
}
```

此函數為具有憑證的用戶端提供增強的安全性，同時保持與信任 IP 範圍內舊版用戶端的相容性。

在部署之前測試 CloudFront 連線函數

您可以使用 TestConnectionFunction API 操作，在 DEVELOPMENT 階段測試 CloudFront Connection Functions。TestConnectionFunction 測試可讓您在發佈至 LIVE 階段之前，使用範例連線事件來驗證函數邏輯。

主題

- [測試程序](#)
- [測試結果](#)
- [連線測試物件](#)

測試程序

若要測試連線函數：

1. 在 DEVELOPMENT 階段建立連線函數
2. 準備代表 TLS 連線事件的測試連線物件
3. 使用 TestConnectionFunction API 操作，以測試資料執行您的函數
4. 檢閱測試結果，包括函數輸出、執行日誌和任何錯誤訊息
5. 視需要更新您的函數程式碼，並重複測試程序

測試結果

當您測試連線函數時，結果包括：

- 函數摘要 – 有關已測試函數的中繼資料

- 運算使用率 – 顯示資源用量的效能指標
- 執行日誌 – 來自函數的主控制台輸出，包括任何記錄陳述式
- 函數輸出 – 函數傳回的結果
- 錯誤訊息 – 執行期間發生的任何執行時間錯誤或例外狀況

連線測試物件

連線測試物件是二進位 Blob（最多 40KB），代表函數將處理的 TLS 連線事件。此物件包含您的函數用來進行身分驗證決策的憑證和連線資訊。

Note

連線測試物件的特定結構和格式是由 CloudFront Connection Functions 執行時間所定義。如需為您的使用案例建立適當測試物件 AWS 支援的詳細資訊，請參閱 CloudFront Functions 文件或聯絡人。

建立連線函數之後，您可以：

- 測試函數 – 使用主控台或 CLI 中的測試功能，透過範例連線事件來驗證您的函數。如需詳細資訊，請參閱連線函數測試。
- 更新函數 – 視需要修改函數程式碼和組態。DEVELOPMENT 階段中的連線函數可以隨時更新。
- 發佈函數 – 準備生產時，發佈函數以將其從 DEVELOPMENT 移至 LIVE 階段。如需詳細資訊，請參閱關聯連線函數。
- 與分佈建立關聯 – 將發佈的函數與啟用 mTLS 的分佈建立關聯，以處理即時連線。如需詳細資訊，請參閱關聯連線函數。

將連線函數與分佈建立關聯

將 Connection Function 發佈至 LIVE 階段之後，您必須將其與啟用 mTLS 的分佈建立關聯，才能處理即時連線。連線函數與分佈層級相關聯，不同於與快取行為相關聯的檢視器請求和檢視器回應函數。

主題

- [關聯需求](#)
- [使用篩選條件組織函數](#)

• [部署考量](#)

關聯需求

若要將連線函數與分佈建立關聯：

- 函數必須處於 LIVE 階段
- 分佈必須已啟用 mTLS
- 分佈必須設定有效的信任存放區
- 每個分佈只能關聯一個連線函數

使用篩選條件組織函數

CloudFront 提供篩選功能，可協助您組織和管理連線函數：

- 分佈 ID 篩選條件 – 尋找與特定分佈相關聯的函數
- 鍵值存放區篩選條件 – 尋找使用特定鍵值存放區進行資料查詢的函數
- 階段篩選條件 – 列出 DEVELOPMENT 或 LIVE 階段中的函數

在不同分佈或開發環境中管理多個連線函數時，請使用這些篩選條件。

部署考量

部署連線函數時，請考慮下列因素：

- 全域部署 – 連線函數部署到全球所有 CloudFront 節點，這可能需要幾分鐘的時間
- 版本管理 – 每個發佈的版本都會建立新的 LIVE 函數來取代先前的版本
- 回復策略 – 透過保留函數程式碼的先前工作版本來規劃回復
- 在生產環境中測試 – 考慮在預備和生產環境中使用個別的分佈

使用 CloudFront Functions 和 KeyValueTypeStore 實作交互 TLS（檢視器）的憑證撤銷

您可以使用 CloudFront Connection Functions 搭配 KeyValueTypeStore 來實作憑證撤銷檢查。這可讓您維護撤銷的憑證序號清單，並在 TLS 交握期間針對此清單檢查用戶端憑證。

若要實作憑證撤銷，您需要這些元件：

- 使用檢視器 mTLS 設定的分佈
- 包含已撤銷憑證序號的 KeyValueStore
- 查詢 KeyValueStore 以檢查憑證狀態的連線函數

當用戶端連線時，CloudFront 會根據信任存放區驗證憑證，然後執行連線函數。您的函數會根據 KeyValueStore 檢查憑證序號，並允許或拒絕連線。

主題

- [步驟 1：為已撤銷的憑證建立 KeyValueStore](#)
- [步驟 2：建立撤銷連線函數](#)
- [步驟 3：測試您的撤銷函數](#)
- [步驟 4：將函數與分佈建立關聯](#)
- [進階撤銷案例](#)

步驟 1：為已撤銷的憑證建立 KeyValueStore

建立 KeyValueStore，以存放連線函數在 mTLS 連線期間可以檢查的撤銷憑證序號。

首先，以 JSON 格式準備您撤銷的憑證序號：

```
{
  "data": [
    {
      "key": "ABC123DEF456",
      "value": ""
    },
    {
      "key": "789XYZ012GHI",
      "value": ""
    }
  ]
}
```

將此 JSON 檔案上傳至 S3 儲存貯體，然後建立 KeyValueStore：

```
aws s3 cp revoked-serials.json s3://your-bucket-name/revoked-serials.json
```

```
aws cloudfront create-key-value-store \  
  --name revoked-serials-kvs \  
  --import-source '{  
    "SourceType": "S3",  
    "SourceARN": "arn:aws:s3:::your-bucket-name/revoked-serials.json"  
  }'
```

等待 KeyValueStore 完成佈建。使用 檢查狀態：

```
aws cloudfront get-key-value-store --name "revoked-serials-kvs"
```

步驟 2：建立撤銷連線函數

建立連線函數，根據 KeyValueStore 檢查憑證序號，以判斷憑證是否遭到撤銷。

建立 Connection Function，根據 KeyValueStore 檢查憑證序號：

```
aws cloudfront create-connection-function \  
  --name "revocation-control" \  
  --connection-function-config file://connection-function-config.json \  
  --connection-function-code file://connection-function-code.txt
```

組態檔案會指定 KeyValueStore 關聯：

```
{  
  "Runtime": "cloudfront-js-2.0",  
  "Comment": "A function that implements revocation control via KVS",  
  "KeyValueStoreAssociations": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "KeyValueStoreArn": "arn:aws:cloudfront::account-id:key-value-store/kvs-id"  
      }  
    ]  
  }  
}
```

Connection Function 程式碼會檢查 KeyValueStore 是否有已撤銷的憑證：

```
import cf from 'cloudfront';
```

```
async function connectionHandler(connection) {
  const kvsHandle = cf.kvs();

  // Get parsed client serial number from client certificate
  const clientSerialNumber = connection.clientCertInfo.serialNumber;

  // Check KVS to see if serial number exists as a key
  const serialNumberExistsInKvs = await kvsHandle.exists(clientSerialNumber);

  // Deny connection if serial number exists in KVS
  if (serialNumberExistsInKvs) {
    console.log("Connection denied - certificate revoked");
    return connection.deny();
  }

  // Allow connections that don't exist in kvs
  console.log("Connection allowed");
  return connection.allow();
}
```

步驟 3：測試您的撤銷函數

使用 CloudFront 主控台以範例憑證測試連線函數。導覽至 主控台 中的連線函數，並使用測試索引標籤。

使用範例憑證進行測試

1. 將 PEM 格式的範例憑證貼入測試界面
2. 選擇性地指定用戶端 IP 地址以測試 IP 型邏輯
3. 選擇測試函數以查看執行結果
4. 檢閱執行日誌以驗證您的函數邏輯

使用有效和已撤銷的憑證進行測試，以確保您的函數正確處理這兩個案例。執行日誌會顯示 console.log 輸出和函數執行期間發生的任何錯誤。

步驟 4：將函數與分佈建立關聯

發佈連線函數後，請將其與啟用 mTLS 的分佈建立關聯，以啟用憑證撤銷檢查。

您可以從分佈設定頁面或從連線函數的關聯分佈資料表建立函數的關聯。導覽至您的分佈設定，捲動至檢視器交互驗證 (mTLS) 區段，選取連線函數，然後儲存變更。

進階撤銷案例

如需更複雜的憑證撤銷要求，請考慮下列其他組態：

主題

- [將憑證撤銷清單 \(CRL\) 轉換為 KeyValueStore 格式](#)
- [處理多個憑證授權機構](#)
- [將自訂資料新增至連線日誌](#)
- [管理 CRL 更新](#)
- [規劃 KeyValueStore 容量](#)

將憑證撤銷清單 (CRL) 轉換為 KeyValueStore 格式

如果您有憑證撤銷清單 (CRL) 檔案，您可以使用 OpenSSL 和 jq 將其轉換為 KeyValueStore JSON 格式：

將 CRL 轉換為 KeyValueStore 格式

從 CRL 檔案擷取序號：

```
openssl crl -text -noout -in rfc5280_CRL.crl | \
  awk '/Serial Number:/ {print $3}' | \
  cut -d'=' -f2 | \
  sed 's/./&:/g;s/:$//' >> serialnumbers.txt
```

將序號轉換為 KeyValueStore JSON 格式：

```
jq -R -s 'split("\n") | map(select(length > 0)) | {data: map({"key": ., "value": ""})}' \
  serialnumbers.txt >> serialnumbers_kvs.json
```

將格式化檔案上傳至 S3，並建立 KeyValueStore，如步驟 1 所述。

處理多個憑證授權機構

當您的 TrustStore 包含多個憑證授權機構 (CAs) 時，請在 KeyValueStore 金鑰中包含發行者資訊，以避免不同 CAs 的憑證之間發生可能具有相同序號的衝突。

對於多 CA 案例，請使用發行者的 SHA1 雜湊和序號的組合做為金鑰：

```
import cf from 'cloudfront';

async function connectionHandler(connection) {
  const kvsHandle = cf.kvs();
  const clientCert = connection.clientCertInfo;

  // Create composite key with issuer hash and serial number
  const issuer = clientCert.issuer.replace(/[^a-zA-Z0-9]/g, '').substring(0, 20);
  const serialno = clientCert.serialNumber;
  const compositeKey = `${issuer}_${serialno}`;

  const cert_revoked = await kvsHandle.exists(compositeKey);

  if (cert_revoked) {
    console.log(`Blocking revoked cert: ${serialno} from issuer: ${issuer}`);
    connection.deny();
  } else {
    connection.allow();
  }
}
```

Note

使用發行者識別符 + 序號會建立較長的金鑰，這可能會減少您可以存放在 KeyValueCollection 中的項目總數。

將自訂資料新增至連線日誌

連線函數可以使用 `logCustomData` 方法將自訂資料新增至 CloudFront 連線日誌。這可讓您在日誌中包含撤銷檢查結果、憑證資訊或其他相關資料。

```
async function connectionHandler(connection) {
  const kvsHandle = cf.kvs();
  const clientSerialNumber = connection.clientCertInfo.serialNumber;
  const serialNumberExistsInKvs = await kvsHandle.exists(clientSerialNumber);

  if (serialNumberExistsInKvs) {
    // Log revocation details to connection logs
    connection.logCustomData(`REVOKED:${clientSerialNumber}:DENIED`);
    console.log("Connection denied - certificate revoked");
    return connection.deny();
  }
}
```

```
}

// Log successful validation
connection.logCustomData(`VALID:${clientSerialNumber}:ALLOWED`);
console.log("Connection allowed");
return connection.allow();
}
```

自訂資料限制為 800 個位元組的有效 UTF-8 文字。如果您超過此限制，CloudFront 會將資料截斷到最近的有效 UTF-8 邊界。

Note

自訂資料記錄只有在為您的分佈啟用連線日誌時才有效。如果未設定連線日誌，logCustomData 方法是無操作。

管理 CRL 更新

憑證授權機構可以發出兩種類型的 CRLs：

- 完整 CRLs：包含所有已撤銷憑證的完整清單
- Delta CRLs：僅列出自上次完整 CRL 以來撤銷的憑證

如需完整的 CRL 更新，請使用更新的資料建立新的 KeyValueStore，並將連線函數關聯重新導向至新的 KeyValueStore。這種方法比計算差異和執行增量更新更簡單。

對於差異 CRL 更新，請使用 update-keys 命令將新的撤銷憑證新增至現有的 KeyValueStore：

```
aws cloudfront update-key-value-store \
  --name "revoked-serials-kvs" \
  --if-match "current-etag" \
  --put file://delta-revoked-serials.json
```

規劃 KeyValueStore 容量

KeyValueStore 的大小限制為 5 MB，並支援最多 1,000 萬個鍵值對。根據您的金鑰格式和資料大小來規劃撤銷清單容量：

- 僅限序號：簡單撤銷檢查的高效儲存

- 發行者識別符 + 序號：多 CA 環境的較長金鑰

對於大型撤銷清單，請考慮實作分層方法，在其中針對不同的憑證類別或期間維護個別的 KeyValueStores。

使用 Lambda@Edge 在邊緣自訂

Lambda@Edge 是 的延伸 AWS Lambda。Lambda@Edge 是運算服務，可讓您執行函數，以自訂 Amazon CloudFront 交付的內容。您可以在 Lambda 主控台內的 AWS 區域美國東部（維吉尼亞北部）撰寫 Node.js 或 Python 函數。

建立函數後，您可以使用 Lambda 主控台或 CloudFront 主控台新增觸發，讓函數在更接近檢視器 AWS 的位置執行，而無需佈建或管理伺服器。或者您可以使用 Lambda 和 CloudFront API 操作，以程式設計方式設定函數和觸發。

Lambda@Edge 會自動擴展，從每天幾個請求自動擴展到每秒數千個請求。在更接近瀏覽者而非原始伺服器 AWS 的位置處理請求，可大幅降低延遲並改善使用者體驗。

Note

gRPC 請求不支援 Lambda@Edge。如需更多詳細資訊，請參閱 [使用 gRPC 搭配 CloudFront 分佈](#)。

主題

- [Lambda@Edge 如何處理請求和回應](#)
- [使用 Lambda@Edge 的方式](#)
- [開始使用 Lambda@Edge 函數 \(主控台\)](#)
- [設定 Lambda@Edge 的 IAM 許可權限和角色](#)
- [撰寫及建立 Lambda@Edge 函數](#)
- [為 Lambda@Edge 函數新增觸發條件](#)
- [測試和偵錯 Lambda@Edge 函數](#)
- [刪除 Lambda@Edge 函數和複本](#)
- [Lambda@Edge 事件結構說明頁面](#)
- [使用請求和回應](#)

- [Lambda@Edge 範例函數](#)

Lambda@Edge 如何處理請求和回應

當您將 CloudFront 分佈與 Lambda@Edge 函數建立關聯時，CloudFront 會攔截在 CloudFront 節點的請求和回應。您可以在發生下列 CloudFront 事件時執行 Lambda 函數：

- 當 CloudFront 接收到來自檢視器的請求 (檢視器請求) 時
- 在 CloudFront 轉寄請求至來源 (原始請求) 之前
- 當 CloudFront 接收到來自來源的回應 (原始回應) 時
- 在 CloudFront 傳回回應給檢視器 (檢視器回應) 之前

如果您使用的是 AWS WAF，則會在套用任何 AWS WAF 規則後執行 Lambda@Edge 檢視器請求。

如需詳細資訊，請參閱[使用請求和回應](#)及[Lambda@Edge 事件結構說明頁面](#)。

使用 Lambda@Edge 的方式

您的 Amazon CloudFront 分佈有許多 Lambda@Edge 處理用途，例如下列範例：

- Lambda 函數可檢查 Cookie 和重新寫入 URL，讓使用者看到不同版本網頁的 A/B 測試。
- CloudFront 可以檢視 User-Agent 標頭來根據檢視器使用的裝置傳回不同的物件給檢視器，其中包含裝置相關資訊。例如，根據檢視器裝置的螢幕尺寸，CloudFront 可以傳回不同的圖片大小。同樣地，該函數可以考慮 Referer 標頭中的值，讓 CloudFront 傳回最低可用解析度圖片給機器人。
- 或者，您可以檢查 Cookie 的其他條件。例如，在銷售衣服的零售網站，如果您使用 Cookie 以顯示使用者選擇什麼顏色的夾克，該 Lambda 函數可以變更請求，讓 CloudFront 傳回所選顏色的夾克圖片。
- 發生 CloudFront 檢視器請求或原始伺服器請求事件時，Lambda 函數可產生 HTTP 回應。
- 函數可以檢查標頭或授權字符，在 CloudFront 轉送請求到您的原始伺服器之前，插入標頭來控制存取您的內容。
- Lambda 函數也可以讓網路呼叫外部資源，確認使用者登入資料，或擷取額外的內容以自訂回應。

如需包含範例程式碼的詳細資訊，請參閱 [Lambda@Edge 範例函數](#)。

如需有關在主控制台設定 Lambda@Edge 的詳細資訊，請參閱 [教學課程：建立基本 Lambda@Edge 函數 \(主控制台\)](#)。

開始使用 Lambda@Edge 函數 (主控台)

Lambda@Edge 可讓您使用 CloudFront 觸發條件來調用 Lambda 函數。當您將 CloudFront 分佈與 Lambda 函數建立關聯時，CloudFront 會在 CloudFront 節點[攔截請求和回應](#)，並執行該函數。Lambda 函數可以提高安全性，或是自訂與檢視器密切相關的資訊來提升效能。

下列清單提供基本概觀，協助瞭解如何透過 CloudFront 建立和使用 Lambda 函數。

概觀：透過 CloudFront 建立和使用 Lambda 函數

1. 在美國東部 (維吉尼亞北部) 區域建立 Lambda 函數。
2. 儲存並發佈有編號的函數版本。

如果您想要變更函數，您必須編輯美國東部 (維吉尼亞北部) 區域內 \$LATEST 版本的函數。然後，在您設定它來搭配 CloudFront 使用之前，請發佈新編號的版本。

3. 將函數與 CloudFront 分佈和快取行為建立關聯。然後，指定可以使函數執行的一或多個 CloudFront 事件 (觸發條件)。例如，您可以建立當 CloudFront 接收來自檢視器的請求時，促使函數執行的觸發條件。
4. 您建立觸發條件時，Lambda 會在全球各地的 AWS 位置建立函數複本。

Tip

如需更多詳細資訊，請參閱[如何建立和更新函數](#)、[事件結構](#)，以及[如何新增 CloudFront 觸發條件](#)。您也可以[在 Lambda@Edge 範例函數](#)中找到更多想法並取得程式碼範例。

如需逐步教學課程，請參閱下列主題：

主題

- [教學課程：建立基本 Lambda@Edge 函數 \(主控台\)](#)

教學課程：建立基本 Lambda@Edge 函數 (主控台)

本教學課程將為您示範如何開始使用 Lambda@Edge，協助您建立及設定在 CloudFront 中執行的 Node.js 函數範例。此項範例會在 CloudFront 檢索檔案時，將 HTTP 安全性標頭新增到回應。(這可以改善網站的安全性和隱私。)

本教學課程不需要您自己的網站。不過您建立自己的 Lambda@Edge 解決方案時，會依照類似的步驟進行，並從相同的選項中選擇。

主題

- [步驟 1：註冊 AWS 帳戶](#)
- [步驟 2：建立 CloudFront 分佈](#)
- [步驟 3：建立函數](#)
- [步驟 4：新增一個 CloudFront 觸發條件來執行該函數](#)
- [步驟 5：驗證函數正常執行](#)
- [步驟 6：排除問題](#)
- [步驟 7：清除範例資源](#)
- [相關資訊](#)

步驟 1：註冊 AWS 帳戶

若您尚未註冊，請先註冊 AWS 帳戶。如需詳細資訊，請參閱[註冊 AWS 帳戶](#)。

步驟 2：建立 CloudFront 分佈

在建立 Lambda@Edge 範例函數之前，您必須有一個可使用、包含提供內容之來源的 CloudFront 環境。

在此範例中，您會建立 CloudFront 分佈，此分佈是使用 Amazon S3 儲存貯體做為分佈的原始伺服器。如果您已有環境可使用，可以略過此步驟。

使用 Amazon S3 原始伺服器建立 CloudFront 分佈

1. 使用一兩個檔案來建立 Amazon S3 儲存貯體，例如範例內容適用的映像檔案。如需說明，請遵循[將您的內容上傳到 Amazon S3](#)中的步驟。請確定您有設定對應的許可，以授予儲存貯體中物件的公有讀取權限。
2. 請依照[建立 CloudFront Web 分佈](#)中的步驟來建立 CloudFront 分佈，並新增您的 S3 儲存貯體當做原始伺服器。如果您已有分佈，可以改為新增儲存貯體當做該分佈的來源。

Tip

請記下您的分佈 ID。稍後您在本教學課程中為函數新增 CloudFront 觸發條件時，您必須在下拉式清單中選擇分佈的 ID，例如 E653W22221KDDL。

步驟 3：建立函數

在這個步驟中，您將從 Lambda 主控台提供的藍圖範本開始建立 Lambda 函數。此函數會新增程式碼來更新您的 CloudFront 分佈中的安全標頭。

建立 Lambda 函式

1. 登入 AWS 管理主控台 並開啟位於 的 AWS Lambda 主控台<https://console.aws.amazon.com/lambda/>。

Important

請確定您位於美國 US-East-1 (維吉尼亞北部) AWS 區域 (us-east-1)。您必須位在此區域，才能建立 Lambda@Edge 函數。

2. 選擇 Create function (建立函數)。
3. 在建立函數頁面上，選擇使用藍圖，然後在搜尋欄位中輸入 **cloudfront** 以篩選 CloudFront 藍圖。

Note

CloudFront 藍圖僅適用於美國東部 1 (維吉尼亞北部) 區域 (us-east-1)。

4. 選擇修改 HTTP 回應標頭藍圖做為函數的範本。
5. 輸入以下有關函數的資訊：
 - 函數名稱：輸入函數的名稱。
 - 執行角色：選擇如何為函數設定許可權限。若要使用建議的基本 Lambda@Edge 許可政策範本，請選擇從 AWS 政策範本建立新角色。
 - 角色名稱：輸入政策範本建立的角色名稱。
 - 政策範本：Lambda 會自動新增基本 Lambda@Edge 許可權限政策範本，因為您選擇 CloudFront 藍圖做為您的函數基礎。此原則範本會新增執行角色許可，讓 CloudFront 能夠在全球的 CloudFront 位置為您執行 Lambda 函數。如需詳細資訊，請參閱[設定 Lambda@Edge 的 IAM 許可權限和角色](#)。
6. 請在頁面底部選擇建立函數。
7. 在顯示的部署至 Lambda@Edge 窗格中選擇取消。(在本教學課程中，您必須先修改函數程式碼，才能將函數部署至 Lambda@Edge。)

8. 向下捲動至頁面的程式碼來源區段。
9. 使用修改原始伺服器傳回之安全性標頭的函數，來取代範本程式碼。例如，您可能會使用與下列類似的程式碼：

```
'use strict';
export const handler = (event, context, callback) => {

    //Get contents of response
    const response = event.Records[0].cf.response;
    const headers = response.headers;

    //Set new headers
    headers['strict-transport-security'] = [{key: 'Strict-Transport-Security',
value: 'max-age= 63072000; includeSubdomains; preload'}];
    headers['content-security-policy'] = [{key: 'Content-Security-Policy', value:
"default-src 'none'; img-src 'self'; script-src 'self'; style-src 'self'; object-
src 'none'"}];
    headers['x-content-type-options'] = [{key: 'X-Content-Type-Options', value:
'nosniff'}];
    headers['x-frame-options'] = [{key: 'X-Frame-Options', value: 'DENY'}];
    headers['x-xss-protection'] = [{key: 'X-XSS-Protection', value: '1;
mode=block'}];
    headers['referrer-policy'] = [{key: 'Referrer-Policy', value: 'same-origin'}];

    //Return modified response
    callback(null, response);
};
```

10. 選擇檔案，然後選擇儲存以儲存更新後的程式碼。
11. 選擇部署。

繼續依照下一節的步驟，新增 CloudFront 觸發條件以執行函數。

步驟 4：新增一個 CloudFront 觸發條件來執行該函數

既然您已擁有用來更新安全標頭的 Lambda 函數，您可以設定 CloudFront 觸發條件來執行函數，在 CloudFront 從您的分佈原始伺服器收到的任何回應中新增標頭。

設定函數的 CloudFront 觸發條件

1. 在 Lambda 主控台頁面的函數概覽頁面中，選擇新增觸發條件。

2. 針對觸發組態選擇 CloudFront。
3. 選擇部署到 Lambda@Edge。
4. 在部署到 Lambda@Edge 面板中，在設定 CloudFront 觸發條件下輸入以下資訊：
 - 分佈：要與函數建立關聯的 CloudFront 分佈 ID。在下拉式清單中選擇分佈 ID。
 - 快取行為：要用於觸發條件的快取行為。在此範例中，請將值的設定保留為 *，這代表分佈的預設快取行為。如需詳細資訊，請參閱 [所有分佈設定參考](#) 主題中的 [快取行為設定](#)。
 - CloudFront 事件：指定函數何時執行的觸發條件。我們希望每當 CloudFront 從原始伺服器傳回回應時，都會執行安全標頭函數。因此請在下拉式清單中選擇原始伺服器回應。如需詳細資訊，請參閱 [為 Lambda@Edge 函數新增觸發條件](#)。
5. 選取確認部署到 Lambda@Edge 核取方塊。
6. 選擇部署以新增觸發條件，並將函數複寫至全球 AWS 位置。
7. 請等候函數完成複寫。這通常需要幾分鐘的時間。

若要檢查複寫是否已完成，請[移至 CloudFront 主控台](#)並檢視您的分佈。等待分佈的狀態是否已從正在部署變為日期和時間 (表示您的函數已複寫完成)。若要驗證函數是否可正常運作，請依照下一節的步驟進行。

步驟 5：驗證函數正常執行

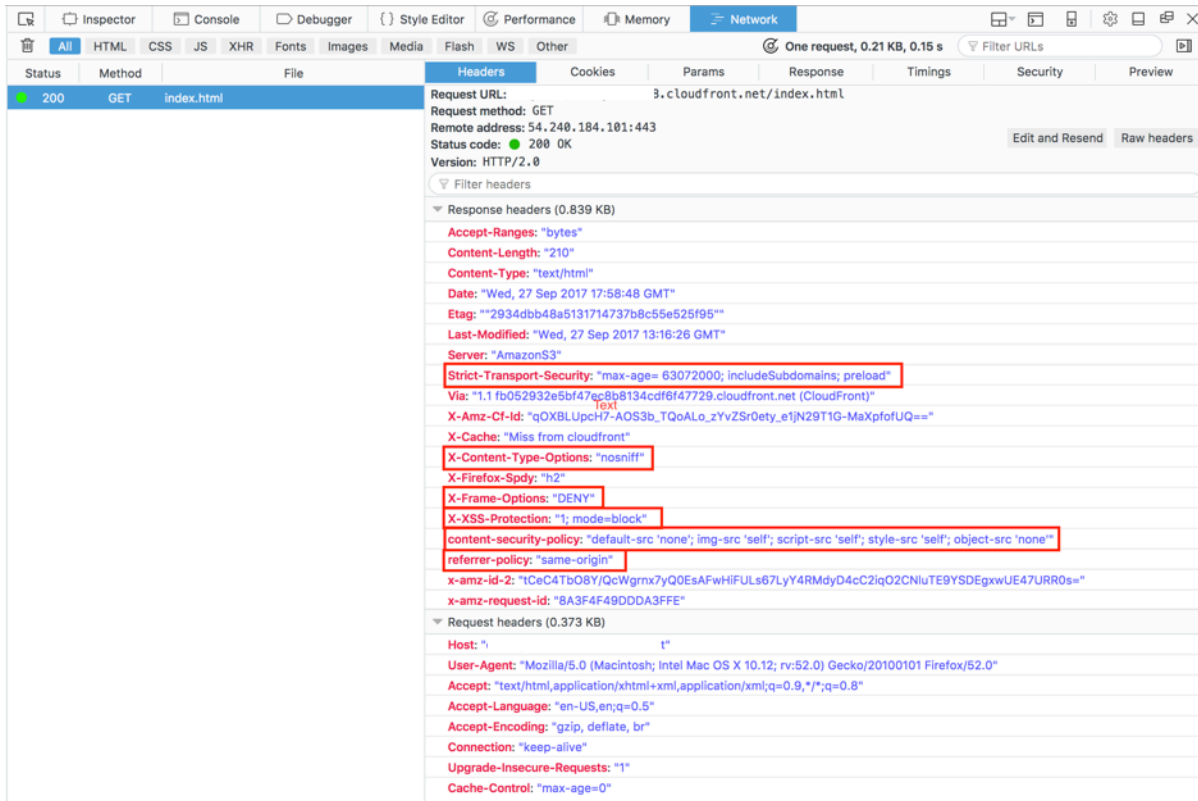
既然您已建立 Lambda 函數並設定觸發條件來為 CloudFront 分佈執行該函數，接下來請進行檢查，以確定函數會如預期執行。在此範例中，我們會檢查 CloudFront 傳回的 HTTP 標頭，以確認安全標頭已新增。

驗證您的 Lambda@Edge 函數有新增安全標頭

1. 在瀏覽器中，輸入 S3 儲存貯體中某個檔案的 URL。例如，您可以使用類似以下的 URL：`https://d1111111abcdef8.cloudfront.net/image.jpg`。

如需用於檔案 URL 中 CloudFront 網域名稱的詳細資訊，請參閱[自訂 CloudFront 中檔案的 URL 格式](#)。
2. 開啟瀏覽器的網頁開發人員工具列。例如，在 Chrome 瀏覽器視窗中，開啟內容 (按一下滑鼠右鍵) 選單，然後選擇 Inspect (檢查)。
3. 選擇 Network (網路) 索引標籤。
4. 重新載入頁面來查看您的影像，然後選擇左窗格中的 HTTP 請求。您會看到 HTTP 標頭顯示在個別的窗格中。

5. 查看 HTTP 標頭清單，以驗證預期的安全標頭包含在清單中。例如，您可能會看到類似於以下螢幕擷取畫面所示的標頭。



如果安全標頭包含在您的標頭清單中，那就太棒了！這表示您已成功建立第一個 Lambda@Edge 函數。如果 CloudFront 傳回錯誤或有其他問題，請繼續下一個步驟來排除問題。

步驟 6：排除問題

如果 CloudFront 傳回錯誤或是未依照預期來新增安全標頭，您可以查看 CloudWatch Logs 來調查函數的執行狀況。請務必使用存放在最 AWS 接近執行函數之位置的日誌。

例如，如果您從倫敦檢視檔案，請試著在 CloudWatch 主控台將區域變更為歐洲 (倫敦)。

檢查您 Lambda@Edge 函數的 CloudWatch Logs

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/cloudwatch/> 開啟 CloudWatch 主控台。
2. 將 Region (區域) 變更為您在瀏覽器中檢視檔案時所顯示的位置。這是函數執行所在的位置。
3. 在左窗格中，選擇 Logs (日誌) 來檢視分佈的日誌。

如需詳細資訊，請參閱[使用 Amazon CloudWatch 監控 CloudFront 指標](#)。

步驟 7：清除範例資源

如果您僅為本教學課程建立 Amazon S3 儲存貯體和 CloudFront 分佈，請刪除您配置 AWS 的資源，以免再產生費用。刪除 AWS 資源後，您新增的任何內容將無法再使用。

工作

- [刪除 S3 儲存貯體](#)
- [刪除 Lambda 函數](#)
- [刪除 CloudFront 分佈](#)

刪除 S3 儲存貯體

刪除 Amazon S3 儲存貯體之前，請務必停用儲存貯體的記錄。否則，在刪除日誌時 AWS，請繼續將日誌寫入您的儲存貯體。

停用儲存貯體的記錄

1. 在以下網址開啟 Amazon S3 主控台：<https://console.aws.amazon.com/s3/>。
2. 選取您的儲存貯體，然後選擇 Properties (屬性)。
3. 從 Properties (屬性) 選擇 Logging (記錄日誌)。
4. 清除 Enabled (已啟用) 核取方塊。
5. 選擇 Save (儲存)。

現在即可刪除儲存貯體。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的[我該如何刪除 S3 儲存貯體？](#)。

刪除 Lambda 函數

如需刪除 Lambda 函數關聯和函數本身 (選用) 的說明，請參閱 [刪除 Lambda@Edge 函數和複本](#)。

刪除 CloudFront 分佈

刪除 CloudFront 分佈前必須先將其停用。已停用的分佈如此即不再有用，也不會產生費用。您隨時都可以啟用之前停用的分佈。刪除停用的分佈之後，它即不再可供使用。

停用和刪除 CloudFront 分佈

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 開啟 CloudFront 主控台。
2. 選取您要停用的分佈，然後選擇 Disable (停用)。
3. 出現確認提示時，請選擇 Yes, Disable (是，停用)。
4. 選取已停用的分佈，然後選擇 Delete (刪除)。
5. 出現確認提示時，選擇 Yes, Delete (是，刪除)。

相關資訊

現在您對於 Lambda@Edge 函數的運作方式已有了基本了解，請閱讀以下內容來進一步了解：

- [Lambda@Edge 範例函數](#)
- [Lambda@Edge 設計最佳實務](#)
- [使用 Lambda@Edge 減少延遲並將運算轉移至邊緣](#)

設定 Lambda@Edge 的 IAM 許可權限和角色

若要設定 Lambda@Edge，您必須具有以下適用於 AWS Lambda 的 IAM 許可權限和角色：

- [IAM 許可權限](#)：這些許可權限可讓您建立 Lambda 函數，並將其與 CloudFront 分佈建立關聯。
- [Lambda 函數執行角色](#) (IAM 角色) – Lambda 服務主體會擔任此角色來執行您的函數。
- [Lambda@Edge 的服務連結角色](#) – 服務連結角色允許特定 AWS 服務 將 Lambda 函數複寫到，AWS 區域 並讓 CloudWatch 使用 CloudFront 日誌檔案。

將 Lambda@Edge 函數與 CloudFront 分佈產生關聯所需的 IAM 許可

除了 Lambda 需要的 IAM 許可權限之外，IAM 使用者還需要下列 IAM 許可權限，以便將 Lambda 函數與 CloudFront 分佈建立關聯：

- `lambda:GetFunction`：授予許可權限以取得 Lambda 函數的組態資訊，以及預先簽署的 URL，以下載包含該函數的 .zip 檔案。
- `lambda:EnableReplication*`：授予許可權限至資源政策，讓 Lambda 複寫服務能夠取得函數程式碼和組態。
- `lambda:DisableReplication*`：授予許可權限至資源政策，以便讓 Lambda 複寫服務可以刪除函數。

 Important

您必須在 `lambda:EnableReplication*` 和 `lambda:DisableReplication*` 動作的結尾新增星號 (*)。

- 對於資源，指定要在 CloudFront 事件發生時執行之函數版本 ARN，如以下範例所示：

```
arn:aws:lambda:us-east-1:123456789012:function:TestFunction:2
```

- `iam:CreateServiceLinkedRole`：授予許可權限建立服務連結角色，供 Lambda@Edge 用來複寫 CloudFront 中的 Lambda 函數。第一次設定 Lambda@Edge 之後，會自動為您建立服務連結角色。您不需要將此許可權限新增至使用 Lambda@Edge 的其他分佈。
- `cloudfront:UpdateDistribution` 或 `cloudfront:CreateDistribution`：授予許可權限以更新或建立分佈。

如需詳細資訊，請參閱下列主題：

- [適用於 Amazon CloudFront 的 Identity and Access Management](#)
- 《AWS Lambda 開發人員指南》中的 [Lambda 資源存取許可權限](#)

服務主體的函數執行角色

您必須建立 `lambda.amazonaws.com` 和 `edgelambda.amazonaws.com` 服務主體可在執行函數時擔任的 IAM 角色。

 Tip

在 Lambda 主控台中建立函數時，您可以選擇使用 AWS 政策範本建立新的執行角色。此步驟會自動新增執行函數所需的 Lambda@Edge 許可權限。請參閱[教學課程：建立簡單 Lambda@Edge 函數中的步驟 5](#)。

如需有關手動建立 IAM 角色的詳細資訊，請參閱《IAM 使用者指南》中的[建立角色及附加政策 \(主控台\)](#)。

Example 範例：角色信任政策

您可在 IAM 主控台的信任關係索引標籤之下新增此角色。請勿在許可權限索引標籤下新增此政策。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "lambda.amazonaws.com",
          "edgelambda.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

如需更多資訊瞭解授予執行角色所需的許可權限，請參閱《AWS Lambda 開發人員指南》中的 [Lambda 資源存取許可權限](#)。

備註

- 根據預設，每當 CloudFront 事件觸發 Lambda 函數時，都會將資料寫入至 CloudWatch Logs。若要使用這些日誌，執行角色需要許可才能將資料寫入 CloudWatch Logs。您能使用預先定義的 AWSLambdaBasicExecutionRole 對執行角色授予許可權限。

如需 CloudWatch Logs 的詳細資訊，請參閱 [the section called “邊緣函數日誌”](#)。

- 如果您的 Lambda 函數程式碼存取其他 AWS 資源，例如從 S3 儲存貯體讀取物件，則執行角色需要執行該動作的許可。

Lambda@Edge 的服務連結角色

Lambda@Edge 使用 IAM [服務連結角色](#)。服務連結角色是直接連結至服務的一種特殊 IAM 角色類型。服務連結角色由服務預先定義，並包含該服務在代表您呼叫其他 AWS 服務時，需要用到的所有權限。

Lambda@Edge 使用以下 IAM 服務連結角色：

- **AWSServiceRoleForLambdaReplicator** – Lambda@Edge 使用此角色讓 Lambda@Edge 將函數複寫至 AWS 區域。

您初次在 CloudFront 中新增 Lambda@Edge 觸發條件時，將會自動建立名為 **AWSServiceRoleForLambdaReplicator** 的角色，以便讓 Lambda@Edge 將函數複寫至 AWS 區域。使用 Lambda@Edge 函數時需要這個角色。AWSServiceRoleForLambdaReplicator 角色的 ARN 類似以下範例：

```
arn:aws:iam::123456789012:role/aws-service-role/
replicator.lambda.amazonaws.com/AWSServiceRoleForLambdaReplicator
```

- **AWSServiceRoleForCloudFrontLogger**：CloudFront 使用此角色將日誌檔案推送至 CloudWatch。您可以使用日誌檔案來偵錯 Lambda@Edge 驗證錯誤。

您新增 Lambda@Edge 函數關聯，讓 CloudFront 將 Lambda@Edge 錯誤日誌檔案推送至 CloudWatch 時，會自動建立名為 **AWSServiceRoleForCloudFrontLogger** 的角色。AWSServiceRoleForCloudFrontLogger 角色的 ARN 看起來類似如下：

```
arn:aws:iam::account_number:role/aws-service-role/
logger.cloudfront.amazonaws.com/AWSServiceRoleForCloudFrontLogger
```

服務連結角色可讓設定及使用 Lambda@Edge 變得更輕鬆，因為您不必手動新增必要的許可。Lambda@Edge 會定義其服務連結角色的許可，而且只有 Lambda@Edge 能夠擔任此角色。已定義的許可包括信任政策和許可政策。許可政策無法連接到其他任何 IAM 實體。

您必須移除任何關聯的 CloudFront 或 Lambda@Edge 資源，然後才能刪除服務連結角色。這有助於保護您的 Lambda@Edge 資源，讓您不會移除在存取作用中資源時仍有需要的服務連結角色。

如需服務連結角色的詳細資訊，請參閱[CloudFront 的服務連結角色](#)。

Lambda@Edge 的服務連結角色許可

Lambda@Edge 使用兩個服務連結角色，分別名為 **AWSServiceRoleForLambdaReplicator** 及 **AWSServiceRoleForCloudFrontLogger**。以下章節說明這些角色的許可。

內容

- [Lambda Replicator 的服務連結角色許可](#)
- [CloudFront Logger 的服務連結角色許可](#)

Lambda Replicator 的服務連結角色許可

這個服務連結的角色可讓 Lambda 將 Lambda@Edge 函式複製到 AWS 區域。

AWSServiceRoleForLambdaReplicator 服務連結角色信任 `replicator.lambda.amazonaws.com` 服務來擔任該角色。

角色許可政策允許 Lambda@Edge 在指定資源上完成下列動作：

- `arn:aws:lambda:*:*:function:*` 的 `lambda:CreateFunction`
- `arn:aws:lambda:*:*:function:*` 的 `lambda>DeleteFunction`
- `arn:aws:lambda:*:*:function:*` 的 `lambda:DisableReplication`
- all AWS resources 的 `iam:PassRole`
- all AWS resources 的 `cloudfront:ListDistributionsByLambdaFunction`

CloudFront Logger 的服務連結角色許可

此服務連結角色可讓 CloudFront 將日誌檔案推送至您的 CloudWatch，以讓您偵錯 Lambda@Edge 驗證錯誤。

AWSServiceRoleForCloudFrontLogger 服務連結角色信任 `logger.cloudfront.amazonaws.com` 服務來擔任該角色。

角色許可政策允許 Lambda@Edge 在指定 `arn:aws:logs:*:*:log-group:/aws/cloudfront/*` 資源上完成下列動作：

- `logs:CreateLogGroup`
- `logs:CreateLogStream`
- `logs:PutLogEvents`

您必須設定許可，允許 IAM 實體 (例如使用者、群組或角色) 刪除 Lambda@Edge 服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的 [服務連結角色許可](#)。

建立 Lambda@Edge 的服務連結角色

一般而言，您不需要手動建立 Lambda@Edge 的服務連結角色。此服務會在以下情境為您自動建立角色：

- 您第一次建立觸發條件時，服務會建立 `AWSServiceRoleForLambdaReplicator` 角色 (如果尚未存在)。這個角色可讓 Lambda 將 Lambda@Edge 函數複製到 AWS 區域。

如果您刪除服務連結角色，則當您在分佈中為 Lambda@Edge 新增觸發條件時，將會重新建立此角色。

- 您更新或建立具有 Lambda@Edge 關聯的 CloudFront 分佈時，服務會建立一個 `AWSServiceRoleForCloudFrontLogger` 角色 (如果該角色尚不存在)。此角色可讓 CloudFront 將日誌檔案推送至 CloudWatch。

如果您刪除服務連結角色，當您更新或建立具有 Lambda@Edge 關聯的 CloudFront 分佈時，將會再次建立此角色。

若要手動建立這些服務連結角色，您可以執行下列 AWS Command Line Interface (AWS CLI) 命令：

建立 `AWSServiceRoleForLambdaReplicator` 角色

- 執行下列命令。

```
aws iam create-service-linked-role --aws-service-name
replicator.lambda.amazonaws.com
```

建立 `AWSServiceRoleForCloudFrontLogger` 角色

- 執行下列命令。

```
aws iam create-service-linked-role --aws-service-name
logger.cloudfront.amazonaws.com
```

編輯 Lambda@Edge 服務連結角色

Lambda@Edge 不允許您編輯 `AWSServiceRoleForLambdaReplicator` 或 `AWSServiceRoleForCloudFrontLogger` 服務連結角色。在服務建立服務連結角色之後，您將無法變

更角色名稱，因為各種實體皆可能會參考角色。不過，您可以使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的[編輯服務連結角色](#)。

AWS 區域 支援 Lambda@Edge 服務連結角色

CloudFront 在下列 AWS 區域中支援將服務連結角色用於 Lambda@Edge：

- 美國東部 (維吉尼亞北部) – us-east-1
- 美國東部 (俄亥俄) – us-east-2
- 美國西部 (加利佛尼亞北部) – us-west-1
- 美國西部 (奧勒岡) – us-west-2
- 亞太地區 (孟買) – ap-south-1
- 亞太地區 (首爾) – ap-northeast-2
- 亞太地區 (新加坡) – ap-southeast-1
- 亞太地區 (雪梨) – ap-southeast-2
- 亞太地區 (東京) – ap-northeast-1
- 歐洲 (法蘭克福) – eu-central-1
- 歐洲 (愛爾蘭) – eu-west-1
- 歐洲 (倫敦) – eu-west-2
- 南美洲 (聖保羅) – sa-east-1

撰寫及建立 Lambda@Edge 函數

若要使用 Lambda@Edge，請撰寫 AWS Lambda 函數的程式碼。為了協助您撰寫 Lambda@Edge 函數，請參閱下列資源：

- [Lambda@Edge 事件結構說明頁面](#)：瞭解與 Lambda@Edge 搭配使用的事件結構。
- [Lambda@Edge 範例函數](#)：範例函數，例如 A/B 測試和產生 HTTP 重新導向。

將 Node.js 或 Python 用於 Lambda@Edge 的程式設計模型和在 AWS 區域中使用 Lambda 的程式設計模型相同。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的[使用 Node.js 建構 Lambda 函數](#)或[使用 Python 建構 Lambda 函數](#)。

在您的 Lambda@Edge 函數中，包含 callback 參數並傳回請求或回應事件的適用物件：

- 請求事件 – 在回應中包含 `cf.request` 物件。

如果您要產生回應，請在回應中包含 `cf.response` 物件。如需詳細資訊，請參閱[在請求觸發條件中產生 HTTP 回應](#)。

- 回應事件 – 在回應中包含 `cf.response` 物件。

在您撰寫自己的程式碼或使用其中一個範例，然後在 Lambda 中建立函數。若要建立函數或編輯現有函數，請參閱下列主題：

主題

- [建立 Lambda@Edge 函數](#)
- [編輯 Lambda 函數](#)

在 Lambda 中建立函數後，您可以設定 Lambda 根據稱為觸發的特定 CloudFront 事件執行函數。如需詳細資訊，請參閱[為 Lambda@Edge 函數新增觸發條件](#)。

建立 Lambda@Edge 函數

若要設定 AWS Lambda 以執行以 CloudFront 事件為基礎的 Lambda 函數，請遵循此程序。

建立 Lambda@Edge 函數

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/lambda/> 開啟 AWS Lambda 主控台。
2. 如果您已擁有一個或多個 Lambda 函數，請選擇 Create function (建立函數)。

如果您未擁有任何函數，請選擇 Get Started Now (立即開始)。

3. 在頁面頂端的「區域」清單中，選擇美國東部 (維吉尼亞北部)。
4. 使用您自己的程式碼建立函數，或以 CloudFront 藍圖為基礎建立函數。

- 若要使用自己的程式碼來建立函數，請選擇 Author from scratch (從頭開始編寫)。
- 若要顯示 CloudFront 藍圖清單，請在篩選條件欄位輸入 cloudfront，然後選擇 Enter。

如果找到想要使用的藍圖，請選擇該藍圖名稱。

5. 在 Basic information (基本資訊) 區段中，指定下列的值：
 - a. 名稱：輸入函數名稱。

- b. 角色：若要快速開始使用，請選擇從範本建立新角色。您也可以選擇選擇現有角色或建立自訂角色，然後依照提示完成本區段的資訊。
 - c. 角色名稱：輸入角色名稱。
 - d. 政策範本：選擇基本邊緣 Lambda 許可權限。
6. 如果您在步驟 4 中選擇 Author from scratch (從頭開始編寫)，請跳到步驟 7。

如果您在步驟 4 選擇藍圖，則可於 cloudfront 區段建立一個觸發，其可以將此函數與 CloudFront 分佈和 CloudFront 事件中的快取相關聯。我們建議您在此處選擇 Remove (移除)，如此函數在建立時就不會有觸發條件。您可以在稍後新增觸發。

 Tip

建議在新增觸發條件之前，先對函數進行測試及除錯。如果您現在新增觸發條件，函式會在您建立函式並完成複製至世界各地的 AWS 位置，並部署對應的分佈時立即執行。

7. 選擇 Create function (建立函數)。

Lambda 會建立兩個版本的函數：\$LATEST 和 Version 1。您只能編輯 \$LATEST 版本，但是主控台最初會顯示 Version 1。

8. 若要編輯函數，請選擇該函數 ARN 下方、靠近頁面頂端的 Version 1 (版本 1)。接著，在 Versions (版本) 索引標籤中，選擇 \$LATEST (\$LATEST)。(如果離開函數再返回，按鈕的標籤會是 Qualifiers (修飾詞)。)
9. 在 Configuration (組態) 索引標籤中，選擇適用的 Code entry type (程式碼項目類型)。然後遵循提示來編輯或上傳程式碼。
10. 針對 Runtime (執行時間)，根據函數的程式碼來選擇值。
11. 在 Tags (標籤) 區段中，新增任何適用的標籤。
12. 選擇 Actions (動作)，然後選擇 Publish new version (發佈新版本)。
13. 輸入函數新版本的說明。
14. 選擇 Publish (發佈)。
15. 對函數進行測試與偵錯。如需有關在 Lambda 主控台進行測試的詳細資訊，請參閱《AWS Lambda 開發人員指南》中的[使用主控台調用 Lambda 函數](#)。
16. 當您準備好將函數用於 CloudFront 事件執行時，請發佈另一個版本，並編輯該函數以新增觸發。如需詳細資訊，請參閱[為 Lambda@Edge 函數新增觸發條件](#)。

編輯 Lambda 函數

建立 Lambda@Edge 函數之後，您可以使用 Lambda 主控台編輯該函數。

備註

- 原始版本標記為 \$LATEST。
- 您只能編輯 \$LATEST 版本。
- 每次您編輯 \$LATEST 版本時，必須發佈新的編號版本。
- 您無法為 \$LATEST 建立觸發。
- 當您發佈新函數版本時，Lambda 不會自動將觸發條件由前一個版本複製到新版本。您必須為新版本重新產生觸發。
- 當您新增 CloudFront 事件觸發條件到函數時，如果針對相同分佈、快取行為及事件之較早版本的相同函數已具有觸發條件，則 Lambda 會刪除較早版本的觸發條件。
- 在您對 CloudFront 分佈進行例如新增觸發等更新後，必須等待變更傳播到節點，您由觸發條件所指定的函數才可運作。

若要編輯 Lambda 函數

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/lambda/> 開啟 AWS Lambda 主控台。
2. 在頁面頂端的「區域」清單中，選擇美國東部 (維吉尼亞北部)。
3. 在函數清單上選取函數名稱。

根據預設，主控台會顯示 \$LATEST 版本。您可以檢視較早的版本 (選擇 Qualifiers (修飾詞))，但是只能編輯 \$LATEST。

4. 在程式碼索引標籤上，針對程式碼項目類型，選擇在瀏覽器中編輯程式碼、上傳 .zip 檔案，或從 Amazon S3 上傳檔案。
5. 選擇 Save (儲存) 或 Save and test (儲存並測試)。
6. 選擇 Actions (動作)，然後選擇 Publish new version (發佈新版本)。
7. 在 Publish new version from \$LATEST (從 \$LATEST 發佈新版本) 對話方塊中，輸入新版本的說明。此說明會與自動產生的版本編號一起顯示在版本清單中。
8. 選擇 Publish (發佈)。

新版本會自動成為最新版本。版本編號會顯示於頁面左上角的版本按鈕上。

 Note

如果您尚未為函數新增觸發條件，請參閱 [為 Lambda@Edge 函數新增觸發條件](#)。

9. 選擇 Triggers (觸發條件) 索引標籤。
10. 選擇 Add trigger (新增觸發條件)。
11. 在新增觸發對話方塊，選擇虛線方塊，再選擇 CloudFront。

 Note

如果您已為函數建立一或多個觸發條件，則 CloudFront 為預設服務。

12. 指定下列值，以指示您希望 Lambda 函數在何時執行。
 - a. 分佈 ID：選擇您要為其新增觸發的分佈 ID。
 - b. 快取行為：選擇快取行為，透過該行為指定您要執行函數的物件。
 - c. CloudFront 事件：選擇促使函數執行的 CloudFront 事件。
 - d. 啟用觸發條件及複寫：選取此核取方塊，讓 Lambda 將函數複寫到全球的 AWS 區域。
13. 選擇提交。
14. 若要為此函數新增更多觸發，請重複操作步驟 10 到 13。

如需更多資訊瞭解在 Lambda 主控台中測試和偵錯函數，請參閱《AWS Lambda 開發人員指南》中的 [使用主控台調用 Lambda 函數](#)。

當您準備好將函數用於 CloudFront 事件執行時，請發佈另一個版本，並編輯該函數以新增觸發。如需詳細資訊，請參閱 [為 Lambda@Edge 函數新增觸發條件](#)。

為 Lambda@Edge 函數新增觸發條件

Lambda@Edge 觸發器是指 CloudFront 分佈、快取行為及促使函數執行的事件的組合。例如，您可以建立觸發條件，在 CloudFront 收到檢視器針對您為分佈設定的特定快取行為提出的請求時，執行函數。您可以指定一或多個 CloudFront 觸發條件。

Tip

當您建立 CloudFront 分佈，您需要指定設定來告知 CloudFront 如何在接收不同的請求時回應。預設設定稱為分佈的預設快取行為。您可以設定其他快取行為來定義 CloudFront 在特定情況下回應的方式，例如在接收請求特定的檔案類型的請求時。如需詳細資訊，請參閱[快取行為設定](#)。

您首次建立 Lambda 函數時，只能指定一個觸發條件。您之後可以使用以下方式將更多觸發條件新增到相同函數：使用 Lambda 主控台或在 CloudFront 主控台上編輯分佈。

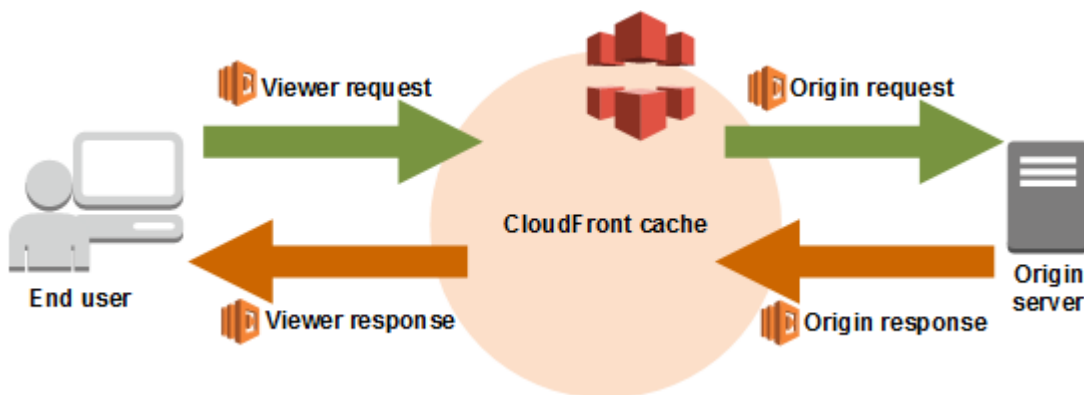
- 如果您要將更多觸發條件新增到同一 CloudFront 分佈的函數中，Lambda 主控台將非常有效。
- 如果要為多個分佈新增觸發條件，使用 CloudFront 主控台較適合，因為這樣更易於找到您要更新的分佈。您也可以同時更新其他的 CloudFront 設定。

主題

- [可以觸發 Lambda@Edge 函數的 CloudFront 事件](#)
- [選擇要觸發函數的事件](#)
- [新增觸發條件到 Lambda@Edge 函數](#)

可以觸發 Lambda@Edge 函數的 CloudFront 事件

對於 Amazon CloudFront 分佈中的每個快取行為，您可以新增最多四個觸發條件 (關聯)，以便在特定 CloudFront 事件發生時執行 Lambda 函數。CloudFront 觸發條件可以根據四個 CloudFront 事件之一，如下圖所示。



可用於觸發 Lambda@Edge 函數的 CloudFront 事件如下：

檢視者請求

函數會在 CloudFront 接獲檢視器請求時執行，然後再檢視請求的物件是否位在 CloudFront 快取。

函數不在以下情況執行：

- 擷取自訂錯誤頁面時。
- CloudFront 自動重新導向 HTTP 請求到 HTTPS ([檢視器通訊協定政策](#) 的值為重新導向 HTTP 到 HTTPS 時)。

原始伺服器請求

函數只在 CloudFront 將請求轉傳到您的原始伺服器時執行。如果請求的物件在 CloudFront 快取中，則不會執行該函數。

原始伺服器回應

函數會在 CloudFront 接獲原始伺服器回應後執行，然後才快取回應中的物件。請注意，即使原始伺服器傳回錯誤，函數仍會執行。

函數不在以下情況執行：

- 當請求的檔案在 CloudFront 快取中，並且未過期。
- 當回應是從原始伺服器請求事件觸發的函數所產生。

檢視者回應

函數會在請求的檔案傳回給檢視器之前執行。請注意，無論檔案是否已在 CloudFront 快取中，函數皆會執行。

函數不在以下情況執行：

- 當原始伺服器傳回 HTTP 狀態碼 400 或更高版本。
- 當傳回自訂錯誤頁面。
- 當回應是被檢視器請求事件觸發的函數所產生。
- CloudFront 自動重新導向 HTTP 請求到 HTTPS ([檢視器通訊協定政策](#) 的值為重新導向 HTTP 到 HTTPS 時)。

當您新增多個觸發條件至相同的快取行為時，可將其用於針對每個觸發條件執行相同或不同的函數。您也可以將相同的函數與一個以上的分佈建立關聯。

Note

CloudFront 事件觸發執行 Lambda 函數時，該函數必須先完成，CloudFront 才能繼續執行。

例如，當 Lambda 函數是由 CloudFront 檢視器請求事件觸發時，CloudFront 在 Lambda 函數執行完畢之前不會將回應傳回給檢視器或轉送請求給原始伺服器。
這表示觸發 Lambda 函數的每個請求都會增加請求延遲，因此您會希望函數執行速度越快越好。

選擇要觸發函數的事件

當您決定要使用哪個 CloudFront 事件來觸發 Lambda 函數，請考慮以下資訊：

我希望 CloudFront 快取由 Lambda 函數更改的物件

如果要快取由 Lambda 函數修改的物件，以便讓 CloudFront 在下次收到請求時能夠從邊緣節點提供物件，請使用原始伺服器請求或原始伺服器回應事件。

這樣可以降低原始伺服器的負載，減少後續請求的延遲，並降低叫用 Lambda@Edge 在後續請求的費用。

例如，如果您想要新增、移除或變更傳回原始伺服器的物件標頭，且您希望 CloudFront 快取結果，請使用原始伺服器的回應事件。

我希望函數執行每個請求

如果要讓函數執行每個 CloudFront 收到用於分佈的請求，請使用檢視器請求或檢視器回應事件。

原始伺服器請求與原始伺服器回應事件，僅會在當請求的物件未快取於節點以及 CloudFront 轉送請求至原始伺服器時發生。

我希望函數變更快取金鑰

如欲變更快取基礎值，請使用檢視器請求事件。

例如，若函數變更其 URL 以包含語言縮寫在路徑中 (例如，因為使用者從下拉式清單中選擇其語言)，請使用檢視器請求事件：

- 檢視器請求中的 URL – `https://https://example.com/en/index.html`
- 當請求來自德國境內 IP 位址時的 URL – `https://example.com/de/index.html`

如果您快取根據 Cookie 或請求標頭，也可以使用檢視器請求事件。

Note

如果函數變更 Cookie 或標頭，可設定 CloudFront 轉送請求適用的部分到原始伺服器。如需詳細資訊，請參閱下列主題：

- [根據 Cookie 快取內容](#)
- [根據請求標頭快取內容](#)

函數會影響原始伺服器回應

如果您希望依影響原始伺服器回應的方法變更請求，請使用原始伺服器請求事件。

一般而言，大多數檢視器請求事件不會轉送至原始伺服器。CloudFront 會使用已在邊緣快取中的物件回應請求。如果函數依原始伺服器請求事件變更請求，CloudFront 會快取對變更的原始伺服器請求的回應。

新增觸發條件到 Lambda@Edge 函數

您可以使用 AWS Lambda 主控台或 Amazon CloudFront 主控台，將觸發條件新增至 Lambda@Edge 函數。

Important

您只能為編號的函數版本 (而非 \$LATEST) 建立觸發條件。

Lambda console

將 CloudFront 事件的觸發條件新增至 Lambda@Edge 函數

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/lambda/> 開啟 AWS Lambda 主控台。
2. 在頁面頂端的「區域」清單中，選擇美國東部 (維吉尼亞北部)。
3. 在 Functions (函數) 頁面上，選擇您要為其新增觸發條件的函數名稱。
4. 在函數概觀頁面上，選擇版本索引標籤。
5. 選擇您要為其新增觸發的版本。

選擇版本之後，按鈕的名稱會變更為 Version: \$LATEST (版本：\$LATEST) 或 Version: (版本：) 版本編號。

6. 選擇 Triggers (觸發條件) 索引標籤。
7. 選擇 Add trigger (新增觸發條件)。

8. 針對觸發組態選擇選取來源，輸入 **cloudfront**，然後選擇 CloudFront。

 Note

如果您已經建立一或多個觸發，則 CloudFront 為預設服務。

9. 指定下列值，以指示您希望 Lambda 函數在何時執行。
 - a. 分佈：選擇您要為其新增觸發的分佈。
 - b. 快取行為：選擇快取行為，透過該行為指定您要執行函數的物件。

 Note

如果您為快取行為指定 *，Lambda 函式則會部署至預設的快取行為。

- c. CloudFront 事件：選擇促使函數執行的 CloudFront 事件。
 - d. 包含本文：如果想要存取您函數中的請求本體，請勾選此核取方塊。
 - e. 確認部署到 Lambda@Edge：選取此核取方塊，以便讓 AWS Lambda 將函數複寫至全球 AWS 區域。
10. 選擇新增。

將已更新的 CloudFront 分佈部署時，函數便開始針對指定的 CloudFront 事件處理請求。若要判斷是否已部署分佈，請在導覽窗格中選擇 Distributions (分佈)。部署分佈時，該分佈狀態欄中的值，會從正在部署變更為部署的日期與時間。

CloudFront console

將 CloudFront 事件的觸發條件新增至 Lambda@Edge 函數

1. 取得您希望新增觸發的 Lambda 函數的 ARN：
 - a. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/lambda/> 開啟 AWS Lambda 主控台。
 - b. 在頁面頂端的區域清單中，選擇美國東部 (維吉尼亞北部)。
 - c. 在函數清單上，選取您想要為其新增觸發的函數名稱。
 - d. 在函數概觀頁面選擇版本索引標籤，然後選擇您想要為其新增觸發條件的編號版本。
 - e. 選擇複製 ARN 按鈕，將 ARN 複製到剪貼簿。Lambda 函數的 ARN 如下所示：

```
arn:aws:lambda:us-east-1:123456789012:function:TestFunction:2
```

最後面的數字 (此範例中為 2 (2)) 是該函數的版本編號。

2. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 開啟 CloudFront 主控台。
3. 在分佈清單中，選取您想要為其新增觸發的分佈 ID。
4. 選擇 Behaviors (動作) 索引標籤。
5. 選擇您想要新增觸發條件的快取動作，然後選擇編輯。
6. 對於函數關聯，請在函數類型清單中選擇 Lambda@Edge，作為您希望函數執行的時間點：針對檢視器請求、檢視器回應、原始伺服器請求或原始伺服器回應。

如需詳細資訊，請參閱[選擇要觸發函數的事件](#)。

7. 在函數 ARN / 名稱文字方塊中，貼上所選事件發生時您想要執行 Lambda 函數的 ARN。這是您從 Lambda 主控台複製的值。
8. 如果想要存取您函數中的請求本文，請勾選包含本文。

如果您只想要替換請求本體，就不需要選取此選項。

9. 若要針對更多事件類型執行同一個函數，請重複步驟 6 和 7。
10. 選擇儲存變更。
11. 若要為此分佈新增更多快取行為的觸發，請重複步驟 5 到 10。

將已更新的 CloudFront 分佈部署時，函數便開始針對指定的 CloudFront 事件處理請求。若要判斷是否已部署分佈，請在導覽窗格中選擇 Distributions (分佈)。部署分佈時，該分佈狀態欄中的值，會從正在部署變更為部署的時間與日期。

測試和偵錯 Lambda@Edge 函數

單獨測試您 Lambda@Edge 函數的程式碼，和進行整合測試，這些動作非常地重要，前者可確保函數能夠完成預期的任務，後者則可確保函數能夠正確地搭配 CloudFront 使用。

在整合測試期間，或是在已經部署函數之後，您可能會需要進行 CloudFront 錯誤的偵錯，例如 HTTP 5xx 錯誤。錯誤可能是從 Lambda 函數傳回的無效回應、觸發函數時的執行錯誤，或是由於 Lambda 服務進行調節所產生的錯誤。本主題中的段落，說明了用來判斷是哪些故障類型造成問題的策略，以及您可採取的問題修正步驟。

 Note

當您在疑難排解錯誤時檢閱 CloudWatch 日誌檔案或指標時，請注意這些檔案或指標會顯示在最 AWS 區域 接近函數執行位置的位置。因此，舉例來說，如果您擁有網站或 Web 應用程式，其使用者位於英國，而您擁有與分佈相關聯的 Lambda 函數，則您必須變更區域，才能檢視倫敦 AWS 區域的 CloudWatch 指標或日誌檔案。如需詳細資訊，請參閱[the section called “判斷 Lambda@Edge 區域”](#)。

主題

- [測試您的 Lambda@Edge 函數](#)
- [在 CloudFront 中識別 Lambda@Edge 函數錯誤](#)
- [針對無效的 Lambda@Edge 函數回應 \(驗證錯誤\) 進行疑難排解](#)
- [進行 Lambda@Edge 函數執行錯誤的疑難排解](#)
- [判斷 Lambda@Edge 區域](#)
- [判斷您的帳戶是否推送日誌到 CloudWatch](#)

測試您的 Lambda@Edge 函數

測試您的 Lambda 函數包括兩個步驟：獨立測試和整合測試。

測試獨立的功能

在將您的 Lambda 函數新增到 CloudFront 之前，請務必先使用 CloudFront 主控台中的測試函數，或利用其他的方法，來進行函數的測試。如需有關在 Lambda 主控台進行測試的詳細資訊，請參閱《AWS Lambda 開發人員指南》中的[使用主控台調用 Lambda 函數](#)。

在 CloudFront 中測試函數的運作

完成整合測試非常地重要，這項測試會建立您的函數與分佈的關聯，並根據 CloudFront 事件執行。請務必以正確的事件觸發函數，並傳回對 CloudFront 有效且正確的回應。例如請確定事件的結構正確、只包含有效的標頭等，以此類推。

當您在 Lambda 主控台中，重複地進行函數的整合測試時，請參閱 Lambda@Edge 教學課程中的步驟，以修改程式碼，或變更呼叫函數的 CloudFront 觸發條件。例如，請確定您使用函式的編號版本，如教學課程的這項步驟中所述：[步驟 4：新增一個 CloudFront 觸發條件來執行該函數](#)。

在變更和部署函數時，請注意，更新的函數與 CloudFront 觸發條件複寫到所有區域時，將需要幾分鐘的時間。這通常需要幾分鐘的時間，但最多可能需要 15 分鐘。

若要檢查複寫是否已完成，請移至 CloudFront 主控台並檢視您的分佈。

如欲檢查您的複寫是否已完成部署

1. 在 <https://console.aws.amazon.com/cloudfront/v4/home> 中開啟 CloudFront 主控台。
2. 選擇分佈名稱。
3. 檢查分佈的狀態是否已從 In Progress (進行中) 變回 Deployed (已部署)，這表示您的函數已複寫完成。接著，請依照下一節的步驟來驗證函數是否正常運作。

請注意，在主控台中進行的測試只會驗證您的函數邏輯，並不會套用 Lambda@Edge 特定的服務配額 (先前稱為限制)。

在 CloudFront 中識別 Lambda@Edge 函數錯誤

確認您的函數邏輯可正常運作之後，當函數在 CloudFront 中執行時，仍然可能會出現 HTTP 5xx 錯誤。有各種原因可能傳回 HTTP 5xx 錯誤，包括 Lambda 函數錯誤或 CloudFront 中的其他問題。

- 如果您使用 Lambda@Edge 函數，您可以使用 CloudFront 主控台內的圖形以協助追蹤造成錯誤的原因，然後試圖修正該錯誤。例如，您可以檢視 HTTP 5xx 錯誤是否由 CloudFront 或 Lambda 函數造成，然後針對特定函數檢視相關的日誌檔案以調查該問題。
- 若要排除 CloudFront 中的一般性 HTTP 錯誤，請參閱下列主題中的故障診斷步驟：[對 CloudFront 中的錯誤回應狀態碼進行疑難排解](#)。

導致 CloudFront 中 Lambda@Edge 函數錯誤的原因

Lambda 函數造成 HTTP 5xx 錯誤的原因有許多種，您應該依據錯誤的類型採取相應的疑難排解步驟。錯誤分類如下：

Lambda 函數執行錯誤

如果因為函數中存在未處理的例外狀況，或程式碼中出現錯誤，而使得 CloudFront 未獲得 Lambda 的回應，會造成執行錯誤。例如，如果程式碼包含回呼 (錯誤)。

無效的 Lambda 函數回應會傳回至 CloudFront

函數執行後，CloudFront 會收到來自 Lambda 的回應。如果回應的物件結構不符合 [Lambda@Edge 事件結構說明頁面](#)，或回應中包含無效的標頭或其他無效的欄位，系統會傳回錯誤。

由於 Lambda 服務配額 (以前稱為限制)，CloudFront 中的執行會受到調節

Lambda 服務會在各區域中調節執行作業，並在您超出配額時傳回錯誤。如需詳細資訊，請參閱 [Lambda@Edge 的配額](#)。

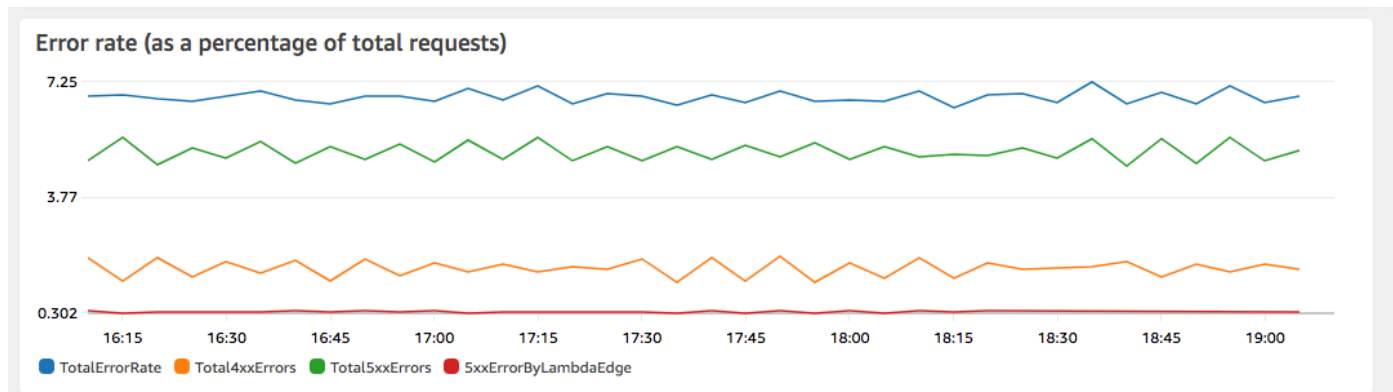
如何判斷故障的類型

為了在您偵錯和解決 CloudFront 傳回的錯誤時協助您決定處理重點，找出為何 CloudFront 會傳回 HTTP 錯誤的原因是很有幫助的。若要開始使用，您可以使用 AWS 管理主控台上 CloudFront 主控台的 Monitoring (監控) 區段中提供的圖表。如需在 CloudFront 主控台的監控區段檢視圖表的詳細資訊，請參閱 [使用 Amazon CloudWatch 監控 CloudFront 指標](#)。

以下圖表在您追縱原始伺服器或 Lambda 函數是否傳回錯誤時特別實用，當錯誤是由於 Lambda 函數造成時，也可縮小問題的類型。

錯誤率圖表

在每一個分佈的 Overview (概觀) 標籤上，您可以檢視的其中一個圖表就是 Error rates (錯誤率) 圖表。此圖表顯示錯誤率佔進入您分配的請求總數的百分比。此圖表顯示總錯誤率，總共 4xx 個錯誤、總共 5xx 個錯誤，以及總共 5xx 個 Lambda 函數的錯誤。根據錯誤類型和磁碟區，您可以採取步驟以針對原因進行調查和故障診斷。



- 如果您看到 Lambda 錯誤，您可以透過查看該函數傳回的特定錯誤類型，以進一步進行調查。Lambda@Edge 錯誤標籤包含了依類型分類的函數錯誤圖表，可協助您找出特定函數的問題。

- 如果您看到 CloudFront 錯誤，您可以進行故障排除並解決原始伺服器錯誤或變更您的 CloudFront 組態。如需詳細資訊，請參閱[對 CloudFront 中的錯誤回應狀態碼進行疑難排解](#)。

執行錯誤和無效函數回應圖表

Lambda@Edge 錯誤標籤包含針對特定分佈 (依類型) 分類 Lambda@Edge 錯誤的圖表。例如一個圖表會依 AWS 區域顯示所有執行錯誤。

為了讓您更輕鬆地對問題進行故障排除，您可以依區域開啟和檢驗特定函數的日誌檔案，以尋找特定問題。

依區域檢視特定函數的日誌檔案

1. 在 Lambda@Edge 錯誤索引標籤的關聯 Lambda@Edge 函數下，選擇函數名稱，然後選擇檢視指標。
2. 接著在出現您函數名稱的頁面上，於右上角選擇檢視函數日誌，然後選擇區域。

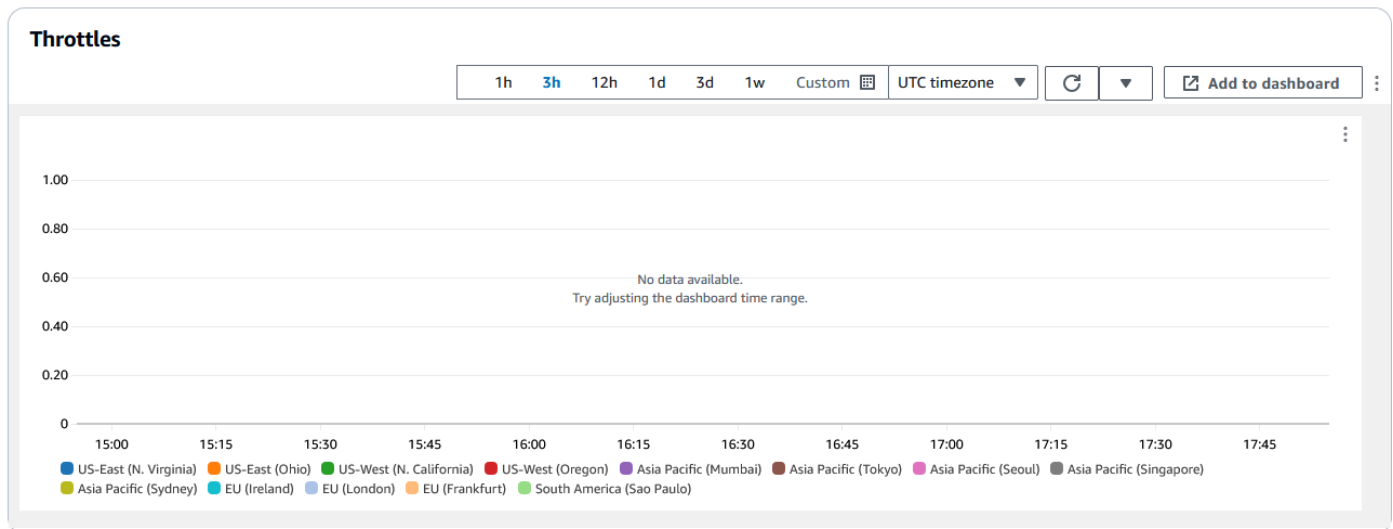
例如，如果您在美國西部 (奧勒岡) 區域的錯誤圖表中看到問題，請從下拉式清單選擇該區域。隨即開啟 Amazon CloudWatch 主控台。

3. 在該區域 CloudWatch 主控台的日誌串流下，選擇日誌串流以檢視函數事件。

此外，請閱讀此章的下列各節，以了解有關故障排除和修復錯誤的更多建議。

調節圖表

Lambda@Edge 錯誤標籤也包含調節圖表。有時，如果您到達區域並行數量配額 (先前稱為限制)，則 Lambda 服務會依每一區域為基礎調節您的函數呼叫。如果出現超過限制錯誤，表示您的函數已到達 Lambda 服務對「區域」中的執行作業所施加的配額。如需詳細資訊，包括如何請求提高配額，請參閱[Lambda@Edge 的配額](#)。



如需如何使用此資訊進行 HTTP 錯誤故障診斷的詳細資訊，請參閱[在 AWS 上針對您的內容交付執行偵錯的四個步驟](#)。

針對無效的 Lambda@Edge 函數回應 (驗證錯誤) 進行疑難排解

如果您找到的問題是 Lambda 驗證錯誤，表示您的 Lambda 函數將無效的回應傳回給 CloudFront。請依照本段落中的指引，採取步驟來審視您的函數，並確認回應符合 CloudFront 的請求。

CloudFront 會以兩種方式驗證來自 Lambda 函數的回應：

- Lambda 回應必須符合所請求的物件結構。錯誤的物件結構範例包括：無法剖析的 JSON、遺漏必要的欄位，以及在回應中包含無效的物件。如需更多資訊，請參閱[Lambda@Edge 事件結構說明頁面](#)。
- 回應必須只包含有效的物件值。如果回應中包含有效的物件，但是具有不支援的值，將會發生錯誤。此種情況的範例包括：新增或更新被列入不允許或唯讀的標頭 (請參閱[對邊緣函數的限制](#))、超過內文大小的上限 (請參閱 Lambda@Edge [錯誤](#) 主題中的對所產生回應的大小限制)，以及無效的字元或值 (請參閱[Lambda@Edge 事件結構說明頁面](#))。

當 Lambda 傳回無效回應至 CloudFront 時，會將錯誤訊息寫入日誌檔案，而 CloudFront 會在執行 Lambda 函數的區域中推送至 CloudWatch。出現無效的回應時，預設的動作是將日誌檔案傳送到 CloudWatch。不過，如果在函數發布之前，就已建立 Lambda 函數與 CloudFront 的關聯，則可能不會針對您的函數啟用這項預設動作。如需詳細資訊，請參閱本主題稍後的判斷您的帳戶是否推送日誌到 CloudWatch。

CloudFront 會將日誌檔案推送到對應您函數執行所在位置的區域 (在與您的分佈具有關聯的日誌群組中)。日誌群組具有下列格式：`/aws/cloudfront/LambdaEdge/DistributionId`，其中 *DistributionId* 是您分佈的 ID。若要判斷您可以在哪個區域中找到 CloudWatch 日誌檔案，請參閱本主題稍後的判斷 Lambda@Edge 區域。

如果錯誤是可重現的，您可以建立新的請求來造成錯誤，然後在失敗的 CloudFront 回應 (X-Amz-Cf-Id 標頭) 中找到該請求的 ID，然後在日誌檔案中找出單一錯誤。日誌檔案記錄所包含的資訊，可協助您找出傳回錯誤的原因，也可以列出對應的 Lambda 請求 ID，來讓您針對單一請求的範圍，分析錯誤的根本原因。

如果錯誤是間歇性出現，您可以利用 CloudFront 存取日誌，針對失敗的請求找出其請求 ID，然後搜尋 CloudWatch Logs，尋找對應的錯誤訊息。如需詳細資訊，請參閱先前的段落判斷故障的類型。

進行 Lambda@Edge 函數執行錯誤的疑難排解

如果問題是 Lambda 的執行錯誤，那麼建立 Lambda 函數的記錄陳述式、將訊息寫入 CloudWatch 日誌檔案 (此日誌檔案會監控您的函數在 CloudFront 中的執行狀況)，然後判斷函數是否如預期運作，這些動作將會有所幫助。接著，您可以在 CloudWatch 日誌檔案中搜尋這些陳述式，來確認您的函數是否正常運作。

Note

即使您未變更您的 Lambda@Edge 函數，Lambda 函數執行環境的更新仍會對其造成影響，並因而傳回執行錯誤。如需有關測試和遷移至較新版本的資訊，請參閱 [Lambda AWS 和 AWS Lambda@Edge 執行環境的近期更新](#)。

判斷 Lambda@Edge 區域

若要檢視您 Lambda@Edge 函數正在接收流量的區域，請在 AWS 管理主控台上的 CloudFront 主控台檢視函數的指標圖形。每個 AWS 區域的指標都會顯示。在同一頁面中，您可以選擇一個區域並檢視該區域的日誌檔，以便調查問題。您必須檢閱正確區域中的 CloudWatch 日誌檔案 AWS，以查看 CloudFront 執行 Lambda 函數時建立的日誌檔案。

如需在 CloudFront 主控台的監控區段檢視圖表的詳細資訊，請參閱[使用 Amazon CloudWatch 監控 CloudFront 指標](#)。

判斷您的帳戶是否推送日誌到 CloudWatch

根據預設，CloudFront 會啟用記錄無效的 Lambda 函式回應，並使用其中一個 [Lambda@Edge 的服務連結角色](#) 將日誌檔案推送至 CloudWatch。如果您擁有 Lambda@Edge 函數，而在無效的 Lambda 函數回應日誌函數推出之前，您已經將此 Lambda@Edge 函數新增到 CloudFront，而當您後續更新 Lambda@Edge 組態時，就會啟用記錄函數 (例如，透過新增 CloudFront 觸發條件)。

您可以執行下列的動作，來確認您的帳戶已啟用推送日誌檔案到 CloudWatch 的功能：

- 檢查日誌是否顯示在 CloudWatch 中：請務必查看執行 Lambda@Edge 函數的區域。如需詳細資訊，請參閱 [判斷 Lambda@Edge 區域](#)。
- 在 IAM 中判斷您的帳戶是否存在相關的服務連結角色：您的帳戶必須擁有 IAM 角色 `AWSServiceRoleForCloudFrontLogger`。如需有關此角色的詳細資訊，請參閱 [Lambda@Edge 的服務連結角色](#)。

刪除 Lambda@Edge 函數和複本

您可以只在函數複本遭 CloudFront 刪除時，系統才會刪除 Lambda@Edge 函數。Lambda 函數的複本將在下列情況自動刪除：

- 當您從所有 CloudFront 分佈移除最後一個函數的關聯後。如果一個以上的分佈使用一個函數，則只有在您從上一個分佈移除函數關聯之後，複本才會刪除。
- 在您刪除最後一個與函數相關聯的分佈後。

複本一般會在幾個小時內刪除。您無法手動刪除 Lambda@Edge 函數複本。這有助於防止刪除仍在使用的複本，以免導致錯誤情況。

Warning

請勿建置在 CloudFront 之外使用 Lambda@Edge 函數複本的應用程式。當這些複本與分佈的關聯遭到移除，或分佈本身遭到刪除時，這些複本就會隨之刪除。外部應用程式所依據使用的複本可能會在無預警的情況下移除，導致應用程式作業失敗。

從 CloudFront 分佈刪除 Lambda@Edge 函數關聯

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。

2. 選擇具有您要刪除之 Lambda@Edge 函數關聯的分佈 ID。
3. 選擇 Behaviors (動作) 索引標籤。
4. 選擇具有您要刪除之 Lambda@Edge 函數關聯的快取行為，然後選擇編輯。
5. 在函數關聯、函數類型下，選擇沒有關聯以刪除 Lambda@Edge 函數關聯。
6. 選擇儲存變更。

從 CloudFront 分發中刪除 Lambda@Edge 函數關聯之後，您可以選擇性地從 AWS Lambda 中刪除 Lambda 函數或函數版本。刪除函數關聯後請等待數小時，以便清除 Lambda@Edge 函數複本。之後，您可以使用 Lambda 主控台、AWS CLI Lambda API 或 AWS SDK 刪除函數。

您也可以刪除特定版本的 Lambda 函數，前提示該版本未與任何 CloudFront 分佈建立關聯。移除 Lambda 函數版本的所有關聯後，請等待數小時。然後您可以刪除函數版本。

Lambda@Edge 事件結構說明頁面

下列主題說明在觸發時，CloudFront 傳遞到 Lambda@Edge 函數的請求和回應事件物件。

主題

- [動態原始伺服器選擇](#)
- [請求事件](#)
- [回應事件](#)

動態原始伺服器選擇

您可以根據路徑和請求物件名稱 (例如 `images/*.jpg`)，使用[快取行為中的路徑模式](#)將請求路由傳送到原始伺服器。使用 Lambda@Edge，您也可以根據其他特性路由傳送請求到原始伺服器，例如請求標頭中的值。

此動態原始伺服器選擇在很多方面是非常有用的。例如，您可以將請求分佈到各個在不同地理區域的原始伺服器，以協助全域負載平衡。或者，您可以選擇性地路由傳送請求到服務特定函數之不同的原始伺服器：機器人處理、SEO 最佳化、驗證，以此類推。如需示範如何使用此功能的程式碼範例，請參閱[以內容為基礎的動態原始伺服器選擇 - 範例](#)。

在 CloudFront 原始伺服器請求事件中，事件結構中的 `origin` 物件包含了關於該原始伺服器的資訊 (會根據路徑的模式，將請求轉送到自訂此原始伺服器)。您可以更新 `origin` 物件的值，將請求路由傳送到不同的原始伺服器。更新 `origin` 物件時，您不需要定義分佈中的原始伺服器。您也可以使用自

訂原始伺服器物件來取代 Amazon S3 原始伺服器物件，反之亦然。不過，您只能為每個請求指定單一原始伺服器；也就是指定自訂原始伺服器或 Amazon S3 原始伺服器，但不能兩者同時指定。

請求事件

下列主題說明 CloudFront 針對[檢視器和原始伺服器請求事件](#)傳遞到 Lambda 函數的結構。這些範例顯示不含任何主體的 GET 請求。在範例之後，會提供檢視器和原始伺服器請求事件中的所有可能欄位清單。

主題

- [檢視者請求範例](#)
- [原始伺服器請求範例](#)
- [請求事件欄位](#)

檢視者請求範例

下列範例顯示檢視器請求事件物件。

```
{
  "Records": [
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "viewer-request",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26C1koUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "host": [
              {
                "key": "Host",
                "value": "d111111abcdef8.cloudfront.net"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "curl/7.66.0"
              }
            ]
          }
        }
      }
    }
  ]
}
```

```

    }
  ],
  "accept": [
    {
      "key": "accept",
      "value": "*/*"
    }
  ]
},
"method": "GET",
"querystring": "",
"uri": "/"
}
}
}
]
}

```

原始伺服器請求範例

下列範例顯示原始伺服器請求事件物件。

```

{
  "Records": [
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "origin-request",
          "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "x-forwarded-for": [
              {
                "key": "X-Forwarded-For",
                "value": "203.0.113.178"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",

```

```
        "value": "Amazon CloudFront"
      }
    ],
    "via": [
      {
        "key": "Via",
        "value": "2.0 2afae0d44e2540f472c0635ab62c232b.cloudfront.net
(CloudFront)"
      }
    ],
    "host": [
      {
        "key": "Host",
        "value": "example.org"
      }
    ],
    "cache-control": [
      {
        "key": "Cache-Control",
        "value": "no-cache"
      }
    ]
  },
  "method": "GET",
  "origin": {
    "custom": {
      "customHeaders": {},
      "domainName": "example.org",
      "keepaliveTimeout": 5,
      "path": "",
      "port": 443,
      "protocol": "https",
      "readTimeout": 30,
      "responseCompletionTimeout": 30,
      "sslProtocols": [
        "TLSv1",
        "TLSv1.1",
        "TLSv1.2"
      ]
    }
  },
  "queryString": "",
  "uri": "/"
}
```



```
    }  
  }  
]  
}
```

請求事件欄位

請求事件物件資料包含在兩個子物件中：`config (Records.cf.config)` 和 `request (Records.cf.request)`。下列清單說明每個子物件的欄位。

Config 物件中的欄位

下列清單說明 `config` 物件 (`Records.cf.config`) 中的欄位。

distributionDomainName (唯讀)

與請求相關的分佈網域名稱。

distributionID (唯讀)

與請求相關的分佈 ID。

eventType (唯讀)

與請求關聯的觸發條件類型：`viewer-request` 或 `origin-request`。

requestId (唯讀)

可獨特識別檢視器至 CloudFront 請求的加密字串。此 `requestId` 值也會在 CloudFront 存取記錄中顯示為 `x-edge-request-id`。如需詳細資訊，請參閱[存取日誌（標準日誌）](#)及[日誌檔案欄位](#)。

請求物件中的欄位

下列清單說明 `request` 物件 (`Records.cf.request`) 中的欄位。

clientIp (唯讀)

提出請求之檢視器的 IP 位址。如果檢視器使用 HTTP Proxy 或負載平衡器傳送請求，此值為代理或負載平衡器的 IP 位址。

標頭 (讀取/寫入)

請求中的標頭。注意下列事項：

- 在 `headers` 物件的金鑰為標準 HTTP 標頭名稱的小寫版本。使用小寫金鑰提供您區分大小寫的標頭值存取權。
- 每個標頭物件 (例如 `headers["accept"]` 或 `headers["host"]`) 都是鍵值組的陣列。針對已知的標頭，陣列會在請求中包含每個值的一組鍵值組。
- `key` 包含在 HTTP 請求中出現的標頭名稱 (區分大小寫)，例如 `Host`、`User-Agent`、`X-Forwarded-For`、`Cookie` 等。
- `value` 包含在 HTTP 請求中出現的標頭值。
- 當 Lambda 函數新增或修改請求標頭，而您不想要包含標頭 `key` 欄位時，`Lambda@Edge` 會使用您提供的標頭名稱來自動插入 `key` 標頭。無論您如何設定標頭名稱格式，各部分插入的標頭索引鍵名稱都會自動以大寫開頭，以連字號 (-) 分隔。

例如，您可以在沒有 `key` 標頭的情況下，新增標頭如下：

```
"user-agent": [  
  {  
    "value": "ExampleCustomUserAgent/1.X.0"  
  }  
]
```

在此範例中，`Lambda@Edge` 會自動插入 `"key": "User-Agent"`。

如需標頭使用限制的詳細資訊，請參閱[對邊緣函數的限制](#)。

method (唯讀)

請求的 HTTP 方法。

querystring (讀取/寫入)

請求中的查詢字串 (如果有的話)。如果請求不包含查詢字串，事件物件仍會包含具有空白值的 `querystring`。如需查詢字串的詳細資訊，請參閱[根據查詢字串參數快取內容](#)。

uri (讀取/寫入)

請求物件的相對路徑。如果 Lambda 函數修改了 `uri` 值，請注意下列事項：

- 全新的 `uri` 值必須以正斜線 (/) 做為開頭。
- 當函數變更 `uri` 值時，這會變更檢視器請求的物件。
- 當函數變更 `uri` 值時，並不會變更請求的快取行為或請求傳送的目標原始伺服器。

body (讀取/寫入)

HTTP 請求的主題。body 結構可包含下列欄位：

inputTruncated (唯讀)

布林值旗標，此旗標會指出 Lambda@Edge 是否截斷了本體。如需詳細資訊，請參閱[使用包含內文選項時的要求內文限制](#)。

action (讀取/寫入)

您想要對本體採取的動作。action 的選項如下：

- read-only: 此為預設值。從 Lambda 函式將回應傳回時，如果 action 為唯讀狀態，則 Lambda@Edge 會忽略對 encoding 或 data 所進行的任何變更。
- replace: 如果想取代傳送到原始伺服器的本體，請指定此選項。

encoding (讀取/寫入)

本體的編碼。當 Lambda@Edge 向 Lambda 函式顯露內文時，會先將內文轉換為 base64-encoding。如果針對 action 選擇 replace 以取代本體，您可以選擇使用 base64 (預設) 或 text 編碼。如果將 encoding 指定為 base64，但內文並非有效的 base64，CloudFront 會傳回錯誤。

data (讀取/寫入)

請求本體的內容。

origin (讀取/寫入) (僅限原始伺服器事件)

傳送請求的目標原始伺服器。origin 結構必須確切地包含一個原始伺服器，這可以是自訂原始伺服器或 Amazon S3 原始伺服器。

根據您指定的原始伺服器類型 (自訂或 Amazon S3 原始伺服器)，您必須在請求中指定下列欄位：

customHeaders (讀取/寫入) (自訂和 Amazon S3 原始伺服器)

(選用) 您可以透過指定的標頭名稱與每個自訂標頭的值對，於請求中包含自訂標頭。您無法新增已列入不允許的標頭，且具有相同名稱的標頭也無法出現在 `Records.cf.request.headers` 中。[請求標頭的相關注意事項](#)也適用於自訂標頭。如需詳細資訊，請參閱[CloudFront 無法新增到原始伺服器請求的自訂標頭及對邊緣函數的限制](#)。

domainName (讀取/寫入) (自訂和 Amazon S3 原始伺服器)

原始伺服器的網域名稱。網域名稱不能空白。

- 適用於自訂原始伺服器：指定 DNS 網域名稱，例如 `www.example.com`。網域名稱不能包含冒號 (:)，而且不能是 IP 位址。網域名稱長度上限為 253 個字元。

- 對於 Amazon S3 原始伺服器 — 指定 Amazon S3 儲存貯體的 DNS 網域名稱，例如 `amzn-s3-demo-bucket.s3.eu-west-1.amazonaws.com`。名稱可以高達 128 個字元，而且必須全部小寫。

path (讀取/寫入) (自訂和 Amazon S3 原始伺服器)

目錄路徑位於需定位內容請求的原始伺服器中。路徑的開頭應為正斜線 (/)，但結尾不應為正斜線 (例如，結尾不應為 `example-path/`)。僅適用於自訂原始伺服器：路徑應為 URL 編碼，且其長度上限為 255 個字元。

keepaliveTimeout (讀取/寫入) (僅限自訂原始伺服器)

CloudFront 在收到上次封包的回應後維持與自訂原始伺服器的連線所需的時間長度 (以秒為單位)。值必須是介於 1 到 120 之間的數字。

port (讀取/寫入) (僅限自訂原始伺服器)

CloudFront 應在您的自訂原始伺服器連線的連線埠。此連結埠必須是 80、443，或是介於 1024–65535 之間的數字。

protocol (讀取/寫入) (僅限自訂原始伺服器)

CloudFront 在連線到原始伺服器時應使用的連線通訊協定。此值可以為 `http` 或 `https`。

readTimeout (讀取/寫入) (自訂和 Amazon S3 原始伺服器)

CloudFront 在傳送請求給原始伺服器後應等候回應的時間長度 (以秒為單位)。這也會指定 CloudFront 在收到回應封包後與收到下一個封包前，應等候的時間長度。值必須是介於 1 到 120 之間的數字。

如果您需要更高的配額，請參閱[每個原始伺服器的回應逾時](#)。

responseCompletionTimeout (讀取/寫入) (自訂和 Amazon S3 原始伺服器)

從 CloudFront 到原始伺服器的請求可以保持開啟並等待回應的時間 (以秒為單位)。如果此時未從原始伺服器收到完整回應，CloudFront 會結束連線。

`responseCompletionTimeout` 值必須大於或等於 `readTimeout` 的值。如果您將此值設定為 0，則會移除您設定的任何先前值，並返回預設值。您也可以從事件請求中刪除 `responseCompletionTimeout` 欄位來完成此操作。

sslProtocols (讀取/寫入) (僅限自訂原始伺服器)

當在您原始伺服器建立 HTTPS 連線時，CloudFront 可以使用的最低 SSL/TLS 通訊協定。可為以下任何一個值：`TLSv1.2`、`TLSv1.1`、`TLSv1` 或 `SSLv3`。

authMethod (讀取/寫入) (僅限 Amazon S3 原始伺服器)

如果您使用[原始存取身分 \(OAI\)](#)，請將此欄位設定為 `origin-access-identity`。如果您未使用 OAI，請將其設定為 `none`。如果您將 `authMethod` 設定為 `origin-access-identity`，有幾項要求如下：

- 您必須指定 `region` (請參閱下列欄位)。
- 當您將請求從某個 Amazon S3 原始伺服器變更為其他原始伺服器時，您必須使用相同的 OAI。
- 當您將請求從某個自訂原始伺服器變更為 Amazon S3 原始伺服器時，您無法使用 OAI。

Note

此欄位不支援[原始存取控制 \(OAC\)](#)。

region (讀取/寫入) (僅限 Amazon S3 原始伺服器)

Amazon S3 儲存貯體 AWS 的區域。只有在您將 `authMethod` 設定為 `origin-access-identity` 時才需要。

回應事件

下列主題說明 CloudFront 針對[檢視器和原始伺服器回應事件](#)傳遞到 Lambda 函數的結構。在範例之後，會提供檢視器和原始伺服器回應事件中的所有可能欄位清單。

主題

- [原始伺服器回應範例](#)
- [檢視者回應範例](#)
- [回應事件欄位](#)

原始伺服器回應範例

下列範例顯示原始伺服器回應事件物件。

```
{
  "Records": [
    {
      "cf": {
```

```

    "config": {
      "distributionDomainName": "d111111abcdef8.cloudfront.net",
      "distributionId": "EDFDVBD6EXAMPLE",
      "eventType": "origin-response",
      "requestId": "4TyzHTaYWb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
    },
    "request": {
      "clientIp": "203.0.113.178",
      "headers": {
        "x-forwarded-for": [
          {
            "key": "X-Forwarded-For",
            "value": "203.0.113.178"
          }
        ],
        "user-agent": [
          {
            "key": "User-Agent",
            "value": "Amazon CloudFront"
          }
        ],
        "via": [
          {
            "key": "Via",
            "value": "2.0 8f22423015641505b8c857a37450d6c0.cloudfront.net
(CloudFront)"
          }
        ],
        "host": [
          {
            "key": "Host",
            "value": "example.org"
          }
        ],
        "cache-control": [
          {
            "key": "Cache-Control",
            "value": "no-cache"
          }
        ]
      },
      "method": "GET",
      "origin": {
        "custom": {

```

```
    "customHeaders": {},
    "domainName": "example.org",
    "keepaliveTimeout": 5,
    "path": "",
    "port": 443,
    "protocol": "https",
    "readTimeout": 30,
    "responseCompletionTimeout": 30,
    "sslProtocols": [
      "TLSv1",
      "TLSv1.1",
      "TLSv1.2"
    ]
  },
  "querystring": "",
  "uri": "/"
},
"response": {
  "headers": [
    "access-control-allow-credentials": [
      {
        "key": "Access-Control-Allow-Credentials",
        "value": "true"
      }
    ],
    "access-control-allow-origin": [
      {
        "key": "Access-Control-Allow-Origin",
        "value": "*"
      }
    ],
    "date": [
      {
        "key": "Date",
        "value": "Mon, 13 Jan 2020 20:12:38 GMT"
      }
    ],
    "referrer-policy": [
      {
        "key": "Referrer-Policy",
        "value": "no-referrer-when-downgrade"
      }
    ]
  ],
```

```
    "server": [
      {
        "key": "Server",
        "value": "ExampleCustomOriginServer"
      }
    ],
    "x-content-type-options": [
      {
        "key": "X-Content-Type-Options",
        "value": "nosniff"
      }
    ],
    "x-frame-options": [
      {
        "key": "X-Frame-Options",
        "value": "DENY"
      }
    ],
    "x-xss-protection": [
      {
        "key": "X-XSS-Protection",
        "value": "1; mode=block"
      }
    ],
    "content-type": [
      {
        "key": "Content-Type",
        "value": "text/html; charset=utf-8"
      }
    ],
    "content-length": [
      {
        "key": "Content-Length",
        "value": "9593"
      }
    ]
  },
  "status": "200",
  "statusDescription": "OK"
}
}
```



```
}
```

檢視者回應範例

下列範例顯示檢視器回應事件物件。

```
{
  "Records": [
    {
      "cf": {
        "config": {
          "distributionDomainName": "d111111abcdef8.cloudfront.net",
          "distributionId": "EDFDVBD6EXAMPLE",
          "eventType": "viewer-response",
          "requestId": "4TyzHTaYwb1GX1qTfsHhEqV6HUDd_BzoBZnwfnvQc_1oF26ClkoUSEQ=="
        },
        "request": {
          "clientIp": "203.0.113.178",
          "headers": {
            "host": [
              {
                "key": "Host",
                "value": "d111111abcdef8.cloudfront.net"
              }
            ],
            "user-agent": [
              {
                "key": "User-Agent",
                "value": "curl/7.66.0"
              }
            ],
            "accept": [
              {
                "key": "accept",
                "value": "*/*"
              }
            ]
          },
          "method": "GET",
          "querystring": "",
          "uri": "/"
        },
        "response": {
          "headers": {
```

```
"access-control-allow-credentials": [  
  {  
    "key": "Access-Control-Allow-Credentials",  
    "value": "true"  
  }  
],  
"access-control-allow-origin": [  
  {  
    "key": "Access-Control-Allow-Origin",  
    "value": "*"   
  }  
],  
"date": [  
  {  
    "key": "Date",  
    "value": "Mon, 13 Jan 2020 20:14:56 GMT"  
  }  
],  
"referrer-policy": [  
  {  
    "key": "Referrer-Policy",  
    "value": "no-referrer-when-downgrade"  
  }  
],  
"server": [  
  {  
    "key": "Server",  
    "value": "ExampleCustomOriginServer"  
  }  
],  
"x-content-type-options": [  
  {  
    "key": "X-Content-Type-Options",  
    "value": "nosniff"  
  }  
],  
"x-frame-options": [  
  {  
    "key": "X-Frame-Options",  
    "value": "DENY"  
  }  
],  
"x-xss-protection": [  
  {
```

```

        "key": "X-XSS-Protection",
        "value": "1; mode=block"
      }
    ],
    "age": [
      {
        "key": "Age",
        "value": "2402"
      }
    ],
    "content-type": [
      {
        "key": "Content-Type",
        "value": "text/html; charset=utf-8"
      }
    ],
    "content-length": [
      {
        "key": "Content-Length",
        "value": "9593"
      }
    ]
  },
  "status": "200",
  "statusDescription": "OK"
}
}
}
]
}

```

回應事件欄位

回應事件物件資料包含在三個子物件中：config (Records.cf.config)、request (Records.cf.request) 和 response (Records.cf.response)。如需請求物件中欄位的詳細資訊，請參閱[請求物件中的欄位](#)。下列清單說明 config 和 response 子物件中的欄位。

Config 物件中的欄位

下列清單說明 config 物件 (Records.cf.config) 中的欄位。

distributionDomainName (唯讀)

與回應關聯的分佈的網域名稱。

distributionID (唯讀)

與回應關聯的分佈 ID。

eventType (唯讀)

與回應關聯的觸發類型：origin-response 或 viewer-response。

requestId (唯讀)

可獨特識別與此回應相關聯之檢視器至 CloudFront 請求的加密字串。此 requestId 值也會在 CloudFront 存取記錄中顯示為 x-edge-request-id。如需詳細資訊，請參閱[存取日誌 \(標準日誌\)](#) 及 [日誌檔案欄位](#)。

回應物件中的欄位

下列清單說明 response 物件 (Records.cf.response) 中的欄位。如需如何使用 Lambda@Edge 函式產生 HTTP 請求的資訊，請參閱[在請求觸發條件中產生 HTTP 回應](#)。

headers (讀取/寫入)

回應中的標頭。注意下列事項：

- 在 headers 物件的金鑰為標準 HTTP 標頭名稱的小寫版本。使用小寫金鑰提供您區分大小寫的標頭值存取權。
- 每個標頭物件 (例如 headers["content-type"] 或 headers["content-length"]) 都是鍵值組的陣列。針對已知的標頭，陣列會在回應中包含每個值的一組鍵值組。
- key 包含在 HTTP 回應中出現的標頭名稱 (區分大小寫)，例如 Content-Type、Content-Length、Cookie 等。
- value 包含在 HTTP 回應中出現的標頭值。
- 當 Lambda 函數新增或修改回應標頭，而您不想要包含標頭 key 欄位時，Lambda@Edge 會使用您提供的標頭名稱來自動插入 key 標頭。無論您如何設定標頭名稱格式，各部分插入的標頭索引鍵名稱都會自動以大寫開頭，以連字號 (-) 分隔。

例如，您可以在沒有 key 標頭的情況下，新增標頭如下：

```
"content-type": [  
  {  
    "value": "text/html;charset=UTF-8"  
  }  
]
```

在此範例中，Lambda@Edge 會自動插入 "key": "Content-Type"。

如需標頭使用限制的詳細資訊，請參閱[對邊緣函數的限制](#)。

status

回應的 HTTP 狀態碼。

statusDescription

回應的 HTTP 狀態說明。

使用請求和回應

若要使用 Lambda@Edge 請求和回應，請參閱下列主題：

主題

- [使用 Lambda@Edge 函數搭配原始伺服器容錯移轉](#)
- [在請求觸發條件中產生 HTTP 回應](#)
- [在原始伺服器回應觸發條件中更新 HTTP 回應](#)
- [選擇包含本文選項來存取請求本文](#)

使用 Lambda@Edge 函數搭配原始伺服器容錯移轉

您可以使用 Lambda@Edge 函數，與您已使用原始伺服器群組來設定的 CloudFront 分佈搭配，例如，針對您設定來確保高可用性的原始伺服器容錯移轉。若要使用 Lambda 函數與原始伺服器群組搭配，請在您建立快取行為時針對原始伺服器群組觸發的原始伺服器請求或原始伺服器回應中指定函數。

如需詳細資訊，請參閱下列內容：

- 建立原始伺服器群組：[建立原始伺服器群組](#)
- 原始伺服器容錯移轉如何使用 Lambda@Edge：[使用原始伺服器容錯移轉與 Lambda@Edge 函數搭配](#)

在請求觸發條件中產生 HTTP 回應

當 CloudFront 收到請求時，您可以使用 Lambda 函數產生 CloudFront 無需轉送回應給原始伺服器，可直接傳回給檢視器的 HTTP 回應。產生 HTTP 回應降低原始伺服器的負載，且通常也可減少檢視器的延遲。

產生 HTTP 回應的一些常見案例包括下列項目：

- 傳回小型網頁給檢視器
- 傳回 HTTP 301 或 302 狀態碼來重新導向使用者到另一個網頁
- 當使用者尚未驗證時，傳回 HTTP 401 狀態碼至檢視器

當下列 CloudFront 事件發生時，Lambda@Edge 函數可以產生 HTTP 回應事件：

檢視器請求事件

當函數被檢視器請求事件觸發時，CloudFront 會將回應傳回檢視器，而且加以不快速获取。

原始伺服器請求事件

當函數被原始伺服器請求事件觸發時，CloudFront 會檢查先前函數產生的邊緣快速获取回應。

- 如果回應在快速获取中，系統不會執行該函數，且 CloudFront 會將快速获取的回應傳回給檢視器。
- 如果回應不在快速获取中，會執行該函數，CloudFront 會將回應傳回給檢視器並加以快速获取。

若要查看產生 HTTP 回應的範本程式碼，請參閱 [Lambda@Edge 範例函數](#)。您也可以在此回應觸發中取代 HTTP 回應。如需詳細資訊，請參閱[在原始伺服器回應觸發條件中更新 HTTP 回應](#)。

程式設計模型

本節說明使用 Lambda@Edge 產生 HTTP 回應的程式設計模型。

主題

- [回應物件](#)
- [錯誤](#)
- [必要欄位](#)

回應物件

以 `result` 方法的 `callback` 參數傳回的回應，需具備下列結構 (請注意，僅 `status` 欄位為必要)。

```
const response = {
  body: 'content',
  bodyEncoding: 'text' | 'base64',
  headers: {
    'header name in lowercase': [{
```

```
        key: 'header name in standard case',
        value: 'header value'
    }},
    ...
},
status: 'HTTP status code (string)',
statusDescription: 'status description'
};
```

回應物件可以包含以下值：

body

如有您希望 CloudFront 在產生的回應中傳回的本文。

bodyEncoding

您在 body 中指定的值的編碼。唯一的有效編碼是 text 和 base64。如果您在 response 物件中加入 body，但省略了 bodyEncoding，則 CloudFront 會將內文視為文字。

如果您將 bodyEncoding 指定為 base64 但本文不是有效的 base64，CloudFront 會傳回錯誤。

headers

您希望 CloudFront 於產生的回應中傳回標頭。注意下列事項：

- 在 headers 物件的金鑰為標準 HTTP 標頭名稱的小寫版本。使用小寫金鑰提供您區分大小寫的標頭值存取權。
- 每個標頭 (例如 headers["accept"] 或 headers["host"]) 是一系列的鍵值組。於已知的標頭，陣列在產生的回應中包含每個值的一組鍵值組。
- key (選用) 為在 HTTP 請求中出現的標頭區分大小寫的名稱，例如 accept 或 host。
- 指定 value 為標頭值。
- 如果您未加入索引鍵/值組的標頭索引鍵部分，Lambda@Edge 會自動使用您提供的標頭名稱，來插入標頭索引鍵。無論您如何安排標頭格式，各部分插入的標頭索引鍵名稱都會自動以大寫開頭，以連字號 (-) 分隔。

例如，您可以不用標頭索引鍵，新增標頭如下：'content-type': [{ value: 'text/html; charset=UTF-8' }]

在這個範例中，Lambda@Edge 建立了下列標頭索引鍵：Content-Type。

如需標頭使用限制的詳細資訊，請參閱[對邊緣函數的限制](#)。

status

HTTP 狀態碼。以字串形式提供狀態碼。CloudFront 使用提供的狀態碼進行以下操作：

- 在回應中傳回
- 當原始伺服器請求事件觸發的函數產生回應時，在 CloudFront 邊緣快取中快取。
- 登入 CloudFront [存取日誌（標準日誌）](#)

如果 status 值不在 200 與 599 之間，CloudFront 會將錯誤傳回至檢視器。

statusDescription

您想要 CloudFront 在回應中傳回的描述能搭配 HTTP 狀態碼。您不需要使用標準的描述，例如 HTTP 200 狀態碼為 OK。

錯誤

以下是產生的 HTTP 回應的可能錯誤。

回應包含本文與指定 204 (無內容) 狀態

當函數是由檢視器請求觸發，當下列二者為 true，CloudFront 會傳回 HTTP 502 狀態碼 (無效的閘道) 至檢視器：

- status 的值是 204 (無內容)
- 回應包含 body 的值

這是因為 Lambda@Edge 強加於 RFC 2616 中選用的限制，也就是 HTTP 204 回應不需要包含訊息本文。

已產生回應的大小限制

由 Lambda 函數產生的回應大小上限取決於觸發函數的事件：

- 檢視器請求事件 – 40 KB
- 原始伺服器請求事件 – 1 MB

如果回應超過允許的大小，CloudFront 會將 HTTP 502 狀態碼 (無效的閘道) 傳回檢視器。

必要欄位

status 欄位是必要的。

所有其他欄位是選用的。

在原始伺服器回應觸發條件中更新 HTTP 回應

當 CloudFront 從原始伺服器收到 HTTP 回應時，如果某個原始伺服器回應觸發關聯到快取行為，您可以修改 HTTP 回應以覆寫原始伺服器傳回的回應。

更新 HTTP 回應的一些常用案例包括下列項目：

- 變更狀態以設定 HTTP 200 狀態碼並建立靜態本文內容，以在原始伺服器傳回錯誤狀態碼 (4xx 和 5xx) 時，傳回給檢視器。如需程式碼範例，請參閱 [範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 200](#)。
- 變更狀態來設定 HTTP 301 或 HTTP 302 狀態碼，以在原始伺服器會傳回錯誤狀態碼 (4xx 和 5xx) 時，將使用者重新導向到另一個網站。如需程式碼範例，請參閱 [範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 302](#)。

Note

函數必須傳回介於 200 (含) 和 599 (含) 之間的狀態值，否則 CloudFront 會將錯誤傳回至檢視器。

您也可以檢閱器與原始伺服器請求事件中取代 HTTP 回應。如需詳細資訊，請參閱[在請求觸發條件中產生 HTTP 回應](#)。

當您使用 HTTP 回應時，Lambda@Edge 不會公開原始伺服器傳回至原始伺服器回應觸發條件的本文。您可以藉由設定為所需的值來產生靜態內容本文，或藉由設定空值來移除函數內的本文。如果您不更新函數的本文欄位，原始伺服器回傳的原始主體會傳回給檢視器。

選擇包含本文選項來存取請求本文

您可以選擇讓 Lambda@Edge 在可寫入的 HTTP 方法 (POST、PUT 和 DELETE 等) 中公開請求的內文，如此您就能在 Lambda 函數中存取該內文。您可以選擇唯讀存取或指定您將替換內文。

若要啟用此選項，請在您為函數建立 CloudFront 觸發條件時 (針對檢視器請求或原始伺服器請求的事件)，選擇 Include Body (包含內文)。如需詳細資訊，請參閱 [為 Lambda@Edge 函數新增觸發條件](#)；或者，若要進一步了解使用包含內文與您的函數，請參閱 [Lambda@Edge 事件結構說明頁面](#)。

您可能會想使用此功能的情境，包括下列的案例：

- 處理 Web 表單，例如「聯絡我們」表單，而不將客戶輸入的資料傳回給原始伺服器。
- 收集由檢視器的瀏覽器所傳送的網站信標資料，並且在邊緣處理這些資料。

如需程式碼範例，請參閱 [Lambda@Edge 範例函數](#)。

Note

如果請求主體的資料大小過大，Lambda@Edge 會將主體截斷。如需大小上限和截斷的詳細資訊，請參閱[使用包含內文選項時的要求內文限制](#)。

Lambda@Edge 範例函數

請參閱下列透過 Amazon CloudFront 使用 Lambda 函數的範例。

Note

如果您為 Lambda@Edge 函數選擇執行時期 Node.js 18 或更新版本，系統會自動為您建立 `index.mjs` 檔案。若要使用下列程式碼範例，請將 `index.mjs` 檔案重新命名為 `index.js`。

主題

- [一般範例](#)
- [產生回應：範例](#)
- [查詢字串：範例](#)
- [根據國家/地區或裝置類型標頭個人化 - 範例](#)
- [以內容為基礎的動態原始伺服器選擇 - 範例](#)
- [更新錯誤狀態：範例](#)
- [存取請求本文：範例](#)

一般範例

下列範例顯示在 CloudFront 中使用 Lambda@Edge 的常見方法。

主題

- [範例：A/B 測試](#)
- [範例：覆寫回應標頭](#)

範例：A/B 測試

您可以使用下列範例測試兩種不同版本的影像，而不需要建立重新導向或變更 URL。此範例會讀取檢視器請求中的 Cookie，並據此修改請求 URL。如果檢視器未傳送具有其中一個預期值的 Cookie，此範例會將檢視器隨機指派給其中一個 URL。

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;
  const headers = request.headers;

  if (request.uri !== '/experiment-pixel.jpg') {
    // do not process if this is not an A-B test request
    callback(null, request);
    return;
  }

  const cookieExperimentA = 'X-Experiment-Name=A';
  const cookieExperimentB = 'X-Experiment-Name=B';
  const pathExperimentA = '/experiment-group/control-pixel.jpg';
  const pathExperimentB = '/experiment-group/treatment-pixel.jpg';

  /*
   * Lambda at the Edge headers are array objects.
   *
   * Client may send multiple Cookie headers, i.e.:
   * > GET /viewerRes/test HTTP/1.1
   * > User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1
   * > Cookie: First=1; Second=2
   * > Cookie: ClientCode=abc
   * > Host: example.com
   *
   * You can access the first Cookie header at headers["cookie"][0].value
   * and the second at headers["cookie"][1].value.
   */
}
```

```

    * Header values are not parsed. In the example above,
    * headers["cookie"][0].value is equal to "First=1; Second=2"
    */
let experimentUri;
if (headers.cookie) {
    for (let i = 0; i < headers.cookie.length; i++) {
        if (headers.cookie[i].value.indexOf(cookieExperimentA) >= 0) {
            console.log('Experiment A cookie found');
            experimentUri = pathExperimentA;
            break;
        } else if (headers.cookie[i].value.indexOf(cookieExperimentB) >= 0) {
            console.log('Experiment B cookie found');
            experimentUri = pathExperimentB;
            break;
        }
    }
}

if (!experimentUri) {
    console.log('Experiment cookie has not been found. Throwing dice...');
    if (Math.random() < 0.75) {
        experimentUri = pathExperimentA;
    } else {
        experimentUri = pathExperimentB;
    }
}

request.uri = experimentUri;
console.log(`Request uri set to "${request.uri}"`);
callback(null, request);
};

```

Python

```

import json
import random

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    if request['uri'] != '/experiment-pixel.jpg':
        # Not an A/B Test

```

```

    return request

    cookieExperimentA, cookieExperimentB = 'X-Experiment-Name=A', 'X-Experiment-
Name=B'
    pathExperimentA, pathExperimentB = '/experiment-group/control-pixel.jpg', '/
experiment-group/treatment-pixel.jpg'

    ...

Lambda at the Edge headers are array objects.

Client may send multiple cookie headers. For example:
> GET /viewerRes/test HTTP/1.1
> User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1
OpenSSL/1.0.1u zlib/1.2.3
> Cookie: First=1; Second=2
> Cookie: ClientCode=abc
> Host: example.com

You can access the first Cookie header at headers["cookie"][0].value
and the second at headers["cookie"][1].value.

Header values are not parsed. In the example above,
headers["cookie"][0].value is equal to "First=1; Second=2"
...

experimentUri = ""

for cookie in headers.get('cookie', []):
    if cookieExperimentA in cookie['value']:
        print("Experiment A cookie found")
        experimentUri = pathExperimentA
        break
    elif cookieExperimentB in cookie['value']:
        print("Experiment B cookie found")
        experimentUri = pathExperimentB
        break

if not experimentUri:
    print("Experiment cookie has not been found. Throwing dice...")
    if random.random() < 0.75:
        experimentUri = pathExperimentA
    else:
        experimentUri = pathExperimentB

```

```
request['uri'] = experimentUri
print(f"Request uri set to {experimentUri}")
return request
```

範例：覆寫回應標頭

以下範例說明如何根據另一個標頭的值變更回應標頭的值。

Node.js

```
export const handler = async (event) => {
  const response = event.Records[0].cf.response;
  const headers = response.headers;

  const headerNameSrc = 'X-Amz-Meta-Last-Modified';
  const headerNameDst = 'Last-Modified';

  if (headers[headerNameSrc.toLowerCase()]) {
    headers[headerNameDst.toLowerCase()] = [{
      key: headerNameDst,
      value: headers[headerNameSrc.toLowerCase()][0].value,
    }];
    console.log(`Response header "${headerNameDst}" was set to ` +
      `"${headers[headerNameDst.toLowerCase()][0].value}"`);
  }

  return response;
};
```

Python

```
import json

def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']
    headers = response['headers']

    header_name_src = 'X-Amz-Meta-Last-Modified'
    header_name_dst = 'Last-Modified'

    if headers.get(header_name_src.lower()):
        headers[header_name_dst.lower()] = [{
```

```
        'key': header_name_dst,
        'value': headers[header_name_src.lower()][0]['value']
    ]}
    print(f'Response header "{header_name_dst}" was set to '
          f'"{headers[header_name_dst.lower()][0]["value"]}"')

    return response
```

產生回應：範例

以下範例示範如何使用 Lambda@Edge 來產生回應。

主題

- [範例：提供靜態內容 \(產生回應\)](#)
- [範例：產生 HTTP 重新導向 \(產生回應\)](#)

範例：提供靜態內容 (產生回應)

以下範例說明如何使用 Lambda 函數提供靜態網站內容，其可減少原始伺服器的負載，並降低整體延遲。

Note

您可以針對檢視器要求及原始伺服器請求事件產生 HTTP 回應。如需詳細資訊，請參閱[the section called “在請求觸發條件中產生 HTTP 回應”](#)。

您也可以在原伺服器回應請求事件中取代或移除 HTTP 回應的主體。如需詳細資訊，請參閱[the section called “在原始伺服器回應觸發條件中更新 HTTP 回應”](#)。

Node.js

```
'use strict';

const content = `
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Simple Lambda@Edge Static Content Response</title>
```

```

    </head>
    <body>
      <p>Hello from Lambda@Edge!</p>
    </body>
  </html>
`;

exports.handler = (event, context, callback) => {
  /*
   * Generate HTTP OK response using 200 status code with HTML body.
   */
  const response = {
    status: '200',
    statusDescription: 'OK',
    headers: {
      'cache-control': [{
        key: 'Cache-Control',
        value: 'max-age=100'
      }],
      'content-type': [{
        key: 'Content-Type',
        value: 'text/html'
      }]
    },
    body: content,
  };
  callback(null, response);
};

```

Python

```

import json

CONTENT = """
<\!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Simple Lambda@Edge Static Content Response</title>
</head>
<body>
  <p>Hello from Lambda@Edge!</p>
</body>

```



```
</html>
"""

def lambda_handler(event, context):
    # Generate HTTP OK response using 200 status code with HTML body.
    response = {
        'status': '200',
        'statusDescription': 'OK',
        'headers': {
            'cache-control': [
                {
                    'key': 'Cache-Control',
                    'value': 'max-age=100'
                }
            ],
            "content-type": [
                {
                    'key': 'Content-Type',
                    'value': 'text/html'
                }
            ]
        },
        'body': CONTENT
    }
    return response
```

範例：產生 HTTP 重新導向 (產生回應)

以下範例說明如何產生 HTTP 重新導向。

Note

您可以針對檢視器要求及原始伺服器請求事件產生 HTTP 回應。如需詳細資訊，請參閱[在請求觸發條件中產生 HTTP 回應](#)。

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    /*
```

```
    * Generate HTTP redirect response with 302 status code and Location header.
    */
const response = {
  status: '302',
  statusDescription: 'Found',
  headers: {
    location: [{
      key: 'Location',
      value: 'https://docs.aws.amazon.com/lambda/latest/dg/lambda-
edge.html',
    }],
  },
};
callback(null, response);
};
```

Python

```
def lambda_handler(event, context):

    # Generate HTTP redirect response with 302 status code and Location header.

    response = {
        'status': '302',
        'statusDescription': 'Found',
        'headers': {
            'location': [{
                'key': 'Location',
                'value': 'https://docs.aws.amazon.com/lambda/latest/dg/lambda-
edge.html'
            }]
        }
    }

    return response
```

查詢字串：範例

以下範例示範以查詢字串使用 Lambda@Edge 的方式。

主題

- [範例：根據查詢字串參數新增標頭](#)

- [範例：標準化查詢字串參數，以提升快取點擊率](#)
- [範例：重新導向未驗證使用者到登入頁面](#)

範例：根據查詢字串參數新增標頭

以下範例說明如何取得查詢字串參數的鍵值對，然後根據這些值新增標頭。

Node.js

```
'use strict';

const querystring = require('querystring');
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /* When a request contains a query string key-value pair but the origin server
     * expects the value in a header, you can use this Lambda function to
     * convert the key-value pair to a header. Here's what the function does:
     * 1. Parses the query string and gets the key-value pair.
     * 2. Adds a header to the request using the key-value pair that the function
     * got in step 1.
     */

    /* Parse request querystring to get javascript object */
    const params = querystring.parse(request.querystring);

    /* Move auth param from querystring to headers */
    const headerName = 'Auth-Header';
    request.headers[headerName.toLowerCase()] = [{ key: headerName, value:
params.auth }];
    delete params.auth;

    /* Update request querystring */
    request.querystring = querystring.stringify(params);

    callback(null, request);
};
```

Python

```
from urllib.parse import parse_qs, urlencode
```

```
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    '''
    When a request contains a query string key-value pair but the origin server
    expects the value in a header, you can use this Lambda function to
    convert the key-value pair to a header. Here's what the function does:
        1. Parses the query string and gets the key-value pair.
        2. Adds a header to the request using the key-value pair that the function
    got in step 1.
    '''

    # Parse request querystring to get dictionary/json
    params = {k : v[0] for k, v in parse_qs(request['querystring']).items()}

    # Move auth param from querystring to headers
    headerName = 'Auth-Header'
    request['headers'][headerName.lower()] = [{'key': headerName, 'value':
params['auth']}]
    del params['auth']

    # Update request querystring
    request['querystring'] = urlencode(params)

    return request
```

範例：標準化查詢字串參數，以提升快取點擊率

以下範例說明如何在 CloudFront 轉送請求至您的原始伺服器之前，做以下變更至查詢字串以提高您的快取點擊率：

- 依字母排序鍵值組的參數名稱。
- 將鍵值組變更為小寫。

如需詳細資訊，請參閱[根據查詢字串參數快取內容](#)。

Node.js

```
'use strict';

const querystring = require('querystring');
```

```

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;
  /* When you configure a distribution to forward query strings to the origin and
   * to cache based on an allowlist of query string parameters, we recommend
   * the following to improve the cache-hit ratio:
   * - Always list parameters in the same order.
   * - Use the same case for parameter names and values.
   *
   * This function normalizes query strings so that parameter names and values
   * are lowercase and parameter names are in alphabetical order.
   *
   * For more information, see:
   * https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
QueryStringParameters.html
   */

  console.log('Query String: ', request.querystring);

  /* Parse request query string to get javascript object */
  const params = querystring.parse(request.querystring.toLowerCase());
  const sortedParams = {};

  /* Sort param keys */
  Object.keys(params).sort().forEach(key => {
    sortedParams[key] = params[key];
  });

  /* Update request querystring with normalized */
  request.querystring = querystring.stringify(sortedParams);

  callback(null, request);
};

```

Python

```

from urllib.parse import parse_qs, urlencode

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    ...

    When you configure a distribution to forward query strings to the origin and
    to cache based on an allowlist of query string parameters, we recommend

```

the following to improve the cache-hit ratio:

Always list parameters in the same order.

- Use the same case for parameter names and values.

This function normalizes query strings so that parameter names and values are lowercase and parameter names are in alphabetical order.

For more information, see:

[https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/QueryStringParameters.html)

[QueryStringParameters.html](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/QueryStringParameters.html)

```
...
print("Query string: ", request["querystring"])

# Parse request query string to get js object
params = {k : v[0] for k, v in parse_qs(request['querystring'].lower()).items()}

# Sort param keys
sortedParams = sorted(params.items(), key=lambda x: x[0])

# Update request querystring with normalized
request['querystring'] = urlencode(sortedParams)

return request
```

範例：重新導向未驗證使用者到登入頁面

以下範例說明，如果尚未輸入他們的登入資料，如何將使用者重新導向至登入頁面。

Node.js

```
'use strict';

function parseCookies(headers) {
  const parsedCookie = {};
  if (headers.cookie) {
    headers.cookie[0].value.split(';').forEach((cookie) => {
      if (cookie) {
        const parts = cookie.split('=');
        parsedCookie[parts[0].trim()] = parts[1].trim();
      }
    });
  }
  return parsedCookie;
}
```

```

}

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;
  const headers = request.headers;

  /* Check for session-id in request cookie in viewer-request event,
   * if session-id is absent, redirect the user to sign in page with original
   * request sent as redirect_url in query params.
   */

  /* Check for session-id in cookie, if present then proceed with request */
  const parsedCookies = parseCookies(headers);
  if (parsedCookies && parsedCookies['session-id']) {
    callback(null, request);
    return;
  }

  /* URI encode the original request to be sent as redirect_url in query params */
  const encodedRedirectUrl = encodeURIComponent(`https://${headers.host[0].value}${request.uri}?${request.querystring}`);
  const response = {
    status: '302',
    statusDescription: 'Found',
    headers: {
      location: [{
        key: 'Location',
        value: `https://www.example.com/signin?redirect_url=${encodedRedirectUrl}`,
      }],
    },
  };
  callback(null, response);
};

```

Python

```

import urllib

def parseCookies(headers):
    parsedCookie = {}
    if headers.get('cookie'):
        for cookie in headers['cookie'][0]['value'].split(';'):

```

```

        if cookie:
            parts = cookie.split('=')
            parsedCookie[parts[0].strip()] = parts[1].strip()
    return parsedCookie

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    '''
    Check for session-id in request cookie in viewer-request event,
    if session-id is absent, redirect the user to sign in page with original
    request sent as redirect_url in query params.
    '''

    # Check for session-id in cookie, if present, then proceed with request
    parsedCookies = parseCookies(headers)

    if parsedCookies and parsedCookies['session-id']:
        return request

    # URI encode the original request to be sent as redirect_url in query params
    redirectUrl = "https://%s%s?%s" % (headers['host'][0]['value'], request['uri'],
    request['querystring'])
    encodedRedirectUrl = urllib.parse.quote_plus(redirectUrl.encode('utf-8'))

    response = {
        'status': '302',
        'statusDescription': 'Found',
        'headers': {
            'location': [{
                'key': 'Location',
                'value': 'https://www.example.com/signin?redirect_url=%s' %
encodedRedirectUrl
            }]
        }
    }
    return response

```

根據國家/地區或裝置類型標頭個人化 - 範例

以下範例說明如何使用 Lambda@Edge 來根據位置或檢視器使用的裝置類型自訂行為。

主題

- [範例：將檢視器請求重新導向到國家/地區特定的 URL](#)
- [範例：根據裝置提供不同的物件版本](#)

範例：將檢視器請求重新導向到國家/地區特定的 URL

以下範例說明如何產生含國家/地區特定 URL 的 HTTP 重新導向回應，並將回應傳回至檢視器。當您希望提供特定國家的回應時，此方法很有用。例如：

- 如果您有國家/地區特定的子網域 (例如 us.example.com 和 tw.example.com)，當檢視器請求 example.com 時，您可以產生一個重新導向回應。
- 如果您在串流視訊，但沒有在特定國家/地區串流此內容的權利，您可以在該國家/地區將使用者重新導向到說明他們為何無法檢視影片的頁面。

注意下列事項：

- 您必須根據 CloudFront-Viewer-Country 標頭設定您的分佈為快取。如需詳細資訊，請參閱[根據選取請求標頭的快取](#)。
- 在檢視器請求事件後，CloudFront 新增 CloudFront-Viewer-Country 標頭。要使用此範例，您必須建立原始伺服器請求事件的觸發。

Node.js

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;
  const headers = request.headers;

  /*
   * Based on the value of the CloudFront-Viewer-Country header, generate an
   * HTTP status code 302 (Redirect) response, and return a country-specific
   * URL in the Location header.
   * NOTE: 1. You must configure your distribution to cache based on the
   *         CloudFront-Viewer-Country header. For more information, see
   *         https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
   */
}
```

```

*           2. CloudFront adds the CloudFront-Viewer-Country header after the
viewer
*           request event. To use this example, you must create a trigger for
the
*           origin request event.
*/

let url = 'https://example.com/';
if (headers['cloudfront-viewer-country']) {
  const countryCode = headers['cloudfront-viewer-country'][0].value;
  if (countryCode === 'TW') {
    url = 'https://tw.example.com/';
  } else if (countryCode === 'US') {
    url = 'https://us.example.com/';
  }
}

const response = {
  status: '302',
  statusDescription: 'Found',
  headers: {
    location: [{
      key: 'Location',
      value: url,
    }],
  },
};
callback(null, response);
};

```

Python

```

# This is an origin request function

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    ...

    Based on the value of the CloudFront-Viewer-Country header, generate an
    HTTP status code 302 (Redirect) response, and return a country-specific
    URL in the Location header.
    NOTE: 1. You must configure your distribution to cache based on the

```

```
CloudFront-Viewer-Country header. For more information, see
https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
2. CloudFront adds the CloudFront-Viewer-Country header after the viewer
request event. To use this example, you must create a trigger for the
origin request event.

...

url = 'https://example.com/'
viewerCountry = headers.get('cloudfront-viewer-country')
if viewerCountry:
    countryCode = viewerCountry[0]['value']
    if countryCode == 'TW':
        url = 'https://tw.example.com/'
    elif countryCode == 'US':
        url = 'https://us.example.com/'

response = {
    'status': '302',
    'statusDescription': 'Found',
    'headers': {
        'location': [{
            'key': 'Location',
            'value': url
        }]
    }
}

return response
```

範例：根據裝置提供不同的物件版本

以下範例說明如何根據使用者使用的裝置類型，提供不同的物件版本，例如行動裝置或平板電腦。注意下列事項：

- 您必須根據 CloudFront-Is-*-Viewer 標頭設定您的分佈為快取。如需詳細資訊，請參閱[根據選取請求標頭的快取](#)。
- 在檢視器請求事件後，CloudFront 新增 CloudFront-Is-*-Viewer 標頭。要使用此範例，您必須建立原始伺服器請求事件的觸發。

Node.js

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;
  const headers = request.headers;

  /*
   * Serve different versions of an object based on the device type.
   * NOTE: 1. You must configure your distribution to cache based on the
   *         CloudFront-Is-*Viewer headers. For more information, see
   *         the following documentation:
   *         https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-
headers
   *         https://docs.aws.amazon.com/console/cloudfront/cache-on-device-type
   *         2. CloudFront adds the CloudFront-Is-*Viewer headers after the viewer
   *         request event. To use this example, you must create a trigger for
the
   *         origin request event.
   */

  const desktopPath = '/desktop';
  const mobilePath = '/mobile';
  const tabletPath = '/tablet';
  const smarttvPath = '/smarttv';

  if (headers['cloudfront-is-desktop-viewer']
    && headers['cloudfront-is-desktop-viewer'][0].value === 'true') {
    request.uri = desktopPath + request.uri;
  } else if (headers['cloudfront-is-mobile-viewer']
    && headers['cloudfront-is-mobile-viewer'][0].value === 'true') {
    request.uri = mobilePath + request.uri;
  } else if (headers['cloudfront-is-tablet-viewer']
    && headers['cloudfront-is-tablet-viewer'][0].value === 'true') {
    request.uri = tabletPath + request.uri;
  } else if (headers['cloudfront-is-smarttv-viewer']
    && headers['cloudfront-is-smarttv-viewer'][0].value === 'true') {
    request.uri = smarttvPath + request.uri;
  }
  console.log(`Request uri set to "${request.uri}"`);

  callback(null, request);
}
```

```
};
```

Python

```
# This is an origin request function
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    headers = request['headers']

    ...

    Serve different versions of an object based on the device type.
    NOTE: 1. You must configure your distribution to cache based on the
           CloudFront-Is-*-Viewer headers. For more information, see
           the following documentation:
           https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
           https://docs.aws.amazon.com/console/cloudfront/cache-on-device-type
           2. CloudFront adds the CloudFront-Is-*-Viewer headers after the viewer
           request event. To use this example, you must create a trigger for the
           origin request event.
    ...

    desktopPath = '/desktop';
    mobilePath = '/mobile';
    tabletPath = '/tablet';
    smarttvPath = '/smarttv';

    if 'cloudfront-is-desktop-viewer' in headers and headers['cloudfront-is-desktop-viewer'][0]['value'] == 'true':
        request['uri'] = desktopPath + request['uri']
    elif 'cloudfront-is-mobile-viewer' in headers and headers['cloudfront-is-mobile-viewer'][0]['value'] == 'true':
        request['uri'] = mobilePath + request['uri']
    elif 'cloudfront-is-tablet-viewer' in headers and headers['cloudfront-is-tablet-viewer'][0]['value'] == 'true':
        request['uri'] = tabletPath + request['uri']
    elif 'cloudfront-is-smarttv-viewer' in headers and headers['cloudfront-is-smarttv-viewer'][0]['value'] == 'true':
        request['uri'] = smarttvPath + request['uri']

    print("Request uri set to %s" % request['uri'])

    return request
```

以內容為基礎的動態原始伺服器選擇 - 範例

以下範例示範如何使用 Lambda@Edge 來根據請求中的資訊路由到不同的原始伺服器。

主題

- [範例：使用原始伺服器請求觸發條件，以從自訂原始伺服器變更到 Amazon S3 原始伺服器](#)
- [範例：使用原始伺服器請求觸發條件，以變更 Amazon S3 原始伺服器區域](#)
- [範例：使用原始伺服器請求觸發條件，變更 Amazon S3 原始伺服器為自訂原始伺服器](#)
- [範例：使用原始伺服器請求觸發條件，將流量自一個 Amazon S3 儲存貯體逐漸傳輸到另一個。](#)
- [範例：使用原始伺服器請求觸發，根據國家/地區標頭變更原始網域名稱](#)

範例：使用原始伺服器請求觸發條件，以從自訂原始伺服器變更到 Amazon S3 原始伺服器

此函數示範如何使用 origin-request 觸發條件，根據請求屬性從自訂原始伺服器變更至內容被擷取的 Amazon S3 原始伺服器。

Node.js

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;

  /**
   * Reads query string to check if S3 origin should be used, and
   * if true, sets S3 origin properties.
   */

  const params = querystring.parse(request.querystring);

  if (params['useS3origin']) {
    if (params['useS3origin'] === 'true') {
      const s3DomainName = 'amzn-s3-demo-bucket.s3.amazonaws.com';

      /* Set S3 origin fields */
      request.origin = {
        s3: {
          domainName: s3DomainName,
```

```

        region: '',
        authMethod: 'origin-access-identity',
        path: '',
        customHeaders: {}
    }
};
request.headers['host'] = [{ key: 'host', value: s3DomainName}];
}
}

callback(null, request);
};

```

Python

```

from urllib.parse import parse_qs

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    '''
    Reads query string to check if S3 origin should be used, and
    if true, sets S3 origin properties
    '''
    params = {k: v[0] for k, v in parse_qs(request['queryString']).items()}
    if params.get('useS3Origin') == 'true':
        s3DomainName = 'amzn-s3-demo-bucket.s3.amazonaws.com'

        # Set S3 origin fields
        request['origin'] = {
            's3': {
                'domainName': s3DomainName,
                'region': '',
                'authMethod': 'origin-access-identity',
                'path': '',
                'customHeaders': {}
            }
        }
        request['headers']['host'] = [{'key': 'host', 'value': s3DomainName}]
    return request

```

範例：使用原始伺服器請求觸發條件，以變更 Amazon S3 原始伺服器區域

此函數示範如何根據請求屬性，使用 origin-request 觸發條件來變更內容被擷取的 Amazon S3 原始伺服器。

在這個範例中，我們使用 CloudFront-Viewer-Country 標頭的值來將 S3 儲存貯體的網域名稱更新為較靠近檢視器的區域內的儲存貯體。這在數種方式中非常受用：

- 當指定的區域更靠近檢視器的國家/地區時，能減少延遲。
- 藉由確保該資料與請求來自位於相同國家/地區的原始伺服器，提供資料主權服務。

要使用此範例，須執行下列項目：

- 您必須根據 CloudFront-Viewer-Country 標頭設定您的分佈為快取。如需詳細資訊，請參閱[根據選取請求標頭的快取](#)。
- 在原始伺服器請求事件中建立此函數的觸發。CloudFront 將在檢視者請求事件後新增 CloudFront-Viewer-Country 標頭，所以若要使用此範例，必須確認函數會執行原始請求。

Note

下列範例程式碼針對您用於原始伺服器的所有 S3 儲存貯體使用相同的原始存取身分 (OAI)。如需更多詳細資訊，請參閱[原始存取身分](#)。

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;

  /**
   * This blueprint demonstrates how an origin-request trigger can be used to
   * change the origin from which the content is fetched, based on request
   * properties.
   * In this example, we use the value of the CloudFront-Viewer-Country header
   * to update the S3 bucket domain name to a bucket in a Region that is closer to
   * the viewer.
   *
   * This can be useful in several ways:
```



```

*      1) Reduces latencies when the Region specified is nearer to the viewer's
*      country.
*      2) Provides data sovereignty by making sure that data is served from an
*      origin that's in the same country that the request came from.
*
* NOTE: 1. You must configure your distribution to cache based on the
*      CloudFront-Viewer-Country header. For more information, see
*      https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-
headers
*      2. CloudFront adds the CloudFront-Viewer-Country header after the
viewer
*      request event. To use this example, you must create a trigger for
the
*      origin request event.
*/

const countryToRegion = {
  'DE': 'eu-central-1',
  'IE': 'eu-west-1',
  'GB': 'eu-west-2',
  'FR': 'eu-west-3',
  'JP': 'ap-northeast-1',
  'IN': 'ap-south-1'
};

if (request.headers['cloudfront-viewer-country']) {
  const countryCode = request.headers['cloudfront-viewer-country'][0].value;
  const region = countryToRegion[countryCode];

  /**
   * If the viewer's country is not in the list you specify, the request
   * goes to the default S3 bucket you've configured.
   */
  if (region) {
    /**
     * If you've set up OAI, the bucket policy in the destination bucket
     * should allow the OAI GetObject operation, as configured by default
     * for an S3 origin with OAI. Another requirement with OAI is to provide
     * the Region so it can be used for the SIGV4 signature. Otherwise, the
     * Region is not required.
     */
    request.origin.s3.region = region;
    const domainName = `amzn-s3-demo-bucket-in-${region}.s3.
    ${region}.amazonaws.com`;
  }
}

```

```

        request.origin.s3.domainName = domainName;
        request.headers['host'] = [{ key: 'host', value: domainName }];
    }
}

callback(null, request);
};

```

Python

```

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    ...

    This blueprint demonstrates how an origin-request trigger can be used to
    change the origin from which the content is fetched, based on request
    properties.

    In this example, we use the value of the CloudFront-Viewer-Country header
    to update the S3 bucket domain name to a bucket in a Region that is closer to
    the viewer.

    This can be useful in several ways:
        1) Reduces latencies when the Region specified is nearer to the viewer's
           country.
        2) Provides data sovereignty by making sure that data is served from an
           origin that's in the same country that the request came from.

    NOTE: 1. You must configure your distribution to cache based on the
           CloudFront-Viewer-Country header. For more information, see
           https://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
        2. CloudFront adds the CloudFront-Viewer-Country header after the viewer
           request event. To use this example, you must create a trigger for the
           origin request event.

    ...

    countryToRegion = {
        'DE': 'eu-central-1',
        'IE': 'eu-west-1',
        'GB': 'eu-west-2',
        'FR': 'eu-west-3',
        'JP': 'ap-northeast-1',
        'IN': 'ap-south-1'
    }

```

```

viewerCountry = request['headers'].get('cloudfront-viewer-country')
if viewerCountry:
    countryCode = viewerCountry[0]['value']
    region = countryToRegion.get(countryCode)

    # If the viewer's country is not in the list you specify, the request
    # goes to the default S3 bucket you've configured
    if region:
        '''
        If you've set up OAI, the bucket policy in the destination bucket
        should allow the OAI GetObject operation, as configured by default
        for an S3 origin with OAI. Another requirement with OAI is to provide
        the Region so it can be used for the SIGV4 signature. Otherwise, the
        Region is not required.
        '''
        request['origin']['s3']['region'] = region
        domainName = 'amzn-s3-demo-bucket-in-{0}.s3.
{0}.amazonaws.com'.format(region)
        request['origin']['s3']['domainName'] = domainName
        request['headers']['host'] = [{'key': 'host', 'value': domainName}]

    return request

```

範例：使用原始伺服器請求觸發條件，變更 Amazon S3 原始伺服器為自訂原始伺服器

此函數示範如何根據請求屬性，使用原始伺服器請求觸發來變更內容被擷取的自訂原始伺服器。

Node.js

```

'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /**
     * Reads query string to check if custom origin should be used, and
     * if true, sets custom origin properties.
     */

    const params = querystring.parse(request.querystring);

```

```

    if (params['useCustomOrigin']) {
        if (params['useCustomOrigin'] === 'true') {

            /* Set custom origin fields*/
            request.origin = {
                custom: {
                    domainName: 'www.example.com',
                    port: 443,
                    protocol: 'https',
                    path: '',
                    sslProtocols: ['TLSv1', 'TLSv1.1'],
                    readTimeout: 5,
                    keepaliveTimeout: 5,
                    customHeaders: {}
                }
            };
            request.headers['host'] = [{ key: 'host', value: 'www.example.com'}];
        }
    }
    callback(null, request);
};

```

Python

```

from urllib.parse import parse_qs

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    # Reads query string to check if custom origin should be used, and
    # if true, sets custom origin properties

    params = {k: v[0] for k, v in parse_qs(request['queryString']).items()}

    if params.get('useCustomOrigin') == 'true':
        # Set custom origin fields
        request['origin'] = {
            'custom': {
                'domainName': 'www.example.com',
                'port': 443,
                'protocol': 'https',
                'path': '',

```

```

        'sslProtocols': ['TLSv1', 'TLSv1.1'],
        'readTimeout': 5,
        'keepaliveTimeout': 5,
        'customHeaders': {}
    }
}
request['headers']['host'] = [{ 'key': 'host', 'value':
'www.example.com' }]

return request

```

範例：使用原始伺服器請求觸發條件，將流量自一個 Amazon S3 儲存貯體逐漸傳輸到另一個。

此函數示範如何以可控的方式，從一個 Amazon S3 儲存貯體逐步傳輸流量到另一個儲存貯體。

Node.js

```

'use strict';

function getRandomInt(min, max) {
    /* Random number is inclusive of min and max*/
    return Math.floor(Math.random() * (max - min + 1)) + min;
}

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const BLUE_TRAFFIC_PERCENTAGE = 80;

    /**
     * This Lambda function demonstrates how to gradually transfer traffic from
     * one S3 bucket to another in a controlled way.
     * We define a variable BLUE_TRAFFIC_PERCENTAGE which can take values from
     * 1 to 100. If the generated randomNumber less than or equal to
    BLUE_TRAFFIC_PERCENTAGE, traffic
     * is re-directed to blue-bucket. If not, the default bucket that we've
    configured
     * is used.
     */

    const randomNumber = getRandomInt(1, 100);

    if (randomNumber <= BLUE_TRAFFIC_PERCENTAGE) {
        const domainName = 'blue-bucket.s3.amazonaws.com';
    }
}

```

```

        request.origin.s3.domainName = domainName;
        request.headers['host'] = [{ key: 'host', value: domainName}];
    }
    callback(null, request);
};

```

Python

```

import math
import random

def getRandomInt(min, max):
    # Random number is inclusive of min and max
    return math.floor(random.random() * (max - min + 1)) + min

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    BLUE_TRAFFIC_PERCENTAGE = 80

    '''
    This Lambda function demonstrates how to gradually transfer traffic from
    one S3 bucket to another in a controlled way.
    We define a variable BLUE_TRAFFIC_PERCENTAGE which can take values from
    1 to 100. If the generated randomNumber less than or equal to
    BLUE_TRAFFIC_PERCENTAGE, traffic
    is re-directed to blue-bucket. If not, the default bucket that we've configured
    is used.
    '''

    randomNumber = getRandomInt(1, 100)

    if randomNumber <= BLUE_TRAFFIC_PERCENTAGE:
        domainName = 'blue-bucket.s3.amazonaws.com'
        request['origin']['s3']['domainName'] = domainName
        request['headers']['host'] = [{'key': 'host', 'value': domainName}]

    return request

```

範例：使用原始伺服器請求觸發，根據國家/地區標頭變更原始網域名稱

此函數示範如何根據 CloudFront-Viewer-Country 標頭來變更原始伺服器的網域名稱，使內容能從靠近檢視者國家/地區的原始伺服器提供。

為您的分佈實施此功能可以擁有如下所述的優勢：

- 當指定的區域更靠近檢視器的國家/地區時，可減少延遲
- 藉由確保該資料與請求來自位於相同國家/地區的原始伺服器來提供資料主權服務

請注意，要啟用此功能，您必須根據 CloudFront-Viewer-Country 標頭設定您的分佈為快取。如需詳細資訊，請參閱[the section called “根據選取請求標頭的快取”](#)。

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    if (request.headers['cloudfront-viewer-country']) {
        const countryCode = request.headers['cloudfront-viewer-country'][0].value;
        if (countryCode === 'GB' || countryCode === 'DE' || countryCode === 'IE' )
        {
            const domainName = 'eu.example.com';
            request.origin.custom.domainName = domainName;
            request.headers['host'] = [{key: 'host', value: domainName}];
        }
    }

    callback(null, request);
};
```

Python

```
def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    viewerCountry = request['headers'].get('cloudfront-viewer-country')
    if viewerCountry:
        countryCode = viewerCountry[0]['value']
        if countryCode == 'GB' or countryCode == 'DE' or countryCode == 'IE':
            domainName = 'eu.example.com'
            request['origin']['custom']['domainName'] = domainName
            request['headers']['host'] = [{'key': 'host', 'value': domainName}]
    return request
```

更新錯誤狀態：範例

以下範例提供指導，說明如何使用 Lambda@Edge 來變更傳回給使用者的錯誤狀態。

主題

- [範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 200](#)
- [範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 302](#)

範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 200

此函數示範如何更新回應狀態為 200 並產生靜態本文內容，以在以下案例傳回給檢視器：

- 函數在原始伺服器回應中觸發。
- 原始伺服器的回應狀態為錯誤狀態碼 (4xx 和 5xx)

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
  const response = event.Records[0].cf.response;

  /**
   * This function updates the response status to 200 and generates static
   * body content to return to the viewer in the following scenario:
   * 1. The function is triggered in an origin response
   * 2. The response status from the origin server is an error status code (4xx or
   5xx)
   */

  if (response.status >= 400 && response.status <= 599) {
    response.status = 200;
    response.statusDescription = 'OK';
    response.body = 'Body generation example';
  }

  callback(null, response);
};
```


Python

```
def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']

    '''
    This function updates the response status to 200 and generates static
    body content to return to the viewer in the following scenario:
    1. The function is triggered in an origin response
    2. The response status from the origin server is an error status code (4xx or
    5xx)
    '''

    if int(response['status']) >= 400 and int(response['status']) <= 599:
        response['status'] = 200
        response['statusDescription'] = 'OK'
        response['body'] = 'Body generation example'
    return response
```

範例：使用原始伺服器請求觸發以更新錯誤狀態碼為 302

此函數示範如何更新 HTTP 狀態碼為 302，以重新導向到另一個由不同原始伺服器設定的路徑 (快取行為)。注意下列事項：

- 函數在原始伺服器回應中觸發。
- 原始伺服器的回應狀態為錯誤狀態碼 (4xx 和 5xx)

Node.js

```
'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;
    const request = event.Records[0].cf.request;

    /**
     * This function updates the HTTP status code in the response to 302, to
     * redirect to another
     * path (cache behavior) that has a different origin configured. Note the
     * following:
     * 1. The function is triggered in an origin response
```

```

    * 2. The response status from the origin server is an error status code (4xx or
    5xx)
    */

    if (response.status >= 400 && response.status <= 599) {
        const redirect_path = `/plan-b/path?${request.querystring}`;

        response.status = 302;
        response.statusDescription = 'Found';

        /* Drop the body, as it is not required for redirects */
        response.body = '';
        response.headers['location'] = [{ key: 'Location', value: redirect_path }];
    }

    callback(null, response);
};

```

Python

```

def lambda_handler(event, context):
    response = event['Records'][0]['cf']['response']
    request = event['Records'][0]['cf']['request']

    '''
    This function updates the HTTP status code in the response to 302, to redirect
    to another
    path (cache behavior) that has a different origin configured. Note the
    following:
    1. The function is triggered in an origin response
    2. The response status from the origin server is an error status code (4xx or
    5xx)
    '''

    if int(response['status']) >= 400 and int(response['status']) <= 599:
        redirect_path = '/plan-b/path?%s' % request['querystring']

        response['status'] = 302
        response['statusDescription'] = 'Found'

        # Drop the body as it is not required for redirects
        response['body'] = ''

```

```
response['headers']['location'] = [{'key': 'Location', 'value':  
redirect_path}]  
  
return response
```

存取請求本文：範例

以下範例說明如何使用 Lambda@Edge 來處理 POST 請求。

Note

若要使用這些範例，您必須在分佈的 Lambda 函數關聯中啟用包含內文選項。依預設不會啟用此功能。

- 若要在 CloudFront 主控台中啟用此設定，請選取 Lambda 函數關聯中包含內文的核取方塊。
- 若要在 CloudFront API 中或使用 啟用此設定 CloudFormation，請在 `true` 中將 `IncludeBody` 欄位設定為 `LambdaFunctionAssociation`。

主題

- [範例：使用請求觸發條件來讀取 HTML 表單](#)
- [範例：使用請求觸發條件來修改 HTML 表單](#)

範例：使用請求觸發條件來讀取 HTML 表單

此函數示範如何處理 HTML 表單 (Web 表單) 所產生 POST 請求的主體，例如「聯絡我們」表單。例如，您可能會有如下的 HTML 表單：

```
<html>  
  <form action="https://example.com" method="post">  
    Param 1: <input type="text" name="name1"><br>  
    Param 2: <input type="text" name="name2"><br>  
    input type="submit" value="Submit">  
  </form>  
</html>
```

至於後面的範例函數，該函數必須在 CloudFront 檢視器請求或原始伺服器請求中觸發。

Node.js

```
'use strict';

const querystring = require('querystring');

/**
 * This function demonstrates how you can read the body of a POST request
 * generated by an HTML form (web form). The function is triggered in a
 * CloudFront viewer request or origin request event type.
 */

exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;

  if (request.method === 'POST') {
    /* HTTP body is always passed as base64-encoded string. Decode it. */
    const body = Buffer.from(request.body.data, 'base64').toString();

    /* HTML forms send the data in query string format. Parse it. */
    const params = querystring.parse(body);

    /* For demonstration purposes, we only log the form fields here.
     * You can put your custom logic here. For example, you can store the
     * fields in a database, such as Amazon DynamoDB, and generate a response
     * right from your Lambda@Edge function.
     */
    for (let param in params) {
      console.log(`For "${param}" user submitted "${params[param]}".\n`);
    }
  }
  return callback(null, request);
};
```

Python

```
import base64
from urllib.parse import parse_qs

...
Say there is a POST request body generated by an HTML such as:

<html>
```

```

<form action="https://example.com" method="post">
  Param 1: <input type="text" name="name1"><br>
  Param 2: <input type="text" name="name2"><br>
  input type="submit" value="Submit">
</form>
</html>

'''

'''
This function demonstrates how you can read the body of a POST request
generated by an HTML form (web form). The function is triggered in a
CloudFront viewer request or origin request event type.
'''

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']

    if request['method'] == 'POST':
        # HTTP body is always passed as base64-encoded string. Decode it
        body = base64.b64decode(request['body']['data'])

        # HTML forms send the data in query string format. Parse it
        params = {k: v[0] for k, v in parse_qs(body).items()}

        '''
        For demonstration purposes, we only log the form fields here.
        You can put your custom logic here. For example, you can store the
        fields in a database, such as Amazon DynamoDB, and generate a response
        right from your Lambda@Edge function.
        '''
        for key, value in params.items():
            print("For %s use submitted %s" % (key, value))

    return request

```

範例：使用請求觸發條件來修改 HTML 表單

此函數示範如何修改 HTML 表單 (Web 表單) 所產生 POST 請求的主體。此函數會在 CloudFront 檢視器請求或原始伺服器請求中觸發。

Node.js

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
  var request = event.Records[0].cf.request;
  if (request.method === 'POST') {
    /* Request body is being replaced. To do this, update the following
    /* three fields:
    *   1) body.action to 'replace'
    *   2) body.encoding to the encoding of the new data.
    *
    *       Set to one of the following values:
    *
    *       text - denotes that the generated body is in text format.
    *             Lambda@Edge will propagate this as is.
    *       base64 - denotes that the generated body is base64 encoded.
    *             Lambda@Edge will base64 decode the data before sending
    *             it to the origin.
    *   3) body.data to the new body.
    */
    request.body.action = 'replace';
    request.body.encoding = 'text';
    request.body.data = getUpdatedBody(request);
  }
  callback(null, request);
};

function getUpdatedBody(request) {
  /* HTTP body is always passed as base64-encoded string. Decode it. */
  const body = Buffer.from(request.body.data, 'base64').toString();

  /* HTML forms send data in query string format. Parse it. */
  const params = querystring.parse(body);

  /* For demonstration purposes, we're adding one more param.
  *
  * You can put your custom logic here. For example, you can truncate long
  * bodies from malicious requests.
  */
  params['new-param-name'] = 'new-param-value';
  return querystring.stringify(params);
}
```

```
}
```

Python

```
import base64
from urllib.parse import parse_qs, urlencode

def lambda_handler(event, context):
    request = event['Records'][0]['cf']['request']
    if request['method'] == 'POST':
        '''
        Request body is being replaced. To do this, update the following
        three fields:
            1) body.action to 'replace'
            2) body.encoding to the encoding of the new data.

            Set to one of the following values:

                text - denotes that the generated body is in text format.
                    Lambda@Edge will propagate this as is.
                base64 - denotes that the generated body is base64 encoded.
                    Lambda@Edge will base64 decode the data before sending
                    it to the origin.
            3) body.data to the new body.
        '''
        request['body']['action'] = 'replace'
        request['body']['encoding'] = 'text'
        request['body']['data'] = getUpdatedBody(request)
    return request

def getUpdatedBody(request):
    # HTTP body is always passed as base64-encoded string. Decode it
    body = base64.b64decode(request['body']['data'])

    # HTML forms send data in query string format. Parse it
    params = {k: v[0] for k, v in parse_qs(body).items()}

    # For demonstration purposes, we're adding one more param

    # You can put your custom logic here. For example, you can truncate long
    # bodies from malicious requests
    params['new-param-name'] = 'new-param-value'
    return urlencode(params)
```

對邊緣函數的限制

下列主題說明適用於 CloudFront Functions 和 Lambda@Edge 的限制。某些限制適用於所有邊緣函數，而其他限制僅適用於 CloudFront Functions 或 Lambda@Edge。

每個主題都提供有關使用 CloudFront 開發和部署邊緣函數時應考慮的限制和約束詳細資訊。

了解這些限制可協助您確保邊緣函數如預期運作，並符合支援的功能。

主題

- [對所有邊緣函數的限制](#)
- [對 CloudFront Functions 的限制](#)
- [對 Lambda@Edge 的限制](#)

如需有關配額 (先前稱為限制) 的詳細資訊，請參閱 [CloudFront Functions 上的配額](#) 和 [Lambda@Edge 的配額](#)。

對所有邊緣函數的限制

下列限制適用於所有邊緣函數，包括 CloudFront Functions 和 Lambda@Edge。

主題

- [AWS 帳戶 擁有權](#)
- [結合 CloudFront Functions 與 Lambda@Edge](#)
- [HTTP 狀態碼](#)
- [HTTP 標頭](#)
- [查詢字串](#)
- [URI](#)
- [URI、查詢字串，以及標頭編碼](#)
- [Microsoft Smooth Streaming](#)
- [標記](#)

AWS 帳戶 擁有權

若要將邊緣函數與 CloudFront 分佈產生關聯，函數和分佈必須由相同的 AWS 帳戶 擁有。

結合 CloudFront Functions 與 Lambda@Edge

下列限制適用於指定的快取行為：

- 每個事件類型 (檢視器請求、原始伺服器請求、原始伺服器回應和檢視器回應) 只能有一個邊緣函數關聯。
- 您無法在檢視器事件 (檢視器請求和檢視器回應) 中結合 CloudFront Functions 和 Lambda@Edge。

允許邊緣函數的所有其他組合。下表說明了允許的組合。

		CloudFront Functions	
		檢視者請求	檢視者回應
Lambda@Edge	檢視者請求	不允許	不允許
	原始伺服器請求	已允許	已允許
	原始伺服器回應	已允許	已允許
	檢視者回應	不允許	不允許

HTTP 狀態碼

當原始伺服器傳回 HTTP 狀態碼 400 (或更高值) 時，CloudFront 不會針對檢視器回應事件調用邊緣函數。

CloudFront 會針對所有原始伺服器回應，呼叫原始伺服器回應事件的 Lambda@Edge 函數，包括在原始伺服器傳回 HTTP 狀態碼 400 (或更高值) 時。如需詳細資訊，請參閱 [在原始伺服器回應觸發條件中更新 HTTP 回應](#)。

HTTP 標頭

某些 HTTP 標頭不允許使用，這意味著這些標頭不會公開給邊緣函數，且函數無法新增這些標頭。其他標頭則設為唯讀，代表函數可以讀取這些標頭，但無法新增、修改或刪除。

主題

- [不允許的標頭](#)
- [唯讀標頭](#)

不允許的標頭

下列 HTTP 標頭不會公開給邊緣函數，且函數無法新增這些標頭。如果您的函數新增這些標頭之一，CloudFront 驗證請求會失敗，且 CloudFront 會傳回 HTTP 狀態碼 502 (無效的閘道) 給檢視器。

- Connection
- Expect
- Keep-Alive
- Proxy-Authenticate
- Proxy-Authorization
- Proxy-Connection
- Trailer
- Upgrade
- X-Accel-Buffering
- X-Accel-Charset
- X-Accel-Limit-Rate
- X-Accel-Redirect
- X-Amz-Cf-*
- X-Amzn-Auth
- X-Amzn-Cf-Billing
- X-Amzn-Cf-Id
- X-Amzn-Cf-Xff
- X-Amzn-Errortype
- X-Amzn-Fle-Profile
- X-Amzn-Header-Count
- X-Amzn-Header-Order
- X-Amzn-Lambda-Integration-Tag
- X-Amzn-RequestId
- X-Cache
- X-Edge-*
- X-Forwarded-Proto

- X-Real-IP

唯讀標頭

下列標頭為唯讀的狀態。您的函數可以讀取這些標頭並將其作為函數邏輯的輸入，但無法變更其值。如果您的函數新增或編輯了唯讀標頭，請求會無法通過 CloudFront 驗證，而且 CloudFront 會傳回 HTTP 狀態碼 502 (無效的閘道) 給檢視器。

檢視器請求事件中的唯讀標頭

下列標頭在檢視器請求事件中為唯讀的狀態。

- Content-Length
- Host
- Transfer-Encoding
- Via

原始伺服器請求事件中的唯讀標頭 (僅限 Lambda@Edge)

下列標頭在原始伺服器請求事件中為唯讀的狀態，僅存在於 Lambda@Edge 中。

- Accept-Encoding
- Content-Length
- If-Modified-Since
- If-None-Match
- If-Range
- If-Unmodified-Since
- Transfer-Encoding
- Via

原始伺服器回應事件中的唯讀標頭 (僅限 Lambda@Edge)

下列標頭在原始伺服器回應事件中為唯讀的狀態，僅存在於 Lambda@Edge 中。

- Transfer-Encoding
- Via

檢視器回應事件中的唯讀標頭

下列標頭在 CloudFront Functions 與 Lambda@Edge 的檢視器回應事件中為唯讀。

- Warning
- Via

下列標頭在 Lambda@Edge 檢視器回應事件中為唯讀狀態。

- Content-Length
- Content-Encoding
- Transfer-Encoding

查詢字串

下列限制適用於讀取、更新或在請求 URI 中建立查詢字串的函數。

- (僅限 Lambda@Edge) 若要存取原始伺服器請求或原始伺服器回應函數中的查詢字串，您的快取政策或原始伺服器請求政策必須針對查詢字串設定為 All (全部)。
- 函數可以為檢視器請求和原始伺服器請求事件建立或更新查詢字串 (原始伺服器請求事件僅存在於 Lambda@Edge 中)。
- 函數可以讀取查詢字串，但無法為原始伺服器回應和檢視器回應事件建立或更新查詢字串 (原始伺服器回應事件僅存在於 Lambda@Edge 中)。
- 如果函數建立或更新查詢字串，將適用下列限制：
 - 查詢字串不可包含空格、控制字元或片段識別碼 (#)。
 - URI 的總大小 (包含查詢字串) 必須小於 8,192 個字元。
 - 我們建議您於 URI 和查詢字串使用 % 編碼。如需更多詳細資訊，請參閱 [URI、查詢字串，以及標頭編碼](#)。

URI

如果函數為請求變更了 URI，這不會改變針對請求進行的快取動作，也不會改變請求轉傳目的地的原始伺服器。

URI 的總大小 (包含查詢字串) 必須小於 8,192 個字元。

URI、查詢字串，以及標頭編碼

傳遞給邊緣函數的 URI、查詢字串以及標頭值是使用 UTF-8 編碼。您的函數應針對其傳回的 URI、查詢字串和標頭值使用 UTF-8 編碼。百分比編碼與 UTF-8 編碼相容。

下列清單說明了 CloudFront 如何處理 URI、查詢字串值以及標頭的編碼：

- 如果請求中的值為 UTF-8 編碼，CloudFront 不需變更值，即可將其轉傳給函數。
- 當請求中的值採用 [ISO-8859-1 編碼](#) 時，CloudFront 會將這些值轉換為 UTF-8 編碼，然後再將其轉傳給函數。
- 當請求中的值採用其他字元編碼時，CloudFront 會假設其為 ISO-8859-1 編碼，並嘗試將 ISO-8859-1 編碼轉換為 UTF-8 編碼。

Important

轉換後的字元中的值可能是原始伺服器請求的不正確轉譯。這可能會導致函數或原始伺服器產生意外結果。

CloudFront 轉傳給您的原始伺服器的 URI、查詢字符串和標頭值，取決於函數是否變更這些值：

- 如果函數未變更 URI、查詢字串或標頭，CloudFront 會將請求所收到的值轉傳給您的原始伺服器。
- 如果函數變更了 URI 查詢字串或標頭，CloudFront 會轉傳以 UTF-8 編碼的值。

Microsoft Smooth Streaming

您無法將邊緣函數與已轉碼為 Microsoft Smooth Streaming 格式之串流媒體檔案的 CloudFront 分佈與搭配使用。

標記

您無法將標籤新增至邊緣函數。如需 CloudFront 標記功能的詳細資訊，請參閱 [標記分佈](#)。

對 CloudFront Functions 的限制

下列限制僅適用於 CloudFront Functions。

內容

- [日誌](#)

- [請求內文](#)
- [搭配 CloudFront KeyValueStore API 使用臨時憑證](#)
- [執行時期](#)
- [運算利用率](#)

如需有關配額 (先前稱為限制) 的詳細資訊，請參閱 [CloudFront Functions 上的配額](#)。

日誌

CloudFront Functions 中的函數日誌在 10 KB 處被截斷。

請求內文

CloudFront Functions 無法存取 HTTP 請求的本文。

搭配 CloudFront KeyValueStore API 使用臨時憑證

您可以使用 AWS Security Token Service (AWS STS) 來產生臨時安全憑證 (也稱為工作階段字符)。工作階段字符可讓您暫時擔任 AWS Identity and Access Management (IAM) 角色，以便存取 AWS 服務。

若要呼叫 [CloudFront KeyValueStore API](#)，請在 AWS STS 中使用區域端點來傳回第 2 版工作階段字符。如果您使用 AWS STS ([sts.amazonaws.com](#)) 的全域端點，AWS STS 會產生第 1 版工作階段字符，而簽署版本 4A (SigV4A) 不支援此字符。因此，您會收到身分驗證錯誤。

若要呼叫 CloudFront KeyValueStore API，您可以使用下列選項：

AWS CLI 和 AWS SDK

您可以設定 AWS CLI 或 AWS SDK 來使用區域 AWS STS 端點。如需詳細資訊，請參閱《AWS SDK 與工具參考指南》中的 [AWS STS 區域和端點](#)。

如需可用 AWS STS 端點的詳細資訊，請參閱《IAM 使用者指南》中的 [區域與端點](#)。

SAML

您可以設定 SAML 以使用區域 AWS STS 端點。如需詳細資訊，請參閱[如何使用區域 SAML 端點進行容錯移轉](#)部落格文章。

SetSecurityTokenServicePreferences API

您可以設定 AWS STS 的全域端點以傳回第 2 版工作階段字符，而不是使用區域 AWS STS 端點。若要執行此操作，請使用 [SetSecurityTokenServicePreferences](#) API 操作來設定您的 AWS 帳戶。

Example 範例：IAM CLI 命令

```
aws iam set-security-token-service-preferences --global-endpoint-token-version v2Token
```

Tip

我們建議您使用 AWS STS 區域端點，而非此選項。區域端點提供更高的可用性和容錯移轉案例。

自訂身分提供者

如果您使用的是自訂身分提供者，且由它負責進行聯合並代為承擔該角色，請使用前述選項之一，以作為負責產生工作階段字符的上層身分提供者。

執行時期

CloudFront Functions 執行時期環境不支援動態程式碼評估，而且會限制對網路、檔案系統、環境變數和計時器的存取。如需更多詳細資訊，請參閱 [限制功能](#)。

Note

若要使用 CloudFront KeyValueCollection，您的 CloudFront 函數必須使用 [JavaScript 執行時期 2.0](#)。

運算利用率

CloudFront Functions 對其可以執行所需的時間有限制，測量為運算利用率。運算利用率是介於 0 到 100 之間的數字，表示函數執行所花費的時間，以所允許時間上限的百分比表示。例如，35 的運算利用率表示函數以所允許時間上限的 35% 完成。

當您[測試函數](#)時，您可以在測試事件的輸出中看到運算利用率值。對於生產函數，您可以在 [CloudFront 主控台](#)中的 [Monitoring \(監控\) 頁面](#) 上，或在 CloudWatch 中檢視[運算利用率指標](#)。

對 Lambda@Edge 的限制

下列限制僅適用於 Lambda@Edge。

內容

- [DNS 解析](#)
- [HTTP 狀態碼](#)
- [Lambda 函數版本](#)
- [Lambda 區域](#)
- [Lambda 角色許可](#)
- [Lambda 功能](#)
- [支援的執行時期](#)
- [CloudFront 標頭](#)
- [使用包含內文選項時的要求內文限制](#)
- [回應逾時和保持連線逾時 \(僅限自訂原始伺服器\)](#)

如需 配額的詳細資訊，請參閱[Lambda@Edge 的配額](#)。

DNS 解析

CloudFront 在執行原始伺服器請求 Lambda@Edge 函數之前，會對原始伺服器網域名稱執行 DNS 解析。如果您網域的 DNS 服務遇到問題，且 CloudFront 無法解析網域名稱以取得 IP 位址，您的 Lambda@Edge 函數將不會調用。CloudFront 將傳回 [HTTP 502 狀態碼 \(無效的閘道\)](#) 給用戶端。如需更多詳細資訊，請參閱 [DNS 錯誤 \(NonS3OriginDnsError\)](#)。

如果您的函數邏輯修改原始網域名稱，CloudFront 將在函數完成執行後對更新的網域名稱執行另一個 DNS 解析。

如需有關管理 DNS 備援的詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的[設定 DNS 備援](#)。

HTTP 狀態碼

無論回應來自原始伺服器或 CloudFront 快取，檢視器回應事件的 Lambda@Edge 函數無法修改回應的 HTTP 狀態碼。

Lambda 函數版本

您必須使用 Lambda 函數的已編號版本，而不是 \$LATEST 或別名。

Lambda 區域

Lambda 函數必須位於美國東部 (維吉尼亞北部) 區域。

Lambda 角色許可

與 Lambda 函數關聯的 IAM 執行角色必須可由服務主體 `lambda.amazonaws.com` 和 `edgelambda.amazonaws.com` 擔任。如需更多詳細資訊，請參閱 [設定 Lambda@Edge 的 IAM 許可權限和角色](#)。

Lambda 功能

Lambda@Edge 不支援下列 Lambda 函數：

- 自動 (預設值) 以外的 [Lambda 執行時期管理組態](#)。
- 用於存取您 VPC 中的資源的 Lambda 函數組態
- [Lambda 函數無效字母佇列](#)。
- [Lambda 環境變數](#) (自動支援預留環境變數除外)
- [透過圖層管理 AWS Lambda 相依項](#) 的 Lambda 函數
- [使用 AWS X-Ray](#)
- Lambda 佈建並行

Note

Lambda@Edge 函數與所有 Lambda 函數共用相同的 [區域並行](#) 功能。如需更多詳細資訊，請參閱 [Lambda@Edge 的配額](#)。

- [使用容器映像建立 Lambda 函數](#)
- [使用 arm64 架構的 Lambda 函數](#)。
- 具有超過 512 MB 暫時性儲存的 Lambda 函數
- 使用 [客戶自管金鑰以加密您的 .zip 部署套件](#)

支援的執行時期

Lambda@Edge 支援最新版本的 Node.js 和 Python 執行時期。如需支援版本及其未來棄用日期的清單，請參閱《AWS Lambda 開發人員指南》中的 [支援的執行時期](#)。

i Tip

- 依照最佳實務，請使用所提供的最新版本執行時期，以獲得效能提升與新功能。
- 您無法使用棄用的 Node.js 版本來建立或更新函數。您只能將現有函數與這些版本與 CloudFront 分佈建立關聯。使用此版本且與分佈相關聯的函數會繼續執行。不過，建議您將函數移至更新版本的 Node.js。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的[執行時期支援政策](#)和 GitHub 上的[Node.js 發行排程](#)。

CloudFront 標頭

Lambda@Edge 函數可以讀取、編輯、移除或新增任何列於 [新增 CloudFront 請求標頭](#) 的 CloudFront 標頭。

i 備註

- 如果您希望 CloudFront 新增這些標頭，則必須將 CloudFront 設定為使用[快取政策](#)或[原始伺服器請求政策](#)來新增標頭。
- CloudFront 會在檢視器請求事件之後新增標頭，這表示檢視器請求中的 Lambda@Edge 函數無法使用這些標頭。標頭僅適用於原始伺服器請求和原始伺服器回應中的 Lambda@Edge 函數。
- 如果檢視器請求包含具有這些名稱的標頭，且您將 CloudFront 設定為使用[快取政策](#)或[伺服器請求政策](#)來新增標頭，則 CloudFront 會覆寫檢視器請求中的標頭值。檢視器對應的函數會從檢視器請求查看標頭值，而原始伺服器對應的函數則會查看 CloudFront 新增的標頭值。
- 若檢視器請求函數新增 CloudFront-Viewer-Country 標頭，它將無法通過驗證，且 CloudFront 會向檢視器傳回 HTTP 狀態碼 502 (無效的閘道)。

使用包含內文選項時的要求內文限制

如果您選擇包含本文選項以向您的 Lambda@Edge 函數公開請求本文，則請您注意，公開或替換的本文部分須遵循下列的資訊和大小配額。

- CloudFront 一律會在將要求內文公開給 Lambda@Edge 之前對其進行 base64 編碼。
- 如果要求內文過大，CloudFront 會在公開給 Lambda@Edge 之前將內文截斷，如下所示：
 - 針對檢視器請求事件，會在 40 KB 處截斷內文。

- 針對原始伺服器請求事件，會在 1 MB 處截斷內文。
- 如果以唯讀方式存取請求內文，CloudFront 會將完整的原始伺服器請求內文傳送給原始伺服器。
- 如果 Lambda@Edge 函數替換了請求本文，則下列大小配額適用於函數所傳回的本文：
 - 如果 Lambda@Edge 函數以純文字格式傳回內文：
 - 針對檢視器請求事件，本文限制為 40 KB。
 - 針對原始伺服器請求事件，本文限制為 1 MB。
 - 如果 Lambda@Edge 函數以 base64 編碼的文字傳回內文：
 - 針對檢視器請求事件，本文限制為 53.2 KB。
 - 針對原始伺服器請求事件，本文限制為 1.33 MB。

Note

如果您的 Lambda@Edge 函數傳回超過這些限制的本文，您的請求將使用 HTTP 502 狀態碼 ([Lambda 驗證錯誤](#)) 失敗。我們建議您更新 Lambda@Edge 函數，讓本文不超過這些限制。

回應逾時和保持連線逾時 (僅限自訂原始伺服器)

如果您使用 Lambda@Edge 函數來設定分佈原始伺服器的回應逾時或持續作用逾時，請確認您正在指定原始伺服器可支援的值。如需更多詳細資訊，請參閱 [回應和保持連線逾時配額](#)。

報告、指標和日誌

CloudFront 提供數種選項來報告、監控和記錄您的 CloudFront 資源。您可完成下列任務：

- 檢視和下載報告，查看 CloudFront 分佈的用量與活動，包括帳單報告、快取統計資料、熱門內容和最佳推薦網站。
- 直接在 CloudFront 主控台或使用 Amazon CloudWatch 監控和追蹤 CloudFront (包括[邊緣運算函數](#))。CloudFront 傳送指標到 CloudWatch 供分佈和邊緣函數 (Lambda@Edge 和 CloudFront Functions) 使用。
- 檢視 CloudFront 分佈透過標準日誌或即時存取日誌收到的檢視器請求的日誌。除了檢視器請求日誌外，您可以使用 CloudWatch Logs 取得 Lambda @Edge 和 CloudFront Functions 兩種邊緣函數的日誌。您也可以使用 AWS CloudTrail 來取得中 CloudFront API 活動的日誌 AWS 帳戶。
- 使用 追蹤 CloudFront 資源的組態變更 AWS Config。

如需上述選項的詳細資訊，請參閱以下主題。

主題

- [AWS CloudFront 的帳單和用量報告](#)
- [檢視主控台中的 CloudFront 報告](#)
- [使用 Amazon CloudWatch 監控 CloudFront 指標](#)
- [CloudFront 和邊緣函數記錄](#)
- [使用 追蹤組態變更 AWS Config](#)

AWS CloudFront 的帳單和用量報告

AWS 為 CloudFront 提供兩個用量報告：

- AWS 帳單報告是您正在使用的所有活動的 AWS 服務 高階檢視，包括 CloudFront。
- AWS 用量報告是特定服務的活動摘要，依小時、天或月彙總。它還包含用量圖表，提供 CloudFront 使用量的圖形表示。

 Note

如同其他 AWS 服務，CloudFront 只會針對您使用的項目向您收費。如需詳細資訊，請參閱 [CloudFront 定價](#)。

主題

- [檢視 CloudFront 的 AWS 帳單報告](#)
- [檢視 CloudFront 的 AWS 用量報告](#)
- [解譯 CloudFront 的 AWS 帳單和用量報告](#)

檢視 CloudFront 的 AWS 帳單報告

您可以在 AWS 帳單與成本管理 主控台的帳單頁面上檢視依服務列出的 AWS 用量和費用摘要。

檢視 AWS 帳單報告

1. 登入 AWS 管理主控台 並在 <https://console.aws.amazon.com/costmanagement/> 開啟 AWS 帳單與成本管理 主控台。
2. 在導覽窗格中，選擇 Bills (帳單)。
3. 選擇一個帳單期間 (例如 2023 年 8 月)。
4. 在依服務收費索引標籤上，選擇 CloudFront，然後展開全域或 AWS 區域 名稱。
5. 若要下載 CSV 格式的詳細帳單報告，請選擇全部下載為 CSV。

如需 AWS 帳單的詳細資訊，請參閱 AWS Billing 《使用者指南》中的 [檢視帳單](#)。

帳單報告包含以下適用於 CloudFront 的值：

- ProductCode : AmazonCloudFront
- UsageType – 下列其中一個值：
 - 識別資料傳輸類型的代碼
 - Invalidations
 - Executions-CloudFrontFunctions
 - KeyValueStore-APIOperations
 - KeyValueStore-EdgeReads

- RealTimeLog-KinesisDataStream
- SSL-Cert-Custom
- ItemDescription：UsageType 的計費費率說明。
- UsageStart Date 及 UsageEndDate：套用用量的日期，採用國際標準時間 (UTC)。
- UsageQuantity：下列其中一個值：
 - 指定時段期間的請求數目
 - 資料傳輸量 (以 GB 為單位)
 - 失效的物件的數量
 - 與已啟用 CloudFront 分佈關聯的 SSL 憑證，按月份依比例分配計算的總和。例如，如果您有一個整個月期已啟用分佈的相關聯憑證和另一個半月期已啟用分佈相關聯憑證，則此值將為 1.5。

檢視 CloudFront 的 AWS 用量報告

AWS 提供比帳單報告更詳細的 CloudFront 用量報告，但比 CloudFront 存取日誌更不詳細。用量報告按小時、日或月提供彙總用量資料；並按地區和使用類型列出操作，例如從澳洲區域轉出的資料。

檢視 AWS 用量報告

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/costmanagement/> 開啟 AWS 帳單與成本管理 主控台。
2. 在導覽窗格中選擇 Cost Explorer。
3. 在新成本與用量報告頁面上的報告參數面板中，選擇報告的日期範圍和精細程度。
4. 在篩選條件、服務下選取 CloudFront。
5. 選取用量類型。
6. 在成本和用量明細下選擇以 CSV 格式下載。

如需 AWS 用量報告的詳細資訊，請參閱AWS 資料匯出 《使用者指南》中的[AWS 用量報告](#)。

CloudFront 用量報告包含以下值：

- 服務 – AmazonCloudFront
- 操作：HTTP 方法。數值包含 DELETE、GET、HEAD、OPTIONS、PATCH、POST 和 PUT。
- UsageType – 下列其中一個值：
 - 識別資料傳輸類型的代碼

- Invalidations
- Executions-CloudFrontFunctions
- KeyValueStore-APIOperations
- KeyValueStore-EdgeReads
- RealTimeLog-KinesisDataStream
- SSL-Cert-Custom
- 資源：與用量相關聯的 CloudFront 分佈之 ID，或是您已經與 CloudFront 分佈建立關聯的 SSL 憑證之憑證 ID。
- StartTime/EndTime：套用用量的日期，採用國際標準時間 (UTC)。
- UsageValue：(1) 在指定期間內的請求數量或 (2) 以位元組為單位的資料傳輸量。

如果您使用 Amazon S3 做為 CloudFront 的原始伺服器，請考慮執行 Amazon S3 的用量報告。但是，如果您將 Amazon S3 用於 CloudFront 分佈原始伺服器以外的用途，可能無法清楚哪些部分可套用於您的 CloudFront 使用量。

Tip

如需 CloudFront 針對物件所接收之每個請求的詳細資訊，請開啟您的分佈的 CloudFront 存取記錄。如需詳細資訊，請參閱[the section called “存取日誌（標準日誌）”](#)。

如需瞭解報告中 CloudFront 費用和用量類型的詳細資訊，請參閱 [the section called “解譯 CloudFront 的 AWS 帳單和用量報告”](#)。

解譯 CloudFront 的 AWS 帳單和用量報告

您取得[帳單報告](#)和[用量報告](#)後，可以使用此主題來瞭解如何解讀帳單上顯示的每項 CloudFront 費用，以及每項費用對應的用量類型。本主題包含可在兩個報告上顯示的代碼和 AWS 區域縮寫。

兩欄中的大多數代碼都包含兩個字母的縮寫，指出活動的位置。在下表中，程式碼中的##會在您的 AWS 帳單和用量報告中替換為下列其中一個兩個字母縮寫：

- AP：香港特別行政區、菲律賓、南韓、台灣和新加坡 (亞太區域)
- AU：澳洲
- CA：加拿大
- EU：歐洲和以色列

- IN：印度
- JP：日本
- ME：中東
- SA：南美洲
- US：美國
- ZA：南非

如需定價方式的詳細資訊 AWS 區域，請參閱 [Amazon CloudFront 定價](#)。

備註

- 此表格不包括將您的物件從 Amazon S3 儲存貯體轉移到 CloudFront 節點的費用。這些費用 (如果有的話) 會列在 AWS 帳單的 AWS 資料傳輸區段。
- 第一欄列出 AWS 帳單報告中顯示的費用，並說明每個費用的含義。
- 第二欄列出出現在 AWS 用量報告中的項目，並顯示帳單費用與用量報告項目之間的關聯性。

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType欄中的值
<i>region</i> -DataTransfer-Out-Bytes	<i>region</i> -Out-Bytes-HTTP-Static:
回應使用者 GET 和 HEAD 請求，從##中的 CloudFront 邊緣站點提供的總位元組數。	透過 HTTP 為 TTL ≥ 3,600 秒的物件提供的位元組數。
	<i>region</i> -Out-Bytes-HTTPS-Static:
	透過 HTTPS 為 TTL ≥ 3,600 秒的物件提供的位元組數。
	<i>region</i> -Out-Bytes-HTTP-Dynamic:
	透過 HTTP 為 TTL < 3,600 秒的物件提供的位元組數。
	<i>region</i> -Out-Bytes-HTTPS-Dynamic:

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType 欄中的值
	透過 HTTPS 為 TTL < 3,600 秒的物件提供的位元組數。 <i>region</i>-Out-Bytes-HTTP-Proxy: 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTP 從 CloudFront 傳回到檢視器的位元組數。 <i>region</i>-Out-Bytes-HTTPS-Proxy: 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTPS 從 CloudFront 傳回到檢視器的位元組數。 這包括透過 gRPC 從 CloudFront 傳回給檢視器的位元組。
<i>region</i>-DataTransfer-Out-OBytes 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求時，從 CloudFront 邊緣節點傳輸到原始伺服器或邊緣函數的總位元組數。這些費用包括將 WebSocket 資料從用戶端傳輸到伺服器的費用。	<i>region</i>-Out-OBytes-HTTP-Proxy 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求時，從 CloudFront 邊緣節點透過 HTTP 傳輸到原始伺服器或邊緣函數的總位元組數。 <i>region</i>-Out-OBytes-HTTPS-Proxy 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求時，從 CloudFront 邊緣節點透過 HTTPS 傳輸到原始伺服器或邊緣函數的總位元組數。 這包括透過 gRPC 從 CloudFront 邊緣節點傳輸到您的原始伺服器或 CloudFront Functions 的位元組數。

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType 欄中的值
<i>region</i>-Requests-Tier1 HTTP GET 和 HEAD 請求數。	<i>region</i>-Requests-HTTP-Static 為物件提供的 HTTP GET 和 HEAD 請求數 (TTL $\geq$ 3,600 秒)。 <i>region</i>-Requests-HTTP-Dynamic 為物件提供的 HTTP GET 與 HEAD 請求數 (TTL $<$ 3,600 秒)。
<i>region</i>-Requests-Tier2-HTTPS HTTPS GET 和 HEAD 請求數。	<i>region</i>-Requests-HTTPS-Static 為物件提供的 HTTPS GET 和 HEAD 請求數 (TTL $\geq$ 3,600 秒)。 <i>region</i>-Requests-HTTPS-Dynamic 為物件提供的 HTTPS GET 與 HEAD 請求數 (TTL $<$ 3,600 秒)。
<i>region</i>-Requests-HTTP-Proxy CloudFront 轉送到您原始伺服器或邊緣函數的 HTTP DELETE、OPTIONS、PATCH、POST 和 PUT 請求數。 還包括 CloudFront 轉送到您的原始伺服器或邊緣函數之 HTTP WebSocket 請求 (具有 Upgrade: websocket 標頭的 GET 請求) 的數量。	<i>region</i>-Requests-HTTP-Proxy 與 CloudFront 帳單中的對應項目相同。

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType 欄中的值
<i>region</i>-Requests-HTTPS-Proxy CloudFront 轉送到您原始伺服器或邊緣函數的 HTTPS DELETE、OPTIONS、PATCH、POST 和 PUT 請求數。 也包含下列請求類型： - CloudFront 轉送到您的原始伺服器或邊緣函數之 HTTPS WebSocket 請求 (具有 Upgrade: websocket 標頭的 GET 請求) 的數量。 - HTTPS gRPC 請求數。	<i>region</i>-Requests-HTTPS-Proxy 與 CloudFront 帳單中的對應項目相同。
<i>region</i>-Requests-HTTPS-Proxy-FLE CloudFront 轉送到您原始伺服器或邊緣函數，並以欄位層級加密的 HTTPS DELETE、OPTIONS、PATCH 和 POST 請求數。	<i>region</i>-Requests-HTTPS-Proxy-FLE 與 CloudFront 帳單中的對應項目相同。
<i>region</i>-Bytes-OriginShield 從原始伺服器傳輸至任何區域邊緣快取的總位元組數，包括做為 Origin Shield 啟用的區域邊緣快取。	<i>region</i>-Bytes-OriginShield 與 CloudFront 帳單中的對應項目相同。
<i>region</i>-OBytes-OriginShield 從任何區域邊緣快取傳輸至原始伺服器的總位元組數，包括做為 Origin Shield 啟用的區域邊緣快取。	<i>region</i>-OBytes-OriginShield 與 CloudFront 帳單中的對應項目相同。

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType欄中的值
<i>region</i>-Requests-OriginShield 轉到 Origin Shield 的請求數做為增量改進層。對於代理至原始伺服器的動態 (非可快取) 請求，Origin Shield 一律為增量改進層。對於可快取的請求，Origin Shield 有時是增量改進層。 如需詳細資訊，請參閱 the section called “預估 Origin Shield 成本”。	<i>region</i>-Requests-OriginShield 與 CloudFront 帳單中的對應項目相同。
無效化 失效物件費用 (從 CloudFront 邊緣節點移除物件)。如需詳細資訊，請參閱支付檔案失效的費用。	無效化 與 CloudFront 帳單中的對應項目相同。
SSL-Cert-Custom 付費使用具有 CloudFront 備用網域名稱 (如 example.com) 的 SSL 憑證的費用，而不使用預設的 CloudFront SSL 憑證和 CloudFront 指派給您的分佈的網域名稱。	SSL-Cert-Custom 與 CloudFront 帳單中的對應項目相同。
RealTimeLog-KinesisDataStream 即時存取日誌產生的行數費用。	RealTimeLog-KinesisDataStream 與 CloudFront 帳單中的對應項目相同。
Executions-CloudFrontFunctions CloudFront Functions 調用次數的費用。	Executions-CloudFrontFunctions 與 CloudFront 帳單中的對應項目相同。
<i>region</i>-Lambda-Edge-Request Lambda@Edge 函數調用次數的費用。	<i>region</i>-Lambda-Edge-Request 與 CloudFront 帳單中的對應項目相同。
<i>region</i>-Lambda-Edge-GB-Second 從調用 Lambda@Edge 函數到傳回或終止時的持續時間費用。	<i>region</i>-Lambda-Edge-GB-Second 與 CloudFront 帳單中的對應項目相同。

AWS 帳單中的 CloudFront 費用	AWS 用量報告中 UsageType 欄中的值
KeyValueStore-EdgeReads 對 CloudFront KeyValueStore 方法、get()、exists() 和 meta() 的讀取呼叫次數費用。如需詳細資訊，請參閱 鍵值存放區的協助程式方法 。	KeyValueStore-EdgeReads 與 CloudFront 帳單中的對應項目相同。
KeyValueStore-APIOperations 呼叫 CloudFront KeyValueStore API 的次數費用。	KeyValueStore-APIOperations 與 CloudFront 帳單中的對應項目相同。

檢視主控台中的 CloudFront 報告

每項報告都提供詳細資訊和視覺化內容，讓您可以最佳化內容交付、識別效能瓶頸，以及做出資料導向決策。無論您要監控快取效率、分析流量模式，還是想更充分瞭解檢視器，都可以使用這些報告來有效地監控和分析 CloudFront 分佈。

您可以在主控台中檢視 CloudFront 活動的下列報告：

主題

- [檢視 CloudFront 快取統計資料報告](#)
- [檢視 CloudFront 熱門物件報告](#)
- [檢視 CloudFront 最佳推薦網站報告](#)
- [檢視 CloudFront 用量報告](#)
- [檢視 CloudFront 檢視器報告](#)

大多數的報告都是根據 CloudFront 存取日誌中的資料，其中包含 CloudFront 收到的每個使用者請求的詳細資訊。您不需要啟用存取日誌來查看報告。如需更多詳細資訊，請參閱 [存取日誌（標準日誌）](#)。

檢視 CloudFront 快取統計資料報告

Amazon CloudFront 快取統計資料報告包含以下資訊：

- 請求總數：所有 HTTP 狀態碼 (例如 200 或 404) 和所有方法 (例如 GET、HEAD 或 POST) 的請求總數
- 依結果類型區分的檢視器請求百分比：針對所選取的 CloudFront 分佈，提供命中數、未命中數和錯誤佔檢視器請求總數的百分比
- 傳輸到檢視器的位元組：總位元組數和未命中的位元組數
- HTTP 狀態碼：根據 HTTP 狀態碼區分的檢視器請求。
- 未完成下載的 GET 請求百分比：未完成下載所請求物件的檢視器 GET 請求數佔總請求數的百分比。

這些統計資訊的資料取自與 CloudFront 存取日誌相同的來源。不過您不需要啟用[存取日誌](#)來查看快取統計資料。

您可以使用每小時或每天的資料點來顯示過去 60 天內指定日期範圍的圖表。您通常可以檢視最近一小時前收到的 CloudFront 請求的資料，但資料偶爾會延遲高達 24 小時。

主題

- [在主控台中檢視 CloudFront 快取統計資料報告](#)
- [以 CSV 格式下載資料](#)
- [快取統計資料圖表與 CloudFront 標準日誌 \(存取日誌\) 中資料的關係](#)

在主控台中檢視 CloudFront 快取統計資料報告

您可以在主控台檢視 CloudFront 快取統計資料報告。

檢視 CloudFront 快取統計資料報告

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽面板中選擇快取統計資料。
3. 在 CloudFront 快取統計資料報告窗格中，針對開始日期和結束日期，選擇您要顯示快取統計資料圖表的日期範圍。可用的範圍取決於您針對 Granularity (精細度) 選擇的值：
 - 每日 – 若要顯示每天一個資料點的圖表，請選擇前 60 天內的任何日期範圍。
 - 每小時 – 若要顯示每小時一個資料點的圖表，請選擇前 60 天內的任意日期範圍 (最多 14 天)。

日期和時間都使用國際標準時間 (UTC)。

4. 對於 Granularity (精細度)，指定要在圖表中顯示每天一個資料點，或每小時一個資料點。如果您指定的日期範圍大於 14 天，則每小時指定一個資料點的選項不可用。
5. 針對 Viewer Location (檢視器位置)，選擇發出檢視器請求的洲別，或是選擇 All Locations (所有位置)。快取統計資料圖表包含 CloudFront 從指定位置收到的請求的資料。
6. 在 Distribution (分佈) 清單中，選擇要在用量圖表中顯示其資料的分佈：
 - 個別分佈 - 此圖表會顯示所選取 CloudFront 分佈的資料。Distribution (分佈) 清單會顯示分佈的分佈 ID 和替代網域名稱 (CNAME) (如果有的話)。如果分佈沒有備用網域名稱，則該清單包括分佈的來源網域名稱。
 - 所有分佈：圖表會針對與目前 AWS 帳戶 帳戶具有關聯的所有分佈 (不包括已刪除的分佈) 顯示摘要資料。
7. 選擇更新。

Tip

- 若要查看圖表中每日或每小時資料點的資料，請將滑鼠指標停留在資料點上。
- 對於顯示傳輸資料的圖表，您可以將垂直擴展變更為 GB、MB 或 KB。

以 CSV 格式下載資料

您可使用 CSV 格式下載快取統計資料報告。本節說明如何下載報告和描述報告中的值。

以 CSV 格式下載快取統計資料報告。

1. 請在檢視快取統計資料報告時選擇 CSV。
2. 在開啟檔案名稱對話方塊中，選擇要開啟或儲存檔案。

有關報告的資訊

報告的前幾行包含以下資訊：

版本

此 CSV 檔案的格式版本。

報告

報告名稱。

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

精細程度

報告中的每一行表示一小時還是一天。

ViewerLocation

該檢視器所請求來源的洲別，或是如果您選擇用於所有位置而下載報告的 ALL。

快取統計資料報告中的資料

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

ViewerLocation

該檢視器所請求來源的洲別，或是如果您選擇用於所有位置而下載報告的 ALL。

TimeBucket

資料所應用的小時或天，以國際標準時間 (UTC) 為準。

RequestCount

所有 HTTP 狀態碼 (例如, 200 或 404) 和所有方法 (例如, GET、HEAD 或 POST) 的請求總數

HitCount

用於從 CloudFront 邊緣快取為提供服務的物件, 其檢視器請求的數量。

MissCount

用於該物件的檢視器請求數量目前不在邊緣快取中, 因此 CloudFront 必須取得從您的原始伺服器取得物件。

ErrorCount

產生錯誤的檢視器請求的數量, 因此 CloudFront 無法為物件提供服務。

IncompleteDownloadCount

檢視器啟動下載物件但未完成的檢視器請求數量。

HTTP2xx

用於 HTTP 狀態碼為 2xx 的值 (成功) 的檢視器請求數量。

HTTP3xx

用於 HTTP 狀態碼 3xx 的值 (需要額外動作) 的檢視器請求數量。

HTTP4xx

用於 HTTP 狀態碼 4xx 的值 (用戶端錯誤) 的檢視器請求數量。

HTTP5xx

用於 HTTP 狀態碼 5xx 的值 (伺服器錯誤) 的檢視器請求數量。

TotalBytes

CloudFront 提供給檢視器的總位元組數, 以回應所有 HTTP 方法的所有請求。

BytesFromMisses

用於在請求時不在節點快取中的物件, 所提供給檢視器的位元組數。該值是從原始伺服器傳輸到 CloudFront 邊緣快取的良好位元組近似值。但是, 它會排除已經在節點快取但已過期的物件的請求。

快取統計資料圖表與 CloudFront 標準日誌 (存取日誌) 中資料的關係

下表顯示了 CloudFront 主控台中的快取統計資料圖表與 CloudFront 存取日誌中的值的對應方式。如需有關 CloudFront 存取日誌的詳細資訊，請參閱 [存取日誌 \(標準日誌\)](#)。

請求總數

此圖表顯示所有 HTTP 狀態碼 (例如，200 或 404) 和所有方法 (例如，GET、HEAD 或 POST) 的請求總數。此圖表中顯示的請求總數等於相同時間段中存取日誌檔中的請求總數。

依結果類型的檢視器請求百分比

此圖表將命中數、未命中數和錯誤顯示為所選 CloudFront 分佈的檢視器請求總數的百分比：

- 命中 – 從 CloudFront 邊緣快取為其提供物件的檢視器請求數量。在存取日誌中，這些都是 `x-edge-response-result-type` 值為 `Hit` 的請求。
- 未命中 – 檢視器請求的物件目前不在邊緣快取中，因此 CloudFront 必須從您的原始伺服器取得物件。在存取日誌中，這些都是 `x-edge-response-result-type` 值為 `Miss` 的請求。
- 錯誤 – 造成錯誤的檢視器請求，因此 CloudFront 並未提供物件。在存取日誌中，這些都是 `x-edge-response-result-type` 值為 `Error`、`LimitExceeded` 或 `CapacityExceeded` 的請求。

圖表並不包括用於在邊緣快取中但已過期的物件所做的重新整理命中請求。在存取日誌中，重新整理命中是 `x-edge-response-result-type` 值為 `RefreshHit` 的請求。

傳輸到檢視器的位元組數

此圖表顯示兩個值：

- 總位元組數 – 為了回應所有 HTTP 方法的所有請求，CloudFront 提供給檢視器的總位元組數。在 CloudFront 存取日誌中，總位元組數是在同一個期間內所有請求的 `sc-bytes` 欄位值總和。
- 未命中的位元組數 – 在請求發出時，針對不在邊緣快取中的物件，提供給檢視器的位元組數。在 CloudFront 存取日誌中，`bytes from misses` (未命中的位元組數) 是 `sc-bytes` 欄位值的總和，這些位元組是用於其 `x-edge-result-type` 值為 `Miss` 的請求。該值是從原始伺服器傳輸到 CloudFront 邊緣快取的良好位元組近似值。但是，它會排除已經在節點快取但已過期的物件的請求。

HTTP 狀態碼

此圖表透過 HTTP 狀態碼顯示檢視器請求。在 CloudFront 存取日誌中，狀態碼會顯示於 `sc-status` 欄位：

- 2xx – 請求已成功。
- 3xx – 需要執行其他動作。例如，301 (永久移除) 表示請求的物件已經移到不同的位置。
- 4xx – 用戶端明顯出錯。例如，404 (未找到) 表示未找到用戶端請求的物件。
- 5xx – 原始伺服器並未完成請求的請求。例如，503 (服務無法使用) 表示原始伺服器目前無法使用。

未完成下載的 GET 請求的百分比

此圖表顯示未完成下載請求物件的檢視器 GET 請求，其表示為請求總數的百分比。一般而言，因為檢視器取消下載，致使下載物件不完全；例如，按下了不同的連結或關閉瀏覽器。在 CloudFront 存取日誌中，這些請求在 200 欄位中的值為 `sc-status`，在 Error 欄位中的值為 `x-edge-result-type`。

檢視 CloudFront 熱門物件報告

檢視 Amazon CloudFront 熱門物件報告，以查看過去 60 天內指定日期範圍內分佈的 50 個最熱門物件。您也可以檢視這些物件的統計資料，包括下列項目：

- 物件的請求總數。
- 命中次數和未命中次數
- 命中率
- 為遺漏提供的位元組數
- 提供的總位元組數
- 未完成下載的數量
- 依 HTTP 狀態碼 (2xx、3xx、4xx 和 5xx) 區分的請求數量

這些統計資訊的資料取自與 CloudFront 存取日誌相同的來源。不過您不需要啟用[存取記錄](#)功能以檢視熱門物件。

主題

- [在主控台中檢視 CloudFront 熱門物件報告](#)
- [CloudFront 如何計算熱門物件統計資料](#)
- [以 CSV 格式下載資料](#)
- [熱門物件報告中的資料如何與 CloudFront 標準日誌 \(存取日誌\) 中的資料相關](#)

在主控台中檢視 CloudFront 熱門物件報告

您可以在主控台中檢視 CloudFront 熱門物件報告。

檢視 CloudFront 分佈的熱門物件

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽面板中按一下熱門物件。
3. 在 CloudFront 熱門物件報告窗格中，針對 開始日期和結束日期，選取您要顯示熱門物件清單的日期範圍。您可以選擇過去 60 天內的任何日期範圍。

日期和時間都使用國際標準時間 (UTC)。

4. 在 Distribution (分佈) 清單中，選擇要顯示其熱門物件清單的分佈。
5. 選擇更新。

CloudFront 如何計算熱門物件統計資料

為了得到前 50 個物件的精確計數，CloudFront 會從午夜開始以 10 分鐘的時間間隔對所有物件請求進行計數，並在後續 24 小時內計算前 150 個物件的累計加總。(CloudFront 還保留 60 天內前 150 個物件的每日總計。)

清單底部附近的物件會不斷上榜或落榜，所以這些物件的總計是近似值。150 個物件清單中位居前列的 50 個物件可能會在清單中上升和下降，但幾乎不會完全掉出清單之外，因此這些物件的總計通常更可靠。

如果有一個物件掉出前 150 個物件的清單，然後在一天後再次升入清單中，則 CloudFront 會針對物件不在清單之中的這段期間，加入預估的請求數量。此預估是根據在該時段內，由任一物件在清單底部所接收到的請求數量。

如果物件在當天較晚時間上升到前 50 個物件，那麼當物件超出前 150 個物件時，CloudFront 所接收到的請求數估計值通常會導致熱門物件報告中的請求數量超過顯示在該物件存取日誌中的請求數量。

以 CSV 格式下載資料

您可使用 CSV 格式下載熱門物件報告。本節說明如何下載報告和描述報告中的值。

以 CSV 格式下載熱門物件報告。

1. 檢視熱門物件報告時，請選擇 CSV。

2. 在開啟檔案名稱對話方塊中，選擇要開啟或儲存檔案。

有關報告的資訊

報告的前幾行包含以下資訊：

版本

此 CSV 檔案的格式版本。

報告

報告名稱。

DistributionID

您為執行報告所分佈的 ID。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

熱門物件報告中的資料

該報告包含以下值：

DistributionID

您為執行報告所分佈的 ID。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

物件

該物件 URL 的最後 500 個字元。

RequestCount

此物件的請求總數。

HitCount

用於從 CloudFront 邊緣快取為提供服務的物件，其檢視器請求的數量。

MissCount

用於該物件的檢視器請求數量目前不在邊緣快取中，因此 CloudFront 必須取得從您的原始伺服器取得物件。

HitCountPct

HitCount 值做為 RequestCount 值的百分比。

BytesFromMisses

當請求時，物件不在節點快取中，為了該物件所提供的檢視器位元組數。

TotalBytes

用於回應所有 HTTP 方法請求的物件，由 CloudFront 所提供給檢視器的總位元組數

IncompleteDownloadCount

檢視器啟動下載但未完成的檢視器請求數量。

HTTP2xx

用於 HTTP 狀態碼為 2xx 的值 (成功) 的檢視器請求數量。

HTTP3xx

用於 HTTP 狀態碼 3xx 的值 (需要額外動作) 的檢視器請求數量。

HTTP4xx

用於 HTTP 狀態碼 4xx 的值 (用戶端錯誤) 的檢視器請求數量。

HTTP5xx

用於 HTTP 狀態碼 5xx 的值 (伺服器錯誤) 的檢視器請求數量。

熱門物件報告中的資料如何與 CloudFront 標準日誌 (存取日誌) 中的資料相關

下表的清單顯示了 CloudFront 主控台中的熱門物件報告中的值與 CloudFront 存取日誌中的值的對應方式。如需有關 CloudFront 存取日誌的詳細資訊，請參閱 [存取日誌 \(標準日誌\)](#)。

URL

檢視器用於存取該物件的 URL 的最後 500 個字元。

請求

物件的請求總數。這個值通常與 CloudFront 存取日誌中的物件的 GET 請求數量密切對應。

命中

用於從 CloudFront 邊緣快取為提供服務的物件，其檢視器請求的數量。在存取日誌中，這些都是 `x-edge-response-result-type` 值為 `Hit` 的請求。

未命中數

該檢視器請求數量用於不在邊緣快取中的物件，因此 CloudFront 必須取得從您的原始伺服器擷取物件。在存取日誌中，這些都是 `x-edge-response-result-type` 值為 `Miss` 的請求。

命中率

Hits (命中) 欄位的值，佔 Requests (請求) 欄位值的百分比。

未命中的位元組

用於在請求時不在節點快取中的物件，所提供給檢視器的位元組數。在 CloudFront 存取日誌中，`bytes from misses` (未命中的位元組數) 是 `sc-bytes` 欄位值的總和，這些位元組是用於其 `x-edge-result-type` 值為 `Miss` 的請求。

總位元組數

在回應所有 HTTP 方法的物件的所有請求時，CloudFront 為檢視器提供的總位元組數。在 CloudFront 存取日誌中，`total bytes` (總位元組數) 是在同一個期間內所有請求的 `sc-bytes` 欄位值總和。

不完整的下載

未完成下載請求物件的檢視器請求數量。一般而言，未下載完全的原因是檢視器取消下載，例如，按下了不同的連結或關閉瀏覽器。在 CloudFront 存取日誌中，這些請求在 `200` 欄位中的值為 `sc-status`，在 `Error` 欄位中的值為 `x-edge-result-type`。

2xx

HTTP 狀態碼為 2xx、Successful 的請求數。在 CloudFront 存取日誌中，狀態碼會顯示於 `sc-status` 欄位。

3xx

其 HTTP 狀態碼為 3xx、Redirection 的請求數量。3xx 狀態碼代表需要執行其他的動作。例如，301 (永久移除) 表示請求的物件已經移到不同的位置。

4xx

其 HTTP 狀態碼為 4xx、Client Error 的請求數量。4xx 狀態碼代表用戶端明顯出錯。例如，404 (未找到) 表示未找到用戶端請求的物件。

5xx

其 HTTP 狀態碼為 5xx、Server Error 的請求數量。5xx 狀態碼代表原始伺服器並未完成請求的要求。例如，503 (服務無法使用) 表示原始伺服器目前無法使用。

檢視 CloudFront 最佳推薦網站報告

CloudFront 最佳推薦網站報告包含過去 60 天內任何日期範圍的下列項目：

- 前 25 名推薦網站 (針對 CloudFront 為您的分佈的所分佈物件發出最多 HTTP 和 HTTPS 請求的網站網域)
- 來自推薦網站的請求數量
- 推薦網站請求數量佔指定期間請求總數的百分比

最佳推薦網站報告的資料與 CloudFront 存取日誌的來源相同。不過您不需要啟用[存取記錄](#)功能以檢視最佳推薦網站。

最佳推薦網站可能是搜尋引擎、直接連結到您物件的其他網站或是您自己的網站。例如若 `https://example.com/index.html` 連結至 10 個圖形，則 `example.com` 是所有 10 個圖形的推薦網站。

Note

如果使用者直接將 URL 輸入到瀏覽器的地址行中，則不會有所請求物件的推薦網站。

主題

- [在主控台中檢視 CloudFront 最佳推薦網站報告](#)
- [CloudFront 如何計算最佳推薦網站統計資料](#)

- [以 CSV 格式下載資料](#)
- [最佳推薦網站報告中的資料如何與 CloudFront 標準日誌 \(存取日誌\) 中的資料相關](#)

在主控台中檢視 CloudFront 最佳推薦網站報告

您可以在主控台中檢視 CloudFront 最佳推薦網站報告。

檢視 CloudFront 分佈的最佳推薦網站

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽面板中選擇最佳推薦網站。
3. 在 CloudFront 最佳推薦網站報告窗格中，針對 開始日期和結束日期，選取您要顯示的最佳推薦網站日期範圍。

日期和時間都使用國際標準時間 (UTC)。

4. 在 Distribution (分佈) 清單中，選擇要顯示其最佳推薦網站清單的分佈。
5. 選擇更新。

CloudFront 如何計算最佳推薦網站統計資料

為了準確計算前 25 個推薦網站，CloudFront 以 10 分鐘的時間間隔對所有物件的請求進行計數，並維持前 75 個推薦網站的計算加總。清單底部附近的推薦網站會不斷上榜或落榜，所以這些推薦網站的總計是近似值。

75 個推薦網站清單中位居前列的 25 個推薦網站可能會在清單中上升和下降，但幾乎不會完全掉出清單之外，因此這些推薦網站的總計通常更可靠。

以 CSV 格式下載資料

您可使用 CSV 格式下載最佳推薦網站報告。本節說明如何下載報告和描述報告中的值。

以 CSV 格式下載最佳推薦網站報告。

1. 檢視最佳推薦網站報告時，請選擇 CSV。
2. 在開啟檔案名稱對話方塊中，選擇要開啟或儲存檔案。

有關報告的資訊

報告的前幾行包含以下資訊：

版本

此 CSV 檔案的格式版本。

報告

報告名稱。

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

最佳推薦網站報告中的資料

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

推薦網站

推薦網站的網域名稱。

RequestCount

在 Referrer (推薦網站) 欄位中網域名稱的請求總數。

RequestsPct

推薦網站所提交的請求數量佔指定期間請求總數的百分比。

最佳推薦網站報告中的資料如何與 CloudFront 標準日誌 (存取日誌) 中的資料相關

下表清單顯示了 CloudFront 主控台的最佳推薦網站報告中的值與 CloudFront 存取日誌中的值的對應方式。如需有關 CloudFront 存取日誌的詳細資訊，請參閱 [存取日誌 \(標準日誌\)](#)。

推薦網站

推薦網站的網域名稱。在存取日誌中，推薦網站列於 `cs(Referer)` 欄位中。

請求計數

從 Referrer (推薦網站) 欄位中的網域名稱，所發出請求的總數。此值通常與 CloudFront 存取日誌中推薦網站的 GET 請求數量密切對應。

要求 %

推薦網站所提交的請求數量佔指定期間請求總數的百分比。如果您擁有超過 25 個推薦網站，則無法根據此表格中的資料來計算 Request % (請求百分比)，因為 Request Count (請求計數) 欄位並未包含在指定期間內的所有請求。

檢視 CloudFront 用量報告

CloudFront 用量報告包含下列資訊：

- 請求數量 – 顯示在所指定 CloudFront 分佈的每個時間間隔內，CloudFront 從選定區域中的節點回應的請求總數。
- 依據通訊協定傳輸的資料和依目的地的資料傳輸 – 兩者顯示在所指定 CloudFront 分佈的每個時間間隔內，從選定區域中的 CloudFront 節點傳輸的資料總量。它們以不同的方式分隔資料，如下所示：
 - 依據通訊協定 — 以通訊協定分隔資料：HTTP 或 HTTPS。
 - 依目的地：依目的地分隔資料：傳送給您的使用者或您的來源。

CloudFront 用量報告是以 CloudFront AWS 用量報告為基礎。此報告不需要任何其他組態。如需更多詳細資訊，請參閱 [檢視 CloudFront 的 AWS 用量報告](#)。

您可以使用每小時或每天的資料點來檢視過去 60 天內指定日期範圍的報告。您通常可以檢視最近 4 小時前收到的 CloudFront 請求的資料，但資料偶爾會延遲高達 24 小時。

如需更多詳細資訊，請參閱 [用量圖表與 CloudFront 用量報告中的資料的關係](#)。

主題

- [檢視主控台中的 CloudFront 用量報告](#)
- [以 CSV 格式下載資料](#)
- [用量圖表與 CloudFront 用量報告中的資料的關係](#)

檢視主控台中的 CloudFront 用量報告

您可以在主控台中檢視 CloudFront 用量報告。

檢視 CloudFront 用量報告

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽窗格中，選擇 Usage Reports (用量報告)。
3. 在 CloudFront 用量報告窗格中，針對開始日期和結束日期，選取您要顯示圖表的日期範圍。可用的範圍取決於您針對 Granularity (精細度) 選擇的值：
 - 每日 — 若要顯示每天一個資料點的圖表，請選擇前 60 天內的任何日期範圍。
 - 每小時 — 若要顯示每小時一個資料點的圖表，請選擇前 60 天內的任意日期範圍 (最多 14 天)。

日期和時間都使用國際標準時間 (UTC)。

4. 對於 Granularity (精細度)，指定要在圖表中顯示每天一個資料點，或每小時一個資料點。如果您指定的日期範圍大於 14 天，則每小時指定一個資料點的選項不可用。
5. 對於帳單區域，選擇有您要檢視的資料的 CloudFront 帳單區域，或者選擇所有區域。使用量圖表包含用於在指定區域中，CloudFront 在節點內所處理請求的資料。CloudFront 處理請求的區域可能會也可能不會與檢視器的位置相對應。

僅選取包含在分佈價格分級中的區域。否則用量圖表可能不會包含任何資料。例如，如果您為分佈選擇了價格分級「200」，則不包含南美洲和澳大利亞的計費區域，因此 CloudFront 通常不會處理來自這些區域的請求。如需價格分級的詳細資訊，請參閱 [CloudFront 定價](#)。

6. 在 Distribution (分佈) 清單中，選擇要在用量圖表中顯示其資料的分佈：

- 個別分佈 - 此圖表會顯示所選取 CloudFront 分佈的資料。Distribution (分佈) 清單會顯示分佈的分佈 ID 和替代網域名稱 (CNAME) (如果有的話)。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。
- 所有分佈 (不包括已刪除的分佈) – 圖表會針對與目前 AWS 帳戶具有關聯的所有分佈 (不包括已刪除的分佈)，顯示摘要的資料。
- 所有已刪除的分佈 - 圖表會針對與目前 AWS 帳戶具有關聯的所有分佈，以及在過去 60 天內刪除的分佈，顯示摘要的資料。

7. 選擇更新圖形。

Tip

- 若要查看圖表中每日或每小時資料點的資料，請將滑鼠指標停留在資料點上。
- 對於顯示傳輸資料的圖表，請注意可以將垂直擴展變更為 GB、MB 或 KB。

以 CSV 格式下載資料

您可使用 CSV 格式下載用量報告。本節說明如何下載報告和描述報告中的值。

以 CSV 格式下載用量報告。

1. 檢視用量報告時，請選擇 CSV。
2. 在開啟檔案名稱對話方塊中，選擇要開啟或儲存檔案。

有關報告的資訊

報告的前幾行包含以下資訊：

版本

此 CSV 檔案的格式版本。

報告

報告名稱。

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告為 ALL，或者如果您為所有已刪除的分佈執行報告，則為 ALL_DELETED。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

精細程度

報告中的每一行表示一小時還是一天。

BillingRegion

如果您選擇下載所有計費區域的報告，則該檢視器請求源發自洲別或 ALL。

用量報告中的資料

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告為 ALL，或者如果您為所有已刪除的分佈執行報告，則為 ALL_DELETED。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

BillingRegion

為其產生報告的 CloudFront 計費區域，或 ALL。

TimeBucket

資料所應用的小時或天，以國際標準時間 (UTC) 為準。

HTTP

在指定的 CloudFront 分佈的每個時間間隔內，CloudFront 從所選區域中的節點回應的 HTTP 請求數量。數值包含：

- GET 與 HEAD 請求的數量，可讓 CloudFront 將資料傳輸給檢視器
- DELETE、OPTIONS、PATCH、POST 和 PUT 請求的數量，可讓 CloudFront 將資料傳輸到原始伺服器

HTTPS

在指定的 CloudFront 分佈的每個時間間隔內，CloudFront 從所選區域中的節點回應的 HTTPS 請求數量。數值包含：

- GET 與 HEAD 請求的數量，可讓 CloudFront 將資料傳輸給檢視器
- DELETE、OPTIONS、PATCH、POST 和 PUT 請求的數量，可讓 CloudFront 將資料傳輸到原始伺服器

HTTPBytes

在指定的 CloudFront 分佈期間內，透過 HTTP 傳輸來自所選計費區域的 CloudFront 節點傳輸的總資料量。數值包含：

- 回應 GET 和 HEAD 請求，從 CloudFront 將資料傳輸給檢視器
- 對於 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，從 CloudFront 將資料傳輸到原始伺服器
- 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，從 CloudFront 將資料傳輸給使用者

HTTPSBytes

在指定的 CloudFront 分佈期間內，透過 HTTPS 傳輸來自所選計費區域的 CloudFront 節點傳輸的總資料量。數值包含：

- 回應 GET 和 HEAD 請求，從 CloudFront 將資料傳輸給檢視器
- 對於 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，從 CloudFront 將資料傳輸到原始伺服器
- 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，從 CloudFront 將資料傳輸給使用者

BytesIn

在指定 CloudFront 分佈的每個時間間隔內，其選定區域中，從 CloudFront 傳輸到您的原始伺服器所 DELETE、OPTIONS、PATCH、POST 和 PUT 請求的總資料量。

BytesOut

在指定 CloudFront 分佈的每個時間間隔內，透過 HTTP 和 HTTPS 從 CloudFront 向選定區域中的檢視器傳輸的總資料量。數值包含：

- 回應 GET 和 HEAD 請求，從 CloudFront 將資料傳輸給檢視器
- 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，從 CloudFront 將資料傳輸給使用者

用量圖表與 CloudFront 用量報告中的資料的關係

以下清單說明 CloudFront 主控台中的使用量圖表如何與 CloudFront 用量報告中的使用量類型欄位中的值相對應。

主題

- [請求數](#)
- [依據通訊協定傳輸的資料](#)
- [依目的地的資料傳輸](#)

請求數

此圖表顯示指定 CloudFront 分佈的每個時間間隔期間，CloudFront 從所選區域中的節點回應的請求總數，依據通訊協定 (HTTP 或 HTTPS) 和類型 (靜態、動態或 Proxy) 來分隔。

HTTP 請求數

- **region**-Requests-HTTP-Static：在 TTL $\geq$ 3600 秒下，為了物件而提供的 HTTP GET 和 HEAD 請求數
- **region**-Requests-HTTP-Dynamic：已為其提供物件的 HTTP GET 與 HEAD 請求數量 (TTL < 3600 秒)
- **region**-Requests-HTTP-Proxy：CloudFront 轉送到原始伺服器的 HTTP DELETE、OPTIONS、PATCH、POST 和 PUT 請求數量

HTTPS 請求數

- **region**-Requests-HTTPS-Static：在 TTL $\geq$ 3600 秒下，為了物件而提供的 HTTPS GET 和 HEAD 請求數
- **region**-Requests-HTTPS-Dynamic：已為其提供物件的 HTTPS GET 與 HEAD 請求數量 (TTL < 3600 秒)

- **region**-Requests-HTTPS-Proxy : CloudFront 轉送到原始伺服器的 HTTPS DELETE、OPTIONS、PATCH、POST 和 PUT 請求數量

依據通訊協定傳輸的資料

此圖表顯示指定 CloudFront 分佈的每個時間間隔內，從所選區域中的 CloudFront 邊緣節點傳輸的資料總量，依據通訊協定 (HTTP 或 HTTPS)、類型 (靜態、動態或 Proxy) 和目的地 (檢視器或原始伺服器) 分隔。

透過 HTTP 傳輸的資料

- **region**-Out-Bytes-HTTP-Static : 在 TTL $\geq$ 3600 秒下，為了物件而透過 HTTP 所提供的位元組
- **region**-Out-Bytes-HTTP-Dynamic : 透過 HTTP 提供給物件的位元組數 (TTL < 3600 秒)
- **region**-Out-Bytes-HTTP-Proxy : 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTP 從 CloudFront 傳回到檢視器的位元組數。
- **region**-Out-Bytes-HTTP-Proxy : 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTP 從 CloudFront 節點傳輸到您的原始伺服器的總位元組數

透過 HTTPS 傳輸的資料

- **region**-Out-Bytes-HTTPS-Static : 在 TTL $\geq$ 3600 秒下，為了物件而透過 HTTPS 所提供的位元組
- **region**-Out-Bytes-HTTPS-Dynamic : 透過 HTTPS 提供給物件的位元組數 (TTL < 3600 秒)
- **region**-Out-Bytes-HTTPS-Proxy : 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTPS 從 CloudFront 傳回到檢視器的位元組數。
- **region**-Out-Bytes-HTTPS-Proxy : 回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTPS 從 CloudFront 節點傳輸到您的原始伺服器的總位元組數

依目的地的資料傳輸

此圖表顯示指定 CloudFront 分佈的每個時間間隔內，從所選區域中的 CloudFront 邊緣節點傳輸的資料總量，依據目的地 (檢視器或原始伺服器)、通訊協定 (HTTP 或 HTTPS) 及類型 (靜態、動態或 Proxy) 分隔。

從 CloudFront 傳輸到您檢視器的資料

- **region**-Out-Bytes-HTTP-Static : 在 TTL $\geq$ 3600 秒下，為了物件而透過 HTTP 所提供的位元組
- **region**-Out-Bytes-HTTPS-Static : 在 TTL $\geq$ 3600 秒下，為了物件而透過 HTTPS 所提供的位元組

- **region**-Out-Bytes-HTTP-Dynamic：透過 HTTP 提供給物件的位元組數 (TTL < 3600 秒)
- **region**-Out-Bytes-HTTPS-Dynamic：透過 HTTPS 提供給物件的位元組數 (TTL < 3600 秒)
- **region**-Out-Bytes-HTTP-Proxy：回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTP 從 CloudFront 傳回到檢視器的位元組數。
- **region**-Out-Bytes-HTTPS-Proxy：回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTPS 從 CloudFront 傳回到檢視器的位元組數。

從 CloudFront 傳輸到您的原始伺服器的資料

- **region**-Out-OBytes-HTTP-Proxy：回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTP 從 CloudFront 節點傳輸到您的原始伺服器的總位元組數
- **region**-Out-OBytes-HTTPS-Proxy：回應 DELETE、OPTIONS、PATCH、POST 和 PUT 請求，透過 HTTPS 從 CloudFront 節點傳輸到您的原始伺服器的總位元組數

檢視 CloudFront 檢視器報告

CloudFront 檢視器報告包含過去 60 天內任何日期範圍的下列資訊：

- 裝置：最常用來存取內容的裝置類型 (例如桌上型電腦或行動裝置)
- 瀏覽器：最常用於存取內容的前 10 個瀏覽器 (例如 Chrome 或 Firefox)
- 作業系統：存取內容時最常使用的前 10 個作業系統 (例如 Linux、macOS 或 Windows)
- 位置：檢視器存取內容時的前 50 名位置 (根據國家/區域或美國州/區域)。
- 您還可以依據每小時資料點檢視過去 60 天內任何日期範圍 (最長 14 天) 內的位置

Note

您不需要啟用[存取記錄](#)功能來查看檢視器圖表和報告。

主題

- [在主控台中檢視檢視器圖表和報告](#)
- [以 CSV 格式下載資料](#)
- [檢視器報告中包含的資料](#)
- [位置報告中的資料如何與 CloudFront 標準日誌 \(存取日誌\) 中的資料相關](#)

在主控台中檢視檢視器圖表和報告

您可以在主控台中檢視 CloudFront 檢視器圖表和報告。

檢視 CloudFront 檢視器圖表和報告

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽面板中選擇檢視器。
3. 在 CloudFront 檢視器窗格中，針對開始日期和結束日期，選取您要顯示檢視器圖表和報告的日期範圍。

如果是位置圖表，可用的範圍取決於您針對 Granularity (精細度) 所選擇的值：

- 每日 – 若要顯示每天一個資料點的圖表，請選擇前 60 天內的任何日期範圍。
- 每小時 – 若要顯示每小時一個資料點的圖表，請選擇前 60 天內的任意日期範圍 (最多 14 天)。

日期和時間都使用國際標準時間 (UTC)。

4. (僅適用於瀏覽器與作業系統圖表) 針對 Grouping (分組)，指定要依據名稱 (Chrome、Firefox) 或名稱和版本 (Chrome 40.0、Firefox 35.0)，來將瀏覽器與作業系統分組。
5. (僅適用於位置圖表) 針對 Granularity (精細度)，指定要在圖表中顯示每天一個資料點，或每個小時一個資料點。如果您指定的日期範圍大於 14 天，則每小時指定一個資料點的選項不可用。
6. (僅適用於位置圖表) 針對 Details (詳細資訊)，指定要依據國家/地區或美國的州別，來顯示排名在前面的位置。
7. 在 Distribution (分佈) 清單中，選擇要在用量圖表中顯示其資料的分佈：
 - 個別分佈 - 此圖表會顯示所選取 CloudFront 分佈的資料。Distribution (分佈) 清單會顯示分佈的分佈 ID 和替代網域名稱 (CNAME) (如果有的話)。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。
 - 所有分佈 (不包括已刪除的分佈) – 圖表會針對與目前 AWS 帳戶具有關聯的所有分佈 (不包括已刪除的分佈)，顯示摘要的資料。
8. 選擇更新。

若要查看圖表中每日或每小時資料點的資料，請將滑鼠指標停留在資料點上。

以 CSV 格式下載資料

您可使用 CSV 格式下載每個檢視器報告。本節說明如何下載報告和描述報告中的值。

以 CSV 格式下載檢視器報告。

1. 查看檢視器報告時，請選擇 CSV。
2. 選擇您想要下載的資料，例如 Devices (裝置) 或 Devices Trends (裝置趨勢)。
3. 在開啟檔案名稱對話方塊中，選擇要開啟或儲存檔案。

檢視器報告中包含的資料

每份報告的前幾行包含以下資訊：

版本

此 CSV 檔案的格式版本。

報告

報告名稱。

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

群組 (僅限瀏覽器和作業系統報告)

該資料是否依照名稱，還是瀏覽器或作業系統的名稱和版本進行分組。

精細程度

報告中的每一行表示一小時還是一天。

詳細資訊 (僅限位置報告)

無論請求按國家/地區或美國州/區域列出。

下列主題說明不同檢視器報告中的資訊。

主題

- [裝置報告](#)
- [裝置趨勢報告](#)
- [瀏覽器報告](#)
- [瀏覽器趨勢報告](#)
- [作業系統報告](#)
- [作業系統趨勢報告](#)
- [位置報告](#)
- [位置趨勢報告](#)

裝置報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

請求

從每個類型的裝置所收到 CloudFront 的請求數量。

RequestsPct

CloudFront 從每種類型裝置所收到的請求數量，佔 CloudFront 從全部裝置所收到請求總數的百分比。

自訂

User-Agent HTTP 標頭的值與其中一個標準裝置類型 (例如, Desktop 或 Mobile) 沒有關聯的請求。

裝置趨勢報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

TimeBucket

適用於小時或天的資料，以國際標準時間 (UTC) 為準。

桌面

在這段期間，CloudFront 從桌上型電腦收到的請求數。

行動應用程式

在這段期間，CloudFront 從行動裝置收到的請求數。行動裝置可以包括平板電腦和行動電話。如果 CloudFront 無法確定請求來自行動裝置或平板電腦，則會將該請求列入 Mobile 欄位值的計算。

智慧型電視

在這段期間，CloudFront 從智慧型電視收到的請求數。

平板電腦

在這段期間，CloudFront 從平板電腦收到的請求數。如果 CloudFront 無法確定請求來自行動裝置或平板電腦，則會將該請求列入 Mobile 欄位值的計算。

不明

User-Agent HTTP 標頭的值與其中一個標準裝置類型 (例如, Desktop 或 Mobile) 沒有關聯的請求。

空白

在這段期間內，CloudFront 所收到其 HTTP User-Agent 標頭未包含值的請求數量。

瀏覽器報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

群組

發送請求給 CloudFront 的瀏覽器或瀏覽器及其版本，這取決於 Grouping 的值。除了瀏覽器名稱外，可能的值包括下列項目：

- 機器人/爬蟲程式 – 對此值的請求，主要來自於將您的內容建立索引的搜尋引擎。
- 空白 – 其 User-Agent HTTP 標頭值為空白的請求。
- 其他 – CloudFront 已辨識出的瀏覽器，但這些並非最熱門的瀏覽器。如果 Bot/Crawler、Empty 和/或 Unknown 不出現在前 9 個值中，那麼它們也會包括在 Other 中。
- 未知 – 其 User-Agent HTTP 標頭的值與標準瀏覽器不具有關聯的請求。在這個類別的大多數請求都來自自訂應用程式或指令碼。

請求

CloudFront 從每個類型的瀏覽器收到的請求數量。

RequestsPct

CloudFront 從每種類型瀏覽器所收到的請求數量，佔 CloudFront 在這段期間內所收到請求總數的百分比。

瀏覽器趨勢報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

TimeBucket

適用於小時或天的資料，以國際標準時間 (UTC) 為準。

(瀏覽器)

報告中的其餘欄位列出了瀏覽器或瀏覽器及其版本，具體取決於 Grouping 的值。除了瀏覽器名稱外，可能的值包括下列項目：

- 機器人/爬蟲程式 – 對此值的請求，主要來自於將您的內容建立索引的搜尋引擎。
- 空白 – 其 User-Agent HTTP 標頭值為空白的請求。
- 其他 – CloudFront 已辨識出的瀏覽器，但這些並非最熱門的瀏覽器。如果 Bot/Crawler、Empty 和/或 Unknown 不出現在前 9 個值中，那麼它們也會包括在 Other 中。
- 未知 – 其 User-Agent HTTP 標頭的值與標準瀏覽器不具有關聯的請求。在這個類別的大多數請求都來自自訂應用程式或指令碼。

作業系統報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

群組

發送請求給 CloudFront 的作業系統或作業系統及其版本，這取決於 Grouping 的值。除了作業系統名稱外，可能的值包括下列項目：

- 機器人/爬蟲程式 – 對此值的請求，主要來自於將您的內容建立索引的搜尋引擎。
- 空白 – 其 User-Agent HTTP 標頭值為空白的請求。
- 其他 – CloudFront 已辨識出的作業系統，但這些並非最熱門的作業系統。如果 Bot/Crawler、Empty 和/或 Unknown 不出現在前 9 個值中，那麼它們也會包括在 Other 中。
- 未知 – 其 User-Agent HTTP 標頭的值與標準瀏覽器不具有關聯的請求。在這個類別的大多數請求都來自自訂應用程式或指令碼。

請求

CloudFront 從每個類型的作業系統收到的請求數量。

RequestsPct

CloudFront 從每種類型作業系統所收到的請求數量，佔 CloudFront 在這段期間內所收到請求總數的百分比。

作業系統趨勢報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

TimeBucket

適用於小時或天的資料，以國際標準時間 (UTC) 為準。

(作業系統)

報告中的其餘欄位列出了作業系統或作業系統及其版本，具體取決於 Grouping 的值。除了作業系統名稱外，可能的值包括下列項目：

- 機器人/爬蟲程式 – 對此值的請求，主要來自於將您的內容建立索引的搜尋引擎。
- 空白 – 其 User-Agent HTTP 標頭值為空白的請求。
- 其他 – CloudFront 已辨識出的作業系統，但這些並非最熱門的作業系統。如果 Bot/Crawler、Empty 和/或 Unknown 不出現在前 9 個值中，那麼它們也會包括在 Other 中。

- 未知 – 其 User-Agent HTTP 標頭中未註明作業系統的請求。

位置報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

LocationCode

CloudFront 收到請求的位置的縮寫。如需有關可能的值的詳細資訊，請參閱[位置報告中的資料如何與 CloudFront 標準日誌 \(存取日誌\) 中的資料相關](#)的位置說明。

LocationName

CloudFront 收到請求的位置的名稱。

請求

CloudFront 從每個位置收到的請求數。

RequestsPct

CloudFront 從每個位置所收到的請求數量，佔 CloudFront 在這段期間內從所有位置收到請求總數的百分比。

TotalBytes

在指定的分佈和期間，CloudFront 為該國家/區域或州的檢視器提供的位元組數。

位置趨勢報告

該報告包含以下值：

DistributionID

為執行報告所分佈的 ID，如果您為所有分佈執行報告，則為 ALL。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

TimeBucket

適用於小時或天的資料，以國際標準時間 (UTC) 為準。

(位置)

報表中的其餘欄位列出了 CloudFront 收到請求的位置。如需有關可能的值的詳細資訊，請參閱[位置報告中的資料如何與 CloudFront 標準日誌 \(存取日誌\) 中的資料相關](#)的位置說明。

位置報告中的資料如何與 CloudFront 標準日誌 (存取日誌) 中的資料相關

下表的清單顯示了 CloudFront 主控台的位置報告中的資料與 CloudFront 存取日誌中的值的對應方式。如需有關 CloudFront 存取日誌的詳細資訊，請參閱[存取日誌 \(標準日誌\)](#)。

位置

檢視器所在國家/地區或美國州。在存取日誌中，c-ip 欄位包含檢視器所正在執行之裝置的 IP 位址。我們使用地理定位資料來識別根據 IP 位址的裝置的地理位置。

如果您是依照國家/地區來顯示位置報告，則國家/地區清單的依據為[ISO 3166-2 國家/地區及其子行政區名稱表示代碼 – 第 2 部分：國家/地區子行政區代碼](#)。國家/地區清單包括以下附加值：

- 匿名代理 – 來自匿名代理的請求。
- 衛星服務供應商 – 來自於衛星服務供應商的請求，此等供應商提供網際網路服務給多個國家/區域。檢視器可能處於詐騙風險很高的國家/地區。
- 歐洲 (未知) – 請求的原始伺服器 IP，位於多個歐洲國家/區域所使用的區塊中。無法確定該請求所來自的國家/地區。CloudFront 使用 歐洲 (未知) 做為預設值。
- 亞太區域 (未知) – 請求的原始伺服器 IP，位於多個亞太區域國家/區域所使用的區塊中。無法確定該請求所來自的國家/地區。CloudFront 使用亞太區域 (未知) 做為預設值。

如果您顯示美國區域的 Locations (位置) 報告，請注意，該報告可能包括美國領土和美國兵力部署的地區。

Note

如果 CloudFront 無法判斷使用者的位置，會在檢視器報告中將位置顯示為「未知」。

請求計數

在指定的分佈和期間，檢視器所在的國家/地區或美國州的請求總數。這個值通常與 CloudFront 存取日誌中來自該國家/區域或州的 IP 位址的 GET 請求數量密切對應。

要求 %

下列其中一項，取決於您針對 Details (詳細資訊) 所選擇的值：

- 國家/區域 – 來自這個國家/區域的請求數量，佔請求總數的百分比。
- 美國 – 來自該州的請求占美國請求總數的百分比。

如果請求來自超過 50 個國家/地區，則無法根據此表格中的資料計算 Request % (請求百分比)，因為 Request Count (請求計數) 欄位並未包含在指定期間內的所有請求。

位元組

在指定的分佈和期間，CloudFront 為該國家/區域或州的檢視器提供的位元組數。若要將這個欄位中的資料顯示變更為 KB、MB 或 GB，請選擇在欄位標頭中的連結。

使用 Amazon CloudWatch 監控 CloudFront 指標

Amazon CloudFront 與 Amazon CloudWatch 整合，並自動為分佈和邊緣函數 (包含 [Lambda@Edge](#) 和 [CloudFront Functions](#)) 發佈運作指標。您可利用這些指標以協助疑難排解、追蹤和偵錯問題。這些指標大部分以一組圖形顯示於 CloudFront 主控台，也可以使用 CloudFront API 或 CLI 存取。所有這些指標皆可在 [CloudWatch 主控台](#) 或透過 CloudWatch API 或 CLI 取得。CloudFront 指標不會計入 [CloudWatch 配額 \(前稱為限制\)](#)，也不會產生任何額外的費用。

除了 CloudFront 發佈的預設指標之外，您還可以開啟其他指標，但需要支付額外費用。額外的指標適用於 CloudFront 發佈，而且必須分別為每個發佈開啟。如需成本的詳細資訊，請參閱 [the section called “估算其他 CloudFront 指標的成本”](#)。

您也可以 CloudFront 主控台、CloudWatch 主控台、API 或 CLI 中，根據這些指標設定警示。例如，您可以根據 5xxErrorRate 指標設定警示，該指標代表回應的 HTTP 狀態碼在 500 到 599 範圍內之所有瀏覽者請求的百分比。當錯誤率在特定時間內達到特定值 (例如連續 5 分鐘達到 5% 的請求) 時，就會觸發警示。您可以在建立警示時指定警示的值及其時間單位。

❗ 備註

- 您在 CloudFront 主控台中建立 CloudWatch 警示時，系統會在美國東部 (維吉尼亞北部) 區域 (us-east-1) 為您建立警示。如果您從 CloudWatch 主控台建立警示，則必須使用相同區域。由於 CloudFront 是全球服務，因此服務指標會傳送到美國東部 (維吉尼亞北部)。
- 建立警示時適用 [標準 CloudWatch 定價](#)。

主題

- [檢視 CloudFront 和邊緣函數指標](#)
- [建立 指標的 警示](#)
- [以 CSV 格式下載指標資料](#)
- [CloudFront 的指標類型](#)

檢視 CloudFront 和邊緣函數指標

您可以在 CloudFront 主控台中檢視有關 CloudFront 發佈和 [邊緣函數](#) 的操作指標。

在 CloudFront 檢視 CloudFront 和邊緣函數指標

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽窗格中，選擇 Monitoring (監控)。
3. 若要檢視特定 CloudFront 發佈或邊緣函數的相關活動圖表，請選擇一個圖表，然後選擇 View distribution metrics (檢視發佈指標) 或 View metrics (檢視指標)。
4. 您可以執行以下動作以自訂圖表：
 - a. 若要變更顯示在圖表中資訊的時間範圍，請選擇 1h (1 小時)、3h (3 小時) 或其他範圍或指定自訂範圍。
 - b. 若要變更圖表中 CloudFront 更新資訊的頻率，請選擇重新整理圖示旁的向下箭頭，然後選擇重新整理速率。預設重新整理速率為 1 分鐘，但您可以選擇其他選項。
5. 若要在 CloudWatch 主控台中檢視 CloudFront 圖表，請選擇 Add to dashboard (新增至儀表板)。在 CloudWatch 主控台中，您必須使用美國東部 (維吉尼亞北部) 區域檢視圖表。

主題

- [預設的 CloudFront 分佈指標](#)
- [開啟其他 CloudFront 分佈指標](#)
- [預設 Lambda@Edge 函數指標](#)
- [預設的 CloudFront Functions 指標](#)

預設的 CloudFront 分佈指標

所有 CloudFront 發佈皆包含下列預設指標，無需額外付費：

請求

CloudFront 收到之所有 HTTP 方法以及 HTTP 和 HTTPS 請求的檢視器請求總數。

下載的位元組數

檢視器針對 GET 和 HEAD 請求下載的位元組總數。

上傳的位元組數

檢視器使用 OPTIONS、POST 和 PUT 請求上傳至 CloudFront 的位元組總數。

4xx 錯誤率

回應的 HTTP 狀態碼為 4xx 之所有瀏覽者請求的百分比。

5xx 錯誤率

回應的 HTTP 狀態碼為 5xx 之所有瀏覽者請求的百分比。

總錯誤率

回應的 HTTP 狀態碼為 4xx 或 5xx 之所有瀏覽者請求的百分比。

在 CloudFront 主控台的監控頁面上，每個 CloudFront 分佈的這些指標都會以圖表顯示。在每個圖表上，總計會以 1 分鐘為間隔來顯示。除了檢視圖形之外，您也可以[以 CSV 檔案格式下載指標報告](#)。

開啟其他 CloudFront 分佈指標

除了預設指標之外，您還可以開啟其他指標，但需要支付額外費用。如需成本的詳細資訊，請參閱[the section called “估算其他 CloudFront 指標的成本”](#)。

必須分別為每個分佈開啟這些額外指標：

快取命中率

CloudFront 從其快取中提供內容的所有可快取請求的百分比。HTTP POST 和 PUT 請求及錯誤不視為可快取請求。

來源延遲

對於原始伺服器提供的請求 (非從 CloudFront 快取提供)，從 CloudFront 收到請求到開始對網路 (非檢視器) 提供回應所花費的總時長。這也稱為第一位元組延遲或第一個位元組的時間。

依狀態碼分類的錯誤率

其回應的 HTTP 狀態碼是 4xx 或 5xx 範圍中之特定程式碼的所有瀏覽者請求的百分比。此指標適用於下列所有錯誤碼：401、403、404、502、503 和 504。

您可以在 CloudFront 主控台中、使用 CloudFormation、使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API 開啟其他指標。

Console

開啟其他指標

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽窗格中，選擇 Monitoring (監控)。
3. 選擇要開啟其他指標的分佈，然後選擇 View distribution metrics (檢視分佈指標)。
4. 選擇 Manage additional metrics (管理其他指標)。
5. 在 Manage additional metrics (管理其他指標) 視窗中，開啟 Enabled (已啟用)。開啟其他指標後，您可以關閉 Manage additional metrics (管理其他指標) 視窗。

開啟其他指標後，它們會顯示在圖表中。在每個圖表上，總計會以 1 分鐘為間隔來顯示。除了檢視圖形之外，您也可以 [以 CSV 檔案格式下載指標報告](#)。

CloudFormation

若要使用 CloudFormation 開啟其他指標，請使用 `AWS::CloudFront::MonitoringSubscription` 資源類型。下列範例顯示用於啟用其他指標的 YAML 格式 CloudFormation 範本語法。

```
Type: AWS::CloudFront::MonitoringSubscription
```

Properties:DistributionId: *EDFDVBD6EXAMPLE***MonitoringSubscription:**

RealtimeMetricsSubscriptionConfig:

RealtimeMetricsSubscriptionStatus: Enabled

CLI

若要使用 AWS Command Line Interface (AWS CLI) 管理其他指標，請使用下列其中一個命令：

開啟分佈的其他指標

- 使用 `create-monitoring-subscription` 命令，如下列範例所示。將 *EDFDVBD6EXAMPLE* 取代為您要啟用其他指標的分佈 ID。

```
aws cloudfront create-monitoring-subscription --  
distribution-id EDFDVBD6EXAMPLE --monitoring-subscription  
RealtimeMetricsSubscriptionConfig={RealtimeMetricsSubscriptionStatus=Enabled}
```

查看是否已針對分佈開啟其他指標

- 使用 `get-monitoring-subscription` 命令，如下列範例所示。將 *EDFDVBD6EXAMPLE* 取代為您正在檢查的分佈 ID。

```
aws cloudfront get-monitoring-subscription --distribution-id EDFDVBD6EXAMPLE
```

關閉分佈的其他指標

- 使用 `delete-monitoring-subscription` 命令，如下列範例所示。將 *EDFDVBD6EXAMPLE* 取代為您要關閉其他指標的分佈 ID。

```
aws cloudfront delete-monitoring-subscription --distribution-id EDFDVBD6EXAMPLE
```

API

若要使用 CloudFront API 管理其他指標，請使用下列其中一項 API 作業。

- 若要開啟分佈的其他指標，請使用 [CreateMonitoringSubscription](#)。
- 若要查看是否已針對分佈開啟其他指標，請使用 [GetMonitoringSubscription](#)。
- 若要關閉分佈的其他指標，請使用 [DeleteMonitoringSubscription](#)。

如需這些 API 操作的詳細資訊，請參閱 AWS SDK 或其他 API 用戶端的 API 參考文件。

估算其他 CloudFront 指標的成本

在美國東部 (維吉尼亞北部) 區域，當您為某個發佈開啟其他指標時，CloudFront 最多會傳送 8 個指標到 CloudWatch。CloudWatch 會按固定的低費率向每個指標收費。此費率每個指標每月僅收取一次 (每個分佈最多 8 個指標)。這是固定費率，因此無論 CloudFront 發佈接收或傳送的請求或回應數量多少，您的成本均維持不變。如需每指標的費率，請參閱 [Amazon CloudWatch 定價頁面](#) 和 [CloudWatch 定價計算工具](#)。當您使用 CloudWatch API 擷取指標時，需另行支付 API 費用。

預設 Lambda@Edge 函數指標

您可以使用 CloudWatch 指標即時監控 Lambda@Edge 函數的問題。使用這些指標無需負擔額外費用。

當您將 Lambda @Edge 函數附加至 CloudFront 發佈中的快取行為，Lambda 會開始自動將指標傳送至 CloudWatch。所有 Lambda 區域均可使用指標，但若要在 CloudWatch 主控台中檢視指標或從 CloudWatch API 取得指標資料，您必須使用美國東部 (維吉尼亞北部) 區域 (us-east-1)。指標群組名稱格式為：AWS/CloudFront/*distribution-ID*，其中 *distribution-ID* 是 Lambda@Edge 函數所關聯的 CloudFront 分佈 ID。如需有關 Amazon CloudWatch 的詳細資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。

在 CloudFront 主控台的監控頁面上，每個 Lambda@Edge 函數的下列預設指標都會以圖表顯示：

- 5xxLambda@Edge 的 錯誤率
- Lambda 執行錯誤
- Lambda 無效回應
- Lambda 節流器

每個圖表都包含了呼叫、錯誤、調節等資訊。在每個圖表上的總計也會以 1 分鐘為間隔，並按 AWS 區域分組來顯示。

如果您看到了一個峰值錯誤而想進行調查，您可以選擇一個函數，然後按 AWS 區域檢視日誌檔，直到您判斷出哪些 AWS 區域的哪些函數造成問題。如需針對 Lambda@Edge 錯誤進行故障診斷的詳細資訊，請參閱：

- [the section called “如何判斷故障的類型”](#)
- [在 AWS 上針對內容交付執行偵錯的四個步驟](#)

預設的 CloudFront Functions 指標

CloudFront Functions 會將運作指標傳送到 Amazon CloudWatch，以便您監控自己的函數。檢視這些指標可協助您疑難排解、追蹤和偵錯問題。CloudFront Functions 會將下列指標發佈至 CloudWatch：

- 叫用 (FunctionInvocations) – 在給定時間段內啟動 (叫用) 函數的次數。
- 驗證錯誤 (FunctionValidationErrors) – 函數在給定時間段內產生的驗證錯誤數。函數成功執行但返回無效的資料 (無效的 [事件物件](#)) 時，就會發生驗證錯誤。
- 執行錯誤 (FunctionExecutionErrors) – 在給定時間段內發生的執行錯誤次數。當函數無法成功完成時，就會發生執行錯誤。
- 運用利用率 (FunctionComputeUtilization) – 執行函數所花費的時間，以所允許時間上限的百分比表示。例如，35 的值表示該函數以所允許時間上限的 35% 完成。此指標是介於 0 到 100 之間的數字。

如果此值達到或接近 100，則函數已使用或接近使用允許的執行時間，且後續請求可能會受到限流。如果您的函數以 80% 以上的使用率執行，建議您檢閱函數，以減少執行時間並改善使用率。例如您可能只想要記錄錯誤、簡化任何複雜的 regex 表達式，或移除複雜 JSON 物件不必要的剖析。

- 調節 (FunctionThrottles) – 在特定期間內調節函數的次數。可能會因下列原因而對函數進行調節：
 - 函數連續超過允許執行的最長時間
 - 函數導致編譯錯誤
 - 每秒請求數異常高

CloudFront KeyValueStore 也會將下列操作指標傳送至 Amazon CloudWatch：

- 讀取請求 (KvsReadRequests)：函數在指定期間內成功從鍵值存放區讀取的次數。
- 讀取錯誤 (KvsReadErrors)：函數在指定期間內無法從鍵值存放區讀取的次數。

所有這些指標都會發佈至美國東部 (維吉尼亞北部) 區域 (us-east-1) 中的 CloudWatch，位於 CloudFront 命名空間中。您也可以在此 CloudWatch 主控台中檢視這些指標。在 CloudWatch 主控台中，您可以按每個函數檢視指標，也可以按每個發佈的每個函數檢視指標。

您也可以使用 CloudWatch 根據這些指標設定警示。例如，您可以根據執行時間 (FunctionComputeUtilization) 指標來設定警示，這代表函數執行所花費的可用時間百分比。執行時間達到特定值且持續特定時間長度時。例如若您連續 15 分鐘選擇 70% 以上的可用時間，就會觸發警示。您可以在建立警示時指定警示的值及其時間單位。

Note

CloudFront Functions 只會針對 LIVE 階段中的函數將指標傳送至 CloudWatch，此類函數會執行以回應生產請求和回應。[測試函數](#)時，CloudFront 不會將任何指標傳送至 CloudWatch。測試輸出包含錯誤、運算利用率和函數日誌 (console.log() 陳述式) 的相關資訊，但此資訊不會傳送至 CloudWatch。

如需如何使用 CloudWatch API 取得這些指標的相關資訊，請參閱 [the section called “CloudFront 指標”](#)。

建立 指標的 警示

您可以在 CloudFront 主控台中設定警示，讓 Amazon Simple Notification Service (Amazon SNS) 根據特定的 CloudFront 指標通知您。

建立指標的警示

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽窗格中，選擇 Alarms (警示)。
3. 選擇建立警示。
4. 針對詳細資訊指定以下內容：
 - a. 警示名稱：警示的名稱。
 - b. 分佈：您要為其建立警示的 CloudFront 分佈。
5. 針對條件指定下列項目：
 - a. 指標：您要為其建立警示的指標。

- b. "IF" <condition>：指定 CloudWatch 觸發警示的閾值，並將通知傳送到 Amazon SNS 主題。
例如，要在 5xx 錯誤率超過 1% 時收到通知，請指定以下內容：

5xx 錯誤率 > 1

- c. "FOR" 連續期間：觸發警示之前必須符合條件的持續時間。您選擇值時，請盡量在值之間取得適當平衡，讓該值不會針對暫時的問題發出警示，但會針對持續或實際問題發出警示。
- d. (選用) 通知：此指標觸發警示時，要傳送通知的目標 Amazon SNS 主題。

6. 選擇 Create alarm (建立警示)。

 備註

- 您輸入條件的值時，請使用無標點符號的整數。例如，若要指定一千，請輸入 **1000**。
- 對於 4xx、5xx 和總錯誤率，您指定的值為百分比。
- 對於請求、下載的位元組和上傳的位元組，您指定的值是單位。例如，1073742000 個位元組。

如需建立 Amazon SNS 主題的詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的[建立 Amazon SNS 主題](#)。

以 CSV 格式下載指標資料

您可以使用 CSV 格式下載 CloudFront 發佈的 CloudWatch 指標資料。

以 CSV 格式下載指標資料

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/cloudfront/v4/home> 的 CloudFront 主控台。
2. 在導覽窗格中，選擇 Monitoring (監控)。
3. 選擇分佈，然後選擇檢視分佈指標。
4. 選擇下載 CSV，然後選擇期間 (例如過去 1 天 (1 小時期間))。
5. 下載檔案之後，請將其開啟以檢視下列資訊。

主題

- [有關報告的資訊](#)

- [指標報告中的資料](#)

有關報告的資訊

報告的前幾行包含以下資訊：

版本

CloudFront 報告版本。

報告

報告名稱。

DistributionID

您執行報表的分佈 ID。

StartDateUTC

您所執行報告日期範圍的開始時間，是以國際標準時間 (UTC) 為準。

EndDateUTC

您執行報告的日期範圍的結束時間，以國際標準時間 (UTC) 為準。

GeneratedTimeUTC

您執行報告的日期和時間，以國際標準時間 (UTC) 為準。

精細程度

在報告中每行的時間段，例如 ONE_MINUTE。

指標報告中的資料

該報告包含以下值：

DistributionID

您執行報表的分佈 ID。

FriendlyName

分佈的備用網域名稱 (CNAME)，如果有的話。如果分佈沒有備用網域名稱，則該清單包括分佈的原始網域名稱。

TimeBucket

適用於小時或天的資料，以國際標準時間 (UTC) 為準。

請求

在這段期間，所有 HTTP 狀態碼 (例如 200、404 等) 和所有方法 (例如 GET、HEAD、POST 等) 的請求總數。

BytesDownloaded

瀏覽者在這段期間為指定分佈下載的位元組數。

BytesUploaded

瀏覽者在這段期間為指定分佈上傳的位元組數。

TotalErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 4xx 或 5xx 錯誤的請求百分比。。

4xxErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 4xx 錯誤的請求百分比。

5xxErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 5xx 錯誤的請求百分比。

如果您已為分佈[開啟其他指標](#)，則報表也會包含下列額外值：

401ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 401 錯誤的請求百分比。

403ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 403 錯誤的請求百分比。

404ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 404 錯誤的請求百分比。

502ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 502 錯誤的請求百分比。

503ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 503 錯誤的請求百分比。

504ErrorRatePct

在這段期間，對於指定分佈，HTTP 狀態碼為 504 錯誤的請求百分比。

OriginLatency

對於原始伺服器提供的請求 (非從 CloudFront 快取提供)，從 CloudFront 收到請求到開始對網路 (非檢視器) 提供回應所花費的總時長 (以毫秒為單位)。這也稱為第一位元組延遲或第一個位元組的時間。

CacheHitRate

CloudFront 從其快取中提供內容的所有可快取請求的百分比。HTTP POST 和 PUT 請求及錯誤不視為可快取請求。

CloudFront 的指標類型

您可使用 CloudWatch API 或 AWS Command Line Interface (AWS CLI) 取得您建置之程式或應用程式中的 CloudFront 指標。您可以使用原始資料來建置自己的自訂儀表板、您自己的警示工具等等。

如需詳細資訊，請參閱《AWS CLI 命令參考》中的 [get-metric-data](#) 或《Amazon CloudWatch API 參考》中的 [GetMetricData](#) API 操作。

主題

- [所有 CloudFront 指標的值](#)
- [CloudFront 分發指標的值](#)
- [CloudFront 函數指標的值](#)

Note

若要從 CloudWatch API 取得 CloudFront 指標，您必須使用美國東部 (維吉尼亞北部) 區域 (us-east-1)。您還必須知道每個指標的特定值和類型。

所有 CloudFront 指標的值

下列值適用於所有 CloudFront 指標：

Namespace

Namespace 的值永遠為 AWS/CloudFront。

維度

每個 CloudFront 指標都有下列維度：

DistributionId

您想要取得指標之目標 CloudFront 分發的 ID。

FunctionName

您要取得指標之目標函數的名稱 (在 CloudFront Functions 中)。

此維度僅適用於函數。

Region

Region 值一律為 Global，因為 CloudFront 是全球服務。

CloudFront 分發指標的值

使用下列清單中的資訊，從 CloudWatch API 取得特定 CloudFront 分發的詳細資訊。其中一些指標只有在您已為分佈開啟其他指標時才能使用。

Note

每個指標僅適用一個統計資料 (Average 或 Sum)。下列清單指定適用於該指標的統計資料。

4xx 錯誤率

回應的 HTTP 狀態碼為 4xx 之所有瀏覽者請求的百分比。

- 指標名稱：4xxErrorRate
- 有效統計資訊：Average
- 單位：Percent

401 錯誤率

回應的 HTTP 狀態碼為 401 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：401ErrorRate
- 有效統計資訊：Average
- 單位：Percent

403 錯誤率

回應的 HTTP 狀態碼為 403 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：403ErrorRate
- 有效統計資訊：Average
- 單位：Percent

404 錯誤率

回應的 HTTP 狀態碼為 404 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：404ErrorRate
- 有效統計資訊：Average
- 單位：Percent

5xx 錯誤率

回應的 HTTP 狀態碼為 5xx 之所有瀏覽者請求的百分比。

- 指標名稱：5xxErrorRate
- 有效統計資訊：Average
- 單位：Percent

502 錯誤率

回應的 HTTP 狀態碼為 502 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：502ErrorRate
- 有效統計資訊：Average
- 單位：Percent

503 錯誤率

回應的 HTTP 狀態碼為 503 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：503ErrorRate
- 有效統計資訊：Average
- 單位：Percent

504 錯誤率

回應的 HTTP 狀態碼為 504 之所有瀏覽者請求的百分比。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：504ErrorRate
- 有效統計資訊：Average
- 單位：Percent

下載的位元組數

檢視器針對 GET 和 HEAD 請求下載的位元組總數。

- 指標名稱：BytesDownloaded
- 有效統計資訊：Sum
- 單位：None

上傳的位元組數

檢視器使用 OPTIONS、POST 和 PUT 請求上傳至 CloudFront 的位元組總數。

- 指標名稱：BytesUploaded
- 有效統計資訊：Sum
- 單位：None

快取命中率

CloudFront 從其快取中提供內容的所有可快取請求的百分比。HTTP POST 和 PUT 請求及錯誤不視為可快取請求。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：CacheHitRate
- 有效統計資訊：Average
- 單位：Percent

來源延遲

對於原始伺服器提供的請求 (非從 CloudFront 快取提供)，從 CloudFront 收到請求到開始對網路 (非檢視器) 提供回應所花費的總時長 (以毫秒為單位)。這也稱為第一位元組延遲或第一個位元組的時間。若要取得此指標，您必須先[開啟其他指標](#)。

- 指標名稱：OriginLatency
- 有效統計資訊：Percentile
- 單位：Milliseconds

 Note

若要從 CloudWatch API 取得 Percentile 統計資料，請使用 ExtendedStatistics 參數，不要使用 Statistics。如需詳細資訊，請參閱 Amazon CloudWatch API 參考中的 [GetMetricStatistics](#)，或 [AWS 開發套件](#) 的參考文件。

請求

CloudFront 收到之所有 HTTP 方法以及 HTTP 和 HTTPS 請求的檢視器請求總數。

- 指標名稱：Requests
- 有效統計資訊：Sum
- 單位：None

總錯誤率

回應的 HTTP 狀態碼為 4xx 或 5xx 之所有瀏覽者請求的百分比。

- 指標名稱：TotalErrorRate
- 有效統計資訊：Average
- 單位：Percent

CloudFront 函數指標的值

使用下列清單中的資訊，從 CloudWatch API 取得特定 CloudFront 函數的詳細資訊。

 Note

每個指標僅適用一個統計資料 (Average 或 Sum)。下列清單指定適用於該指標的統計資料。

呼叫

在給定時間段內啟動 (叫用) 函數的次數。

- 指標名稱：FunctionInvocations

- 有效統計資訊：Sum
- 單位：None

驗證錯誤

函數在給定時間段內產生的驗證錯誤數。函數成功執行但返回無效的資料 (無效的事件物件) 時，就會發生驗證錯誤。

- 指標名稱：FunctionValidationErrors
- 有效統計資訊：Sum
- 單位：None

執行錯誤

在給定時間段內發生的執行錯誤次數。當函數無法成功完成時，就會發生執行錯誤。

- 指標名稱：FunctionExecutionErrors
- 有效統計資訊：Sum
- 單位：None

運算利用率

執行函數所花費的時間 (0-100)，以所允許時間上限的百分比表示。例如，35 的值表示該函數以所允許時間上限的 35% 完成。

- 指標名稱：FunctionComputeUtilization
- 有效統計資訊：Average
- 單位：Percent

調節

在特定期間內調節函數的次數。

- 指標名稱：FunctionThrottles
- 有效統計資訊：Sum
- 單位：None

CloudFront 和邊緣函數記錄

Amazon CloudFront 提供不同類型的日誌記錄。您可以記錄發送到 CloudFront 分發的檢視器請求，也可以在 AWS 帳戶中記錄 CloudFront 服務活動 (API 活動)。您也可以從 CloudFront Functions 和 Lambda@Edge 函數取得日誌。

記錄請求

CloudFront 提供了以下方法來記錄分發的請求。

存取日誌（標準日誌）

CloudFront 存取日誌提供有關對分佈提出的每個請求的詳細記錄。您可以將日誌用於各種情境，例如安全性和存取稽核。

CloudFront 存取日誌會交付至您指定的交付目的地。

視需要使用存取日誌：

- 歷史分析和報告
- 安全稽核和合規要求
- 符合成本效益的長期日誌保留

如需詳細資訊，請參閱[存取日誌（標準日誌）](#)。

即時存取日誌

CloudFront 即時存取日誌會在收到請求的幾秒鐘內交付，並提供對分發提出請求的相關資訊。您可以選擇即時存取日誌的取樣率，也就是您想要接收即時存取日誌記錄的請求百分比。此外，您還可以選擇要在日誌記錄中接收的特定欄位。即時存取日誌非常適合即時監控內容交付效能。

CloudFront 即時存取日誌會交付至 Amazon Kinesis Data Streams 中您選擇的資料串流。除了您使用 Kinesis Data Streams 所產生的費用之外，CloudFront 還會收取即時存取日誌的費用。

視需要使用即時存取日誌：

- 即時監控和提醒
- 即時儀表板和營運洞見

如需詳細資訊，請參閱[使用即時存取日誌](#)。

連線日誌

連線日誌提供伺服器與用戶端之間連線的詳細資訊，用於啟用 mTLS 的分佈。連線日誌提供用戶端憑證資訊的可見性、mTLS 身分驗證失敗的原因，以及是否允許或拒絕連線。

如同存取日誌（標準日誌），連線日誌會傳送到您指定的交付目的地。

 Note

若要啟用連線日誌，您必須先為分發[啟用 mTLS](#)。

需要時使用連線日誌：

- 在 TLS 交握期間成功或不成功連線的原因
- 用戶端憑證資訊的可見性

如需詳細資訊，請參閱[使用連線日誌的可觀測性](#)。

記錄邊緣函數

您可以使用 Amazon CloudWatch Logs 取得 Lambda @Edge 和 CloudFront Functions 兩種邊緣函數的日誌。您可以使用 CloudWatch 主控台或 CloudWatch Logs API 存取日誌檔案。如需詳細資訊，請參閱[the section called “邊緣函數日誌”](#)。

記錄服務活動

您可以使用 AWS CloudTrail 來記錄帳戶中 AWS 的 CloudFront 服務活動 (API 活動)。CloudTrail 提供 CloudFront 中使用者、角色 AWS 或服務所採取 API 動作的記錄。您可以利用 CloudTrail 所收集的資訊來判斷向 CloudFront 發出的 API 請求，以及發出請求的 IP 位址、人員、時間和其他詳細資訊。

如需詳細資訊，請參閱[使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫](#)。

如需有關日誌記錄的詳細資訊，請參閱下列主題：

主題

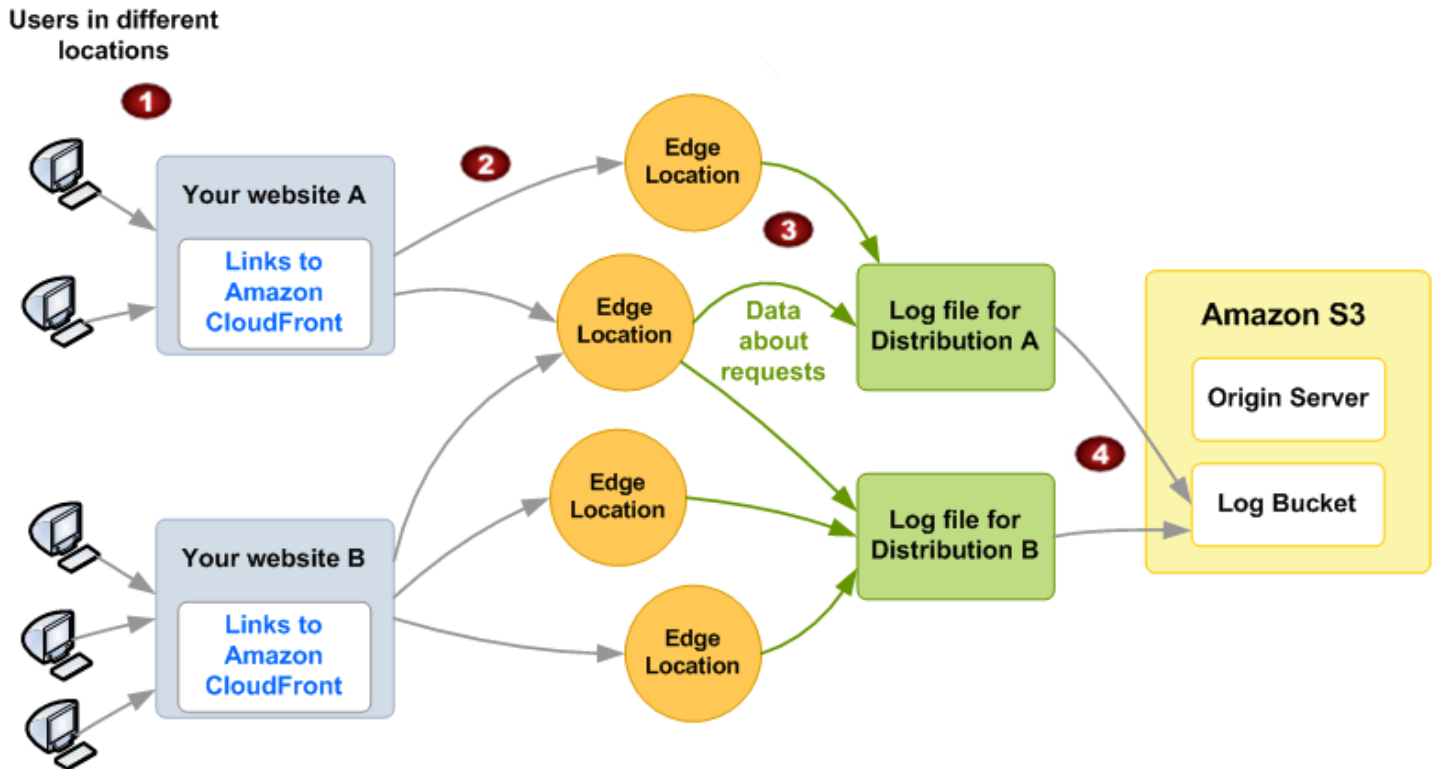
- [存取日誌（標準日誌）](#)
- [使用即時存取日誌](#)
- [邊緣函數日誌](#)
- [使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫](#)

存取日誌（標準日誌）

您可以設定 CloudFront 以建立日誌檔案，其中包含有關 CloudFront 收到的每個使用者 (檢視器) 請求的詳細資訊。這些稱為存取日誌，也稱為標準日誌。

每個日誌包含收到請求的請求時間、處理時間、請求路徑和伺服器回應等資訊。您可以使用這些存取日誌來分析回應時間，並針對問題進行故障診斷。

下圖顯示 CloudFront 如何記錄有關物件請求的資訊。在此範例中，分佈會設定為將存取日誌傳送至 Amazon S3 儲存貯體。



1. 在此範例中，您有兩個網站，A 和 B 以及兩個對應的 CloudFront 分佈。使用者使用與分佈相關聯的 URL 請求物件。
2. CloudFront 將每個請求路由到適當的節點。
3. CloudFront 寫入有關每個請求的資料，記錄特定於該分佈的檔案。在此範例中，與分佈 A 相關的請求資訊會進入分佈 A 的日誌檔案，與分佈 B 相關的請求資訊會進入分佈 B 的日誌檔案。
4. CloudFront 定期將分佈的日誌檔案儲存在您啟用記錄時指定的 Amazon S3 儲存貯體中。然後 CloudFront 在新的分佈日誌檔案中啟動儲存有關後續請求的資訊。

如果在指定的某小時內沒有檢視器存取您的內容，您不會收到該小時的任何日誌檔案。

Note

我們建議您使用日誌，了解內容請求的性質，而不是像完全考量所有請求。CloudFront 依最佳作業基礎交付存取日誌。在實際處理請求之後，才可能長時間交付特定請求的日誌項目，在極

少數的情況下，有可能完全不會交付日誌項目。從存取日誌省略日誌項目時，存取日誌中的項目數量與顯示於 AWS 帳單和使用量報告中的用量會不相符。

CloudFront 支援兩種版本的標準記錄。標準記錄 (舊版) 僅支援將您的存取日誌傳送至 Amazon S3。標準記錄 (v2) 支援其他交付目的地。您可以為分佈設定兩種或其中一種記錄選項。如需詳細資訊，請參閱下列主題：

主題

- [設定標準記錄 \(v2\)](#)
- [設定標準記錄 \(舊式\)](#)
- [標準記錄參考](#)

Tip

CloudFront 也提供即時存取日誌，提供您即時對分佈提出請求的相關資訊（日誌會在收到請求後的幾秒鐘內交付）。您可以使用即時存取日誌，根據內容交付效能來監控、分析和採取行動。如需詳細資訊，請參閱[使用即時存取日誌](#)。

設定標準記錄 (v2)

您可以在建立或更新分佈時啟用存取日誌（標準日誌）。標準記錄 (v2) 包含下列功能：

- 將存取日誌傳送至 Amazon CloudWatch Logs、Amazon Data Firehose 與 Amazon Simple Storage Service (Amazon S3)。
- 選取您想要的日誌欄位。您也可以選取[即時存取日誌欄位的子集](#)。
- 選取其他[輸出日誌檔案格式](#)。

如果您使用的是 Amazon S3，您有下列選用功能：

- 將日誌傳送至選擇加入 AWS 區域。
- 使用分割組織您的日誌。
- 啟用與 Hive 相容的檔案名稱。

如需詳細資訊，請參閱[傳送日誌至 Amazon S3](#)。

如要開始使用標準記錄，請完成以下步驟：

1. 為 AWS 服務 將接收日誌的指定 設定所需的許可。
2. 從 CloudFront 主控台或 CloudWatch API 設定標準記錄。
3. 檢視您的存取日誌。

 Note

- 如果您啟用標準記錄 (v2)，這不會影響或變更標準記錄 (舊版)。除了使用標準記錄 (v2) 之外，您還可以繼續使用分佈的標準記錄 (舊版)。如需詳細資訊，請參閱[設定標準記錄 \(舊式\)](#)。
- 如果您已啟用標準記錄 (舊版)，而且想要啟用 Amazon S3 的標準記錄 (v2)，建議您指定不同的 Amazon S3 儲存貯體，或使用相同儲存貯體中的個別路徑 (例如，使用日誌字首或分割)。這可協助您追蹤哪些日誌檔案與哪些分佈相關聯，並防止日誌檔案互相覆寫。

許可

CloudFront 使用 CloudWatch 提供的日誌來傳遞存取日誌。若要這樣做，您需要指定 的許可，AWS 服務 才能啟用記錄交付。

若要查看每個記錄目的地的必要許可，請在 Amazon CloudWatch Logs 使用者指南中選擇下列主題。

- [CloudWatch Logs](#)
- [Firehose](#)
- [Amazon S3](#)

在設定記錄目的地的許可後，您便可以為分佈啟用標準記錄。

 Note

CloudFront 支援將存取日誌傳送到不同的 AWS 帳戶 (跨帳戶)。若要啟用跨帳戶交付，帳戶 (您的帳戶和接收帳戶) 必須具有必要的許可。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 [啟用跨帳戶交付的標準記錄](#) 一節或[跨帳戶交付範例](#)。

啟用標準記錄

若要啟用標準記錄，您可以使用 CloudFront 主控台或 CloudWatch API。

內容

- [啟用標準記錄 \(CloudFront 主控台\)](#)
- [啟用標準記錄 \(CloudWatch API\)](#)

啟用標準記錄 (CloudFront 主控台)

啟用 CloudFront 分佈的標準記錄 (主控台)

1. 使用 CloudFront 主控台以[更新現有的分佈](#)。
 2. 選擇 Logging (日誌記錄) 索引標籤。
 3. 選擇新增，然後選擇要接收日誌的服務：
 - CloudWatch Logs
 - Firehose
 - Amazon S3
 4. 針對目的地，請選取您服務的資源。如果您尚未建立資源，您可以選擇建立或查看下列文件。
 - 針對 CloudWatch Logs，輸入[日誌群組名稱](#)。
 - 對於 Firehose，輸入 [Firehose 交付串流](#)。
 - 針對 Amazon S3，輸入[儲存貯體名稱](#)。
- 
- #### Tip
- 若要指定字首，請在儲存貯體名稱後面輸入字首，例如 `amzn-s3-demo-bucket.s3.amazonaws.com/MyLogPrefix`。如果您未指定字首，CloudFront 會自動為您新增字首。如需詳細資訊，請參閱[傳送日誌至 Amazon S3](#)。
5. 對於其他設定 – 選用，您可以指定下列選項：
 - a. 針對欄位選擇，選取您要交付至目的地的日誌欄位名稱。您可以選取[存取日誌欄位](#)和[即時存取日誌欄位](#)的子集。
 - b. (僅限 Amazon S3) 對於分割，指定分割日誌檔案資料的路徑。

- c. (僅限 Amazon S3) 對於 Hive 相容檔案格式，您可以選取核取方塊以使用 Hive 相容 S3 路徑。這有助於簡化將新資料載入 Hive 相容工具的過程。
- d. 針對輸出格式，請指定您偏好的格式。

 Note

如果您選擇 Parquet，此選項會產生將存取日誌轉換為 Apache Parquet 的 CloudWatch 費用。如需詳細資訊，請參閱 [CloudWatch 定價中的付費日誌](#) 一節。

- e. 對於欄位分隔符號，指定如何分隔日誌欄位。
6. 完成更新或建立分佈的步驟。
7. 若要新增另一個目的地，請重複步驟 3–6。
8. 在日誌頁面中，確認分佈旁的標準日誌狀態為已啟用。
9. (選用) 若要啟用 Cookie 記錄，請選擇管理、設定並開啟 Cookie 記錄，然後選擇儲存變更。

 Tip

Cookie 記錄是全域設定，適用於分佈的所有標準日誌。您無法覆寫個別交付目的地的此設定。

如需標準記錄交付和日誌欄位的詳細資訊，請參閱 [標準記錄參考](#)。

啟用標準記錄 (CloudWatch API)

您也可以使用 CloudWatch API 為您的分佈啟用標準記錄。

 備註

- 呼叫 CloudWatch API 以啟用標準記錄時，即使您想要啟用跨區域交付到另一個目的地，您也必須指定美國東部 (維吉尼亞北部) 區域 (us-east-1)。例如，如果您想要將存取日誌傳送至歐洲 (愛爾蘭) 區域的 S3 儲存貯體 (eu-west-1)，請使用 us-east-1 區域中的 CloudWatch API。
- 在標準日誌中包含 Cookie 還有其他選項。在 CloudFront API 中，這是 IncludeCookies 參數。如果您透過使用 CloudWatch API 設定存取記錄，並指定要包含 Cookie，則必須使用

CloudFront 主控台或 CloudFront API 來更新您的分佈以包含 Cookie。否則，CloudFront 無法傳送 Cookie 到您的日誌目的地。如需詳細資訊，請參閱[Cookie 記錄](#)。

啟用分佈的標準記錄 (CloudWatch API)

1. 建立分佈之後，請取得 Amazon Resource Name (ARN)。

您可以從 CloudFront 主控台的分佈頁面找到 ARN，也可以使用 [GetDistribution](#) API 操作。分佈 ARN 遵循以下格式：arn:aws:cloudfront::123456789012:distribution/d1111111abcdef8

2. 接著，使用 CloudWatch [PutDeliverySource](#) API 操作來建立分佈的交付來源。
 - a. 輸入交付來源的名稱。
 - b. 傳遞分佈的 resourceArn。
 - c. 對於 logType，指定 ACCESS_LOGS 為收集的日誌類型。
 - d. Example Example AWS CLI put-delivery-source 命令

以下是為分佈設定交付來源的範例。

```
aws logs put-delivery-source --name S3-delivery --resource-arn
arn:aws:cloudfront::123456789012:distribution/d1111111abcdef8 --log-type
ACCESS_LOGS
```

輸出

```
{
  "deliverySource": {
    "name": "S3-delivery",
    "arn": "arn:aws:logs:us-east-1:123456789012:delivery-source:S3-delivery",
    "resourceArns": [
      "arn:aws:cloudfront::123456789012:distribution/d1111111abcdef8"
    ],
    "service": "cloudfront",
    "logType": "ACCESS_LOGS"
  }
}
```

3. 使用 [PutDeliveryDestination](#) API 操作來設定存放日誌的位置。

- a. 針對 `destinationResourceArn`，指定目的地的 ARN。這可以是 CloudWatch Logs 日誌群組、Firehose 交付串流，或 Amazon S3 儲存貯體。
- b. 針對 `outputFormat`，指定日誌的輸出格式。
- c. Example Example AWS CLI `put-delivery-destination` 命令

以下是將交付目的地設定為 Amazon S3 儲存貯體的範例。

```
aws logs put-delivery-destination --name S3-destination --delivery-destination-configuration destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket
```

輸出

```
{
  "name": "S3-destination",
  "arn": "arn:aws:logs:us-east-1:123456789012:delivery-destination:S3-destination",
  "deliveryDestinationType": "S3",
  "deliveryDestinationConfiguration": {
    "destinationResourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

Note

如果您要跨帳戶傳遞日誌，則必須使用 [PutDeliveryDestinationPolicy](#) API 操作，以便將 AWS Identity and Access Management (IAM) 政策指派給目的地。IAM 政策允許從一個帳戶交付到另一個帳戶。

4. 使用 [CreateDelivery](#) API 操作，將交付來源連結至您在先前步驟中建立的目的地。此 API 操作會將交付來源與最終目的地建立關聯。
 - a. 針對 `deliverySourceName`，指定來源名稱。
 - b. 針對 `deliveryDestinationArn`，指定交付目的地的 ARN。
 - c. 對於 `fieldDelimiter`，指定要分隔每個日誌欄位的字串。
 - d. 針對 `recordFields`，指定您想要的日誌欄位。
 - e. 如果您使用的是 S3，請指定是否使用 `enableHiveCompatiblePath` 和 `suffixPath`。

Example AWS CLI create-delivery 命令範例

以下為建立交付的範例。

```
aws logs create-delivery --delivery-source-name cf-delivery --delivery-destination-arn arn:aws:logs:us-east-1:123456789012:delivery-destination:S3-destination
```

輸出

```
{
  "id": "abcNegnBoTR123",
  "arn": "arn:aws:logs:us-east-1:123456789012:delivery:abcNegnBoTR123",
  "deliverySourceName": "cf-delivery",
  "deliveryDestinationArn": "arn:aws:logs:us-east-1:123456789012:delivery-destination:S3-destination",
  "deliveryDestinationType": "S3",
  "recordFields": [
    "date",
    "time",
    "x-edge-location",
    "sc-bytes",
    "c-ip",
    "cs-method",
    "cs(Host)",
    "cs-uri-stem",
    "sc-status",
    "cs(Referer)",
    "cs(User-Agent)",
    "cs-uri-query",
    "cs(Cookie)",
    "x-edge-result-type",
    "x-edge-request-id",
    "x-host-header",
    "cs-protocol",
    "cs-bytes",
    "time-taken",
    "x-forwarded-for",
    "ssl-protocol",
    "ssl-cipher",
    "x-edge-response-result-type",
    "cs-protocol-version",
```

```

        "fle-status",
        "fle-encrypted-fields",
        "c-port",
        "time-to-first-byte",
        "x-edge-detailed-result-type",
        "sc-content-type",
        "sc-content-len",
        "sc-range-start",
        "sc-range-end",
        "c-country",
        "cache-behavior-path-pattern"
    ],
    "fieldDelimiter": ""
}

```

5. 在 CloudFront 主控台的日誌頁面上，確認分佈旁的標準日誌狀態已啟用。

如需標準記錄交付和日誌欄位的詳細資訊，請參閱 [標準記錄參考](#)。

Note

若要使用 啟用 CloudFront 的標準記錄 (v2) AWS CloudFormation，您可以使用下列 CloudWatch Logs 屬性：

- [Delivery](#)
- [DeliveryDestination](#)
- [DeliverySource](#)

ResourceArn 是 CloudFront 分佈，且 LogType 必須是 ACCESS_LOGS 作為支援的日誌類型。

啟用跨帳戶交付的標準記錄

如果您為 啟用標準記錄，AWS 帳戶 並且想要將存取日誌交付給另一個帳戶，請確定您已正確設定來源帳戶和目的地帳戶。具有 CloudFront 分佈的來源帳戶會將其存取日誌傳送至目的地帳戶。

在此範例程序中，來源帳戶 (**111111111111**) 會將其存取日誌傳送至目的地帳戶 (**222222222222**) 中的 Amazon S3 儲存貯體。若要將存取日誌傳送至目的地帳戶中的 Amazon S3 儲存貯體，請使用 AWS CLI。

設定目的地帳戶

對於目的地帳戶，請完成下列程序。

設定目的地帳戶

1. 若要建立日誌交付目的地，您可以輸入下列 AWS CLI 命令。此範例使用 *MyLogPrefix* 字串為您的存取日誌建立字首。

```
aws logs put-delivery-destination --name cloudfront-delivery-destination --  
delivery-destination-configuration "destinationResourceArn=arn:aws:s3:::amzn-s3-  
demo-bucket-cloudfront-logs/MyLogPrefix"
```

輸出

```
{  
  "deliveryDestination": {  
    "name": "cloudfront-delivery-destination",  
    "arn": "arn:aws:logs:us-east-1:222222222222:delivery-  
destination:cloudfront-delivery-destination",  
    "deliveryDestinationType": "S3",  
    "deliveryDestinationConfiguration": {"destinationResourceArn":  
"arn:aws:s3:::amzn-s3-demo-bucket-cloudfront-logs/MyLogPrefix"}  
  }  
}
```

Note

如果您指定的 S3 儲存貯體沒有字首，則 CloudFront 會自動將 AWSLogs/*<account-ID>*/CloudFront 附加為字首，顯示在 S3 交付目的地的 `suffixPath` 中。如需詳細資訊，請參閱 [S3DeliveryConfiguration](#)。

2. 新增日誌交付目的地的資源政策，以允許來源帳戶建立日誌交付。

在下列政策中，以來源帳戶 ID 取代 *111111111111*，並從步驟 1 的輸出指定交付目的地 ARN。

JSON

```
{  
  "Version": "2012-10-17",
```



```

    "Statement": [
      {
        "Sid": "AllowCreateDelivery",
        "Effect": "Allow",
        "Principal": {"AWS": "111111111111"},
        "Action": ["logs:CreateDelivery"],
        "Resource": "arn:aws:logs:us-east-1:222222222222:delivery-destination:cloudfront-delivery-destination"
      }
    ]
  }
}

```

3. 儲存檔案，例如 `deliverypolicy.json`。
4. 若要將先前的政策連接至交付目的地，請輸入下列 AWS CLI 命令。

```
aws logs put-delivery-destination-policy --delivery-destination-name cloudfront-delivery-destination --delivery-destination-policy file://deliverypolicy.json
```

5. 將下列陳述式新增至目的地 Amazon S3 儲存貯體政策，取代資源 ARN 和來源帳戶 ID。此政策允許 `delivery.logs.amazonaws.com` 服務主體執行 `s3:PutObject` 動作。

```

{
  "Sid": "AWSLogsDeliveryWrite",
  "Effect": "Allow",
  "Principal": {"Service": "delivery.logs.amazonaws.com"},
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-cloudfront-logs/*",
  "Condition": {
    "StringEquals": {
      "s3:x-amz-acl": "bucket-owner-full-control",
      "aws:SourceAccount": "111111111111"
    },
    "ArnLike": {"aws:SourceArn": "arn:aws:logs:us-east-1:111111111111:delivery-source:*"}
  }
}

```

6. 如果您使用 AWS KMS 做為儲存貯體，請將下列陳述式新增至 KMS 金鑰政策，以將許可授予 `delivery.logs.amazonaws.com` 服務主體。

```

{
  "Sid": "Allow Logs Delivery to use the key",

```

```

    "Effect": "Allow",
    "Principal": {"Service": "delivery.logs.amazonaws.com"},
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {"aws:SourceAccount": "111111111111"},
        "ArnLike": {"aws:SourceArn": "arn:aws:logs:us-east-1:111111111111:delivery-source:*"}
    }
}

```

設定來源帳戶

設定目的地帳戶之後，請依照此程序建立交付來源，並啟用來源帳戶中分佈的記錄。

設定來源帳戶

1. 建立 CloudFront 標準記錄的交付來源，以便您可以將日誌檔案傳送至 CloudWatch Logs。

您可以輸入下列 AWS CLI 命令，取代名稱和分佈 ARN。

```

aws logs put-delivery-source --name s3-cf-delivery --resource-arn
arn:aws:cloudfront::111111111111:distribution/E1TR1RHV123ABC --log-type
ACCESS_LOGS

```

輸出

```

{
  "deliverySource": {
    "name": "s3-cf-delivery",
    "arn": "arn:aws:logs:us-east-1:111111111111:delivery-source:s3-cf-delivery",
    "resourceArns":
    ["arn:aws:cloudfront::111111111111:distribution/E1TR1RHV123ABC"],
    "service": "cloudfront",
    "logType": "ACCESS_LOGS"
  }
}

```

```
}  
}
```

2. 建立交付以對應來源帳戶的日誌交付來源和目的地帳戶的日誌交付目的地。

在下列 AWS CLI 命令中，從[步驟 1：設定目的地帳戶中的輸出指定交付目的地](#) ARN。

```
aws logs create-delivery --delivery-source-name s3-cf-delivery --  
delivery-destination-arn arn:aws:logs:us-east-1:222222222222:delivery-  
destination:cloudfront-delivery-destination
```

輸出

```
{  
  "delivery": {  
    "id": "OPmOpLahVzhx1234",  
    "arn": "arn:aws:logs:us-east-1:111111111111:delivery:OPmOpLahVzhx1234",  
    "deliverySourceName": "s3-cf-delivery",  
    "deliveryDestinationArn": "arn:aws:logs:us-east-1:222222222222:delivery-  
destination:cloudfront-delivery-destination",  
    "deliveryDestinationType": "S3",  
    "recordFields": [  
      "date",  
      "time",  
      "x-edge-location",  
      "sc-bytes",  
      "c-ip",  
      "cs-method",  
      "cs(Host)",  
      "cs-uri-stem",  
      "sc-status",  
      "cs(Referer)",  
      "cs(User-Agent)",  
      "cs-uri-query",  
      "cs(Cookie)",  
      "x-edge-result-type",  
      "x-edge-request-id",  
      "x-host-header",  
      "cs-protocol",  
      "cs-bytes",  
      "time-taken",  
      "x-forwarded-for",  
      "ssl-protocol",  
    ]  
  }  
}
```

```
        "ssl-cipher",
        "x-edge-response-result-type",
        "cs-protocol-version",
        "fle-status",
        "fle-encrypted-fields",
        "c-port",
        "time-to-first-byte",
        "x-edge-detailed-result-type",
        "sc-content-type",
        "sc-content-len",
        "sc-range-start",
        "sc-range-end",
        "c-country",
        "cache-behavior-path-pattern"
    ],
    "fieldDelimiter": "\t"
}
```

3. 驗證您的跨帳戶交付是否成功。

- a. 從##帳戶登入 CloudFront 主控台，然後選擇您的分佈。在記錄索引標籤的類型下，您會看到為 S3 跨帳戶日誌交付建立的項目。
- b. 從###帳戶登入 Amazon S3 主控台，然後選擇您的 Amazon S3 儲存貯體。您會在儲存貯體名稱中看到字首 *MyLogPrefix*，以及傳送到該資料夾的任何存取日誌。

輸出檔案格式

根據您選擇的交付目的地，您可以為日誌檔指定下列其中一種格式：

- JSON
- Plain
- w3c
- Raw
- Parquet (僅限 Amazon S3)

 Note

您只能在第一次建立交付目的地時設定輸出格式。這無法在稍後更新。若要變更輸出格式，請刪除交付項目，然後建立另一種格式。

如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考指南》中的 [PutDeliveryDestination](#)。

編輯標準記錄設定

您可以使用 [CloudFront 主控台](#) 或 CloudWatch API 啟用或停用記錄和更新其他日誌設定。記錄設定的變更在 12 小時內生效。

如需詳細資訊，請參閱下列主題：

- 若要使用 CloudFront 主控台更新分佈，請參閱 [更新分佈](#)。
- 若要使用 CloudFront API 更新分佈，請參閱《Amazon CloudFront API 參考》中的 [UpdateDistribution](#)。
- 如需 CloudWatch Logs API 操作的詳細資訊，請參閱《[Amazon CloudWatch Logs API 參考](#)》。

存取日誌欄位

您可以選取標準記錄 (舊版) 支援的相同日誌欄位。如需詳細資訊，請參閱 [日誌檔案欄位](#)。

此外，您可以選取下列 [即時存取日誌欄位](#)。

1. **timestamp(ms)** – 以毫秒為單位的時間戳記。
2. **origin-fb1** – CloudFront 與您的原始伺服器之間第一個位元組延遲的秒數。
3. **origin-lb1** – CloudFront 與您的原始伺服器之間最後一個位元組延遲的秒數。
4. **asn** – 檢視器的自治系統編號 (ASN)。
5. **c-country** – 國家/地區代碼代表檢視器的地理位置，由檢視器的 IP 位址決定。如需國家/地區代碼的清單，請參閱 [ISO 3166-1 alpha-2](#)。
6. **cache-behavior-path-pattern** – 識別符合檢視器請求之快取行為的路徑模式。

將日誌傳送至 CloudWatch Logs

若要將日誌傳送至 CloudWatch Logs，請建立或使用現有的 CloudWatch Logs 日誌群組。如需有關設定 CloudWatch Logs 日誌群組的詳細資訊，請參閱 [使用日誌群組和日誌串流](#)。

建立日誌群組之後，您必須擁有允許標準記錄的必要許可。如需有關必要許可的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的[傳送至 CloudWatch Logs 的日誌](#)。

備註

- 當您指定 CloudWatch Logs 日誌群組的名稱時，請僅使用 regex 模式 `[\w-]`。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [PutDeliveryDestination](#) API 操作。
- 確認您的日誌群組資源政策未超過大小限制。請參閱 CloudWatch Logs 主題中的 [日誌群組資源政策大小限制考量](#) 一節。

傳送至 CloudWatch Logs 的存取日誌範例

```
{
  "date": "2024-11-14",
  "time": "21:34:06",
  "x-edge-location": "S0F50-P2",
  "asn": "16509",
  "timestamp(ms)": "1731620046814",
  "origin-fbl": "0.251",
  "origin-lbl": "0.251",
  "x-host-header": "d111111abcdef8.cloudfront.net",
  "cs(Cookie)": "examplecookie=value"
}
```

傳送日誌至 Firehose

若要將日誌傳送至 Firehose，請建立或使用現有的 Firehose 交付串流。然後，指定 Firehose 交付串流做為日誌交付目的地。您必須在美國東部 (維吉尼亞北部) `us-east-1` 區域中指定 Firehose 交付串流。

如需建立交付串流的相關資訊，請參閱[建立 Amazon Data Firehose 交付串流](#)。

建立交付串流後，您必須擁有允許標準記錄的必要許可。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的[傳送至 Firehose 的日誌](#)。

Note

當您指定 Firehose 串流的名稱時，請僅使用 regex 模式 `[\w-]`。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [PutDeliveryDestination](#) API 操作。

傳送至 Firehose 的存取日誌範例

```
{"date":"2024-11-15","time":"19:45:51","x-edge-location":"S0F50-P2","asn":"16509","timestamp(ms)":"1731699951183","origin-fbl":"0.254","origin-lbl":"0.254","x-host-header":"d1111111abcdef8.cloudfront.net","cs(Cookie)":"examplecookie=value"}
{"date":"2024-11-15","time":"19:45:52","x-edge-location":"S0F50-P2","asn":"16509","timestamp(ms)":"1731699952950","origin-fbl":"0.125","origin-lbl":"0.125","x-host-header":"d1111111abcdef8.cloudfront.net","cs(Cookie)":"examplecookie=value"}
```

傳送日誌至 Amazon S3

若要將存取日誌傳送至 Amazon S3，請建立或使用現有的 S3 儲存貯體。當您在 CloudFront 中啟用記錄時，請指定儲存貯體名稱。如需建立儲存貯體的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [建立儲存貯體](#)。

建立儲存貯體之後，您必須擁有允許標準記錄的必要許可。如需詳細資訊，請參閱「Amazon CloudWatch Logs 使用者指南」中的 [傳送至 Amazon S3 的日誌](#)。

- 啟用記錄後，AWS 會自動為您新增必要的儲存貯體政策。
- 您也可以[在選擇加入 AWS 區域](#)中使用 S3 儲存貯體。

Note

如果您已啟用標準記錄 (舊版)，而且想要啟用 Amazon S3 的標準記錄 (v2)，建議您指定不同的 Amazon S3 儲存貯體，或使用相同儲存貯體中的個別路徑 (例如，使用日誌字首或分割)。這可協助您追蹤哪些日誌檔案與哪些分佈相關聯，並防止日誌檔案互相覆寫。

主題

- [指定 S3 儲存貯體](#)

- [分割](#)
- [與 Hive 相容的檔案名稱格式](#)
- [存取日誌的範例路徑](#)
- [傳送至 Amazon S3 的存取日誌範例](#)

指定 S3 儲存貯體

當您指定 S3 儲存貯體做為交付目的地時，請注意下列事項。

S3 儲存貯體名稱只能使用 regex 模式 `[\w-]`。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [PutDeliveryDestination](#) API 操作。

如果您為 S3 儲存貯體指定字首，您的日誌會顯示在該路徑下。如果您不指定字首，CloudFront 會自動為您附加 `AWSLogs/{account-id}/CloudFront` 字首。

如需詳細資訊，請參閱[存取日誌的範例路徑](#)。

分割

當 CloudFront 將存取日誌傳送到 S3 儲存貯體時，您可以使用分割來組織存取日誌。這有助於您根據需要的路徑來組織和找出存取日誌。

您可以使用下列變數來建立資料夾路徑。

- `{DistributionId}` 或 `{distributionid}`
- `{yyyy}`
- `{MM}`
- `{dd}`
- `{HH}`
- `{accountid}`

您可以使用任意數量的變數，並指定路徑中的資料夾名稱。接著，CloudFront 會使用此路徑，在 S3 儲存貯體中為您建立資料夾結構。

範例

- `my_distribution_log_data/{DistributionId}/logs`

- `/cloudfront/{DistributionId}/my_distribution_log_data/{yyyy}/{MM}/{dd}/{HH}/logs`

Note

您可以使用任一變數做為尾碼路徑中的分佈 ID。不過，如果您要將存取日誌傳送到 AWS Glue，則必須使用 `{distributionid}` 變數，因為 AWS Glue 預期分割區名稱為小寫。在 CloudFront 中更新現有的日誌組態，以 `{DistributionId}` 取代 `{distributionid}`。

與 Hive 相容的檔案名稱格式

您可以使用此選項，讓包含交付存取日誌的 S3 物件使用允許與 Apache Hive 整合的字首結構。如需詳細資訊，請參閱 [CreateDelivery](#) API 操作。

Example 範例

```
/cloudfront/DistributionId={DistributionId}/my_distribution_log_data/year={yyyy}/month={MM}/day={dd}/hour={HH}/logs
```

如需分割和 Hive 相容選項的詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [S3DeliveryConfiguration](#) 元素。

存取日誌的範例路徑

當您指定 S3 儲存貯體做為目的地時，您可以使用下列選項來建立存取日誌的路徑：

- 包含或不含字首的 Amazon S3 儲存貯體
- 透過使用 CloudFront 提供的變數或輸入您自己變數的分割
- 啟用 Hive 相容選項

下表顯示您的存取日誌在儲存貯體中，依您選擇的選項所顯示的方式。

具有字首的 Amazon S3 儲存貯體

Amazon S3 儲存貯體名稱。	您在尾碼路徑中指定的分割區	已更新尾碼路徑	已啟用 Hive 相容？	存取日誌會傳送至
amzn-s3-demo-bucket/MyLogPrefix	無	無	否	amzn-s3-demo-bucket/MyLogPrefix/
amzn-s3-demo-bucket/MyLogPrefix	myFolderA/	myFolderA/	否	amzn-s3-demo-bucket/MyLogPrefix/myFolderA/
amzn-s3-demo-bucket/MyLogPrefix	myFolderA/{yyyy}	myFolderA/{yyyy}	是	amzn-s3-demo-bucket/MyLogPrefix/myFolderA/year=2025

沒有字首的 Amazon S3 儲存貯體

Amazon S3 儲存貯體名稱。	您在尾碼路徑中指定的分割區	已更新尾碼路徑	已啟用 Hive 相容？	存取日誌會傳送至
amzn-s3-demo-bucket	無	AWSLogs/{account-id}/CloudFront/	否	amzn-s3-demo-bucket/AWSLogs / <i><your-account-ID></i> / CloudFront/
amzn-s3-demo-bucket	myFolderA/	AWSLogs/{account-id}	否	amzn-s3-demo-bucket

Amazon S3 儲存貯體名稱。	您在尾碼路徑中指定的分割區	已更新尾碼路徑	已啟用 Hive 相容？	存取日誌會傳送至
		d}/CloudFront/myFolderA/		t/AWSLogs / <i><your-account-ID></i> / CloudFront/myFolderA/
amzn-s3-demo-bucket	myFolderA/	AWSLogs/{account-id}/CloudFront/myFolderA/	是	amzn-s3-demo-bucket/AWSLogs /aws-account-id= <i><your-account-ID></i> / CloudFront/myFolderA/
amzn-s3-demo-bucket	myFolderA/{yyyy}	AWSLogs/{account-id}/CloudFront/myFolderA/{yyyy}	是	amzn-s3-demo-bucket/AWSLogs /aws-account-id= <i><your-account-ID></i> / CloudFront/myFolderA/year=2025

AWS 帳戶 ID 做為分割區

Amazon S3 儲存貯體名稱。	您在尾碼路徑中指定的分割區	已更新尾碼路徑	已啟用 Hive 相容？	存取日誌會傳送至
amzn-s3-demo-bucket	無	AWSLogs/{account-id}	是	amzn-s3-demo-bucket

Amazon S3 儲存貯體名稱。	您在尾碼路徑中指定的分割區	已更新尾碼路徑	已啟用 Hive 相容？	存取日誌會傳送至
		d}/CloudFront/		t/AWSLogs /aws-account-id=<your-account-ID> / CloudFront/
amzn-s3-demo-bucket	myFolderA/{accountid}	AWSLogs/{account-id}/CloudFront/myFolderA/{accountid}	是	amzn-s3-demo-bucket/AWSLogs/aws-account-id=<your-account-ID>/CloudFront/myFolderA/accountid= <your-account-ID>

備註

- {account-id} 變數會保留給 CloudFront。如果您指定沒有字首的 Amazon S3 儲存貯體，CloudFront 會自動將此變數新增至尾碼路徑。如果您的日誌與 Hive 相容，此變數會顯示為 aws-account-id。
- 您可以使用 {accountid} 變數，讓 CloudFront 將您的帳戶 ID 新增至尾碼路徑。如果您的日誌與 Hive 相容，此變數會顯示為 accountid。
- 如需尾碼路徑的詳細資訊，請參閱 [S3DeliveryConfiguration](#)。

傳送至 Amazon S3 的存取日誌範例

```
#Fields: date time x-edge-location asn timestamp(ms) x-host-header cs(Cookie)
```

```
2024-11-14    22:30:25    S0F50-P2    16509    1731623425421
d1111111abcdef8.cloudfront.net    examplecookie=value2
```

停用標準記錄

如果您不再需要分佈的標準記錄，您可以停用該記錄。

停用標準記錄

1. 登入 CloudFront 主控台。
2. 選擇分佈，然後選擇分佈 ID。
3. 選擇記錄，然後在存取日誌目的地下，選取目的地。
4. 選擇管理，然後選擇刪除。
5. 如果您有多個標準記錄，請重複上一個步驟。

Note

當您從 CloudFront 主控台刪除標準記錄時，此動作只會刪除交付和交付目的地。它不會從您的刪除交付來源 AWS 帳戶。若要刪除交付來源，請在 `aws logs delete-delivery-source --name DeliverySourceName` 命令中指定交付來源名稱。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [DeleteDeliverySource](#)。

疑難排解

使用下列資訊來修正使用 CloudFront 標準記錄 (v2) 時的常見問題。

交付來源已存在

當您為分佈啟用標準記錄時，您可以建立交付來源。然後，您可以使用該交付來源來建立您想要的目的地類型交付：CloudWatch Logs、Firehose、Amazon S3。目前，每個分佈只能有一個交付來源。如果您嘗試為相同的分佈建立另一個交付來源，則會出現下列錯誤訊息。

```
This ResourceId has already been used in another Delivery Source in this account
```

若要建立另一個交付來源，請先刪除現有的交付來源。如需詳細資訊，請參閱《Amazon CloudWatch Logs API 參考》中的 [DeleteDeliverySource](#)。

我變更了尾碼路徑，而 Amazon S3 儲存貯體無法接收我的日誌

如果您啟用標準記錄 (v2) 並指定無字首的儲存貯體 ARN，CloudFront 會將下列預設值附加到尾碼路徑：AWSLogs/{account-id}/CloudFront。如果您使用 CloudFront 主控台或 [UpdateDeliveryConfiguration](#) API 操作來指定不同的尾碼路徑，則必須更新 Amazon S3 儲存貯體政策以使用相同的路徑。

Example 範例：更新尾碼路徑

1. 您的預設尾碼路徑為 AWSLogs/{account-id}/CloudFront，而您將其取代為 myFolderA。
2. 由於您的新尾碼路徑與 Amazon S3 儲存貯體政策中指定的路徑不同，因此不會交付您的存取日誌。
3. 您可以執行下列任一步驟：
 - 將 Amazon S3 儲存貯體許可從 amzn-s3-demo-bucket/AWSLogs/<your-account-ID>/CloudFront/* 更新為 amzn-s3-demo-bucket/myFolderA/*。
 - 更新您的記錄組態以再次使用預設尾碼：AWSLogs/{account-id}/CloudFront

如需詳細資訊，請參閱[許可](#)。

刪除日誌檔案

CloudFront 不會自動從您的目的地刪除日誌檔案。如需有關刪除日誌檔案的詳細資訊，請參閱下列主題：

Amazon S3

- [Amazon Simple Storage Service 主控台使用者指南](#)中的刪除物件

CloudWatch Logs

- Amazon CloudWatch Logs 使用者指南中的[使用日誌群組和日誌串流](#)
- Amazon CloudWatch Logs API 參考中的 [DeleteLogGroup](#)

Firehose

- Amazon Data Firehose API 參考中的 [DeleteDeliveryStream](#)

定價

CloudFront 不收取啟用標準日誌的費用。不過，您可能需要支付交付、擷取、儲存或存取的費用，視您選取的日誌交付目的地而定。如需詳細資訊，請參閱 [Amazon CloudWatch Logs 定價](#)。在付費方案下，選擇日誌索引標籤，然後在已付費日誌下，查看每個交付目的地的資訊。

如需每個定價的詳細資訊 AWS 服務，請參閱下列主題：

- [Amazon CloudWatch Logs 定價](#)
- [Amazon Data Firehose 定價](#)
- [Amazon S3 定價](#)

Note

雖然儲存和存取日誌檔案會產生 Amazon S3 費用，但將日誌傳送到 Amazon S3 不會產生額外費用。如果您啟用 Parquet 選項將存取日誌轉換為 Apache Parquet，此選項會產生 CloudWatch 費用。如需詳細資訊，請參閱 [CloudWatch 定價中的付費日誌](#) 一節。

設定標準記錄 (舊式)

備註

- 本主題適用於先前版本的標準記錄。如需最新版本，請參閱 [設定標準記錄 \(v2\)](#)。
- 如果您已啟用標準記錄 (舊版)，而且想要啟用 Amazon S3 的標準記錄 (v2)，建議您指定不同的 Amazon S3 儲存貯體，或使用相同儲存貯體中的個別路徑 (例如，使用日誌字首或分割)。這可協助您追蹤哪些日誌檔案與哪些分佈相關聯，並防止日誌檔案互相覆寫。

若要開始使用標準記錄 (舊版)，請完成下列步驟：

1. 選擇將接收日誌的 Amazon S3 儲存貯體，並新增必要許可。
2. 從 CloudFront 主控台或 CloudFront API 設定標準記錄 (舊版)。您只能選擇 Amazon S3 儲存貯體來接收您的日誌。
3. 檢視您的存取日誌。

為標準日誌選擇 Amazon S3 儲存貯體

當您啟用分佈記錄時，請指定您要 CloudFront 存放日誌檔案存的 Amazon S3 儲存貯體。如果您是使用 Amazon S3 做為原始伺服器，我們建議您為日誌檔案使用個別的儲存貯體。

指定您要 CloudFront 儲存存取記錄的 Amazon S3 儲存貯體，例如 `amzn-s3-demo-bucket.s3.amazonaws.com`。

您可以在同一個儲存貯體存放多個分佈的日誌檔案。啟用記錄時，您可以指定檔案名稱的選用字首，以便您可以繼續追蹤哪個日誌檔案與那個分佈關聯。

關於選擇 S3 儲存貯體

- 您的儲存貯體必須啟用存取控制清單 (ACL)。如果您從 CloudFront 主控台選擇未啟用 ACL 的儲存貯體，則會出現錯誤訊息。請參閱 [許可](#)。
- 不要選擇 [S3 物件擁有權](#) 設定為強制執行儲存貯體擁有者的 Amazon S3 儲存貯體。該設定會停用儲存貯體及其中物件的 ACL，這可防止 CloudFront 將記錄檔傳送至儲存貯體。
- 請勿在下列 AWS 區域 中選擇 Amazon S3 儲存貯體。CloudFront 不會將標準日誌交付到這些區域中的儲存貯體：
 - 非洲 (開普敦)
 - 亞太區域 (香港)
 - 亞太區域 (海德拉巴)
 - 亞太區域 (雅加達)
 - 亞太區域 (墨爾本)
 - 加拿大西部 (卡加利)
 - 歐洲 (米蘭)
 - 歐洲 (西班牙)
 - 歐洲 (蘇黎世)
 - 以色列 (特拉維夫)
 - 中東 (巴林)
 - 中東 (阿拉伯聯合大公國)

許可

Important

從 2023 年 4 月開始，您必須對用於 CloudFront 標準日誌的新 S3 儲存貯體啟用 S3 ACL。您可以在[建立儲存貯體](#)時啟用 ACL，或為[現有儲存貯體](#)啟用 ACL。

有關變更的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[新 S3 儲存貯體的預設設定常見問答集](#)和 AWS 新聞部落格中的[預告：Amazon S3 安全性變更將於 2023 年 4 月推出](#)。

您的 AWS 帳戶 必須有下列您指定給日誌檔案的儲存貯體之許可：

- 儲存貯體的 ACL 必須授予您 FULL_CONTROL。如果您是儲存貯體擁有者，您的帳戶會有預設的該許可。如果您不是，儲存貯體擁有者則必須更新儲存貯體的 ACL。
- s3:GetBucketAcl
- s3:PutBucketAcl

儲存貯體的 ACL

當您建立或更新分佈及啟用記錄時，CloudFront 使用這些許可更新儲存貯體的 ACL，提供 awslogsdelivery 帳戶 FULL_CONTROL 許可。awslogsdelivery 帳戶將日誌檔案寫入儲存貯體。如果您的帳戶沒有更新 ACL 所需的許可，則建立或更新分佈會失敗。

在某些情況下，如果您以程式設計的方式提交請求建立儲存貯體，但有指定名稱的儲存貯體已存在，則 S3 在儲存貯體上重設許可為預設值。如果您配置 CloudFront 在 S3 儲存貯體儲存存取日誌，且停止在該儲存貯體中取得日誌，請檢查儲存貯體上的許可，確保 CloudFront 擁有必要的許可。

還原儲存貯體的 ACL

如果您移除 awslogsdelivery 帳戶的許可，CloudFront 便無法將日誌儲存到 S3 儲存貯體。若要啟用 CloudFront，使其開始為分佈儲存日誌，請執行以下其中一個步驟，即可還原 ACL 許可：

- 在 CloudFront 中停用分佈的記錄功能，然後再次啟用。如需更多詳細資訊，請參閱[標準記錄](#)。
- 在 Amazon S3 主控台中瀏覽至 S3 儲存貯體，接著新增許可，即可為 awslogsdelivery 手動新增 ACL 許可。若要為 awslogsdelivery 新增 ACL，您必須提供該帳戶的正式 ID，如下所示：

```
c4c1ede66af53448b93c283ce9448c4ba468c9432aa01d700d3878632f77d2d0
```

如需將 ACL 新增至 S3 儲存貯體的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[設定 ACL](#)。

每個日誌檔案的 ACL

除了儲存貯體上的 ACL，每個日誌檔案上都有一個 ACL。每個日誌檔案上儲存貯體擁有者擁有 FULL_CONTROL 許可，分佈擁有者 (如果與儲存貯體擁有者不同) 沒有許可，且 awslogsdelivery 帳戶擁有讀取和寫入許可。

停用記錄

如果您停用記錄，CloudFront 不會刪除儲存貯體或日誌檔案的 ACL。您可以視需要刪除 ACL。

適用於 SSE-KMS 儲存貯體的必要金鑰政策

如果標準日誌的 S3 儲存貯體透過客戶自管金鑰來使用 AWS KMS keys (SSE-KMS) 實現伺服器端加密，則必須將下列陳述式新增至客戶自管金鑰的金鑰政策。這可讓 CloudFront 將日誌寫入儲存貯體。您無法在 AWS 受管金鑰中使用 SSE-KMS，因為 CloudFront 不能夠將日誌檔案寫入儲存貯體。

```
{
  "Sid": "Allow CloudFront to use the key to deliver logs",
  "Effect": "Allow",
  "Principal": {
    "Service": "delivery.logs.amazonaws.com"
  },
  "Action": "kms:GenerateDataKey*",
  "Resource": "*"
}
```

如果標準日誌的 S3 儲存貯體使用 SSE-KMS 搭配 [S3 儲存貯體金鑰](#)，您還需要將 kms:Decrypt 權限新增至政策陳述式。在此情況下，完整的政策陳述式如下所示。

```
{
  "Sid": "Allow CloudFront to use the key to deliver logs",
  "Effect": "Allow",
  "Principal": {
    "Service": "delivery.logs.amazonaws.com"
  },
  "Action": [
    "kms:GenerateDataKey*",
```

```
    "kms:Decrypt"  
  ],  
  "Resource": "*"   
}
```

Note

當您為 S3 儲存貯體啟用 SSE-KMS 時，請為客戶自管金鑰指定完整的 ARN。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[搭配 AWS KMS keys \(SSE-KMS\) 使用伺服器端加密](#)。

啟用標準記錄 (舊版)

若要啟用標準日誌，請使用 CloudFront 主控台或 CloudFront API。

內容

- [啟用標準記錄 \(舊版\) \(CloudFront 主控台\)](#)
- [啟用標準記錄 \(舊版\) \(CloudFront API\)](#)

啟用標準記錄 (舊版) (CloudFront 主控台)

啟用 CloudFront 分佈的標準日誌 (主控台)

1. 使用 CloudFront 主控台建立[新的分佈](#)或[更新現有的分佈](#)。
2. 在標準日誌區段的日誌交付中，選擇開啟。
3. (選用) 對於 Cookie 記錄，如果您想要在日誌中包含 Cookie，請選擇開啟。如需更多詳細資訊，請參閱 [Cookie 記錄](#)。

Tip

Cookie 記錄是全域設定，適用於分佈的所有標準日誌。您無法覆寫個別交付目的地的此設定。

4. 針對交付至區段，指定 Amazon S3 (舊版)。
5. 指定您的 Amazon S3 儲存貯體。如果您還沒有儲存貯體，您可以選擇建立或查看文件來[建立儲存貯體](#)。

6. (選擇性) 針對日誌字首功能，請指定您希望 CloudFront 在此分佈的存取日誌檔案名稱前加入的字首 (如果有)，例如 `exampleprefix/`。結尾的斜線 (/) 是可選的，但建議簡化瀏覽日誌檔案。如需更多詳細資訊，請參閱 [日誌字首](#)。
7. 完成更新或建立分佈的步驟。
8. 在日誌頁面中，確認分佈旁的標準日誌狀態為已啟用。

如需標準記錄交付和日誌欄位的詳細資訊，請參閱 [標準記錄參考](#)。

啟用標準記錄 (舊版) (CloudFront API)

您也可以使用 CloudFront API 為您的分佈啟用標準日誌。

啟用分佈的標準日誌 (CloudFront API)

- 使用 [CreateDistribution](#) 或 [UpdateDistribution](#) API 操作，並設定 [LoggingConfig](#) 物件。

編輯標準記錄設定

您可以啟用或停用記錄、變更存放日誌的 Amazon S3 儲存貯體，以及透過使用 [CloudFront 主控台](#) 或 CloudFront API 來變更日誌檔案的前綴。記錄設定的變更在 12 小時內生效。

如需詳細資訊，請參閱下列主題：

- 若要使用 CloudFront 主控台更新分佈，請參閱[更新分佈](#)。
- 若要使用 CloudFront API 更新分佈，請參閱 Amazon CloudFront API 參考中的 [UpdateDistribution](#)。

傳送日誌至 Amazon S3

當您將日誌傳送至 Amazon S3 時，您的日誌會以下列格式顯示。

檔案名稱格式

CloudFront 儲存在 Amazon S3 儲存貯體的每個日誌檔案名稱使用以下檔案名稱格式：

<optional prefix>/<distribution ID>.YYYY-MM-DD-HH.unique-ID.gz

使用國際標準時間 (UTC) 的日期與時間。

例如，如果您使用 `example-prefix` 做為前綴，且您的分佈 ID 為 `EMLARXS9EXAMPLE`，則檔案名稱看起來像這樣：

```
example-prefix/EMLARXS9EXAMPLE.2019-11-14-20.RT4KCN4SGK9.gz
```

啟用分佈的記錄時，您可以指定檔案名稱的選用字首，以便您可以繼續追蹤哪個日誌檔案與那個分佈關聯。如果您包含日誌檔案前綴的值，且您的前綴不是以正斜線 (/) 結尾，則 CloudFront 會自動附加一個正斜線。如果您的前綴確實以正斜線結束，CloudFront 就不會加以新增。

在檔案名稱尾端的 `.gz` 表示 CloudFront 使用 gzip 已壓縮了日誌檔案。

標準日誌檔案格式

日誌檔案中的每個項目會提供有關單一檢視器請求的詳細資訊。日誌檔案具有下列特性：

- 使用 [W3C 延伸日誌檔案格式](#)。
- 包含索引標籤分隔值。
- 包含不一定按時間順序排列的記錄。
- 包含兩個標頭行：一個有檔案格式版本，而另一個則列出包含在每個記錄的 W3C 欄位。
- 在欄位值中包含空格和特定其他字元的 URL 編碼對等字元。

以下字元會使用 URL 編碼的對等字元：

- ASCII 字元碼 0 到 32 (含)
- ASCII 字元碼 127 及更高
- 下表中的所有字元

URL 編碼標準定義於 [RFC 1738](#)。

URL 編碼的值	字元
%3C	<
%3E	>
%22	"
%23	#
%25	%

URL 編碼的值	字元
%7B	{
%7D	}
%7C	
%5C	\
%5E	^
%7E	~
%5B	[
%5D	]
%60	`
%27	'
%20	空格

刪除日誌檔案

CloudFront 不會自動從您的 Amazon S3 儲存貯體中刪除日誌檔案。如需有關從 Amazon S3 儲存貯體刪除日誌檔案的資訊，請參閱《Amazon Simple Storage Service 主控台使用者指南》中的[刪除物件](#)。

定價

標準記錄是 CloudFront 的一項選用功能。CloudFront 不收取啟用標準日誌的費用。不過，在 Amazon S3 上的存放和存取檔案會依一般 Amazon S3 費用計費。您可隨時將其刪除。

如需 Amazon S3 定價的詳細資訊，請參閱 [Amazon S3 定價](#)。

如需 CloudFront 定價的詳細資訊，請參閱 [CloudFront 定價](#)。

標準記錄參考

下列各節適用於標準記錄 (v2) 和標準記錄 (舊版)。

主題

- [日誌檔案交付的時間點](#)
- [請求 URL 或標頭超過大小上限時，記錄請求的方式](#)
- [日誌檔案欄位](#)
- [分析日誌](#)

日誌檔案交付的時間點

CloudFront 交付分佈的日誌在一小時內就高達數次。一般而言，日誌檔案包含有關請求的資訊，CloudFront 在特定期間內收到此請求。CloudFront 通常會將該時段的日誌檔案，在出現於日誌中事件的一小時內交付到您的目的地。不過，請注意，一個時段的部分或全部日誌檔案項目有時會延遲高達 24 小時。當日誌項目延遲時，CloudFront 將它們儲存在日誌檔案中，其中檔案名稱包含發生請求的日期和時段，而不是交付檔案時的日期和時間。

建立日誌檔案時，CloudFront 為分佈從所有節點整合資訊，此節點在日誌檔案涵蓋的時段間收到物件的請求。

根據 CloudFront 為與分佈關聯的物件收到多少請求，CloudFront 可以儲存一個時段內的多個檔案。

您啟用記錄後，CloudFront 會開始可靠地交付存取日誌大約四個小時。該時段之前，您可能會收到幾個存取日誌。

Note

如果在時段內沒有使用者請求您的物件，您便不會收到該時段的任何日誌檔案。

請求 URL 或標頭超過大小上限時，記錄請求的方式

若所有請求標頭 (包含 cookie) 的總大小超過 20 KB，或是 URL 超過 8192 個位元組，CloudFront 便無法完整剖析請求且無法記錄請求。因為沒有記錄請求，您將無法在日誌檔案中看到傳回的 HTTP 錯誤狀態代碼。

如果要求主體超過大小上限，則會記錄請求，包含 HTTP 錯誤狀態代碼。

日誌檔案欄位

分佈的日誌檔案包含 33 個欄位。下列清單依序包含每個欄位名稱，以及該欄位中資訊的描述。

1. **date**

事件發生的日期格式 YYYY-MM-DD。例如 2019-06-30。使用國際標準時間 (UTC) 的日期與時間。對於 WebSocket 連線，這是連線關閉的日期。

2. **time**

CloudFront 伺服器完成回應至請求的時間 (以 UTC 表示)，例如 01:42:39。對於 WebSocket 連線，這是連線關閉的時間。

3. **x-edge-location**

提供請求的節點。一個三字母代碼和任意指派的數字會辨識每個節點，例如 DFW3。三字母代碼通常會對應節點所在地理位置附近機場的國際航空運輸協會 (IATA) 機場代碼。(未來這些縮寫可能會改變。)

4. **sc-bytes**

回應請求時伺服器提供給檢視器的總位元數，包括標頭。對於 WebSocket 與 gRPC 連線，這是透過連線從伺服器傳送到用戶端的位元組總數。

5. **c-ip**

檢視器的 IP 位址，該檢視器已執行請求，例如 192.0.2.183 或 2001:0db8:85a3::8a2e:0370:7334。如果檢視器使用 HTTP 代理或負載平衡器傳送請求，此欄位值則為代理或負載平衡器的 IP 位址。另請參閱 x-forwarded-for 欄位。

6. **cs-method**

從檢視器接收到的 HTTP 請求方法。

7. **cs(Host)**

CloudFront 分佈的網域名稱(例如，d111111abcdef8.cloudfront.net)。

8. **cs-uri-stem**

識別路徑和物件的請求 URL 部分 (例如 /images/cat.jpg)。URL 中包含問號 (?)，而且查詢字串不包含在日誌中。

9. **sc-status**

包含以下其中一個值：

- 伺服器回應的 HTTP 狀態碼 (例如 200)。
- 000，這表示檢視器在伺服器回應請求之前已關閉連線。如果檢視器在伺服器開始傳送回應之後關閉連線，此欄位會包含伺服器開始傳送回應的 HTTP 狀態碼。

10.cs(Referer)

請求中的 Referer 標頭值。發出請求的網域名稱。常見推薦網站，包含搜尋引擎、其他直接連結到您物件的網站，以及您自己的網站。

11.cs(User-Agent)

請求中的 User-Agent 標頭值。識別請求來源的 User-Agent 標頭，例如提交請求的裝置與瀏覽器類型，或如果請求來自搜尋引擎，則識別是哪一個搜尋引擎。

12.cs-uri-query

請求 URI 的查詢字串部分 (如果有)。

當 URL 不包含查詢字串時，此欄位的值是一個連字號 (-)。如需詳細資訊，請參閱[根據查詢字串參數快取內容](#)。

13.cs(Cookie)

請求中的 Cookie 標頭，包括名稱值對和關聯的屬性。

如果您啟用 Cookie 記錄，無論您選擇將哪一個 Cookie 轉送到原始伺服器，CloudFront 都會在所有請求中記錄 Cookie。當請求不包含 Cookie 標頭時，此欄位的值是一個連字號 (-)。如需 Cookie 的詳細資訊，請參閱[根據 Cookie 快取內容](#)。

14x-edge-result-type

在最後一個位元組離開伺服器之後，伺服器如何將回應分類。在某些情況下，結果類型會在伺服器準備好傳送回應的時間，以及完成傳送回應的時間中發生改變。另請參閱 x-edge-response-result-type 欄位。

例如，在 HTTP 串流中，假設伺服器在快取中找到串流的區段。在這種情況下，這個欄位的值通常是 Hit。不過，如果在伺服器已交付整個區段之前，檢視器關閉檢視器，則最終結果類型 (此欄位的值) 為 Error。

WebSocket 與 gRPC 連線會有的此欄位值為 Miss，因為內容無法快取，且會透過代理直接傳至原始伺服器。

可能的值包括：

- Hit – 該伺服器從快取提供物件給檢視器。
- RefreshHit – 該伺服器在邊緣快取中找到物件，但物件已過期，因此伺服器會聯絡原始伺服器，以確認快取具有該物件的最新版本。

- **Miss** – 快取中的物件無法滿足請求，因此會將請求轉送至原始伺服器，並將結果傳回至檢視器。
- **LimitExceeded** – 請求遭拒，因為超過 CloudFront 配額 (先前稱為限制)。
- **CapacityExceeded** – 伺服器會傳回 HTTP 503 狀態碼，因為在請求提供物件時沒有足夠的容量。
- **Error** – 通常，這表示請求導致客戶端錯誤 (`sc-status` 欄位的值在 4xx 範圍內) 或伺服器錯誤 (`sc-status` 欄位的值在 5xx 範圍內)。如果 `sc-status` 欄位的值是 200，或者如果該欄位的值是 **Error** 並且 `x-edge-response-result-type` 欄位的值不是 **Error**，這代表 HTTP 請求成功，但用戶端在接收所有位元組之前中斷連線。
- **Redirect** – 伺服器會根據分佈設定，將檢視器從 HTTP 重新引導至 HTTPS。
- **LambdaExecutionError** – 與分佈相關聯的 `Lambda@Edge` 函數由於格式不正確的關聯、函數逾時、AWS 相依性問題或其他一般可用性問題而未完成。

15x-edge-request-id

可獨特識別請求的不透明字串。CloudFront 也會在 `x-amz-cf-id` 回應標頭中傳送此字串。

16x-host-header

包含在該請求 `Host` 標頭的檢視器值。如果您在物件 URL 中使用 CloudFront 網域名稱 (例如 `d111111abcdef8.cloudfront.net`)，則此欄位會包含該網域名稱。如果您使用物件 URL 中的備用網域名稱 (CNAME)，例如 `www.example.com`，則此欄位包含此備用網域名稱。

如果您使用備用網域名稱，請參閱網域名稱的欄位 7 中的 `cs(Host)`，此網域名稱與分佈相關聯。

17cs-protocol

檢視器請求的通訊協定 (`http`、`https`、`grpc`、`ws` 或 `wss`)。

18cs-bytes

檢視器包含在請求中的資料位元組總數，包括標頭。對於 `WebSocket` 與 `gRPC` 連線，這是連線時從用戶端傳送到伺服器的位元組總數。

19time-taken

從伺服器收到檢視者請求，到伺服器將回應的最後一個位元組寫入輸出佇列的秒數 (以千分之一秒為單位，例如 0.082)，這會在伺服器上測量。從檢視器來看，取得完整回應的總時間會比該值來的長，因為網路延遲和 TCP 緩衝。

20x-forwarded-for

如果檢視器使用 HTTP 代理或負載平衡器傳送請求，`c-ip` 欄位值則為代理或負載平衡器的 IP 位址。在這種情況下，此欄位是產生請求的檢視器 IP 位址。此欄位可以包含多個逗號分隔的 IP 位址。每個 IP 位址可以是 IPv4 位址 (例如 192.0.2.183) 或 IPv6 位址 (例如 2001:0db8:85a3::8a2e:0370:7334)。

如果檢視器無法使用 HTTP 代理或負載平衡器，則此欄位值是一個連字號 (-)。

21 `ssl-protocol`

當請求使用 HTTPS 時，此欄位會包含檢視器和伺服器為傳輸請求和回應而交涉的 SSL/TLS 通訊協定。如需可能值的清單，請參閱 [檢視器和 CloudFront 之間支援的通訊協定和密碼](#) 中支援的 SSL/TLS 通訊協定。

當欄位 17 中的 `cs-protocol` 為 `http` 時，此欄位的值是一個連字號 (-)。

22 `ssl-cipher`

當請求使用 HTTPS 時，此欄位會包含檢視器和伺服器為加密請求和回應而交涉的 SSL/TLS 密碼。如需可能值的清單，請參閱 [檢視器和 CloudFront 之間支援的通訊協定和密碼](#) 中支援的 SSL/TLS 密碼。

當欄位 17 中的 `cs-protocol` 為 `http` 時，此欄位的值是一個連字號 (-)。

23 `x-edge-response-result-type`

在將回應傳回至檢視器之前，伺服器如何將回應分類。另請參閱 `x-edge-result-type` 欄位。可能的值包括：

- `Hit` – 該伺服器從快取提供物件給檢視器。
- `RefreshHit` – 該伺服器在邊緣快取中找到物件，但物件已過期，因此伺服器會聯絡原始伺服器，以確認快取具有該物件的最新版本。
- `Miss` – 快取中的物件無法滿足請求，因此伺服器會將請求轉送至原始伺服器，並將結果傳回至檢視器。
- `LimitExceeded` – 請求遭拒，因為超過 CloudFront 配額 (先前稱為限制)。
- `CapacityExceeded` – 該伺服器會傳回 503 錯誤，因為在請求提供物件時沒有足夠的容量。
- `Error` – 通常，這表示請求導致客戶端錯誤 (`sc-status` 欄位的值在 4xx 範圍內) 或伺服器錯誤 (`sc-status` 欄位的值在 5xx 範圍內)。

如果 `x-edge-response-result-type` 欄位的值為 `Error` 且此欄位的值不為 `Error`，則用戶端在完成下載之前中斷連線。

- **Redirect** – 伺服器會根據分佈設定，將檢視器從 HTTP 重新引導至 HTTPS。
- **LambdaExecutionError** – 與分佈相關聯的 Lambda@Edge 函數由於格式不正確的關聯、函數逾時、AWS 相依性問題或其他一般可用性問題而未完成。

24.cs-protocol-version

檢視器在請求中指定的 HTTP 版本。可能的值包括

HTTP/0.9、HTTP/1.0、HTTP/1.1、HTTP/2.0 及 HTTP/3.0。

25.file-status

為分佈配置[欄位層級加密](#)時，此欄位包含可指出要求主體是否已成功處理的代碼。當伺服器成功處理要求主體時，會加密指定欄位中的值，並將請求轉送至原始伺服器，此欄位的值為 `Processed`。在這種情況下，`x-edge-result-type` 的值仍然可以表示用戶端或伺服器端的錯誤。

此欄位可能的值包含：

- **ForwardedByContentType** – 伺服器無須剖析與加密便將請求轉送到原始伺服器，因為沒有配置任何內容類型。
- **ForwardedByQueryArgs** – 伺服器無需剖析或加密便將請求轉送到原始伺服器，因為請求包含查詢參數，此參數不在欄位層級加密的組態裡。
- **ForwardedDueToNoProfile** – 伺服器無需剖析或加密便將請求轉送到原始伺服器，因為在欄位層級加密的組態裡沒有指定設定檔。
- **MalformedContentTypeClientError** – 因為 `Content-Type` 標頭的值不是有效格式，因此伺服器拒絕請求並將 HTTP 400 狀態碼傳回至檢視器。
- **MalformedInputClientError** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為要求主體不是有效格式。
- **MalformedQueryArgsClientError** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為查詢參數空白或不是有效格式。
- **RejectedByContentType** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為在欄位層級加密的組態中沒有指定內容類型。
- **RejectedByQueryArgs** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為在欄位層級加密的組態中沒有指定查詢參數。
- **ServerError** – 原始伺服器傳回錯誤。

如果請求超過欄位層級的加密配額 (先前稱為限制)，此欄位會包含下列其中一個錯誤碼，而伺服器會將 HTTP 狀態碼傳回給檢視器 400。如需目前欄位層級加密的配額的詳細資訊，請參閱[欄位層級加密的配額](#)。

- `FieldLengthLimitClientError` – 配置為加密的欄位已超過允許的長度上限。
- `FieldNumberLimitClientError` – 將分佈配置為加密的請求所包含的欄位數超過了允許的欄位數。
- `RequestLengthLimitClientError` – 當配置了欄位層級加密時，要求主體的長度超過允許的長度上限。

如果不為分佈設定欄位層級加密，則此欄位的值是一個連字號 (-)。

26.fle-encrypted-fields

邊緣伺服器加密並轉送至原始伺服器的[欄位層級加密](#)欄位數目。CloudFront 伺服器會在加密資料時將已處理的請求串流至原始伺服器，因此即使 `fle-status` 的值為錯誤，此欄位也可以具有值。

如果不為分佈設定欄位層級加密，則此欄位的值是一個連字號 (-)。

27.c-port

來自檢視器之請求的連接埠號碼。

28.time-to-first-byte

接收請求與寫入回應的第一個位元組之間的秒數 (如伺服器上所測量)。

29.x-edge-detailed-result-type

此欄位包含與 `x-edge-result-type` 欄位相同的值，但下列情況除外：

- 該物件已從 [Origin Shield](#) 層提供給檢視器時，此欄位包含 `OriginShieldHit`。
- 當物件不在 CloudFront 快取中，且回應是由[原始伺服器請求 Lambda @Edge 函數](#)產生時，此欄位會包含 `MissGeneratedResponse`。
- 當 `x-edge-result-type` 欄位的值為 `Error` 時，此欄位會包含下列其中一個值，以及有關該錯誤的詳細資訊：
 - `AbortedOrigin` – 該伺服器發生原始伺服器問題。
 - `ClientCommError` – 檢視器的回應因伺服器與檢視器之間發生通訊問題而遭到中斷。
 - `ClientGeoBlocked` – 分佈設定為拒絕來自檢視者地理位置的請求。
 - `ClientHungUpRequest` – 檢視器在傳送請求的同時提早停止。
 - `Error` – 發生錯誤，其錯誤類型不符合任何其他類別。當此伺服器提供來自快取的錯誤回應時，可能會發生此錯誤類型。
 - `InvalidRequest` – 該伺服器收到來自檢視器的無效請求。
 - `InvalidRequestBlocked` – 對請求的資源的存取遭到封鎖。

- `InvalidRequestCertificate` – 分佈不符合建立 HTTPS 連線的 SSL/TLS 憑證。
- `InvalidRequestHeader` — 請求包含無效的標頭。
- `InvalidRequestMethod` – 分佈未設定為處理使用的 HTTP 請求方法。當分佈只支援可快取的請求時會發生此情況。
- `OriginCommError` – 在連接到原始伺服器或從原始伺服器讀取資料時，請求逾時。
- `OriginConnectError` – 該伺服器無法連線到原始伺服器。
- `OriginContentRangeLengthError` – 原始伺服器回應中的 `Content-Length` 標頭不符合 `Content-Range` 標頭中的長度。
- `OriginDnsError` – 該伺服器無法解析原始伺服器的網域名稱。
- `OriginError` – 原始伺服器傳回不正確的回應。
- `OriginHeaderTooBigError` – 原始伺服器傳回的標頭對邊緣伺服器太大，因而無法處理。
- `OriginInvalidResponseError` – 原始伺服器傳回無效的回應。
- `OriginReadError` – 該伺服器無法從原始伺服器讀取。
- `OriginWriteError` – 該伺服器無法寫入原始伺服器。
- `OriginZeroSizeObjectError` – 從原始伺服器傳送大小為零的物件，因而導致錯誤。
- `SlowReaderOriginError` – 檢視器讀取訊息過慢，因而導致原始伺服器錯誤。

30`sc-content-type`

回應的 HTTP `Content-Type` 標頭值。

31`sc-content-len`

回應的 HTTP `Content-Length` 標頭值。

32`sc-range-start`

當回應包含 HTTP `Content-Range` 標頭時，此欄位包含範圍起始值。

33`sc-range-end`

當回應包含 HTTP `Content-Range` 標頭時，此欄位包含範圍結束值。

34`distribution-tenant-id`

分佈租用戶的 ID。

35`connection-id`

存取日誌（標準日誌）
TLS 連線的唯一識別符。

您必須為分發啟用 mTLS，才能取得此欄位的資訊。如需詳細資訊，請參閱[相互 TLS \(mTLS\) 檢視器](#)。

以下是分佈的範例日誌檔案。

```
#Version: 1.0
#Fields: date time x-edge-location sc-bytes c-ip cs-method cs(Host) cs-uri-stem sc-
status cs(Referer) cs(User-Agent) cs-uri-query cs(Cookie) x-edge-result-type x-edge-
request-id x-host-header cs-protocol cs-bytes time-taken x-forwarded-for ssl-protocol
ssl-cipher x-edge-response-result-type cs-protocol-version fle-status fle-encrypted-
fields c-port time-to-first-byte x-edge-detailed-result-type sc-content-type sc-
content-len sc-range-start sc-range-end
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
SOX4xwn4XV6Q4rgb7XiVG0Hms_BG1TAC4KyHmureZmBNrjGdRLiNIQ== d111111abcdef8.cloudfront.net
https 23 0.001 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.001 Hit
text/html 78 - -
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
k6WGMNkEzR5BEM_SaF47gjtX9zBD02m3490Y2an0QPEaUum1Z0Lrow== d111111abcdef8.cloudfront.net
https 23 0.000 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.000 Hit
text/html 78 - -
2019-12-04 21:02:31 LAX1 392 192.0.2.100 GET d111111abcdef8.cloudfront.net /
index.html 200 - Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;
%20x64)%20AppleWebKit/537.36%20(KHTML,%20like
%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
f37nTMVvnKvV2ZSvEsivup_c2kZ7VXzYdjC-GUQZ5qNs-89BlWazbw== d111111abcdef8.cloudfront.net
https 23 0.001 - TLSv1.2 ECDHE-RSA-AES128-GCM-SHA256 Hit HTTP/2.0 - - 11040 0.001 Hit
text/html 78 - -
2019-12-13 22:36:27 SEA19-C1 900 192.0.2.200 GET d111111abcdef8.cloudfront.net /
favicon.ico 502 http://www.example.com/ Mozilla/5.0%20(Windows
%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Error
1pkpNfBQ39sYMnjjUQjmH2w1wdJnbHYTbag21o_30fcQgPzdL2RSSQ== www.example.com http 675
0.102 - - - Error HTTP/1.1 - - 25260 0.102 OriginDnsError text/html 507 - -
2019-12-13 22:36:26 SEA19-C1 900 192.0.2.200 GET d111111abcdef8.cloudfront.net / 502
- Mozilla/5.0%20(Windows%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
```



```
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Error
3AqrZGCnF_g0-5K0vfA7c9XLcf4YGvMFSeFdIetR1N_2y8jSis8Zxg== www.example.com http 735
0.107 - - - Error HTTP/1.1 - - 3802 0.107 OriginDnsError text/html 507 - -
2019-12-13 22:37:02 SEA19-C2 900 192.0.2.200 GET d111111abcdef8.cloudfront.net / 502
- curl/7.55.1 - - Error kBkDzGnceVtWHqSCqBUqtA_cEs2T3tFUBbnBNkB9El_uVRhHgcZfcw==
www.example.com http 387 0.103 - - - Error HTTP/1.1 - - 12644 0.103 OriginDnsError
text/html 507 - -
```

分析日誌

由於您每個小時可以收到多個存取日誌，我們建議您將指定期間內收到的所有日誌檔案整合成一個檔案。然後，您可以更準確且完整地分析該期間的資料。

使用 [Amazon Athena](#) 是分析存取日誌的其中一種方法。Athena 是一種互動式查詢服務，可協助您分析 AWS 服務的資料，包括 CloudFront。若要進一步了解，請參閱 Amazon Athena 使用者指南中的[查詢 Amazon CloudFront 日誌](#)。

此外，以下 AWS 部落格文章討論一些分析存取日誌的方法。

- [Amazon CloudFront 請求記錄](#) (適用於透過 HTTP 傳送的內容)
- [增強型 CloudFront Logs](#)，現在提供查詢字串

使用即時存取日誌

使用 CloudFront 即時存取日誌，您可以即時取得對分佈提出之請求的相關資訊（日誌會在收到請求後的幾秒內交付）。您可以使用即時存取日誌，根據內容交付效能來監控、分析和採取行動。

CloudFront 即時存取日誌是可設定的。您可以選擇：

- 即時日誌的取樣率，也就是您希望接收即時存取日誌記錄的請求百分比。
- 您想要在日誌中接收的特定欄位。
- 您想要接收即時日誌的特定快取行為 (路徑模式)。

CloudFront 即時存取日誌會交付至 Amazon Kinesis Data Streams 中您選擇的資料串流。您可以建立自己的 [Kinesis 資料串流取用者](#)，或使用 Amazon Data Firehose 將日誌資料傳送到 Amazon Simple Storage Service (Amazon S3)、Amazon Redshift、Amazon OpenSearch Service (OpenSearch Service) 或第三方日誌處理服務。

除了您使用 Kinesis Data Streams 所產生的費用之外，CloudFront 還會收取即時存取日誌的費用。如需定價的詳細資訊，請參閱 [Amazon CloudFront 定價](#) 和 [Amazon Kinesis Data Streams 定價](#)。

⚠ Important

我們建議您使用日誌，了解內容請求的性質，而不是像完全考量所有請求。CloudFront 會盡力提供即時存取日誌。在實際處理請求之後，才可能長時間交付特定請求的日誌項目，在極少數的情況下，有可能完全不會交付日誌項目。從即時存取日誌中省略日誌項目時，即時存取日誌中的項目數量將與 AWS 帳單和用量報告中顯示的用量不符。

主題

- [建立和使用即時存取日誌組態](#)
- [了解即時存取日誌組態](#)
- [建立 Kinesis Data Streams 取用者](#)
- [即時存取日誌疑難排解](#)

建立和使用即時存取日誌組態

若要即時取得對分佈提出之請求的相關資訊，您可以使用即時存取日誌組態。日誌會在收到請求後的幾秒內交付。您可以使用 AWS Command Line Interface (AWS CLI) 或 CloudFront API，在 CloudFront 主控台中建立即時存取日誌組態。

若要使用即時存取日誌組態，您可以將它連接到 CloudFront 分佈中的一或多個快取行為。

Console

建立即時存取日誌組態

1. 登入 AWS 管理主控台，並在位於的 CloudFront CloudFront 主控台中開啟日誌頁面<https://console.aws.amazon.com/cloudfront/v4/home?#/logs>。
2. 選擇即時組態索引標籤。
3. 選擇建立組態。
4. 請在名稱輸入組態名稱。
5. 請在取樣率輸入您希望接收日誌記錄的請求百分比。
6. 對於欄位，選擇要在即時存取日誌中接收的欄位。
 - 若要包含日誌的所有 [CMCD 欄位](#)，請選擇 CMCD 所有鍵。
7. 針對端點，選擇一或多個 Kinesis 資料串流來接收即時存取日誌。

Note

CloudFront 即時存取日誌會交付至您在 Kinesis Data Streams 中指定的資料串流。若要讀取和分析即時存取日誌，您可以建置自己的 Kinesis 資料串流取用者。您也可以使用 Firehose 將日誌資料傳送至 Amazon S3、Amazon Redshift、Amazon OpenSearch Service 或第三方日誌處理服務。

8. 對於 IAM 角色，請選擇建立新服務角色或選擇現有角色。您必須具有建立 IAM 角色的許可。
9. （選用）針對分佈，選擇要連接至即時存取日誌組態的 CloudFront 分佈和快取行為。
10. 選擇建立組態。

如果成功，主控台會顯示您剛建立之即時存取日誌組態的詳細資訊。

如需詳細資訊，請參閱[了解即時存取日誌組態](#)。

AWS CLI

若要使用 建立即時存取日誌組態 AWS CLI，請使用 `aws cloudfront create-realtime-log-config` 命令。您可以使用輸入檔案來提供命令的輸入參數，而不是將每個個別參數指定為命令列輸入。

建立即時存取日誌組態（具有輸入檔案的 CLI）

1. 使用下列命令建立一個名為 `rtl-config.yaml` 的檔案，其中包含 `create-realtime-log-config` 命令的所有輸入參數。

```
aws cloudfront create-realtime-log-config --generate-cli-skeleton yaml-input >
rtl-config.yaml
```

2. 開啟您剛才建立且命名為 `rtl-config.yaml` 的檔案。編輯 檔案以指定您想要的即時存取日誌組態設定，然後儲存檔案。注意下列事項：
 - 對於 `StreamType`，唯一有效的值為 `Kinesis`。

如需即時長組態設定的詳細資訊，請參閱[了解即時存取日誌組態](#)。

3. 使用以下命令，使用 `rtl-config.yaml` 檔案的輸入參數建立即時存取日誌組態。

```
aws cloudfront create-realtime-log-config --cli-input-yaml file://rtl-  
config.yaml
```

如果成功，命令的輸出會顯示您剛建立之即時存取日誌組態的詳細資訊。

將即時存取日誌組態連接至現有分佈（具有輸入檔案的 CLI）

1. 使用下列命令來儲存您想要更新之 CloudFront 分佈的分佈組態。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront get-distribution-config --id distribution_ID --output yaml >  
dist-config.yaml
```

2. 開啟您剛才建立且命名為 dist-config.yaml 的檔案。編輯 檔案，對您要更新的每個快取行為進行下列變更，以使用即時存取日誌組態。
 - 在快取行為中，新增名為 RealtimeLogConfigArn 的欄位。對於欄位的值，請使用您要連接到此快取行為的即時存取日誌組態的 ARN。
 - 將 ETag 欄位重新命名為 IfMatch，但不要變更欄位的值。

完成後儲存檔案。

3. 使用下列命令來更新分佈，以使用即時存取日誌組態。將 *distribution_ID* 取代為分佈的 ID。

```
aws cloudfront update-distribution --id distribution_ID --cli-input-yaml file://  
dist-config.yaml
```

如果成功，命令的輸出會顯示您剛才更新的分佈詳細資料。

API

若要使用 CloudFront API 建立即時存取日誌組態，請使用 [CreateRealtimeLogConfig](#) API 操作。如需您在此 API 呼叫中指定參數的詳細資訊，請參閱 [了解即時存取日誌組態](#) 和 AWS SDK 或其他 API 用戶端的 API 參考文件。

建立即時存取日誌組態之後，您可以使用下列其中一個 API 操作將其連接至快取行為：

- 若要將它附加到現有分佈中的快取行為，請使用 [UpdateDistribution](#)。
- 若要將它附加到新分佈中的快取行為，請使用 [CreateDistribution](#)。

對於這兩個 API 操作，請在快取行為的 `RealtimeLogConfigArn` 欄位中提供即時存取日誌組態的 ARN。如需您在這些 API 呼叫中指定之其他欄位的詳細資訊，請參閱 [所有分佈設定參考](#) 和 AWS SDK 或其他 API 用戶端的 API 參考文件。

了解即時存取日誌組態

若要使用 CloudFront 即時存取日誌，請先建立即時存取日誌組態。即時存取日誌組態包含您要接收哪些日誌欄位、日誌記錄取樣率，以及您要交付日誌之 Kinesis 資料串流的相關資訊。

具體而言，即時存取日誌組態包含下列設定：

內容

- [Name](#)
- [抽樣頻率](#)
- [欄位](#)
- [端點 \(Kinesis Data Streams\)](#)
- [IAM 角色](#)

Name

識別即時存取日誌組態的名稱。

抽樣頻率

取樣率是介於 1 到 100（包含）之間的整數，可決定傳送至 Kinesis Data Streams 的檢視器請求百分比，做為即時存取日誌記錄。若要在即時存取日誌中包含每個檢視器請求，請為取樣率指定 100。您可以選擇較低的取樣率來降低成本，同時在即時存取日誌中仍收到代表性的請求資料範例。

欄位

每個即時存取日誌記錄中包含的欄位清單。每個日誌記錄最多包含 40 個欄位，而且您可以選擇接收所有可用欄位，或只接收監視和分析效能所需的欄位。

下列清單包含每個欄位名稱以及該欄位中資訊的描述。這些欄位會依它們交付給 Kinesis Data Streams 日誌中顯示的順序列出。

欄位 46-63 是[常見的媒體用戶端資料 \(CMCD\)](#)，可由媒體播放器用戶端隨每個請求傳送至 CDN。您可以使用此資料來瞭解每個請求，例如媒體類型 (音訊、影片)、播放速率和串流長度。這些欄位只有在傳送至 CloudFront 時，才會出現在您的即時存取日誌中。

1. **timestamp**

邊緣伺服器完成回應請求的日期和時間。

2. **c-ip**

檢視器的 IP 位址，該檢視器已執行請求，例如 192.0.2.183 或 2001:0db8:85a3::8a2e:0370:7334。如果檢視器使用 HTTP 代理或負載平衡器傳送請求，此欄位值則為代理或負載平衡器的 IP 位址。另請參閱 x-forwarded-for 欄位。

3. **s-ip**

提供請求的 CloudFront 伺服器 IP 位址，例如 192.0.2.183 或 2001:0db8:85a3::8a2e:0370:7334。

4. **time-to-first-byte**

接收請求與寫入回應的第一個位元組之間的秒數 (如伺服器上所測量)。

5. **sc-status**

伺服器回應的 HTTP 狀態碼 (例如 200)。

6. **sc-bytes**

回應請求時伺服器提供給檢視器的總位元數，包括標頭。對於 WebSocket 與 gRPC 連線，這是透過連線從伺服器傳送到用戶端的位元組總數。

7. **cs-method**

從檢視器接收到的 HTTP 請求方法。

8. **cs-protocol**

檢視器請求的通訊協定 (http、https、grpcs、ws 或 wss)。

9. **cs-host**

包含在該請求 Host 標頭的檢視器值。如果您在物件 URL 中使用 CloudFront 網域名稱 (例如 d111111abcdef8.cloudfront.net)，則此欄位會包含該網域名稱。如果您使用物件 URL 中的備用網域名稱 (CNAME)，例如 www.example.com，則此欄位包含此備用網域名稱。

10. **cs-uri-stem**

整個請求 URL，包括查詢字串 (如果存在)，但不包括網域名稱。例如 `/images/cat.jpg?mobile=true`。

 Note

在[標準日誌](#)中，該 `cs-uri-stem` 值不包括查詢字串。

11cs-bytes

檢視器包含在請求中的資料位元組總數，包括標頭。對於 WebSocket 與 gRPC 連線，這是連線時從用戶端傳送到伺服器的位元組總數。

12x-edge-location

提供請求的節點。一個三字母代碼和任意指派的數字會辨識每個節點，例如 DFW3。三字母代碼通常會對應節點所在地理位置附近機場的國際航空運輸協會 (IATA) 機場代碼。(未來這些縮寫可能會改變。)

13x-edge-request-id

可獨特識別請求的不透明字串。CloudFront 也會在 `x-amz-cf-id` 回應標頭中傳送此字串。

14x-host-header

CloudFront 分發的網域名稱(例如，`d111111abcdef8.cloudfront.net`)。

15time-taken

從伺服器收到檢視者請求，到伺服器將回應的最後一個位元組寫入輸出佇列的秒數 (以千分之一秒為單位，例如 0.082)，這會在伺服器上測量。從檢視器來看，取得完整回應的總時間會比該值來的長，因為網路延遲和 TCP 緩衝。

16cs-protocol-version

檢視器在請求中指定的 HTTP 版本。可能的值包括 `HTTP/0.9`、`HTTP/1.0`、`HTTP/1.1`、`HTTP/2.0` 及 `HTTP/3.0`。

17c-ip-version

請求的 IP 版本 (IPv4 或 IPv6)。

18cs-user-agent

請求中的 User-Agent 標頭值。識別請求來源的 User-Agent 標頭，例如提交請求的裝置與瀏覽器類型，或如果請求來自搜尋引擎，則識別是哪一個搜尋引擎。

19.cs-referer

請求中的 Referer 標頭值。發出請求的網域名稱。常見推薦網站，包含搜尋引擎、其他直接連結到您物件的網站，以及您自己的網站。

20.cs-cookie

請求中的 Cookie 標頭，包括名稱值對和關聯的屬性。

Note

此欄位會截斷為 800 位元組。

21.cs-uri-query

請求 URI 的查詢字串部分 (如果有)。

22x-edge-response-result-type

在將回應傳回至檢視器之前，伺服器如何將回應分類。另請參閱 x-edge-result-type 欄位。可能的值包括：

- Hit – 該伺服器從快取提供物件給檢視器。
- RefreshHit – 該伺服器在邊緣快取中找到物件，但物件已過期，因此伺服器會聯絡原始伺服器，以確認快取具有該物件的最新版本。
- Miss – 快取中的物件無法滿足請求，因此伺服器會將請求轉送至原始伺服器，並將結果傳回至檢視器。
- LimitExceeded – 請求遭拒，因為超過 CloudFront 配額 (先前稱為限制)。
- CapacityExceeded – 該伺服器會傳回 503 錯誤，因為在請求提供物件時沒有足夠的容量。
- Error – 通常，這表示請求導致客戶端錯誤 (sc-status 欄位的值在 4xx 範圍內) 或伺服器錯誤 (sc-status 欄位的值在 5xx 範圍內)。

如果 x-edge-result-type 欄位的值為 Error 且此欄位的值不為 Error，則用戶端在完成下載之前中斷連線。

- Redirect – 伺服器會根據分佈設定，將檢視器從 HTTP 重新引導至 HTTPS。
- LambdaExecutionError – 由於關聯格式不正確、函數逾時、AWS 相依性問題或其他一般可用性問題，與分佈相關聯的 Lambda@Edge 函數未完成。

23x-forwarded-for

如果檢視器使用 HTTP 代理或負載平衡器傳送請求，`c-ip` 欄位值則為代理或負載平衡器的 IP 位址。在這種情況下，此欄位是產生請求的檢視器 IP 位址。此欄位可以包含多個逗號分隔的 IP 位址。每個 IP 位址可以是 IPv4 位址 (例如 192.0.2.183) 或 IPv6 位址 (例如 2001:0db8:85a3::8a2e:0370:7334)。

24ssl-protocol

當請求使用 HTTPS 時，此欄位會包含檢視器和伺服器為傳輸請求和回應而交涉的 SSL/TLS 通訊協定。如需可能值的清單，請參閱 [檢視器和 CloudFront 之間支援的通訊協定和密碼](#) 中支援的 SSL/TLS 通訊協定。

25ssl-cipher

當請求使用 HTTPS 時，此欄位會包含檢視器和伺服器為加密請求和回應而交涉的 SSL/TLS 密碼。如需可能值的清單，請參閱 [檢視器和 CloudFront 之間支援的通訊協定和密碼](#) 中支援的 SSL/TLS 密碼。

26x-edge-result-type

在最後一個位元組離開伺服器之後，伺服器如何將回應分類。在某些情況下，結果類型會在伺服器準備好傳送回應的時間，以及完成傳送回應的時間中發生改變。另請參閱 `x-edge-response-result-type` 欄位。

例如，在 HTTP 串流中，假設伺服器在快取中找到串流的區段。在這種情況下，這個欄位的值通常是 `Hit`。不過，如果在伺服器已交付整個區段之前，檢視器關閉檢視器，則最終結果類型 (此欄位的值) 為 `Error`。

WebSocket 與 gRPC 連線會有的此欄位值為 `Miss`，因為內容無法快取，且會透過代理直接傳至原始伺服器。

可能的值包括：

- `Hit` – 該伺服器從快取提供物件給檢視器。
- `RefreshHit` – 該伺服器在邊緣快取中找到物件，但物件已過期，因此伺服器會聯絡原始伺服器，以確認快取具有該物件的最新版本。
- `Miss` – 快取中的物件無法滿足請求，因此會將請求轉送至原始伺服器，並將結果傳回至檢視器。
- `LimitExceeded` – 請求遭拒，因為超過 CloudFront 配額 (先前稱為限制)。
- `CapacityExceeded` – 伺服器會傳回 HTTP 503 狀態碼，因為在請求提供物件時沒有足夠的容量。

- **Error** – 通常，這表示請求導致客戶端錯誤 (sc-status 欄位的值在 4xx 範圍內) 或伺服器錯誤 (sc-status 欄位的值在 5xx 範圍內)。如果 sc-status 欄位的值是 200，或者如果該欄位的值是 Error 並且 x-edge-response-result-type 欄位的值不是 Error，這代表 HTTP 請求成功，但用戶端在接收所有位元組之前中斷連線。
- **Redirect** – 伺服器會根據分佈設定，將檢視器從 HTTP 重新引導至 HTTPS。
- **LambdaExecutionError** – 由於關聯格式不正確、函數逾時、AWS 相依性問題或其他一般可用性問題，與分佈相關聯的 Lambda@Edge 函數未完成。

27.fle-encrypted-fields

邊緣伺服器加密並轉送至原始伺服器的[欄位層級加密](#)欄位數目。CloudFront 伺服器會在加密資料時將已處理的請求串流至原始伺服器，因此即使 fle-status 的值為錯誤，此欄位也可以具有值。

28.fle-status

為分發配置[欄位層級加密](#)時，此欄位包含可指出請求本文是否已成功處理的代碼。當伺服器成功處理要求主體時，會加密指定欄位中的值，並將請求轉送至原始伺服器，此欄位的值為 Processed。在這種情況下，x-edge-result-type 的值仍然可以表示用戶端或伺服器端的錯誤。

此欄位可能的值包含：

- **ForwardedByContentType** – 伺服器無須剖析與加密便將請求轉送到原始伺服器，因為沒有配置任何內容類型。
- **ForwardedByQueryArgs** – 伺服器無需剖析或加密便將請求轉送到原始伺服器，因為請求包含查詢參數，此參數不在欄位層級加密的組態裡。
- **ForwardedDueToNoProfile** – 伺服器無需剖析或加密便將請求轉送到原始伺服器，因為在欄位層級加密的組態裡沒有指定設定檔。
- **MalformedContentTypeClientError** – 因為 Content-Type 標頭的值不是有效格式，因此伺服器拒絕請求並將 HTTP 400 狀態碼傳回至檢視器。
- **MalformedInputClientError** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為要求主體不是有效格式。
- **MalformedQueryArgsClientError** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為查詢參數空白或不是有效格式。
- **RejectedByContentType** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為在欄位層級加密的組態中沒有指定內容類型。
- **RejectedByQueryArgs** – 伺服器拒絕請求且將 HTTP 400 狀態碼傳回給檢視器，因為在欄位層級加密的組態中沒有指定查詢參數。
- **ServerError** – 原始伺服器傳回錯誤。

如果請求超過欄位層級的加密配額 (先前稱為限制)，此欄位會包含下列其中一個錯誤碼，而伺服器會將 HTTP 狀態碼傳回給檢視器 400。如需目前欄位層級加密的配額的詳細資訊，請參閱[欄位層級加密的配額](#)。

- `FieldLengthLimitClientError` – 配置為加密的欄位已超過允許的長度上限。
- `FieldNumberLimitClientError` – 將分佈配置為加密的請求所包含的欄位數超過了允許的欄位數。
- `RequestLengthLimitClientError` – 當配置了欄位層級加密時，請求本文的長度超過允許的長度上限。

29 `sc-content-type`

回應的 HTTP Content-Type 標頭值。

30 `sc-content-len`

回應的 HTTP Content-Length 標頭值。

31 `sc-range-start`

當回應包含 HTTP Content-Range 標頭時，此欄位包含範圍起始值。

32 `sc-range-end`

當回應包含 HTTP Content-Range 標頭時，此欄位包含範圍結束值。

33 `c-port`

來自檢視器之請求的連接埠號碼。

34 `x-edge-detailed-result-type`

此欄位包含與 `x-edge-result-type` 欄位相同的值，但下列情況除外：

- 該物件已從 [Origin Shield](#) 層提供給檢視器時，此欄位包含 `OriginShieldHit`。
- 當物件不在 CloudFront 快取中，且回應是由[原始伺服器請求 Lambda @Edge 函數](#)產生時，此欄位會包含 `MissGeneratedResponse`。
- 當 `x-edge-result-type` 欄位的值為 `Error` 時，此欄位會包含下列其中一個值，以及有關該錯誤的詳細資訊：
 - `AbortedOrigin` – 該伺服器發生原始伺服器問題。
 - `ClientCommError` – 檢視器的回應因伺服器與檢視器之間發生通訊問題而遭到中斷。
 - `ClientGeoBlocked` – 分佈設定為拒絕來自檢視者地理位置的請求。

- `ClientHungUpRequest` – 檢視器在傳送請求的同時提早停止。
- `Error` – 發生錯誤，其錯誤類型不符合任何其他類別。當此伺服器提供來自快取的錯誤回應時，可能會發生此錯誤類型。
- `InvalidRequest` – 該伺服器收到來自檢視器的無效請求。
- `InvalidRequestBlocked` – 對請求的資源的存取遭到封鎖。
- `InvalidRequestCertificate` – 分佈不符合建立 HTTPS 連線的 SSL/TLS 憑證。
- `InvalidRequestHeader` – 請求包含無效的標頭。
- `InvalidRequestMethod` – 分佈未設定為處理使用的 HTTP 請求方法。當分佈只支援可快取的請求時會發生此情況。
- `OriginCommError` – 在連接到原始伺服器或從原始伺服器讀取資料時，請求逾時。
- `OriginConnectError` – 該伺服器無法連線到原始伺服器。
- `OriginContentRangeLengthError` – 原始伺服器回應中的 `Content-Length` 標頭不符合 `Content-Range` 標頭中的長度。
- `OriginDnsError` – 該伺服器無法解析原始伺服器的網域名稱。
- `OriginError` – 原始伺服器傳回不正確的回應。
- `OriginHeaderTooBigError` – 原始伺服器傳回的標頭對邊緣伺服器太大，因而無法處理。
- `OriginInvalidResponseError` – 原始伺服器傳回無效的回應。
- `OriginReadError` – 該伺服器無法從原始伺服器讀取。
- `OriginWriteError` – 該伺服器無法寫入原始伺服器。
- `OriginZeroSizeObjectError` – 從原始伺服器傳送大小為零的物件，因而導致錯誤。
- `SlowReaderOriginError` – 檢視器讀取訊息過慢，因而導致原始伺服器錯誤。

35.c-country

國家/地區代碼代表檢視者的地理位置，由檢視器的 IP 位址決定。如需國家/地區代碼的清單，請參閱 [ISO 3166-1 alpha-2](#)。

36.cs-accept-encoding

檢視器請求中的 `Accept-Encoding` 標頭值。

37.cs-accept

檢視器請求中的 `Accept` 標頭值。

38.cache-behavior-path-pattern

識別符合檢視器請求之快取行為的路徑模式。

39.cs-headers

檢視器請求中的 HTTP 標頭 (名稱和值)。

Note

此欄位會截斷為 800 位元組。

40.cs-header-names

檢視器請求中 HTTP 標頭 (非值) 的名稱。

Note

此欄位會截斷為 800 位元組。

41.cs-headers-count

檢視器請求中的 HTTP 標頭數目。

42.primary-distribution-id

啟用連續部署時，此 ID 會識別哪個分佈是目前分佈中的主要分佈。

43.primary-distribution-dns-name

啟用連續部署時，此值會顯示與目前 CloudFront 分佈相關的主要網域名稱 (例如 d111111abcdef8.cloudfront.net)。

44.origin-fbl

CloudFront 與您的原始伺服器之間第一個位元組延遲的秒數。

45.origin-lbl

CloudFront 與您的原始伺服器之間最後一個位元組延遲的秒數。

46.asn

檢視器的自治系統編號 (ASN)。

47.

 即時存取日誌中的 CMCD 欄位

如需這些欄位的詳細資訊，請參閱《[CTA 規格 Web 應用程式影片生態系統 - 常見媒體用戶端資料 CTA-5004](#)》文件。

48.cmcd-encoded-bitrate

所請求音訊或視訊物件的編碼位元率。

49.cmcd-buffer-length

所請求媒體物件的緩衝區長度。

50.cmcd-buffer-starvation

緩衝區是否在先前請求與物件請求之間的某個時間點出現缺乏情況。這可能會導致播放器處於重新緩衝狀態，可能會使視訊或音訊播放停滯。

51.cmcd-content-id

識別目前內容的唯一字串。

52.cmcd-object-duration

請求物件的播放持續時間 (以毫秒為單位)。

53.cmcd-deadline

此物件第一個範例必須可用的請求時間期限，以避免緩衝區執行不足狀態或其他播放問題。

54.cmcd-measured-throughput

用戶端與伺服器之間的輸送量，由用戶端測量。

55.cmcd-next-object-request

下一個請求物件的相對路徑。

56.cmcd-next-range-request

如果下一個請求是一部分物件請求，此字串表示要請求的位元組範圍。

57.cmcd-object-type

正在請求的目前物件媒體類型。

58.cmcd-playback-rate

1 表示即時、2 表示雙速、0 表示未播放。

59.**cmcd-requested-maximum-throughput**

用戶端認為足以交付資產的請求最大輸送量。

60.**cmcd-streaming-format**

定義目前請求的串流格式。

61.**cmcd-session-id**

識別目前播放工作階段的 GUID。

62.**cmcd-stream-type**

識別客群可用性的記號。v = 所有客群都可用。l = 客群會隨著時間而可用。

63.**cmcd-startup**

如果在緩衝區清空事件之後啟動、尋找或復原期間緊急需要物件，則會包含沒有值的鍵。

64.**cmcd-top-bitrate**

用戶端可播放的最高位元率轉譯。

65.**cmcd-version**

此版本規格用於解譯定義的鍵名稱和值。如果省略此鍵，用戶端和伺服器必須解譯版本 1 定義的值。

66.**r-host**

此欄位會針對原始請求傳送，顯示用於提供物件的原始伺服器網域。如果發生錯誤，您可以使用此欄位來尋找上次嘗試的原始伺服器，例如：*cd8jhdejh6a*.mediapackagev2.us-east-1.amazonaws.com。

67.**sr-reason**

此欄位提供選取原始伺服器的原因。對主要原始伺服器的請求成功時，此欄位為空白。

如果發生原始伺服器容錯移轉，欄位將包含導致容錯移轉的 HTTP 錯誤代碼，例如 Failover:403 或 Failover:502。在原始伺服器容錯移轉的情況下，如果重試的請求也失敗，且您尚未設定自訂錯誤頁面，則 r-status 會指出第二個原始伺服器的回應。不過如果您已設定自訂錯誤頁面與原始伺服器容錯移轉，則如果請求失敗且改為傳回自訂錯誤頁面，這將包含第二個原始伺服器的回應。

如果沒有發生原始伺服器容錯移轉，但發生媒體品質感知彈性 (MQAR) 原始伺服器選擇，則會將此記錄為 MediaQuality。如需詳細資訊，請參閱[媒體品質感知彈性](#)。

68x-edge-mqcs

此欄位指出 CloudFront 從 MediaPackage v2 的 CMSD 回應標頭中檢索媒體區段的媒體品質可信度分數 (MQCS) (範圍：0 – 100)。此欄位適用於符合具有啟用 MQAR 原始伺服器群組之快取行為的請求。除了原始請求之外，CloudFront 還會對也從其快取提供的媒體區段記錄此欄位。如需詳細資訊，請參閱[媒體品質感知彈性](#)。

69distribution-tenant-id

分佈租用戶的 ID。

70connection-id

TLS 連線的唯一識別符。

您必須為您的分佈啟用 mTLS，才能取得此欄位的資訊。如需詳細資訊，請參閱[相互 TLS \(mTLS\) 檢視器](#)。

端點 (Kinesis Data Streams)

端點包含您要傳送即時日誌的 Kinesis Data Streams 相關資訊。您提供資料串流的 Amazon Resource Name (ARN)。

如需建立 Kinesis Data Streams 的詳細資訊，請參閱《Amazon Kinesis Data Streams 開發人員指南》中的下列主題。

- [建立和管理串流](#)
- [使用 執行基本 Kinesis Data Streams 操作 AWS CLI](#)
- [建立串流](#) (使用 適用於 Java 的 AWS SDK)

當您建立資料串流時，您需要指定分片的數目。使用下列資訊來協助您估計需要的碎片數量。

估計 Kinesis 資料串流的碎片數量

1. 計算 (或估計) 您的 CloudFront 分發每秒收到的請求數。

您可以使用 [CloudFront 使用量報告](#) (在 CloudFront 主控台中) 和 [CloudFront 指標](#) (在 CloudFront 和 Amazon CloudWatch 主控台中) 來協助您計算每秒的請求數。

2. 決定單一即時存取日誌記錄的典型大小。

一般而言，單一日誌記錄大約是 500 個位元組。包含所有可用欄位的大型記錄一般約為 1 KB。

如果不確定日誌記錄的大小，您可以啟用取樣率較低 (例如 1%) 的即時日誌，然後使用 Kinesis Data Streams 中的監控資料，來計算平均記錄大小 (總傳入位元組數除以記錄總數)。

3. 在 [Amazon Kinesis Data Streams 定價頁面上](#)，AWS 定價計算工具選擇立即建立您的自訂預估。

- 在計算器中輸入每秒請求 (記錄) 的數量。
- 輸入單一日誌記錄的平均記錄大小。
- 選擇顯示計算。

定價計算器會顯示您需要的碎片數量並預估成本。

IAM 角色

AWS Identity and Access Management (IAM) 角色，提供 CloudFront 許可，以將即時存取日誌交付至 Kinesis 資料串流。

當您使用 CloudFront 主控台建立即時存取日誌組態時，您可以選擇建立新的服務角色，讓主控台為您建立 IAM 角色。

當您使用 AWS CloudFormation 或 CloudFront API (AWS CLI 或 SDK) 建立即時存取日誌組態時，您必須自行建立 IAM 角色並提供角色 ARN。若要自行建立 IAM 角色，請使用下列政策。

IAM 角色信任原則

若要使用以下 IAM 角色信任政策，請將 **111122223333** 取代為您的 AWS 帳戶號碼。此政策中的 Condition 元素有助於防止 [混淆代理人問題](#)，因為 CloudFront 只能代表 中的分佈擔任此角色 AWS 帳戶。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```



```
        "Service": "cloudfront.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        }
      }
    }
  ]
}
```

未加密資料串流的 IAM 角色許可原則

若要使用下列政策，請將 ***arn:aws:kinesis:us-east-2:123456789012:stream/StreamName*** 取代為 Kinesis 資料串流的 ARN。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesis:DescribeStreamSummary",
        "kinesis:DescribeStream",
        "kinesis:PutRecord",
        "kinesis:PutRecords"
      ],
      "Resource": [
        "arn:aws:kinesis:us-east-2:123456789012:stream/StreamName"
      ]
    }
  ]
}
```

已加密資料串流的 IAM 角色許可原則

若要使用以下政策，請將 *arn:aws:kinesis:us-east-2:123456789012:stream/StreamName* 取代為 Kinesis 資料串流的 ARN，將 *arn:aws:kms:us-east-2:123456789012:key/e58a3d0b-fe4f-4047-a495-ae03cc73d486* 取代為 AWS KMS key 的 ARN。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesis:DescribeStreamSummary",
        "kinesis:DescribeStream",
        "kinesis:PutRecord",
        "kinesis:PutRecords"
      ],
      "Resource": [
        "arn:aws:kinesis:us-east-2:123456789012:stream/StreamName"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey"
      ],
      "Resource": [
        "arn:aws:kms:us-east-2:123456789012:key/e58a3d0b-fe4f-4047-a495-ae03cc73d486"
      ]
    }
  ]
}
```

建立 Kinesis Data Streams 取用者

若要讀取和分析即時存取日誌，您可以建置或使用 Kinesis Data Streams 取用者。當您為 CloudFront 即時日誌建置取用者時，請務必了解每個即時存取日誌記錄中的欄位一律以相同順序交付，如 [欄位](#) 一節所列。請確定建立您的取用程式來適應此固定訂單。

例如，請考慮僅包含以下三個欄位的即時存取日誌組態：time-to-first-byte、sc-status 和 c-country。在這個案例中，最後一個欄位 c-country，永遠是每個日誌中的欄位編號 3。不過，如果您稍後將欄位新增至即時存取日誌組態，則記錄中每個欄位的位置可能會變更。

例如，如果您將欄位 sc-bytes 和 time-taken 新增至即時存取日誌組態，這些欄位會根據 [欄位](#) 區段中顯示的順序插入每個日誌記錄中。所有五個欄位的產生順序為 time-to-first-byte、sc-status、sc-bytes、time-taken 和 c-country。該 c-country 欄位原本是欄位編號 3，但現在是欄位編號 5。請確定您的取用者應用程式可以處理日誌記錄中變更位置的欄位，以防您將欄位新增至即時存取日誌組態。

即時存取日誌疑難排解

建立即時存取日誌組態後，您可能會發現沒有記錄（或並非所有記錄）交付至 Kinesis Data Streams。在此情況下，您應該先確認您的 CloudFront 分發是否正在收到檢視器請求。如果是，您可以檢查下列設定來繼續故障排除。

IAM 角色許可

為了將即時存取日誌記錄交付至 Kinesis 資料串流，CloudFront 會在即時存取日誌組態中使用 IAM 角色。請確定角色信任原則和角色許可原則符合 [IAM 角色](#) 中顯示的原則。

Kinesis Data Streams 調節

如果 CloudFront 以比串流更快的速度將即時存取日誌記錄寫入 Kinesis 資料串流，Kinesis Data Streams 可能會調節來自 CloudFront 的請求。在此情況下，您可以增加 Kinesis 資料串流中的碎片數量。每個碎片可支援最高每秒 1,000 筆記錄的寫入數目，最高每秒 1 MB 的資料寫入上限。

邊緣函數日誌

您可以使用 Amazon CloudWatch Logs 取得 [Lambda@Edge](#) 和 [CloudFront Functions](#) 兩種邊緣函數的日誌。您可以使用 CloudWatch 主控台或 CloudWatch Logs API 存取日誌檔案。

⚠ Important

我們建議您使用日誌，了解內容請求的性質，而不是像完全考量所有請求。CloudFront 依最佳作業基礎交付邊緣函數日誌。在實際處理請求之後，才可能長時間交付特定請求的日誌項目，在極少數的情況下，有可能完全不會交付日誌項目。從邊緣函數日誌省略日誌項目時，邊緣函數日誌中的項目數量與顯示於 AWS 帳單和使用量報告中的用量會不相符。

主題

- [Lambda @Edge 日誌](#)
- [CloudFront Functions 日誌](#)

Lambda @Edge 日誌

Lambda@Edge 會自動將函數日誌傳送至 CloudWatch Logs，並在函數被調用的 AWS 區域 中建立日誌串流。當您在 AWS Lambda 中建立或修改函數時，您可以使用預設 CloudWatch 日誌群組名稱或自訂名稱。

- 預設日誌群組名稱為 `/aws/lambda/<FunctionName>`，其中 `<FunctionName>` 是您在建立函數時指定的名稱。將日誌傳送至 CloudWatch 時，Lambda@Edge 會自動將 `us-east-1` 字首新增至函數名稱，因此日誌群組名稱為 `/aws/lambda/us-east-1.<FunctionName>`。此字首對應至建立函數 AWS 區域 的。此字首仍然是日誌群組名稱的一部分，即使在其他調用函數的區域中也是如此。
- 如果您指定自訂日誌群組名稱 (例如 `/MyLogGroup`)，則 Lambda@Edge 將不會新增區域字首。日誌群組名稱在調用函數的所有其他區域中保持不變。

📘 Note

如果您建立自訂日誌群組並指定與預設 `/aws/lambda/<FunctionName>` 相同的名稱，Lambda@Edge 會將 `us-east-1` 字首新增至函數名稱。

除了自訂日誌群組名稱之外，Lambda@Edge 函數還支援 JSON 和純文字日誌格式，以及日誌層級篩選。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的 [設定 Lambda 函數的進階記錄控制項](#)。

Note

Lambda@Edge 根據請求數量和日誌的大小節流日誌。

您必須檢閱正確 區域中的 CloudWatch 日誌檔案，才能檢視 Lambda@Edge 函數日誌檔案。若要檢視您 Lambda@Edge 函數正在執行的區域，請在 CloudFront 主控台檢視函數的指標圖形。指標將依每個 區域進行顯示。在同一頁面中，您可以選擇一個區域，然後檢視該區域的日誌檔以調查問題。

若要進一步了解如何使用 CloudWatch Logs 搭配 Lambda@Edge 函數，請參閱下列的主題：

- 如需在 CloudFront 主控台的監控區段檢視圖表的詳細資訊，請參閱[the section called “使用 Amazon CloudWatch 監控 CloudFront 指標”](#)。
- 如需將資料傳送至 CloudWatch Logs 所需權限的相關資訊，請參閱 [the section called “設定 IAM 許可權限和角色”](#)。
- 如需將記錄新增至 Lambda@Edge 函數的資訊，請參閱 AWS Lambda 開發人員指南中的 [Node.js 中的 AWS Lambda 函數日誌記錄](#) 或 [Python 中的 AWS Lambda 函數日誌記錄](#)。
- 如需 CloudWatch 記錄配額 (之前稱為限制) 的資訊，請參閱 Amazon CloudWatch Logs 使用者指南中的 [CloudWatch Logs 配額](#)。

CloudFront Functions 日誌

如果 CloudFront Functions 的程式碼包含 `console.log()` 陳述式，CloudFront Functions 會自動將這些日誌行傳送至 CloudWatch Logs。如果沒有任何 `console.log()` 陳述式，則不會傳送任何內容到 CloudWatch Logs。

CloudFront Functions 始終在美國東部 (維吉尼亞北部) 區域 (us-east-1) 建立日誌串流，而無論是哪個節點執行該函數。日誌串流名稱的格式為 YYYY/M/D/*UUID*。

日誌群組名稱使用下列格式：

- 對於快取行為層級的 CloudFront Functions，格式為 `/aws/cloudfront/function/<FunctionName>`
- 對於分佈層級的 CloudFront Functions (連線函數)，格式為 `/aws/cloudfront/connection-function/<FunctionName>`

<FunctionName> 是您在建立函數時提供給該函數的名稱。

Example 檢視器請求

以下顯示傳送至 CloudWatch Logs 的範例日誌訊息。每行都會以唯一識別 CloudFront 請求的 ID 開頭。訊息以包含 CloudFront 分佈 ID 的 START 行開頭，並以 END 行結尾。START 和 END 行之間是函數中 `console.log()` 陳述式產生的日誌行。

```
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== START DistributionID:
E3E5D42GADAXZZ
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== Example function log output
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== END
```

Example 連線請求

以下顯示傳送至 CloudWatch Logs 的範例日誌訊息。每行都會以唯一識別 CloudFront 請求的 ID 開頭。訊息以包含 CloudFront 分佈 ID 的 START 行開頭，並以 END 行結尾。START 和 END 行之間是函數中 `console.log()` 陳述式產生的日誌行。

```
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== START DistributionID:
E3E5D42GADA123
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== 1.2.3.4
U7b4hR_RaxMADupvKAvr8_m9gsGXvioUggLV50yq-vmAtH8HADpjhw== END
```

Note

CloudFront Functions 只會針對 LIVE 階段中的函數將日誌傳送至 CloudWatch，此類函數會執行以回應生產請求和回應。[測試函數](#)時，CloudFront 不會將任何日誌傳送至 CloudWatch。測試輸出包含錯誤、運算利用率和函數日誌 (`console.log()` 陳述式) 的相關資訊，但此資訊不會傳送至 CloudWatch。

CloudFront Functions 使用 AWS Identity and Access Management (IAM) [服務連結角色](#)，將日誌傳送至您帳戶中的 CloudWatch Logs。「服務連結角色」是一種直接連結至 AWS 服務的 IAM 角色。服務連結角色是由服務預先定義，並包含服務 AWS 服務 為您呼叫其他 所需的所有許可。CloudFront Functions 使用 `AWSServiceRoleForCloudFrontLogger` 服務連結角色。如需有關此角色的詳細資訊，請參閱 [the section called “Lambda@Edge 的服務連結角色”](#) (Lambda@Edge 會使用相同的服務連結角色)。

當函數失敗並出現驗證錯誤或執行錯誤時，資訊會記錄在[標準日誌](#)和[即時存取日誌](#)中。如需特定錯誤的資訊，請參閱 `x-edge-result-type`、`x-edge-response-result-type` 和 `x-edge-detailed-result-type` 欄位。

使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫

CloudFront 整合了 [AWS CloudTrail](#)：這是一種記錄使用者、角色或 AWS 服務 所採取動作的服務。CloudTrail 會將 CloudFront 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 CloudFront 主控台的呼叫，以及對 CloudFront API 操作發出的程式碼呼叫。您可以利用 CloudTrail 收集的資訊來判斷向 CloudFront 發出的請求，以及發出請求的 IP 位址、時間和其他詳細資訊。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是否使用根使用者還是使用者憑證提出。
- 請求是否代表 IAM Identity Center 使用者提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務 服務提出。

當您建立帳戶時，CloudTrail 會在 AWS 帳戶中啟用，而且您將自動獲得 CloudTrail 事件歷史記錄的存取權。CloudTrail 事件歷史記錄為 AWS 區域中過去 90 天記錄的管理事件，提供可檢視、可搜尋、可下載且不可變的記錄。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」中的[使用 CloudTrail 事件歷史記錄](#)。檢視事件歷史記錄不會產生 CloudTrail 費用。

若要不持續記錄在您 AWS 帳戶中過去 90 天的事件，請建立追蹤或 [CloudTrail Lake](#) 事件資料存放區。

CloudTrail 追蹤

追蹤能讓 CloudTrail 將日誌檔交付至 Amazon S3 儲存貯體。使用 AWS 管理主控台建立的所有追蹤為多區域。您可以使用 AWS CLI 建立單一或多區域追蹤。由於您要擷取帳戶所有 AWS 區域內的活動，因此建議建立多區域追蹤。如果您建立單一區域追蹤，您只能檢視追蹤 AWS 區域中記錄的事件。如需追蹤的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的[為您的 AWS 帳戶建立追蹤](#)和[為組織建立追蹤](#)。

您可以透過建立追蹤，免費將持續管理事件的一個複本從 CloudTrail 傳遞至您的 Amazon S3 儲存貯體，但這樣做會產生 Amazon S3 儲存費用。如需 CloudTrail 定價的詳細資訊，請參閱 [AWS CloudTrail 定價](#)。如需 Amazon S3 定價的相關資訊，請參閱 [Amazon S3 定價](#)。

CloudTrail Lake 事件資料存放區

CloudTrail Lake 讓您能夠對事件執行 SQL 型查詢。CloudTrail Lake 會將分列式 JSON 格式的現有事件轉換為 [Apache ORC](#) 格式。ORC 是一種單欄式儲存格式，針對快速擷取資料進行了最佳化。系統會將事件彙總到事件資料存放區中，事件資料存放區是事件的不可變集合，其依據為您透過套用 [進階事件選取器](#) 選取的條件。套用於事件資料存放區的選取器控制哪些事件持續存在並可供您查詢。如需 CloudTrail Lake 的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 [使用 AWS CloudTrail Lake](#)。

CloudTrail Lake 事件資料存放區和查詢會產生費用。建立事件資料存放區時，您可以選擇要用於事件資料存放區的 [定價選項](#)。此定價選項將決定擷取和儲存事件的成本，以及事件資料存放區的預設和最長保留期。如需 CloudTrail 定價的詳細資訊，請參閱 [AWS CloudTrail 定價](#)。

Note

CloudFront 服務是一項全球性的服務。CloudTrail 會記錄美國東部 (維吉尼亞北部) 區域中的 CloudFront 事件。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 [全域服務事件](#)。

如果您使用 AWS Security Token Service 來使用臨時安全憑證，則對 us-west-2 等區域端點的呼叫會在 CloudTrail 中記錄到其適當的區域。

如需 CloudFront 端點的詳細資訊，請參閱 AWS 一般參考 中的 [CloudFront 端點和配額](#)。

CloudTrail 中的 CloudFront 資料事件

[資料事件](#) 提供在資源上或資源中執行的資源操作相關資訊 (例如讀取或寫入 CloudFront 分佈)。這些也稱為資料平面操作。資料事件通常是大量資料的活動。根據預設，CloudTrail 不會記錄資料事件。CloudTrail 事件歷史記錄不會記錄資料事件。

資料事件需支付額外的費用。如需 CloudTrail 定價的詳細資訊，請參閱 [AWS CloudTrail 定價](#)。

您可以使用 CloudTrail 主控台、AWS CLI 或 CloudTrail API 操作來記錄 CloudFront 資源類型的資料事件。如需如何記錄資料事件的詳細資訊，請參閱 AWS CloudTrail 使用者指南中的 [使用 AWS 管理主控台 記錄資料事件](#) 和 [使用 AWS Command Line Interface 記錄資料事件](#)。

下表列出您可以記錄其資料事件的 CloudFront 資源類型。資料事件類型 (主控台) 資料行會顯示從 CloudTrail 主控台上的資料事件類型清單中選擇的值。resources.type 值資料行會顯示 resources.type 值，您需在使用 AWS CLI 或 CloudTrail API 設定進階事件選取器時指定該值。記錄到 CloudTrail 的資料 API 資料行會針對資源類型顯示記錄到 CloudTrail 的 API 呼叫。

資料事件類型 (主控台)	resources.type 值	記錄到 CloudTrail 的資料 API
CloudFront KeyValueStore	AWS::CloudFront::KeyValueStore	• DeleteKeys • DescribeKeyValueStore • GetKey • ListKeys • PutKeys • UpdateKeys

您可以設定進階事件選取器來篩選 `eventName`、`readOnly` 和 `resources.ARN` 欄位，以僅記錄對您重要的事件。如需這些欄位的詳細資訊，請參閱AWS CloudTrail API 參考中的[AdvancedFieldSelector](#)。

CloudTrail 中的 CloudFront 管理事件

[管理事件](#)提供在 AWS 帳戶 中的資源上所執行的管理作業相關資訊。這些也稱為控制平面操作。根據預設，CloudTrail 記錄管理事件。

Amazon CloudFront 日誌會將所有 CloudFront 控制平面操作記錄為管理事件。如需 CloudFront 記錄到 CloudTrail 的 Amazon CloudFront 控制平面操作清單，請參閱 [Amazon CloudFront API 參考](#)。

CloudFront 事件範例

一個事件代表任何來源提出的單一請求，並包含請求 API 操作的相關資訊、操作的日期和時間、請求參數等。CloudTrail 日誌檔案不是公有 API 呼叫的已排序堆疊追蹤，因此事件不會以任何特定順序顯示。

內容

- [範例：UpdateDistribution](#)
- [範例：UpdateKeys](#)

範例：UpdateDistribution

以下範例顯示的 CloudTrail 事件會示範 [UpdateDistribution](#) 操作。

對 CloudFront API 進行呼叫時，`eventSource` 是 `cloudfront.amazonaws.com`。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE:role-session-name",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/role-session-name",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admin",
        "accountId": "111122223333",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2024-02-02T19:23:50Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2024-02-02T19:26:01Z",
  "eventSource": "cloudfront.amazonaws.com",
  "eventName": "UpdateDistribution",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "52.94.133.137",
  "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/121.0.0.0 Safari/537.36",
  "requestParameters": {
    "distributionConfig": {
      "defaultRootObject": "",
      "aliases": {
        "quantity": 3,
        "items": [
          "alejandro_rosalez.awsps.myinstance.com",
          "cross-testing.alejandro_rosalez.awsps.myinstance.com",
          "*.alejandro_rosalez.awsps.myinstance.com"
        ]
      }
    },
    "cacheBehaviors": {
      "quantity": 0,

```

```

        "items": []
    },
    "httpVersion": "http2and3",
    "originGroups": {
        "quantity": 0,
        "items": []
    },
    "viewerCertificate": {
        "minimumProtocolVersion": "TLSv1.2_2021",
        "cloudFrontDefaultCertificate": false,
        "acmCertificateArn": "arn:aws:acm:us-east-1:111122223333:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "sSLSupportMethod": "sni-only"
    },
    "webACLId": "arn:aws:wafv2:us-east-1:111122223333:global/webacl/testing-acl/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "customErrorResponses": {
        "quantity": 0,
        "items": []
    },
    "logging": {
        "includeCookies": false,
        "prefix": "",
        "enabled": false,
        "bucket": ""
    },
    "priceClass": "PriceClass_All",
    "restrictions": {
        "geoRestriction": {
            "restrictionType": "none",
            "quantity": 0,
            "items": []
        }
    },
    "isIPV6Enabled": true,
    "callerReference": "1578329170895",
    "continuousDeploymentPolicyId": "",
    "enabled": true,
    "defaultCacheBehavior": {
        "targetOriginId": "d111111abcdef8",
        "minTTL": 0,
        "compress": false,
        "maxTTL": 31536000,
        "functionAssociations": {

```

```
        "quantity": 0,
        "items": []
    },
    "trustedKeyGroups": {
        "quantity": 0,
        "items": [],
        "enabled": false
    },
    "smoothStreaming": false,
    "fieldLevelEncryptionId": "",
    "defaultTTL": 86400,
    "lambdaFunctionAssociations": {
        "quantity": 0,
        "items": []
    },
    "viewerProtocolPolicy": "redirect-to-https",
    "forwardedValues": {
        "cookies": {"forward": "none"},
        "queryStringCacheKeys": {
            "quantity": 0,
            "items": []
        },
        "queryString": false,
        "headers": {
            "quantity": 1,
            "items": ["*"]
        }
    },
    "trustedSigners": {
        "items": [],
        "enabled": false,
        "quantity": 0
    },
    "allowedMethods": {
        "quantity": 2,
        "items": [
            "HEAD",
            "GET"
        ],
        "cachedMethods": {
            "quantity": 2,
            "items": [
                "HEAD",
                "GET"
            ]
        }
    }
}
```

```

        ]
      }
    },
    "staging": false,
    "origins": {
      "quantity": 1,
      "items": [
        {
          "originPath": "",
          "connectionTimeout": 10,
          "customOriginConfig": {
            "originReadTimeout": 30,
            "hTTPSPort": 443,
            "originProtocolPolicy": "https-only",
            "originKeepaliveTimeout": 5,
            "hTTPPort": 80,
            "originSslProtocols": {
              "quantity": 3,
              "items": [
                "TLSv1",
                "TLSv1.1",
                "TLSv1.2"
              ]
            }
          },
          "id": "d1111111abcdef8",
          "domainName": "d1111111abcdef8.cloudfront.net",
          "connectionAttempts": 3,
          "customHeaders": {
            "quantity": 0,
            "items": []
          },
          "originShield": {"enabled": false},
          "originAccessControlId": ""
        }
      ]
    },
    "comment": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "id": "EDFDVBD6EXAMPLE",
  "ifMatch": "E1RTLUR9YES760"
},
"responseElements": {

```

```

"distribution": {
  "activeTrustedSigners": {
    "quantity": 0,
    "enabled": false
  },
  "id": "EDFDVBD6EXAMPLE",
  "domainName": "d111111abcdef8.cloudfront.net",
  "distributionConfig": {
    "defaultRootObject": "",
    "aliases": {
      "quantity": 3,
      "items": [
        "alejandro_rosalez.awsps.myinstance.com",
        "cross-testing.alejandro_rosalez.awsps.myinstance.com",
        "*.alejandro_rosalez.awsps.myinstance.com"
      ]
    },
    "cacheBehaviors": {"quantity": 0},
    "httpVersion": "http2and3",
    "originGroups": {"quantity": 0},
    "viewerCertificate": {
      "minimumProtocolVersion": "TLSv1.2_2021",
      "cloudFrontDefaultCertificate": false,
      "aCMCertificateArn": "arn:aws:acm:us-east-1:111122223333:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "sLSupportMethod": "sni-only",
      "certificateSource": "acm",
      "certificate": "arn:aws:acm:us-east-1:111122223333:certificate/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
    },
    "webACLId": "arn:aws:wafv2:us-east-1:111122223333:global/webacl/testing-acl/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "customErrorResponses": {"quantity": 0},
    "logging": {
      "includeCookies": false,
      "prefix": "",
      "enabled": false,
      "bucket": ""
    },
    "priceClass": "PriceClass_All",
    "restrictions": {
      "geoRestriction": {
        "restrictionType": "none",
        "quantity": 0
      }
    }
  }
}

```

```

    }
  },
  "isIPv6Enabled": true,
  "callerReference": "1578329170895",
  "continuousDeploymentPolicyId": "",
  "enabled": true,
  "defaultCacheBehavior": {
    "targetOriginId": "d1111111abcdef8",
    "minTTL": 0,
    "compress": false,
    "maxTTL": 31536000,
    "functionAssociations": {"quantity": 0},
    "trustedKeyGroups": {
      "quantity": 0,
      "enabled": false
    },
  },
  "smoothStreaming": false,
  "fieldLevelEncryptionId": "",
  "defaultTTL": 86400,
  "lambdaFunctionAssociations": {"quantity": 0},
  "viewerProtocolPolicy": "redirect-to-https",
  "forwardedValues": {
    "cookies": {"forward": "none"},
    "queryStringCacheKeys": {"quantity": 0},
    "queryString": false,
    "headers": {
      "quantity": 1,
      "items": ["*"]
    }
  },
  },
  "trustedSigners": {
    "enabled": false,
    "quantity": 0
  },
  },
  "allowedMethods": {
    "quantity": 2,
    "items": [
      "HEAD",
      "GET"
    ],
  },
  "cachedMethods": {
    "quantity": 2,
    "items": [
      "HEAD",

```

```

        "GET"
      ]
    }
  },
  "staging": false,
  "origins": {
    "quantity": 1,
    "items": [
      {
        "originPath": "",
        "connectionTimeout": 10,
        "customOriginConfig": {
          "originReadTimeout": 30,
          "hTTPSPort": 443,
          "originProtocolPolicy": "https-only",
          "originKeepaliveTimeout": 5,
          "hTTPPort": 80,
          "originSslProtocols": {
            "quantity": 3,
            "items": [
              "TLSv1",
              "TLSv1.1",
              "TLSv1.2"
            ]
          }
        },
        "id": "d111111abcdef8",
        "domainName": "d111111abcdef8.cloudfront.net",
        "connectionAttempts": 3,
        "customHeaders": {"quantity": 0},
        "originShield": {"enabled": false},
        "originAccessControlId": ""
      }
    ]
  },
  "comment": "HIDDEN_DUE_TO_SECURITY_REASONS"
},
"aliasICPRecordals": [
  {
    "cNAME": "alejandro_rosalez.awsps.myinstance.com",
    "iCPRecordalStatus": "APPROVED"
  },
  {

```



```

        "cNAME": "cross-testing.alejandro_rosalez.awsps.myinstance.com",
        "iCPRecordalStatus": "APPROVED"
    },
    {
        "cNAME": "/*.alejandro_rosalez.awsps.myinstance.com",
        "iCPRecordalStatus": "APPROVED"
    }
],
"arn": "arn:aws:cloudfront::111122223333:distribution/EDFDVBD6EXAMPLE",
"status": "InProgress",
"lastModifiedTime": "Feb 2, 2024 7:26:01 PM",
"activeTrustedKeyGroups": {
    "enabled": false,
    "quantity": 0
},
"inProgressInvalidationBatches": 0
},
"eTag": "E1YHBLAB2BJY1G"
},
"requestID": "4e6b66f9-d548-11e3-a8a9-73e33example",
"eventID": "5ab02562-0fc5-43d0-b7b6-90293example",
"readOnly": false,
"eventType": "AwsApiCall",
"apiVersion": "2020_05_31",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management",
"tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "cloudfront.amazonaws.com"
},
"sessionCredentialFromConsole": "true"
}

```

範例：UpdateKeys

以下範例顯示的 CloudTrail 事件會示範 [UpdateKeys](#) 操作。

對 CloudFront KeyVaultStore API 進行呼叫時，eventSource 是 `edgekeyvaluestore.amazonaws.com` 而不是 `cloudfront.amazonaws.com`。

```
{
```

```

"eventVersion": "1.09",
"userIdentity": {
  "type": "AssumedRole",
  "principalId": "AIDACKCEVSQ6C2EXAMPLE:role-session-name",
  "arn": "arn:aws:sts::111122223333:assumed-role/Admin/role-session-name",
  "accountId": "111122223333",
  "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
  "sessionContext": {
    "sessionIssuer": {
      "type": "Role",
      "principalId": "AIDACKCEVSQ6C2EXAMPLE",
      "arn": "arn:aws:iam::111122223333:role/Admin",
      "accountId": "111122223333",
      "userName": "Admin"
    },
    "attributes": {
      "creationDate": "2023-11-01T23:41:14Z",
      "mfaAuthenticated": "false"
    }
  }
},
"eventTime": "2023-11-01T23:41:28Z",
"eventSource": "edgekeyvaluestore.amazonaws.com",
"eventName": "UpdateKeys",
"awsRegion": "us-east-1",
"sourceIPAddress": "3.235.183.252",
"userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/121.0.0.0 Safari/537.36",
"requestParameters": {
  "kvsARN": "arn:aws:cloudfront::111122223333:key-value-store/a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111",
  "ifMatch": "KV306B1CX531EBP",
  "deletes": [
    {"key": "key1"}
  ]
},
"responseElements": {
  "itemCount": 0,
  "totalSizeInBytes": 0,
  "eTag": "KVDC9VEVZ71ZG0"
},
"requestID": "5ccf104c-acce-4ea1-b7fc-73e33example",
"eventID": "a0b1b5c7-906c-439d-9925-90293example",
"readOnly": false,

```

```
"resources": [
  {
    "accountId": "111122223333",
    "type": "AWS::CloudFront::KeyValueStore",
    "ARN": "arn:aws:cloudfront::111122223333:key-value-store/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
],
"eventType": "AwsApiCall",
"managementEvent": false,
"recipientAccountId": "111122223333",
"eventCategory": "Data",
"tlsDetails": {
  "tlsVersion": "TLSv1.3",
  "cipherSuite": "TLS_AES_128_GCM_SHA256",
  "clientProvidedHostHeader": "111122223333.cloudfront-kvs.global.api.aws"
}
}
```

如需有關 CloudTrail 記錄內容的資訊，請參閱《AWS CloudTrail 使用者指南》中的 [CloudTrail record contents](#)。

使用 追蹤組態變更 AWS Config

若要記錄和評估 AWS 資源的組態，您可以使用 AWS Config，這可讓您詳細檢視 分佈的組態。這包含資源彼此之間的關係和之前的組態方式，所以您可以檢閱其中的長期變化。

您也可以使用 AWS Config 記錄 CloudFront 分佈設定的組態變更。您可以擷取對於分佈狀態、價格分級、來源、地理限制設定以及 Lambda@Edge 組態的變更。

Note

AWS Config 不會記錄 CloudFront 串流分佈的鍵/值標籤。

內容

- [AWS Config 使用 CloudFront 設定](#)
- [檢視 CloudFront 組態歷史記錄](#)
- [使用 AWS Config 規則評估 CloudFront 組態](#)

AWS Config 使用 CloudFront 設定

設定時 AWS Config，您可以選擇記錄所有支援 AWS 的資源，或僅記錄一些指定的資源，例如僅記錄 CloudFront 的變更。如需獲得支援的 CloudFront 資源清單，請參閱《AWS Config 開發人員指南》中支援資源類型主題的 [Amazon CloudFront](#) 章節。

備註

- 若要追蹤對 CloudFront 分佈進行的組態變更，您必須登入位於美國東部 (維吉尼亞北部) AWS 區域的 CloudFront 主控台。
- 使用記錄資源時，可能會延遲 AWS Config。只有在發現資源後才會 AWS Config 記錄資源。

Console

AWS Config 使用 CloudFront 設定

1. 登入 AWS 管理主控台 並開啟 [AWS Config 主控台](#)。
2. 選擇 Get Started Now (立即開始)。
3. 在設定頁面上，針對要記錄的資源類型，指定您要 AWS Config 記錄 AWS 的資源類型。如果您只想要記錄 CloudFront 的變更，請選擇特定類型，然後在 CloudFront 中，選擇您想追蹤其變更的分佈或串流分發。

若要新增或變更想追蹤的分佈，請在完成初始設定後，選擇左側的 Settings (設定)。

4. 指定其他必要選項 AWS Config：設定通知、指定組態資訊的位置，以及新增評估資源類型的規則。

如需詳細資訊，請參閱《AWS Config 開發人員指南》中的 [AWS Config 使用主控台設定](#)。

AWS CLI

若要使用 AWS Config 設定 CloudFront AWS CLI，請參閱《AWS Config 開發人員指南》中的 [AWS Config 使用 AWS CLI 設定](#)。

AWS Config API

若要使用 AWS Config API AWS Config 設定 CloudFront，請參閱 API 參考中的 [StartConfigurationRecorder](#) API 操作。AWS Config

檢視 CloudFront 組態歷史記錄

AWS Config 開始記錄分佈的組態變更後，您可以取得為 CloudFront 設定的任何分佈的組態歷史記錄。

您可以透過以下方式來檢視組態歷史記錄。

Console

對於每個記錄的資源，您可以查看時間軸頁面，其中提供組態詳細資訊的歷史記錄。若要查看此頁面，請選擇專用執行個體頁面中的 Config Timeline (組態時間軸) 欄內的灰色圖示。

如需詳細資訊，請參閱 [《開發人員指南》中的在 AWS Config 主控台中檢視組態詳細資訊](#)。AWS Config

AWS CLI

若要取得您所有分佈的清單，請執行 [list-discovered-resources](#) 命令，如以下範例所示。

```
aws configservice list-discovered-resources --resource-type
AWS::CloudFront::Distribution
```

若要取得特定時間間隔內的分佈組態詳細資訊，請執行 [get-resource-config-history](#) 命令。

如需詳細資訊，請參閱 AWS Config 開發人員指南中的 [使用 CLI 檢視組態詳細資訊](#)。

AWS Config API

若要取得您所有的分佈清單，請使用 [ListDiscoveredResources](#) API 操作。

若要取得特定時間間隔內的分佈組態詳細資訊，請使用 [GetResourceConfigHistory](#) API 操作。如需詳細資訊，請參閱 [AWS Config API 參考](#)。

使用 AWS Config 規則評估 CloudFront 組態

您可以使用 AWS Config 規則，根據所需的組態來評估組態。例如，AWS Config Rules 可協助您評估 CloudFront 資源是否符合常見的安全最佳實務。您可以選擇在組態變更時觸發的受管規則，例如檢視器政策 HTTPS、啟用 SNI、啟用 OAC、啟用原始伺服器容錯移轉、AWS WAF WebACL 或 AWS Shield Advanced 資源政策。

受管規則可以依您選擇的頻率定期執行評估。AWS Firewall Manager 依賴 AWS Config 進行自動提醒和修復。如需詳細資訊，請參閱 [《AWS Config 開發人員指南》中的使用 AWS Config 規則評估資源](#) 和 [AWS Config 受管規則清單](#)。

Amazon CloudFront 的安全性

的雲端安全 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，該架構專為符合最安全敏感組織的需求而建置。

安全性是 AWS 與您之間의 共同責任。[共同責任模型](#) 將此描述為雲端的安全和雲端內的安全：

- 雲端的安全性 – AWS 負責保護在 AWS 雲端中執行 AWS 服務的基礎設施。AWS 也為您提供可安全使用的服務。在 [AWS 合規計畫](#) 中，第三方稽核員會定期測試並驗證我們的安全功效。若要瞭解適用於 Amazon CloudFront 的合規計畫，請參閱[合規計畫範圍內的 AWS 服務](#)。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對資料敏感度、組織要求，以及適用法律和法規等其他因素負責。

本文件有助於您了解如何在使用 CloudFront 時套用共同責任模型。下列主題將顯示如何配置 CloudFront 以達到您的安全和合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 CloudFront 資源。

主題

- [Amazon CloudFront 的資料保護](#)
- [適用於 Amazon CloudFront 的 Identity and Access Management](#)
- [在 Amazon CloudFront 中進行記錄和監控](#)
- [Amazon CloudFront 的合規驗證](#)
- [Amazon CloudFront 中的復原功能](#)
- [Amazon CloudFront 的基礎設施安全性](#)

Amazon CloudFront 的資料保護

AWS [共同責任模型](#) 適用於 Amazon CloudFront 中的資料保護。如此模型所述，AWS 負責保護執行所有的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱[資料隱私權常見問答集](#)。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的[AWS 共同責任模型和 GDPR](#) 部落格文章。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《使用者指南》中的[使用 CloudTrail 追蹤](#)。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱[聯邦資訊處理標準 \(FIPS\) 140-3](#)。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 CloudFront 或使用主控台、API AWS CLI或 AWS SDKs的其他 AWS 服務 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

Amazon CloudFront 提供了多種選項，以協助您保護其所提供之內容：

- 設定 HTTPS 連線。
- 設定欄位層級加密，以在傳輸期間為特定資料提供額外的安全性。
- 限制存取內容，以使特定人員或特定區域人員才能檢視內容。

下列主題說明了關於選項的詳細資訊。

主題

- [傳輸中加密](#)
- [靜態加密](#)
- [限制存取內容](#)

傳輸中加密

若要在傳輸過程中加密資料，您可配置 Amazon CloudFront 以請求檢視器使用 HTTPS 來請求您的檔案，如此當 CloudFront 與檢視器進行通訊時，連線即可加密。您也可以配置 CloudFront 使用 HTTPS，來從原始伺服器取得檔案，如此當 CloudFront 與原始伺服器進行通訊時，連線即可加密。

如需詳細資訊，請參閱 [透過 CloudFront 使用 HTTPS](#)。

欄位層級加密與 HTTPS 一起新增額外的安全層，可讓您在整個系統處理過程中保護特定的資料，以便只有特定應用程式才能看到它。透過在 CloudFront 中配置欄位層級加密，您可安全地將使用者提交敏感資訊上傳到您的 Web 伺服器。您用戶端提供的敏感資訊會在更靠近使用者節點時進行加密。該敏感資訊會在整個應用程式堆疊中保持加密，以確保只有需要資料 (而且具有將資料解密的登入資料) 的應用程式才能夠這樣做。

如需詳細資訊，請參閱 [使用欄位層級加密來協助保護敏感資料](#)。

CloudFront API 端點 `cloudfront.amazonaws.com`、`cloudfront-fips.amazonaws.com` 僅接受 HTTPS 流量。這表示當您使用 CloudFront API 傳送和接收資訊時，您的資料 (包括分發組態、快取政策和原始伺服器請求政策、金鑰群組和公有金鑰，以及 CloudFront 函數中的函數程式碼) 始終會在傳輸過程中加密。此外，傳送至 CloudFront API 端點的所有請求都會使用 AWS 登入資料登入並登入 AWS CloudTrail。

CloudFront Functions 中的函數程式碼和組態在複製到節點連接點 (POP) 以及 CloudFront 使用的其他儲存位置之間時，一律會在傳輸過程中加密。

靜態加密

CloudFront Functions 中的函數程式碼和組態始終以加密格式儲存在邊緣節點 POP 上，以及 CloudFront 使用的其他儲存位置中。

限制存取內容

許多透過網際網路分佈內容的公司想要限制使用者子集的文件、業務資料、媒體串流，或內容的存取許可。為了使用 Amazon CloudFront 安全的提供此內容，您可以執行一或多個以下項目：

使用簽章的 URL 或 Cookie

您可以使用簽署的 URL 或簽署的 Cookie 以透過 CloudFront 來提供此私人內容，以限制存取僅對特定使用者開放的內容，例如已付費的使用者。如需詳細資訊，請參閱 [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)。

限制對 Amazon S3 儲存貯體中內容的存取

如果您使用例如 CloudFront 簽署的 URL 或簽署的 Cookie 來限制存取內容，您也不希望人們使用檔案的直接 URL 來檢視檔案。反之，您想要他們僅使用 CloudFront URL 來存取檔案，以達保護之目的。

如果您使用 Amazon S3 儲存貯體做為 CloudFront 分發的原始伺服器，您可設定原始存取控制 (OAC)，可限制對 S3 儲存貯體的存取。如需詳細資訊，請參閱[the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

限制存取 Application Load Balancer 所提供的內容

當您在 ELB 中使用 CloudFront 搭配 Application Load Balancer 做為原始伺服器時，您可以設定 CloudFront 以防止使用者直接存取 Application Load Balancer。這讓使用者只能透過 CloudFront 存取 Application Load Balancer，確保您能夠享有使用 CloudFront 的好處。如需詳細資訊，請參閱[限制對 Application Load Balancers 的存取](#)。

使用 AWS WAF Web ACLs

您可以使用 AWS WAF Web 應用程式防火牆服務來建立 Web 存取控制清單 (Web ACL)，以限制對內容的存取。根據您指定的條件，例如請求源自於此的 IP 地址或查詢字串的值，CloudFront 會使用請求的內容或 HTTP 403 狀態碼 (禁止) 回應請求。如需詳細資訊，請參閱[使用 AWS WAF 保護](#)。

使用地理限制

您可以使用地理限制功能 (也稱為地理封鎖)，來防止特定地理位置的使用者，存取您透過 CloudFront 分發所提供的內容。在設定地理限制時，您有幾個選項可選擇。如需詳細資訊，請參閱[限制您內容的地理分佈](#)。

適用於 Amazon CloudFront 的 Identity and Access Management

AWS Identity and Access Management (IAM) 是一種 AWS 服務，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員會控制誰經過身分身分驗證 (已登入) 並得到授權 (具有許可) 來使用 CloudFront 資源。IAM 是您可以免費使用 AWS 服務的。

主題

- [目標對象](#)
- [使用身分驗證](#)
- [使用政策管理存取權](#)

- [Amazon CloudFront 如何與 IAM 搭配運作](#)
- [Amazon CloudFront 的身分型政策範例](#)
- [AWS Amazon CloudFront 的 受管政策](#)
- [使用適用於 CloudFront 的服務連結角色](#)
- [Amazon CloudFront 身分識別和存取疑難排解](#)

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

- 服務使用者 — 若無法存取某些功能，請向管理員申請所需許可 (請參閱 [Amazon CloudFront 身分識別和存取疑難排解](#))
- 服務管理員 — 負責設定使用者存取權並提交相關許可請求 (請參閱 [Amazon CloudFront 如何與 IAM 搭配運作](#))
- IAM 管理員 — 撰寫政策以管理存取控制 (請參閱 [Amazon CloudFront 的身分型政策範例](#))

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的[如何登入您的 AWS 帳戶](#)。

對於程式設計存取，AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的[API 請求的AWS 第 4 版簽署程序](#)。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分可完整存取所有 AWS 服務 和資源。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的[需要根使用者憑證的任務](#)。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取。

聯合身分是您企業目錄、Web 身分提供者的使用者，或是 AWS 服務使用身分來源的憑證 Directory Service 存取的使用者。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的[什麼是 IAM Identity Center？](#)。

IAM 使用者和群組

[IAM 使用者](#)是一種身分，具備單一人員或應用程式的特定許可。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的[要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取](#)。

[IAM 群組](#)會指定 IAM 使用者集合，使管理大量使用者的許可更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 使用者的使用案例](#)。

IAM 角色

[IAM 角色](#)是一種具特定許可的身分，可提供臨時憑證。您可以透過[從使用者切換到 IAM 角色（主控台）](#)或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[擔任角色的方法](#)。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的快帳戶資源存取](#)。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS，會評估這些政策。大多數政策會以 JSON 文件 AWS 形式存放在中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的[JSON 政策概觀](#)。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的[透過客戶管理政策定義自訂 IAM 許可](#)。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策與內嵌政策之間選擇，請參閱《IAM 使用者指南》中的[在受管政策和內嵌政策間選擇](#)。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策與 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。您必須在資源型政策中[指定主體](#)。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 實體許可界限](#)。
- 服務控制政策 (SCP) — 為 AWS Organizations 中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[服務控制政策](#)。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[資源控制政策 \(RCP\)](#)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參《IAM 使用者指南》中的[工作階段政策](#)。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多個政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的[政策評估邏輯](#)。

Amazon CloudFront 如何與 IAM 搭配運作

在您使用 IAM 管理 CloudFront 的存取權之前，請了解搭配 CloudFront 使用的 IAM 功能有哪些。

您可以搭配 Amazon CloudFront 使用的 IAM 功能

IAM 功能	CloudFront 支援
身分型政策	是

IAM 功能	CloudFront 支援
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵 (服務特定)	是
ACL	否
ABAC(政策中的標籤)	部分
臨時憑證	是
轉送存取工作階段 (FAS)	否
服務角色	否
服務連結角色	是

若要全面了解 CloudFront 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱 [《AWS IAM 使用者指南》](#) 中的與 IAM 搭配使用的 [服務](#)。

適用於 CloudFront 的身分型政策

支援身分型政策：是

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的[透過客戶管理政策定義自訂 IAM 許可](#)。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 [IAM JSON 政策元素參考](#)。

CloudFront 的身分型政策範例

若要檢視 CloudFront 身分類型政策的範例，請參閱 [Amazon CloudFront 的身分型政策範例](#)。

CloudFront 內的資源型政策

支援資源型政策：否

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中[指定主體](#)。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的快帳戶資源存取](#)。

CloudFront 的政策動作

支援政策動作：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 Action 元素描述您可以用來允許或拒絕政策中存取的動作。政策會使用動作來授予執行相關聯動作的許可。

若要查看 CloudFront 動作的清單，請參閱服務授權參考中的[Amazon CloudFront 定義的動作](#)。

CloudFront 中的政策動作會在動作前使用以下前綴：

```
cloudfront
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [  
    "cloudfront:action1",  
    "cloudfront:action2"  
]
```

若要檢視 CloudFront 身分類型政策的範例，請參閱[Amazon CloudFront 的身分型政策範例](#)。

CloudFront 的政策資源

支援政策資源：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。最佳實務是使用其 [Amazon Resource Name \(ARN\)](#) 來指定資源。若動作不支援資源層級許可，使用萬用字元 (*) 表示該陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 CloudFront 資源類型清單及其 ARN，請參閱服務授權參考中的 [Amazon CloudFront 定義的資源](#)。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 [Amazon CloudFront 定義的動作](#)。

若要檢視 CloudFront 身分類型政策的範例，請參閱 [Amazon CloudFront 的身分型政策範例](#)。

CloudFront 的政策條件索引鍵

支援服務特定政策條件金鑰：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素會根據定義的條件，指定陳述式的執行時機。您可以建立使用 [條件運算子](#) 的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的 [AWS 全域條件內容索引鍵](#)。

如要查看 CloudFront 條件索引鍵的清單，請參閱《服務授權參考》中的 [Amazon CloudFront 條件索引鍵](#)。若要了解您可以搭配哪些動作和資源使用條件金鑰，請參閱 [Amazon CloudFront 定義的動作](#)。

若要檢視 CloudFront 身分類型政策的範例，請參閱 [Amazon CloudFront 的身分型政策範例](#)。

CloudFront 的 ACL

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

ABAC 搭配 CloudFront

支援 ABAC (政策中的標籤)：部分

屬性型存取控制 (ABAC) 是一種授權策略，根據稱為標籤的屬性定義許可權。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在主體的標籤符合資源上的標籤時允許操作。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的[條件元素](#)中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的[使用 ABAC 授權定義許可](#)。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的[使用屬性型存取控制 \(ABAC\)](#)。

CloudFront 僅對分佈支援 ABAC。

將暫時登入資料與 CloudFront 搭配使用

支援臨時憑證：是

臨時登入資料提供 AWS 資源的短期存取權，當您使用聯合或切換角色時，會自動建立。AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的臨時安全憑證與可與 IAM 搭配運作的 AWS 服務](#)。

CloudFront 的轉送存取工作階段

支援轉寄存取工作階段 (FAS)：否

轉送存取工作階段 (FAS) 使用呼叫的委託人許可 AWS 服務，並結合 AWS 服務請求向下游服務提出請求。如需提出 FAS 請求時的政策詳細資訊，請參閱[轉發存取工作階段](#)。

CloudFront 的服務角色

支援服務角色：否

服務角色是服務擔任的[IAM 角色](#)，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[建立角色以委派許可給 AWS 服務](#)。

Warning

變更服務角色的許可有可能會讓 CloudFront 功能出現故障。只有 CloudFront 提供指引時，才能編輯服務角色。

CloudFront 的服務連結角色

支援服務連結角色：是

服務連結角色是連結至的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的中 AWS 帳戶，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

CloudFront 使用服務連結角色為您執行動作。如需有關建立或管理 CloudFront 服務連結角色的詳細資訊，請參閱 [使用適用於 CloudFront 的服務連結角色](#)。如需有關建立或管理 Lambda@Edge 服務連結角色的詳細資訊，請參閱 [Lambda@Edge 的服務連結角色](#)。

如需建立或管理服務連結角色的詳細資訊，請參閱[可搭配 IAM 運作的AWS 服務](#)。在資料表中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

Amazon CloudFront 的身分型政策範例

根據預設，使用者和角色不具備建立或修改 CloudFront 資源的權限。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。

如需了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的[建立 IAM 政策 \(主控台\)](#)。

如需 CloudFront 所定義之動作和資源類型的詳細資訊，包括每種資源類型的 ARN 格式，請參閱《服務授權參考》中 [Amazon CloudFront 適用的動作、資源和條件索引鍵](#)。

主題

- [政策最佳實務](#)
- [允許使用者檢視他們自己的許可](#)
- [透過程式設計方式存取 CloudFront 的許可](#)
- [使用 CloudFront 主控台所需的許可](#)
- [客戶受管政策範例](#)

政策最佳實務

以身分為基礎的政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 CloudFront 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱《IAM 使用者指南》中的 [AWS 受管政策](#) 或 [任務職能的 AWS 受管政策](#)。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱《IAM 使用者指南》中的 [IAM 中的政策和許可](#)。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 使用服務動作 AWS 服務，您也可以使用條件來授予存取，例如 CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM JSON 政策元素：條件](#)。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的 [使用 IAM Access Analyzer 驗證政策](#)。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的 [透過 MFA 的安全 API 存取](#)。

如需 IAM 中最佳實務的相關資訊，請參閱《IAM 使用者指南》中的 [IAM 安全最佳實務](#)。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控制台或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ]
    }
  ]
}
```

```
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
  },
  {
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
      "iam:GetGroupPolicy",
      "iam:GetPolicyVersion",
      "iam:GetPolicy",
      "iam:ListAttachedGroupPolicies",
      "iam:ListGroupPolicies",
      "iam:ListPolicyVersions",
      "iam:ListPolicies",
      "iam:ListUsers"
    ],
    "Resource": "*"
  }
]
```

透過程式設計方式存取 CloudFront 的許可

以下說明許可政策。Sid (陳述式 ID) 為選用。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAllCloudFrontPermissions",
      "Effect": "Allow",
      "Action": ["cloudfront:*"],
      "Resource": "*"
    }
  ]
}
```

此政策授予執行所有 CloudFront 操作的許可，足夠以程式設計方式存取 CloudFront。如果您使用主控台存取 CloudFront，請參閱[使用 CloudFront 主控台所需的許可](#)。

如需指定以授予或拒絕使用每個動作的許可的動作和 ARN 清單，請參閱《服務授權參考》中的 [Amazon CloudFront 的動作、資源及條件索引鍵](#)。

使用 CloudFront 主控台所需的許可

若要授予 CloudFront 主控台的完整存取權，可在以下許可政策中授予許可：

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "acm:ListCertificates",
        "cloudfront:*",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:PutMetricAlarm",
        "cloudwatch:GetMetricStatistics",
        "elasticloadbalancing:DescribeLoadBalancers",
        "iam:ListServerCertificates",
        "sns:ListSubscriptionsByTopic",
        "sns:ListTopics",
        "waf:GetWebACL",
        "waf:ListWebACLs"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:PutBucketPolicy"
      ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

需要許可的原因如下：

acm:ListCertificates

當您使用 CloudFront 主控台建立和更新分佈時，若想要將 CloudFront 設定為在檢視器與 CloudFront 之間或 CloudFront 與原始伺服器之間必須使用 HTTPS，許可會讓您可查看 ACM 憑證清單。

如果不是使用 CloudFront 主控台，不需要此許可。

cloudfront:*

讓您可執行所有 CloudFront 動作。

cloudwatch:DescribeAlarms 和 **cloudwatch:PutMetricAlarm**

讓您可在 CloudFront 主控台中建立和檢視 CloudWatch 警示。另請參閱 `sns:ListSubscriptionsByTopic` 和 `sns:ListTopics`。

如果不是使用 CloudFront 主控台，不需要這些許可。

cloudwatch:GetMetricStatistics

讓 CloudFront 可在 CloudFront 主控台中轉譯 CloudWatch 指標。

如果不是使用 CloudFront 主控台，不需要此許可。

elasticloadbalancing:DescribeLoadBalancers

建立和更新分佈時，可讓您在可用原始伺服器清單中檢視 ELB 負載平衡器的清單。

如果不是使用 CloudFront 主控台，不需要此許可。

iam:ListServerCertificates

當您使用 CloudFront 主控台建立和更新分佈時，若想要將 CloudFront 設定為在檢視器與 CloudFront 之間或 CloudFront 與原始伺服器之間必須使用 HTTPS，許可會讓您可查看 IAM 憑證存放區中的憑證清單。

如果不是使用 CloudFront 主控台，不需要此許可。

s3:ListAllMyBuckets

當您建立和更新分佈，可以執行以下操作：

- 檢視在可用的原始伺服器清單中的 S3 儲存貯體清單
- 檢視可以保存存取記錄的 S3 儲存貯體清單

如果不是使用 CloudFront 主控台，不需要此許可。

S3:PutBucketPolicy

當您建立或更新限制存取 S3 儲存貯體的分佈時，許可會讓使用者可更新儲存貯體政策來授予對 CloudFront 原始存取身分的存取權。如需詳細資訊，請參閱[the section called “使用原始存取身分 \(舊版，不建議使用\)”](#)。

如果不是使用 CloudFront 主控台，不需要此許可。

sns:ListSubscriptionsByTopic 和 **sns:ListTopics**

當您在 CloudFront 主控台中建立 CloudWatch 警示時，讓您可為通知選擇 SNS 主題。

如果不是使用 CloudFront 主控台，不需要這些許可。

waf:GetWebACL 和 **waf:ListWebACLs**

可讓您在 CloudFront 主控台中檢視 AWS WAF Web ACLs 的清單。

如果不是使用 CloudFront 主控台，不需要這些許可。

CloudFront 主控台的僅限許可動作

您可在 [CloudFront 安全防護優惠組合](#) 頁面執行下列 CloudFront 動作。下列 API 動作並非由您的程式碼呼叫，也不會包含在 AWS CLI 和 AWS SDKs 中。

Action	Description
CreateSavingsPlan	授予許可權限以建立新 Savings Plan。
GetSavingsPlan	授予許可權限以獲取 Savings Plan。
ListRateCards	授予許可權限以列出帳戶的 CloudFront 費率卡。
ListSavingsPlans	授予許可權限以列出帳戶中的 Savings Plans。
ListUsages	授予許可權限以列出 CloudFront 用量。
UpdateSavingsPlan	授予許可權限以更新 Savings Plan。

備註

- 如需 CloudFront Savings Plans 的詳細資訊，請參閱 [Amazon CloudFront 常見問答集](#) 的 CloudFront 安全防護優惠組合區段。
- 如果您為 CloudFront 建立 Savings Plan，並想在稍後刪除，請聯絡 [AWS 支援](#)。

客戶受管政策範例

您可以建立自訂 IAM 政策，授予執行 CloudFront API 動作的相關許可。您可以將這些自訂政策連接至需要指定許可的 IAM 使用者或群組。這些政策會在您使用 CloudFront API、AWS 開發套件或 AWS CLI 時運作。以下範例示範幾個常用案例的許可。如需將 CloudFront 完整存取權授予使用者的政策，請參閱 [使用 CloudFront 主控台所需的許可](#)。

範例

- [範例 1：允許讀取存取所有分佈](#)
- [範例 2：允許建立、更新和刪除分佈](#)
- [範例 3：允許建立和列出失效](#)
- [範例 4：允許建立分佈](#)

範例 1：允許讀取存取所有分佈

以下許可政策授予使用者可在 CloudFront 主控台中查看所有分佈的許可：

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "acm:ListCertificates",
        "cloudfront:GetDistribution",
        "cloudfront:GetDistributionConfig",
        "cloudfront:ListDistributions",
        "cloudfront:ListCloudFrontOriginAccessIdentities",
        "elasticloadbalancing:DescribeLoadBalancers",
```

```

        "iam:ListServerCertificates",
        "sns:ListSubscriptionsByTopic",
        "sns:ListTopics",
        "waf:GetWebACL",
        "waf:ListWebACLs"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:ListAllMyBuckets"
    ],
    "Resource": "arn:aws:s3:::*"
  }
]
}

```

範例 2：允許建立、更新和刪除分佈

以下許可政策讓使用者可使用 CloudFront 主控台建立、更新和刪除分佈：

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "acm:ListCertificates",
        "cloudfront:CreateDistribution",
        "cloudfront:DeleteDistribution",
        "cloudfront:GetDistribution",
        "cloudfront:GetDistributionConfig",
        "cloudfront:ListDistributions",
        "cloudfront:UpdateDistribution",
        "cloudfront:ListCloudFrontOriginAccessIdentities",
        "elasticloadbalancing:DescribeLoadBalancers",
        "iam:ListServerCertificates",
        "sns:ListSubscriptionsByTopic",
        "sns:ListTopics",

```



```

        "waf:GetWebACL",
        "waf:ListWebACLs"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:ListAllMyBuckets",
      "s3:PutBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::*"
  }
]
}

```

`cloudfront:ListCloudFrontOriginAccessIdentities` 許可讓使用者可將許可自動授予現有原始存取身分，以存取 Amazon S3 儲存貯體中的物件。如果您還希望使用者能夠建立原始存取身分，則還需要允許 `cloudfront:CreateCloudFrontOriginAccessIdentity` 許可。

範例 3：允許建立和列出失效

以下許可政策可讓使用者建立和失效清單。它包括對 CloudFront 分佈的讀取存取權，因為您會先顯示分佈的設定再來建立和查看失效：

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "acm:ListCertificates",
        "cloudfront:GetDistribution",
        "cloudfront:GetStreamingDistribution",
        "cloudfront:GetDistributionConfig",
        "cloudfront:ListDistributions",
        "cloudfront:ListCloudFrontOriginAccessIdentities",
        "cloudfront:CreateInvalidation",
        "cloudfront:GetInvalidation",

```

```

        "cloudfront:ListInvalidations",
        "elasticloadbalancing:DescribeLoadBalancers",
        "iam:ListServerCertificates",
        "sns:ListSubscriptionsByTopic",
        "sns:ListTopics",
        "waf:GetWebACL",
        "waf:ListWebACLs"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "s3:ListAllMyBuckets"
    ],
    "Resource": "arn:aws:s3:::*"
}
]
}

```

範例 4：允許建立分佈

以下許可權限政策可授予使用者許可權限，以便在 CloudFront 主控台中建立和列出所有分佈：對於 CreateDistribution 動作，請指定 Resource 的萬用字元 (*)，而非分佈 ARN (arn:aws:cloudfront::123456789012:distribution/*) 的萬用字元。如需有關 Resource 元素的詳細資訊，請參閱《IAM 使用者指南》中的 [IAM JSON 政策元素：資源](#)。

JSON

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "cloudfront:CreateDistribution",
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",

```

```
        "Action": "cloudfront:ListDistributions",
        "Resource": "*"
    }
]
```

AWS Amazon CloudFront 的 受管政策

若要新增許可給使用者、群組和角色，使用 AWS 受管政策比自行撰寫政策更容易。[建立 IAM 客戶受管政策](#)需要時間和專業知識，而受管政策可為您的使用者提供其所需的許可。若要快速開始使用，您可以使用我們的 AWS 受管政策。這些政策涵蓋常見的使用案例，並可在您的 AWS 帳戶中使用。如需 AWS 受管政策的詳細資訊，請參閱《IAM 使用者指南》中的 [AWS 受管政策](#)。

AWS 服務會維護和更新 AWS 受管政策。您無法變更 AWS 受管政策中的許可。服務偶爾會在 AWS 受管政策中新增其他許可以支援新功能。此類型的更新會影響已連接政策的所有身分識別 (使用者、群組和角色)。當新功能啟動或新許可可用時，服務很可能會更新 AWS 受管政策。服務不會從 AWS 受管政策移除許可，因此政策更新不會破壞您現有的許可。

此外，AWS 支援跨多個 服務之任務函數的受管政策。例如，ReadOnlyAccess AWS 受管政策提供所有 AWS 服務和資源的唯讀存取權。當服務啟動新功能時，會為新操作和資源 AWS 新增唯讀許可。如需任務職能政策的清單和說明，請參閱 IAM 使用者指南中 [有關任務職能的 AWS 受管政策](#)。

主題

- [AWS 受管政策 : CloudFrontReadOnlyAccess](#)
- [AWS 受管政策 : CloudFrontFullAccess](#)
- [AWS 受管政策 : AWSCloudFrontLogger](#)
- [AWS 受管政策 : AWSLambdaReplicator](#)
- [AWS 受管政策 : AWSCloudFrontVPCOriginServiceRolePolicy](#)
- [AWS 受管政策的 CloudFront 更新](#)

AWS 受管政策：CloudFrontReadOnlyAccess

您可以將 CloudFrontReadOnlyAccess 政策連接到 IAM 身分。此政策允許 CloudFront 資源的唯讀許可。它還允許與 CloudFront 相關的其他 AWS 服務資源的唯讀許可，並在 CloudFront 主控台中可見。

許可詳細資訊

此政策包含以下許可。

- `cloudfront:Describe*` – 允許主體取得有關 CloudFront 資源中繼資料的資訊。
- `cloudfront:Get*` – 允許主參與者取得 CloudFront 資源的詳細資訊和組態。
- `cloudfront:List*` – 允許主參與者取得 CloudFront 資源的清單。
- `cloudfront-keyvaluestore:Describe*` – 允許主體取得有關鍵值存放區的資訊。
- `cloudfront-keyvaluestore:Get*` – 允許主體取得鍵值存放區的詳細資訊和組態。
- `cloudfront-keyvaluestore:List*` – 允許主體取得鍵值存放區的清單。
- `acm:DescribeCertificate` – 允許主體取得 ACM 憑證的詳細資訊。
- `acm:ListCertificates` – 允許主參與者取得 ACM 憑證清單。
- `iam:ListServerCertificates` – 允許主參與者取得存放在 IAM 的伺服器憑證清單。
- `route53:List*` – 允許主參與者取得 Route 53 資源的清單。
- `waf:ListWebACLs` – 允許主參與者取得 AWS WAF 的 Web ACL 清單。
- `waf:GetWebACL` – 允許主參與者取得 AWS WAF Web ACL 的詳細資訊。
- `wafv2:ListWebACLs` – 允許主參與者取得 AWS WAF 的 Web ACL 清單。
- `wafv2:GetWebACL` – 允許主參與者取得 AWS WAF Web ACL 的詳細資訊。
- `pricingplanmanager:GetSubscription` – 允許主體唯讀存取，以取得有關定價計劃訂閱的詳細資訊。
- `pricingplanmanager:ListSubscriptions` : 允許主體唯讀存取以列出定價計劃訂閱。
- `ec2:DescribeIpamPools` – 允許主體取得 IPAM 集區的詳細資訊。
- `ec2:GetIpamPoolCidrs` – 允許主體取得佈建至 IPAM 集區的 CIDRs。

若要檢視此政策的許可權限，請參閱《AWS 受管政策參考》中的 [CloudFrontReadOnlyAccess](#)。

AWS 受管政策：CloudFrontFullAccess

您可以將 CloudFrontFullAccess 政策連接到 IAM 身分。此政策允許對 CloudFront 資源的管理許可。它還允許與 CloudFront 相關的其他 AWS 服務資源的唯讀許可，並在 CloudFront 主控台中可見。

許可詳細資訊

此政策包含以下許可。

- `s3:ListAllMyBuckets` – 允許主參與者取得所有 Amazon S3 儲存貯體的清單。
- `acm:DescribeCertificate` – 允許主體取得 ACM 憑證的詳細資訊。
- `acm:ListCertificates` – 允許主參與者取得 ACM 憑證清單。
- `acm:RequestCertificate` – 允許主體從 ACM 請求受管憑證。
- `cloudfront:*` – 允許主參與者在所有 CloudFront 資源上執行所有動作。
- `cloudfront-keyvaluestore:*` - 允許主體對鍵值存放區執行所有動作。
- `iam:ListServerCertificates` – 允許主參與者取得存放在 IAM 的伺服器憑證清單。
- `waf:ListWebACLs` – 允許主參與者取得 AWS WAF 的 Web ACL 清單。
- `waf:GetWebACL` – 允許主參與者取得 AWS WAF Web ACL 的詳細資訊。
- `waf:CreateWebACLs` – 允許主體在 中建立 Web ACL AWS WAF。
- `wafv2:ListWebACLs` – 允許主參與者取得 AWS WAF 的 Web ACL 清單。
- `wafv2:GetWebACL` – 允許主參與者取得 AWS WAF Web ACL 的詳細資訊。
- `kinesis:ListStreams` – 允許主參與者取得 Amazon Kinesis 串流的清單。
- `elasticloadbalancing:DescribeLoadBalancers` - 允許主體取得 ELB 中負載平衡器的詳細資訊。
- `kinesis:DescribeStream` – 允許主參與者取得 Kinesis 串流的詳細資訊。
- `iam:ListRoles` – 允許主參與者取得 IAM 中角色的清單。
- `pricingplanmanager:AssociateResourcesToSubscription` - 允許主體將資源與訂閱建立關聯。這可讓資源涵蓋在訂閱的定價計劃中。
- `pricingplanmanager:CancelSubscription` - 允許主體取消現有的訂閱。
- `pricingplanmanager:CancelSubscriptionChange` - 允許主體在套用變更之前取消現有訂閱的待定變更，例如計劃升級。
- `pricingplanmanager:CreateSubscription` - 允許主體建立定價計劃的訂閱。
- `pricingplanmanager:DisassociateResourcesFromSubscription` - 允許主體移除資源與現有訂閱之間的關聯。
- `pricingplanmanager:UpdateSubscription` - 允許主體修改現有的訂閱，例如變更定價計劃。

- `pricingplanmanager:GetSubscription` – 允許主體唯讀存取，以取得有關定價計劃訂閱的詳細資訊。
- `pricingplanmanager:ListSubscriptions` : 允許主體唯讀存取以列出定價計劃訂閱。
- `ec2:DescribeInstances` - 允許主體取得 Amazon EC2 執行個體的詳細資訊。
- `ec2:DescribeInternetGateways` - 允許主體取得 Amazon EC2 中網際網路閘道的詳細資訊。
- `ec2:DescribeIpamPools` – 允許主體取得 IPAM 集區的詳細資訊。
- `ec2:GetIpamPoolCidrs` – 允許主體取得佈建至 IPAM 集區的 CIDRs。

若要檢視此政策的許可權限，請參閱《AWS 受管政策參考》中的 [CloudFrontFullAccess](#)。

Important

如果希望 CloudFront 能夠建立和儲存存取日誌，必須授予額外的許可。如需詳細資訊，請參閱[許可](#)。

AWS 受管政策：AWSCloudFrontLogger

您無法將 AWSCloudFrontLogger 政策連接至您的 IAM 身分。此政策會連接到服務連結角色，而此角色可讓 CloudFront 代表您執行動作。如需詳細資訊，請參閱[the section called “Lambda@Edge 的服務連結角色”](#)。

此政策可讓 CloudFront 將日誌檔案推送至 Amazon CloudWatch。如需此政策中包含之許可的詳細資訊，請參閱 [the section called “CloudFront Logger 的服務連結角色許可”](#)。

若要檢視此政策的許可權限，請參閱《AWS 受管政策參考指南》中的 [AWSCloudFrontLogger](#)。

AWS 受管政策：AWSLambdaReplicator

您無法將 AWSLambdaReplicator 政策連接至您的 IAM 身分。此政策會連接到服務連結角色，而此角色可讓 CloudFront 代表您執行動作。如需詳細資訊，請參閱[the section called “Lambda@Edge 的服務連結角色”](#)。

此政策允許 CloudFront 在 中建立、刪除和停用函數，AWS Lambda 以複寫 Lambda@Edge 函數 AWS 區域。如需此政策中包含之許可的詳細資訊，請參閱 [the section called “Lambda Replicator 的服務連結角色許可”](#)。

若要檢視此政策的許可權限，請參閱《AWS 受管政策參考指南》中的 [AWSLambdaReplicator](#)。

AWS 受管政策：AWSCloudFrontVPCOriginServiceRolePolicy

您無法將 AWSCloudFrontVPCOriginServiceRolePolicy 政策附加至您的 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 CloudFront 代表您執行動作。如需詳細資訊，請參閱[使用適用於 CloudFront 的服務連結角色](#)。

此政策允許 CloudFront 代表您管理 EC2 彈性網路介面和安全群組。如需此政策中包含之許可的詳細資訊，請參閱 [the section called “適用於 CloudFront VPC Origins 的服務連結角色許可權限”](#)。

若要檢視此政策的許可權限，請參閱《AWS 受管政策參考指南》中的 [AWSCloudFrontVPCOriginServiceRolePolicy](#)。

AWS 受管政策的 CloudFront 更新

檢視自此服務開始追蹤這些變更以來 CloudFront AWS 受管政策更新的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 CloudFront [文件歷程記錄](#) 頁面上的 RSS 摘要。

變更	描述	Date
CloudFrontReadOnlyAccess – 更新現有政策	CloudFront 新增了 Amazon EC2 的許可。 新的許可允許主體使用 ec2:DescribeIpamPools 和 ec2:GetIpamPoolCidrs 動作。	2025 年 11 月 24 日
CloudFrontFullAccess – 更新現有政策	CloudFront 新增了 Amazon EC2 的許可。 新的許可允許主體使用 ec2:DescribeIpamPools 和 ec2:GetIpamPoolCidrs 動作。	2025 年 11 月 24 日
CloudFrontFullAccess – 更新現有政策	CloudFront 新增了建立新 AWS WAF ACL 資源的許可，	2025 年 11 月 18 日

變更	描述	Date
	並將建立、更新、刪除和讀取許可新增至 AWS 定價計劃管理員。	
CloudFrontFullAccess – 更新現有政策	CloudFront 新增了建立新 AWS WAF ACL 資源的許可，並將建立、更新、刪除和讀取許可新增至 AWS 定價計劃管理員。	2025 年 11 月 18 日
CloudFrontReadOnlyAccess – 更新現有政策	CloudFront 新增了對 AWS Pricing Plan Manager 的唯讀存取的新許可。	2025 年 11 月 18 日
CloudFrontReadOnlyAccess – 更新現有政策	CloudFront 新增了對 AWS Pricing Plan Manager 的唯讀存取的新許可。	2025 年 11 月 18 日
CloudFrontReadOnlyAccess – 更新現有政策	CloudFront 新增了 ACM 的新許可權限。 新的許可權限可讓主體取得 ACM 憑證的詳細資訊。	2025 年 4 月 28 日
CloudFrontFullAccess – 更新現有政策	CloudFront 新增了 ACM 的新許可權限。 新的許可權限可讓主體取得 ACM 憑證的詳細資訊，以及向 ACM 請求受管憑證。	2025 年 4 月 28 日

變更	描述	Date
CloudFrontFullAccess – 更新現有政策	CloudFront 新增了 Amazon EC2 和 ELB 的新許可。 新的許可可讓 CloudFront 取得 ELB 中負載平衡器和 Amazon EC2 中執行個體和網際網路閘道的詳細資訊。	2024 年 11 月 20 日
AWSCloudFrontVPCOriginServiceRolePolicy : 新政策	CloudFront 新增了新的政策。 此政策允許 CloudFront 代表您管理 EC2 彈性網路介面和安全群組。	2024 年 11 月 20 日
CloudFrontReadOnlyAccess 和 CloudFrontFullAccess – 對兩個現有政策的更新。	CloudFront 針對鍵值存放區新增了新的許可權限。 新許可權限可讓使用者取得有關鍵值存放區的資訊，並對其採取動作。	2023 年 12 月 19 日
CloudFrontReadOnlyAccess - 更新為現有政策	CloudFront 新增了描述 CloudFront 函數的新許可權限。 此許可權限能讓使用者、群組或角色讀取函數的相關資訊和中繼資料，但不能讀取函數的程式碼。	2021 年 9 月 8 日
CloudFront 開始追蹤變更	CloudFront 開始追蹤其 AWS 受管政策的變更。	2021 年 9 月 8 日

使用適用於 CloudFront 的服務連結角色

Amazon CloudFront 使用 AWS Identity and Access Management (IAM) [服務連結角色](#)。服務連結角色是一種獨特的 IAM 角色，可直接連結至 CloudFront。服務連結角色由 CloudFront 預先定義，並包含該服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 CloudFront 更為簡單，因為您不必手動新增必要的許可權限。CloudFront 定義其服務連結角色的許可權限，除非另有定義，否則僅有 CloudFront 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 CloudFront 資源，避免您不小心移除資源的存取許可權限。

如需有關支援服務連結角色的其他服務的資訊，請參閱[AWS 使用 IAM 的服務](#)，並在服務連結角色欄中尋找具有是的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

適用於 CloudFront VPC Origins 的服務連結角色許可權限

CloudFront VPC Origins 使用名為 `AWSServiceRoleForCloudFrontVPCOrigin` 的服務連結角色：可讓 CloudFront 代表您管理 EC2 彈性網路介面和安全群組。

`AWSServiceRoleForCloudFrontVPCOrigin` 服務連結角色信任下列服務以擔任角色：

- `vpcorigin.cloudfront.amazonaws.com`

名為 `AWSCloudFrontVPCOriginServiceRolePolicy` 的角色許可權限政策，允許 CloudFront VPC Origins 在指定資源上完成下列動作：

- 動作：`arn:aws:ec2:*:*:network-interface/*` 上的 `ec2:CreateNetworkInterface`
- 動作：`arn:aws:ec2:*:*:subnet/*` 和 `arn:aws:ec2:*:*:security-group/*` 上的 `ec2:CreateNetworkInterface`
- 動作：`arn:aws:ec2:*:*:security-group/*` 上的 `ec2:CreateSecurityGroup`
- 動作：`arn:aws:ec2:*:*:vpc/*` 上的 `ec2:CreateSecurityGroup`
- 動作：`supported AWS resources that have the aws:ResourceTag/aws.cloudfront.vpcorigin tag enabled` 上的 `ec2:ModifyNetworkInterfaceAttribute`、`ec2>DeleteNetworkInterface`、`ec2>DeleteSecurityGroup` 和 `ec2:UnassignIpv6Addresses`

- 動作：all AWS resources that the actions support 上的 ec2:DescribeNetworkInterfaces、ec2:DescribeSecurityGroups、ec2:DescribeInstances 和 ec2:DescribeAddresses
- 動作：arn:aws:ec2:*:*:security-group/* 和 arn:aws:ec2:*:*:network-interface/* 上的 ec2:CreateTags
- 動作：all AWS resources that the actions support 上的 elasticloadbalancing:DescribeLoadBalancers、elasticloadbalancing:DescribeListeners 和 elasticloadbalancing:DescribeTargetGroups

您必須設定許可，以允許您的使用者、群組或角色建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的[服務連結角色許可](#)。

建立適用於 CloudFront VPC Origins 的服務連結角色

您不需要手動建立服務連結角色，當您在 AWS 管理主控台、AWS CLI 或 AWS API 中建立 VPC 原始伺服器時，CloudFront VPC Origins 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。您建立 VPC 來源時，CloudFront VPC Origins 會再次為您建立服務連結角色。

編輯適用於 CloudFront VPC Origins 的服務連結角色

CloudFront VPC Origins 不允許您編輯 AWSServiceRoleForCloudFrontVPCOrigin 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的[編輯服務連結角色](#)。

刪除 CloudFront VPC Origins 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

Note

如果您嘗試刪除資源時 CloudFront 服務正在使用該角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

刪除 AWSServiceRoleForCloudFrontVPCOrigin 使用的 CloudFront VPC Origins 資源

- 刪除您帳戶中的 VPC 來源資源。
 - CloudFront 可能需要一些時間才能從您的帳戶刪除資源。如果您無法立即刪除服務連結角色，請等待然後再試一次。

使用 IAM 手動刪除服務連結角色

使用 IAM 主控台 AWS CLI、或 AWS API 來刪除 AWSServiceRoleForCloudFrontVPCOrigin 服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [刪除服務連結角色](#)。

CloudFront VPC Origins 服務連結角色的支援區域

CloudFront VPC Origins 不支援在提供服務的每個區域中使用服務連結角色。您可以在下列區域中使用 AWSServiceRoleForCloudFrontVPCOrigin 角色。

區域名稱	區域身分	CloudFront 支援
美國東部 (維吉尼亞北部)	us-east-1	是
美國東部 (俄亥俄)	us-east-2	是
美國西部 (加利佛尼亞北部)	us-west-1 (AZ usw1-az2 除外)	是
美國西部 (奧勒岡)	us-west-2	是
Africa (Cape Town)	af-south-1	是
亞太地區 (香港)	ap-east-1	是
亞太地區 (雅加達)	ap-southeast-3	是
亞太地區 (墨爾本)	ap-southeast-4	是
亞太區域 (孟買)	ap-south-1	是
亞太地區 (海德拉巴)	ap-south-2	是
亞太區域 (大阪)	ap-northeast-3	是

區域名稱	區域身分	CloudFront 支援
亞太區域 (首爾)	ap-northeast-2	是
亞太區域 (新加坡)	ap-southeast-1	是
亞太區域 (雪梨)	ap-southeast-2	是
亞太地區 (東京)	ap-northeast-1 (AZ apne1-az3 除外)	是
加拿大 (中部)	ca-central-1 (AZ cac1-az3 除外)	是
加拿大西部 (卡加利)	ca-west-1	是
歐洲 (法蘭克福)	eu-central-1	是
歐洲 (愛爾蘭)	eu-west-1	是
歐洲 (倫敦)	eu-west-2	是
歐洲 (米蘭)	eu-south-1	是
歐洲 (巴黎)	eu-west-3	是
歐洲 (西班牙)	eu-south-2	是
Europe (Stockholm)	eu-north-1	是
歐洲 (蘇黎世)	eu-central-2	是
以色列 (特拉維夫)	il-central-1	是
Middle East (Bahrain)	me-south-1	是
中東 (阿拉伯聯合大公國)	me-central-1	是
南美洲 (聖保羅)	sa-east-1	是

Amazon CloudFront 身分識別和存取疑難排解

請使用以下資訊來協助您診斷和修復使用 CloudFront 和 IAM 時發生的常見問題。

主題

- [我未獲授權，不得在 CloudFront 中執行動作](#)
- [我未獲授權執行 iam:PassRole](#)
- [我想要允許以外的人員 AWS 帳戶 存取我的 CloudFront 資源](#)

我未獲授權，不得在 CloudFront 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在 mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 `cloudfront:GetWidget` 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
cloudfront:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 `cloudfront:GetWidget` 動作存取 *my-example-widget* 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我未獲授權執行 iam:PassRole

如果您收到錯誤，告知您未獲授權執行 `iam:PassRole` 動作，您的政策必須更新，允許您將角色傳遞給 CloudFront。

有些 AWS 服務可讓您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 marymajor 的 IAM 使用者嘗試使用主控台在 CloudFront 中執行動作時，發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞給服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許以外的人員 AWS 帳戶 存取我的 CloudFront 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 CloudFront 是否支援這些功能，請參閱 [Amazon CloudFront 如何與 IAM 搭配運作](#)。
- 若要了解如何 AWS 帳戶 在您擁有的 資源之間提供存取權，請參閱《[IAM 使用者指南](#)》中的在[您擁有 AWS 帳戶 的另一個 中提供存取權給](#) IAM 使用者。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《[IAM 使用者指南](#)》中的[將存取權提供給第三方 AWS 帳戶 擁有](#)。
- 如需了解如何透過聯合身分提供存取權，請參閱《[IAM 使用者指南](#)》中的[將存取權提供給在外部進行身分驗證的使用者 \(聯合身分\)](#)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《[IAM 使用者指南](#)》中的 [IAM 中的跨帳戶資源存取](#)。

在 Amazon CloudFront 中進行記錄和監控

監控是維護 CloudFront 和 AWS 解決方案可用性和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時更輕鬆地偵錯。AWS 提供數種工具來監控 CloudFront 資源和活動，以及回應潛在事件：

Amazon CloudWatch 警示

您可以使用 CloudWatch 警示觀察單一指標一段指定的時間。如果指標超過指定的閾值，會傳送一則通知至 Amazon SNS 主題或 AWS Auto Scaling 政策。當指標處於特定狀態時，CloudWatch 警示不會叫用動作。必須是狀態已變更並維持了所指定的時間長度，才會呼叫動作。

如需詳細資訊，請參閱[使用 Amazon CloudWatch 監控 CloudFront 指標](#)。

AWS CloudTrail 日誌

CloudTrail 提供由使用者、角色或 CloudFront 中的 AWS 服務所採取的 API 動作記錄。您可以利用 CloudTrail 所收集的資訊來判斷向 CloudFront 發出的 API 請求，以及發出請求的 IP 位址、人員、時間和其他詳細資訊。

如需詳細資訊，請參閱[使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫](#)。

CloudFront 標準日誌和即時存取日誌

CloudFront 日誌提供了對分佈進行請求的詳細記錄。這些日誌對許多應用程式來說都是很有用的資料。舉例來說，日誌資訊在安全與存取稽核中相當實用。

如需詳細資訊，請參閱[存取日誌（標準日誌）](#)及[建立和使用即時存取日誌組態](#)。

邊緣函數日誌

CloudFront Functions 和 Lambda @Edge) 兩種邊緣函數產生的日誌會直接傳送到 Amazon CloudWatch Logs，而且不會由 CloudFront 儲存在任何地方。CloudFront Functions 使用 AWS Identity and Access Management (IAM) [服務連結角色](#)，將客戶產生的日誌直接傳送到您帳戶中的 CloudWatch Logs。

如需詳細資訊，請參閱[邊緣函數日誌](#)。

CloudFront 主控台報告

CloudFront 主控台包含各種報告，包括快取統計資料報告、熱門物件報告和最佳推薦網站報告。大部分的 CloudFront 主控台報告都是以 CloudFront 存取日誌中的資料為基礎，其中包含有關 CloudFront 收到的每個使用者請求的詳細資訊。不過，您不需要啟用存取日誌來查看報告。

如需更多詳細資訊，請參閱[檢視主控台中的 CloudFront 報告](#)。

Amazon CloudFront 的合規驗證

在多個合規計畫中，第三方稽核人員會評估 Amazon CloudFront 的安全與 AWS 合規。這些包括 SOC、PCI、HIPAA 等。

如需特定合規計劃範圍內 AWS 的服務清單，請參閱[AWS 合規計劃範圍內的服務](#)。如需一般資訊，請參閱[AWS 合規計劃](#)。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱[在中下載報告 AWS Artifact](#)。

您使用 CloudFront 時的合規責任，取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。AWS 會提供以下資源協助您處理合規事宜：

- [安全與合規快速入門指南](#) – 這些部署指南討論架構考量，並提供在其中部署以安全與合規為重心的基準環境的步驟 AWS。
- [上的 HIPAA 安全與合規架構 AWS](#) – 此白皮書說明公司如何使用 AWS 來建立符合 HIPAA 規範的應用程式。

HIPAA AWS 合規計劃包含 CloudFront (不包括透過 CloudFront Embedded POPs 交付的內容) 作為符合 HIPAA 資格的服務。如果您有已執行的商業夥伴增補合約 (BAA) AWS, 您可以使用 CloudFront (不包括透過 CloudFront Embedded POPs 交付內容) 來交付包含受保護醫療資訊 (PHI) 的內容。如需詳細資訊, 請參閱 [HIPAA 合規](#)。

- [AWS 合規資源](#) – 此工作手冊和指南集合可能適用於您的產業和位置。
- [AWS Config](#) – AWS 此服務會評估資源組態符合內部實務、產業準則和法規的程度。
- [AWS Security Hub CSPM](#) – AWS 此服務使用安全控制來評估資源組態和安全標準, 以協助您遵守各種合規架構。如需使用 Security Hub CSPM 評估 CloudFront 資源的詳細資訊, 請參閱 AWS Security Hub CSPM 《使用者指南》中的 [Amazon CloudFront 控制項](#)。

CloudFront 合規最佳實務

本節可在您使用 Amazon CloudFront 提供內容時, 提供最佳實務和建議。

如果您根據 [AWS 共同的責任模型](#) 來執行 PCI 合規或 HIPAA 合規的工作負載, 我們建議您記錄過去 365 天的 CloudFront 用量資料, 以供將來稽核之用。若要記錄用量資料, 您可以執行以下操作:

- 啟用 CloudFront 存取日誌。如需詳細資訊, 請參閱 [存取日誌 \(標準日誌\)](#)。
- 擷取傳送到 CloudFront API 的請求。如需詳細資訊, 請參閱 [使用 AWS CloudTrail 以日誌記錄 Amazon CloudFront API 呼叫](#)。

此外, 請參閱以下有關 CloudFront 如何符合 PCI DSS 和 SOC 標準的詳細資訊。

支付卡產業資料安全標準 (PCI DSS)

CloudFront (不包括透過 CloudFront 嵌入式 POP 交付的內容) 支援處理、儲存、傳輸商家或服務供應商的信用卡資料, 並且已驗證符合支付卡產業 (PCI) 資料安全標準 (DSS)。如需 PCI DSS 的詳細資訊, 包括如何請求 AWS PCI 合規套件的副本, 請參閱 [PCI DSS 第 1 級](#)。

為了安全性最佳實務, 我們建議您不要在 CloudFront 節點快取中快取信用卡資訊。例如, 您可以將原始伺服器設定為在包含信用卡資訊 (例如信用卡號碼的最後四碼及持卡人聯絡資訊) 的回應中包含 `Cache-Control: no-cache="####"` 標題。

系統和組織控制 (SOC)

CloudFront (不包括透過 CloudFront 嵌入式 POP 交付的內容) 符合系統和組織控制 (SOC) 措施的規範, 包括 SOC 1、SOC 2 和 SOC 3。SOC 報告是獨立的第三方檢查報告, 示範如何 AWS 實現關鍵

合規控制和目標。這些稽核可確保執行恰當得宜的安全防禦措施與程序，以針對可能影響到客戶與公司資料安全性、機密性和可用性的風險，提供安全防護。這些第三方稽核的結果可在 [AWS SOC 合規網站上](#) 取得，您可以在其中檢視已發佈的報告，以取得支援 AWS 操作和合規之控制項的詳細資訊。

Amazon CloudFront 中的復原功能

AWS 全球基礎設施是以 AWS 區域與可用區域為中心建置的。AWS 區域提供多個分開且隔離的實際可用區域，它們以低延遲、高輸送量和高度備援聯網功能相互連結。透過可用區域，您所設計與操作的應用程式和資料庫，就能夠在可用區域之間自動容錯移轉，而不會發生中斷。可用區域的可用性、容錯能力和擴充能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域與可用區域的詳細資訊，請參閱 [AWS 全球基礎設施](#)。

CloudFront 原始伺服器容錯移轉

除了支援 AWS 全球基礎設施外，Amazon CloudFront 還提供原始伺服器容錯移轉功能，協助支援您的資料復原需求。CloudFront 是一項全球性服務，可透過稱為「節點」或「連接點」(POP) 的資料中心全球網路交付您的內容。如果您的內容尚未在節點中進行快取，CloudFront 會從您識別為內容最終版本來源的原始伺服器中擷取該內容。

您可以設定 CloudFront 的原始伺服器容錯移轉，為特定案例改善復原功能和提高可用性。若要開始使用，請建立一個原始伺服器群組，在其中指定 CloudFront 的主要原始伺服器和輔助原始伺服器。當主要原始伺服器傳回特定的 HTTP 狀態碼失敗回應時，CloudFront 會自動切換到輔助原始伺服器。如需更多詳細資訊，請參閱 [透過 CloudFront 原始伺服器容錯移轉最佳化高可用性](#)。

Amazon CloudFront 的基礎設施安全性

Amazon CloudFront 是一項受管服務，受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及 AWS 如何保護基礎設施的相關資訊，請參閱 [AWS 雲端安全](#)。若要使用基礎設施安全性的最佳實務來設計您的 AWS 環境，請參閱安全支柱 AWS 架構良好的框架中的 [基礎設施保護](#)。

您可使用 AWS 發佈的 API 呼叫透過網路存取 CloudFront。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman) 或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

CloudFront Functions 在 AWS 帳戶之間使用高度安全的隔離屏障，確保客戶環境能夠安全抵禦 Spectre 和 Meltdown 等旁路攻擊。Functions 無法存取或修改屬於其他客戶的資料。Functions 在專用

的 CPU 上運行於專用的單一執行緒程序中，而不需要超執行緒。在任何給定的 CloudFront 節點連接點 (POP) 中，CloudFront Functions 一次只為一位客戶提供服務，而且會在函數執行之間清除所有客戶特定資料。

疑難排解

使用本節來疑難排解您在設定 Amazon CloudFront 分佈內容時可能遇到的常見問題。

每個主題提供有關識別常見問題根本原因的詳細指導，以及解決這些問題的逐步說明。

主題

- [故障診斷分佈問題](#)
- [對 CloudFront 中的錯誤回應狀態碼進行疑難排解](#)
- [負載測試 CloudFront](#)

故障診斷分佈問題

使用此處的資訊協助診斷和修復使用 Amazon CloudFront 分佈來設定網站或應用程式時可能遇到的憑證錯誤、存取遭拒問題或其他常見問題。

主題

- [CloudFront 傳回 Access Denied 錯誤](#)
- [當我嘗試新增替代網域名稱時，CloudFront 傳回 InvalidViewerCertificate 錯誤](#)
- [當我嘗試新增 CNAME 時，CloudFront 會傳回 DNS 記錄設定不正確的錯誤](#)
- [我無法在我的分佈中檢視檔案](#)
- [錯誤訊息：Certificate: <certificate-id> is being used by CloudFront](#)

CloudFront 傳回 Access Denied 錯誤

如果您使用 Amazon S3 儲存貯體做為 CloudFront 分佈的原始伺服器，您可能會在下列範例中看到存取遭拒 (403) 錯誤訊息。

內容

- [您從 Amazon S3 原始伺服器指定遺失的物件](#)
- [您的 Amazon S3 原始伺服器缺少 IAM 許可](#)
- [您使用的憑證無效或沒有足夠的許可](#)

您從 Amazon S3 原始伺服器指定遺失的物件

驗證您儲存貯體中請求的物件是否存在。物件名稱區分大小寫。輸入無效的物件名稱可能會傳回存取遭拒錯誤碼。

例如，如果您遵循 [CloudFront 教學課程](#) 來建立基本分佈，您可以建立 Amazon S3 儲存貯體做為原始伺服器並上傳範例 `index.html` 檔案。

在 Web 瀏覽器中，如果您輸入 `https://d111111abcdef8.cloudfront.net/INDEX.HTML` 而不是 `https://d111111abcdef8.cloudfront.net/index.html`，您可能會看到類似訊息，因為 URL 路徑中的 `index.html` 檔案區分大小寫。

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>22Q367AHT7Y1ABCD</RequestId>
<HostId>
ABCDE/Vg+7PSNa/d/IffQ8Fb92TGQ0KH0ZwG5iEKbc6+e06DdMS1ZW+ryB9GFRIVtS66rSSy6So=
</HostId>
</Error>
```

您的 Amazon S3 原始伺服器缺少 IAM 許可

確認您已選取正確的 Amazon S3 儲存貯體做為原始網域和名稱。原始伺服器 (Amazon S3) 必須具有正確的許可。

如果您未指定正確的許可，檢視器可能會出現 `AccessDenied` 訊息。

當您從 Amazon S3 分發內容，而且也使用 AWS Key Management Service (AWS KMS) 服務端加密 (SSE-KMS) 時，您需要為 KMS 金鑰和 Amazon S3 儲存貯體指定額外的 IAM 許可。您的 CloudFront 分佈需要這些許可才能使用 KMS 金鑰，用於加密原始伺服器 Amazon S3 儲存貯體。

Amazon S3 儲存貯體政策的組態允許 CloudFront 分佈擷取加密物件以進行內容交付。

驗證您的 Amazon S3 儲存貯體和 KMS 金鑰許可

1. 確認您使用的 KMS 金鑰與您的 Amazon S3 儲存貯體用於預設加密的金鑰相同。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [搭配 AWS KMS \(SSE-KMS\) 使用伺服器端加密](#)。
2. 確認儲存貯體中的物件已使用相同的 KMS 金鑰加密。您可以從 Amazon S3 儲存貯體中選取任何物件，並檢查伺服器端加密設定以驗證 KMS 金鑰 ARN。

3. 編輯 Amazon S3 儲存貯體政策，以授予 CloudFront 從 Amazon S3 儲存貯體呼叫 GetObject API 操作的許可。如需使用原始存取控制的 Amazon S3 儲存貯體政策範例，請參閱 [授予 CloudFront 存取 S3 儲存貯體的許可](#)。
4. 編輯 KMS 金鑰政策，以授予 CloudFront 對 Encrypt、Decrypt 和 GenerateDataKey* 執行動作的許可。若要符合最低權限許可，請指定 Condition 元素，允許只有指定的 CloudFront 分佈可以執行列出的動作。您可以自訂現有政策 AWS KMS 的政策。如需 KMS 金鑰政策的範例政策，請參閱 [SSE-KMS](#)。

如果您使用的是原始存取身分 (OAI) 而非 OAC，Amazon S3 儲存貯體的許可會略有不同，因為您授與身分的許可，而不是 AWS 服務。如需詳細資訊，請參閱[授予原始存取身分的許可，以讀取 Amazon S3 儲存貯體中的檔案](#)。

如果您仍然無法檢視分佈中的檔案，請參閱 [我無法在我的分佈中檢視檔案](#)。

您使用的憑證無效或沒有足夠的許可

如果您使用不正確或過期的 AWS SCT 憑證 (存取金鑰和私密金鑰)，或 IAM 角色或使用者缺少在 CloudFront 資源上執行動作所需的許可，則可能會出現存取遭拒錯誤訊息。如需存取遭拒錯誤訊息的詳細資訊，請參閱《IAM 使用者指南》中的[對拒絕存取錯誤訊息進行疑難排解](#)。

如需有關 IAM 如何與 CloudFront 搭配使用的詳細資訊，請參閱 [適用於 Amazon CloudFront 的 Identity and Access Management](#)。

當我嘗試新增替代網域名稱時，CloudFront 傳回 InvalidViewerCertificate 錯誤

若 CloudFront 在您嘗試新增替代網域名稱 (CNAME) 到分佈時傳回 InvalidViewerCertificate 錯誤，請檢閱以下資訊來協助針對問題進行故障診斷。此錯誤可指出必須解決以下其中一個問題，才能成功新增替代網域名稱。

以下錯誤會根據 CloudFront 檢查授權以新增替代網域名稱的順序列出。這可協助您根據 CloudFront 傳回的錯誤，針對問題進行故障診斷；您可以得知哪些驗證檢查已成功完成。

沒有將憑證連接到您的分佈。

若要新增替代網域名稱 (CNAME)，您必須將信任、有效的憑證連接到您的分佈。請檢閱需求、取得足以滿足需求的有效憑證、將它連接到您的分佈，然後再試一次。如需詳細資訊，請參閱 [使用備用網域名稱的需求](#)。

您已連接憑證的憑證鏈結中有太多憑證。

您在憑證鏈結中最多只能有五個憑證。請減少鏈結中的憑證數，然後再試一次。

憑證鏈結中包含一或多個在目前日期無效的憑證。

您已新增憑證的憑證鏈結中有一或多個無效憑證，可能是因為憑證尚未有效或已過期。請檢查您憑證鏈結中憑證內的 Not Valid Before (生效日期) 和 Not Valid After (失效日期) 欄位，確保根據您所列出的日期，所有憑證都是有效的。

您已連接的憑證沒有經過信任的憑證授權單位 (CA) 簽章。

您連接到 CloudFront 以驗證您替代網域名稱的憑證不能是自我簽章憑證。它必須經過信任 CA 的簽章。如需詳細資訊，請參閱 [使用備用網域名稱的需求](#)。

您連接的憑證格式不正確

包含在憑證中的網域名稱和 IP 位址格式，以及憑證本身的格式必須遵循憑證的標準。

發生 CloudFront 內部錯誤。

CloudFront 因內部問題遭到封鎖，無法進行憑證的驗證檢查。在此情況下，CloudFront 會傳回 HTTP 500 狀態碼，指出 CloudFront 連接憑證時發生內部問題。請等待幾分鐘，然後再次嘗試新增替代網域名稱和憑證。

您已連接的憑證並未涵蓋您嘗試新增的替代網域名稱。

針對每個您新增的替代網域名稱，CloudFront 需要您連接來自信任憑證授權單位 (CA) 的有效 SSL/TLS 憑證，且該憑證必須涵蓋網域名稱，以驗證您已獲得授權使用它。請更新您的憑證，以包含能涵蓋您嘗試新增之 CNAME 的網域名稱。如需搭配萬用字元使用網域名稱的詳細資訊和範例，請參閱 [使用備用網域名稱的需求](#)。

當我嘗試新增 CNAME 時，CloudFront 會傳回 DNS 記錄設定不正確的錯誤

當您有指向 CloudFront 分佈的現有萬用字元 DNS 項目時，如果您嘗試新增具有更特定名稱的新 CNAME，可能會遇到下列錯誤：

```
One or more aliases specified for the distribution includes an incorrectly configured DNS record that points to another CloudFront distribution. You must update the DNS record to correct the problem.
```

發生此錯誤是因為 CloudFront 針對 CNAME 查詢 DNS，且萬用字元 DNS 項目解析為另一個分佈。

若要解決此問題，請先建立另一個分佈，然後建立指向新分佈的 DNS 項目。最後，新增更具體的 CNAME。如需如何新增 CNAME 的詳細資訊，請參閱 [新增替代網域名稱](#)。

我無法在我的分佈中檢視檔案

如果無法查看 CloudFront 分佈裡的檔案，請參閱下列一些常見解決方案的主題。

您是否同時註冊 CloudFront 和 Amazon S3？

若要將 Amazon CloudFront 與 Amazon S3 原始伺服器搭配使用，您必須同時註冊 CloudFront 和 Amazon S3。如需註冊 CloudFront 和 Amazon S3 的詳細資訊，請參閱 [設定您的 AWS 帳戶](#)。

Amazon S3 儲存貯體與物件許可是否設定正確？

如果使用的是有 Amazon S3 原始伺服器的 CloudFront，則內容的原始版本會存放在 S3 儲存貯體。若要將內容提供給檢視器，建議您使用 CloudFront Origin Access Control (OAC) 來保護 Amazon S3 儲存貯體存取。這表示您的 S3 儲存貯體只能透過 CloudFront 存取。OAC 會透過 CloudFront 控制檢視器存取和安全交付。如需 OAC 的詳細資訊，請參閱 [the section called “限制對 Amazon S3 原始伺服器的存取”](#)。

如需管理儲存貯體存取的詳細資訊，請參閱《Amazon S3 使用者指南》中的 [封鎖對 Amazon S3 儲存體的公開存取權](#)。

物件屬性與儲存貯體屬性無關。您必須明確授予 Amazon S3 儲存貯體中每個物件的權限。物件不會從儲存貯體繼承屬性，必須獨立於儲存貯體設定物件屬性。

替代網域名稱 (CNAME) 是否正確設定？

如果網域名稱已有現有的 CNAME 記錄，請更新該記錄或將其替換為指向分佈網域名稱的新記錄。

此外，請確保 CNAME 記錄指向分佈的網域名稱，而不是 Amazon S3 儲存貯體。可確認 DNS 系統中的 CNAME 記錄指向分佈的網域名稱。若要執行此作業，請使用像 dig 這樣的 DNS 工具。

以下範例顯示了一個名為 `images.example.com` 的網域名稱的 dig 請求和回應的相關部分。在 ANSWER SECTION，查看包含 CNAME 的列。如果 CNAME 右側的值為 CloudFront 分佈的網域名稱，則網域名稱的 CNAME 記錄設定正確。如果是 Amazon S3 原始伺服器儲存貯體或某些其他網域名稱，則 CNAME 記錄設定便不正確。

```
[prompt]> dig images.example.com

; <<> DiG 9.3.3rc2 <<> images.example.com
;; global options: printcmd
```



```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0
;; QUESTION SECTION:
;images.example.com.      IN      A
;; ANSWER SECTION:
images.example.com. 10800 IN CNAME d1111111abcdef8.cloudfront.net.
...
...
```

如需 CNAME 的詳細資訊，請參閱[透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#)。

是否參考了適用於 CloudFront 分佈的正確 URL？

確保參照的 URL 使用 CloudFront 分佈的網域名稱 (或 CNAME)，而不是 Amazon S3 儲存貯體或自訂原始伺服器。

是否需要協助故障診斷自訂原始伺服器？

如果您需要 AWS 協助您對自訂原始伺服器進行故障診斷，我們可能需要檢查請求中的 X-Amz-Cf-Id 標頭項目。如果尚未記錄這些項目，可能要考慮未來的記錄。如需詳細資訊，請參閱 [the section called “使用 Amazon EC2 \(或其他自訂原始伺服器\)”](#)。如需進一步協助，請參閱 [AWS 支援中心](#)。

錯誤訊息： Certificate: <certificate-id> is being used by CloudFront

問題：嘗試從 IAM 憑證存放區刪除 SSL/TLS 憑證，您會取得訊息「Certificate: <certificate-id> is being used by CloudFront。」。

解決方案：每個 CloudFront 分佈皆必須與預設 CloudFront 憑證或自訂 SSL/TLS 憑證相關聯。在刪除 SSL/TLS 憑證之前，您必須輪換憑證 (將目前的自訂 SSL/TLS 憑證替換成另一個自訂 SSL/TLS 憑證)，或從使用自訂 SSL/TLS 憑證還原為使用預設 CloudFront 憑證。若要進行修復，請完成以下其中一個程序中的步驟：

- [輪換 SSL/TLS 憑證](#)
- [從自訂 SSL/TLS 憑證還原到預設 CloudFront 憑證](#)

對 CloudFront 中的錯誤回應狀態碼進行疑難排解

如果 CloudFront 從原始來源請求一個物件，且原始來源傳回 HTTP 4xx 或 5xx 狀態碼，則 CloudFront 與原始來源之間的通訊會有問題。

本主題也包含使用 Lambda@Edge 或 CloudFront Functions 時這些狀態碼的疑難排解步驟。

下列主題提供這些錯誤回應背後潛在原因的詳細說明，並提供如何診斷和解決基礎問題的逐步指引。

主題

- [HTTP 400 狀態碼 \(錯誤的請求\)](#)
- [HTTP 401 狀態碼 \(未授權\)](#)
- [HTTP 403 狀態碼 \(方法無效\)](#)
- [HTTP 403 狀態碼 \(許可遭拒\)](#)
- [HTTP 404 狀態碼 \(找不到\)](#)
- [HTTP 412 狀態碼 \(先決條件失敗\)](#)
- [HTTP 500 狀態碼 \(內部伺服器錯誤\)](#)
- [HTTP 502 狀態碼 \(無效的閘道\)](#)
- [HTTP 503 狀態碼 \(服務無法使用\)](#)
- [HTTP 504 狀態碼 \(閘道逾時\)](#)

HTTP 400 狀態碼 (錯誤的請求)

當用戶端在請求中傳送一些無效資料時，CloudFront 會傳回 400 錯誤的請求，例如承載或參數中的遺失或不正確的內容。這也可以代表一般用戶端錯誤。

Amazon S3 原始伺服器傳回 400 錯誤

如果您使用 Amazon S3 原始伺服器搭配 CloudFront 分佈，分佈可能會傳送錯誤回應，其中包含 HTTP 狀態碼 400 錯誤的請求，以及類似下列的訊息：

授權標頭格式錯誤；區域 '*<AWS Region>*' 錯誤；預期 '*<AWS Region>*'

例如：

授權標頭格式不正確；區域 'us-east-1' 是錯誤的；應該是 'us-west-2'

在下列情況下可能會發生這個問題：

1. 您的 CloudFront 分佈的來源是 Amazon S3 儲存貯體。
2. 您已將 S3 儲存貯體從一個 AWS 區域移至另一個區域。也就是說，您已刪除 S3 儲存貯體，稍後您建立了一個具有相同儲存貯體名稱的新儲存貯體，但位於與原始 S3 儲存貯體所在位置不同的 AWS 區域中。

若要修正此錯誤，請更新您的 CloudFront 分佈，以便在儲存貯體的目前 AWS 區域中尋找 S3 儲存貯體。

更新您的 CloudFront 分佈

1. 登入 AWS 管理主控台 並開啟位於 的 CloudFront 主控台 <https://console.aws.amazon.com/cloudfront/v4/home>。
2. 選擇產生此錯誤的分佈。
3. 選擇 Origins and Origin Groups (原始伺服器 and 原始伺服器群組)。
4. 尋找您所移動 S3 儲存貯體的原始來源。選取此原始來源旁邊的核取方塊，然後選擇 Edit (編輯)。
5. 請選擇 Yes, Edit (是，編輯)。在選擇 Yes, Edit (是、編輯) 之前，不需要變更任何設定。

當您完成這些步驟時，CloudFront 會重新部署您的分佈。在部署分佈時，您會在上次修改的資料欄下看到部署中狀態。部署完成過一段時間後，您應該不會再收到 AuthorizationHeaderMalformed 錯誤回應。

Application Load Balancer 原始伺服器傳回 400 錯誤

如果您使用 Application Load Balancer 原始伺服器搭配 CloudFront 分佈，400 錯誤的可能原因包括下列項目：

- 用戶端傳送不符合 HTTP 規格的格式錯誤請求。
- 請求標頭超過每個請求行 16 KB、每個單一標頭 16 KB，或整個請求標頭 64 KB 的限制。
- 用戶端在傳送完整請求內文之前關閉了連線。

HTTP 401 狀態碼 (未授權)

401 未經授權的回應狀態碼表示用戶端請求尚未完成，因為其缺少所請求資源的有效身分驗證憑證。此狀態碼會與 HTTP WWW-Authenticate 回應標頭一起傳送，其中包含用戶端在提示使用者進行身分驗證憑證後如何再次請求資源的相關資訊。如需詳細資訊，請參閱 [401 未授權](#)。

在 CloudFront 中，如果您的原始伺服器預期 Authorization 標頭會驗證請求，CloudFront 需要將 Authorization 標頭轉送到原始伺服器，以避免 401 未經授權的錯誤。當 CloudFront 將檢視器請求轉送至您的原始伺服器時，CloudFront 預設會移除某些檢視器標頭，包括 Authorization 標頭。為了確保您的原始伺服器始終收到原始伺服器請求中的 Authorization 標頭，您有以下選項：

- 使用快取政策將 Authorization 標頭新增至快取金鑰。快取金鑰中的所有標頭都會自動包含在原始伺服器請求中。如需詳細資訊，請參閱[使用政策控制快取金鑰](#)。
- 使用將所有檢視器標頭轉寄至原始伺服器的原始伺服器請求政策。您無法在原始伺服器請求政策中個別轉寄 Authorization 標頭，但當您轉寄所有檢視器標頭時，CloudFront 會在檢視器請求中包含 Authorization 標頭。CloudFront 提供此使用案例的受管 AllViewer 原始伺服器請求政策。如需詳細資訊，請參閱[使用受管原始伺服器請求政策](#)。

如需詳細資訊，請參閱[如何設定 CloudFront 將授權標頭轉送至原始伺服器？](#)

HTTP 403 狀態碼 (方法無效)

如果您嘗試使用 CloudFront 分佈中未指定的 HTTP 方法，CloudFront 會傳回 403 (方法無效) 錯誤。您可以為分佈指定下列其中一個選項：

- CloudFront 只轉送 GET 和 HEAD 請求。
- CloudFront 只轉送 GET、HEAD 和 OPTIONS 請求。
- CloudFront 轉送 GET、HEAD、OPTIONS、PUT、PATCH、POST 和 DELETE 請求。(如果您選擇此選項，您可能需要限制存取 Amazon S3 儲存貯體或自訂的原始伺服器，以避免使用者執行您不希望他們執行的操作。例如，您可能不希望使用者具有刪除您原始伺服器物件的許可。

HTTP 403 狀態碼 (許可遭拒)

HTTP 403 錯誤表示用戶端未獲授權存取請求的資源。用戶端了解請求，但無法授權檢視器存取。以下是 CloudFront 傳回此狀態碼時的常見原因：

主題

- [替代 CNAME 設定不正確](#)
- [AWS WAF 在 CloudFront 分佈或原始伺服器設定](#)
- [自訂原始伺服器傳回 403 錯誤](#)
- [Amazon S3 原始伺服器傳回 403 錯誤](#)
- [地理限制傳回 403 錯誤](#)
- [已簽署 URL 或已簽署 Cookie 組態傳回 403 錯誤](#)
- [堆疊分佈導致 403 錯誤](#)

替代 CNAME 設定不正確

確認您已為分佈指定正確的 CNAME。若要使用替代 CNAME 而非預設 CloudFront URL：

1. 在 DNS 中建立 CNAME 記錄，將 CNAME 指向 CloudFront 分佈 URL。
2. 在 CloudFront 分佈組態中新增 CNAME。

如果您建立 DNS 記錄，但未在 CloudFront 分佈組態中新增 CNAME，則請求會傳回 403 錯誤。如需設定自訂 CNAME 的詳細資訊，請參閱 [透過新增備用網域名稱 \(CNAME\) 使用自訂 URL](#)。

AWS WAF 在 CloudFront 分佈或原始伺服器設定

當 AWS WAF 位於用戶端和 CloudFront 之間時，CloudFront 無法區分原始伺服器傳回的 403 錯誤碼和封鎖請求 AWS WAF 時傳回的 403 錯誤碼。

若要尋找 403 狀態碼的來源，請檢查您的 AWS WAF Web 存取控制清單 (ACL) 規則是否有封鎖的請求。如需詳細資訊，請參閱下列主題：

- [AWS WAF Web 存取控制清單 \(Web ACLs\)](#)
- [測試和調整您的 AWS WAF 防護](#)

自訂原始伺服器傳回 403 錯誤

如果您使用的是自訂原始伺服器，且原始伺服器具有自訂防火牆組態，則可能會看到 403 錯誤。若要進行疑難排解，請直接向原始伺服器提出請求。如果您可以在沒有 CloudFront 的情況下複寫錯誤，則原始伺服器會導致 403 錯誤。

如果自訂原始伺服器造成錯誤，請檢查原始伺服器日誌以識別可能導致錯誤的因素。如需詳細資訊，請參閱下列疑難排解主題：

- [如何對 API Gateway 的 HTTP 403 錯誤進行疑難排解？](#)
- [如何對 Application Load Balancer 的 HTTP 403 禁止錯誤進行疑難排解？](#)

Amazon S3 原始伺服器傳回 403 錯誤

您可能會收到 403 錯誤，原因如下：

- CloudFront 無法存取 Amazon S3 儲存貯體。如果您的分佈未啟用原始存取身分 (OAI) 或原始存取控制 (OAC)，且儲存貯體為私有，則可能會發生這種情況。

- 請求 URL 中指定的路徑不正確。
- 請求的物件不存在。
- 主機標頭已使用 REST API 端點轉送。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 [HTTP 託管標頭儲存貯體規格](#)。
- 您設定了自訂錯誤頁面。如需詳細資訊，請參閱[當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤](#)。

地理限制傳回 403 錯誤

如果您啟用了地理限制功能 (有時也稱為地理封鎖) 來防止特定地理位置的使用者存取您透過 CloudFront 分佈所分佈的內容，封鎖使用者時會收到 403 錯誤。

如需詳細資訊，請參閱[限制您內容的地理分佈](#)。

已簽署 URL 或已簽署 Cookie 組態傳回 403 錯誤

如果您為分佈的行為組態啟用限制檢視器存取，則不使用已簽署 Cookie 或已簽署 URL 的請求會導致 403 錯誤。如需詳細資訊，請參閱下列主題：

- [使用已簽署 URL 和已簽署 Cookie 提供私有內容](#)
- [如何對與 CloudFront 中已簽署 URL 或已簽署 Cookie 相關的問題進行疑難排解？](#)

堆疊分佈導致 403 錯誤

如果您在對原始端點的請求鏈中有兩個以上的分佈，CloudFront 會傳回 403 錯誤。我們不建議將一個分佈放在另一個分佈前面。

HTTP 404 狀態碼 (找不到)

當用戶端嘗試存取不存在的資源時，CloudFront 會傳回 404 (找不到) 錯誤。如果您的 CloudFront 分佈收到此錯誤，常見原因如下：

- 資源不存在。
- URL 不正確。
- 傳回 404 的自訂原始伺服器。
- 傳回 404 的自訂錯誤頁面。(任何錯誤碼都可能翻譯為 404。) 如需詳細資訊，請參閱[當您已設定自訂錯誤頁面時，CloudFront 如何處理錯誤](#)。

- 自訂錯誤頁面意外刪除，導致 404，因為請求會尋找已刪除的自訂錯誤頁面。如需詳細資訊，請參閱[您尚未設定自訂錯誤頁面時 CloudFront 如何處理錯誤](#)。
- 原始伺服器路徑不正確。如果填入原始伺服器路徑，其值會附加到瀏覽器每個請求的路徑，然後再將請求轉送到原始伺服器。如需詳細資訊，請參閱[原始伺服器路徑](#)。

HTTP 412 狀態碼 (先決條件失敗)

當目標資源的存取遭拒時，CloudFront 會傳回 412 (先決條件失敗) 錯誤代碼。在某些情況下，伺服器設定為僅在滿足特定條件後才接受請求。如果不符合任何指定的條件，則伺服器不允許用戶端存取指定的資源。反之，伺服器會以 412 錯誤代碼回應。

CloudFront 中 412 錯誤的常見原因包括：

- 未滿足 If-Unmodified-Since 或 If-None-Match 標頭所定義的條件時，對 GET 或 HEAD 以外的方法提出條件請求。在這種情況下，無法提出請求，通常是上傳或修改資源。
- CloudFront [UpdateDistribution](#) API 操作中一或多個請求欄位中的條件評估為 false。

HTTP 500 狀態碼 (內部伺服器錯誤)

HTTP 500 狀態碼 (內部伺服器錯誤) 表示伺服器遇到無法滿足請求的非預期條件。以下是 Amazon CloudFront 中 500 錯誤的常見原因。

主題

- [原始伺服器傳回 500 錯誤至 CloudFront](#)

原始伺服器傳回 500 錯誤至 CloudFront

您的原始伺服器可能會傳回 500 錯誤至 CloudFront。如需詳細資訊，請參閱下列疑難排解主題：

- 如果 Amazon S3 傳回 500 錯誤，請參閱[如何對 Amazon S3 的 HTTP 500 或 503 錯誤進行疑難排解？](#)
- 如果 API Gateway 傳回 500 錯誤，請參閱「[如何對 API Gateway REST API 的 5xx 錯誤進行疑難排解？](#)」。
- 如果 ELB 傳回 500 錯誤，請參閱《Application Load Balancer 使用者指南》中的[HTTP 500：內部伺服器錯誤](#)。

如果上述清單無法解決 500 錯誤，問題可能是 CloudFront 連接點傳回內部伺服器錯誤。您可以聯絡 [支援](#) 尋求協助。

HTTP 502 狀態碼 (無效的閘道)

HTTP 502 狀態碼 (無效的閘道) 表示 CloudFront 無法為請求的物件提供服務，因為它無法連接到原始伺服器。

如果您使用的是 Lambda@Edge，問題可能出自 Lambda 驗證錯誤。如果收到 NonS3OriginDnsError 錯誤碼的 HTTP 502 錯誤，表示存在 DNS 組態問題，導致 CloudFront 無法連線到原始伺服器。

主題

- [CloudFront 與自訂原始伺服器間的 SSL/TLS 溝通失敗](#)
- [原始伺服器無法回應受支援的加密/通訊協定](#)
- [原始伺服器的 SSL/TLS 憑證已過期、無效、已自我簽署，或憑證鏈順序錯誤](#)
- [在原始伺服器設定中原始伺服器無法回應指定的連接埠](#)
- [Lambda 驗證錯誤](#)
- [CloudFront 函數驗證錯誤](#)
- [DNS 錯誤 \(NonS3OriginDnsError\)](#)
- [Application Load Balancer 原始伺服器 502 錯誤](#)
- [API Gateway 原始伺服器 502 錯誤](#)

CloudFront 與自訂原始伺服器間的 SSL/TLS 溝通失敗

如果您使用的自訂原始伺服器需要在 CloudFront 和原始伺服器之間使用 HTTPS，不相符的網域名稱可能會導致錯誤。原始伺服器的 SSL/TLS 憑證必須包含符合您為 CloudFront 分佈指定的原始網域或原始請求 Host 標頭的網域名稱。

如果網域名稱不符合，則 SSL/TLS 的交握便失敗，且 CloudFront 會傳回 HTTP 狀態碼 502 (無效的閘道) 給檢視器，並將 X-Cache 標頭設定成 Error from cloudfront。

若要判斷憑證中的網域名稱是否與分佈或 Host 標頭中的 原始伺服器網域相符，您可以使用線上 SSL 檢查或 OpenSSL。如果網域名稱不相符，您有兩個選項：

- 取得包括適用的網域名稱的新 SSL/TLS 憑證。

如果您使用 AWS Certificate Manager (ACM)，請參閱AWS Certificate Manager [《使用者指南》中的請求公有憑證](#)以請求新的憑證。

- 變更分佈組態，因此 CloudFront 不再嘗試使用 SSL 與您的原始伺服器連線。

線上 SSL 檢查

若要尋找 SSL 測試工具，請搜尋網際網路「線上 ssl 檢查。」通常需要指定網域名稱，且此工具會傳回有關 SSL/TLS 憑證的各種資訊。確認憑證在 Common Name (通用名稱) 或 Subject Alternative Names (主體別名) 欄位中包含您的網域名稱。

OpenSSL

若要協助故障診斷來自 CloudFront 的 HTTP 502 錯誤，您可以使用 OpenSSL 嘗試與原始伺服器建立 SSL/TLS 連線。如果 OpenSSL 無法建立連線，可能表示原始伺服器的 SSL/TLS 組態發生問題。如果 OpenSSL 能夠建立連線，它會傳回原始伺服器憑證的相關資訊，包括憑證的通用名稱 (Subject CN 欄位) 和主體別名 (Subject Alternative Name 欄位)。

使用下列 OpenSSL 命令來測試與原始伺服器的連線 (將####取代為原始伺服器的網域名稱，例如 example.com)：

```
openssl s_client -connect origin domain name:443
```

如果下列為真：

- 原始伺服器支援有多個 SSL/TLS 憑證的多個網域名稱
- 您的分佈設定為將 Host 標頭轉送至原始伺服器

然後將 -servername 選項新增至 OpenSSL 命令，如下列範例所示 (將 *CNAME* 取代為您的分佈中設定的 CNAME)：

```
openssl s_client -connect origin domain name:443 -servername CNAME
```

原始伺服器無法回應受支援的加密/通訊協定

CloudFront 使用密碼加密和通訊協定連線至原始伺服器。如需 CloudFront 支援的密碼加密和通訊協定清單，請參閱 [the section called “CloudFront 和原始伺服器之間支援的通訊協定和密碼”](#)。如果原始伺服器在 SSL/TLS 交換中不回應這些密碼加密或通訊協定的其中之一，則 CloudFront 連線失敗。您可以使用如 [SSL Labs](#) (SSL 實驗室) 等線上工具來驗證原始伺服器是否支援密碼加密和通訊協定。在 Hostname (主機名稱) 欄位中輸入原始伺服器的網域名稱，然後選擇 Submit (提交)。從測試來檢閱

Common names (通用名稱) 和 Alternative names (替代名稱) 欄位，查看它們是否符合原始伺服器的網域名稱。測試完成後，在測試結果中找到 Protocols (通訊協定) 與 Cipher Suites (密碼套件) 區段，查看原始伺服器支援哪些密碼加密或通訊協定。將其與 [the section called “CloudFront 和原始伺服器之間支援的通訊協定和密碼”](#) 的清單進行比較。

原始伺服器的 SSL/TLS 憑證已過期、無效、已自我簽署，或憑證鏈順序錯誤

如果原始伺服器傳回以下內容時，則 CloudFront 失去 TCP 連線，傳回 HTTP 狀態碼 502 (無效的閘道)，且將 X-Cache 標頭設定為 Error from cloudfront：

- 過期的憑證
- 無效的憑證
- 已自我簽署的憑證
- 憑證鏈順序錯誤

Note

若完整的憑證鏈結 (包含中繼憑證) 不存在，CloudFront 便會卸除 TCP 連線。

如需有關在自訂原始伺服器上安裝 SSL/TLS 憑證的詳細資訊，請參閱 [the section called “需要 HTTPS 到自訂原始伺服器”](#)。

在原始伺服器設定中原始伺服器無法回應指定的連接埠

當您在 CloudFront 分佈上建立原始伺服器時，可以設定 CloudFront 用於連線至原始伺服器以傳送 HTTP 和 HTTPS 流量的連接埠。根據預設，這些都是 TCP 80/443。您可以選擇修改這些連接埠。如果原始伺服器為了任何原因拒絕這些連接埠上的流量，或者如果後端伺服器沒有回應連接埠，則 CloudFront 連線失敗。

若要故障診斷這些問題，請檢查基礎設施中執行的任何防火牆和驗證它們不會封鎖受支援的 IP 範圍。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 [AWS IP 位址範圍](#)。此外，驗證 Web 伺服器是否可在原始伺服器上執行。

Lambda 驗證錯誤

如果您使用 Lambda@Edge，HTTP 502 狀態碼可能表示您的 Lambda 函數回應的格式不正確或包含無效的內容。如需針對 Lambda@Edge 錯誤進行故障診斷的詳細資訊，請參閱 [測試和偵錯 Lambda@Edge 函數](#)。

CloudFront 函數驗證錯誤

如果您使用的是 CloudFront 函數，HTTP 502 狀態碼可以表示 CloudFront 函數正在嘗試新增、刪除或變更唯讀標頭。此錯誤不會在測試期間顯示，但會在您部署函數並執行請求後顯示。若要解決此錯誤，請檢查並更新您的 CloudFront 函數。如需詳細資訊，請參閱[更新函數](#)。

DNS 錯誤 (NonS3originDnsError)

具有 NonS3originDnsError 錯誤碼的 HTTP 502 錯誤表示存在 DNS 組態問題，導致 CloudFront 無法連線到原始伺服器。如果您從 CloudFront 收到此錯誤訊息，請確定原始伺服器的 DNS 組態正確且正常運作。

當 CloudFront 收到逾期或不在快取中的物件請求，會向原始伺服器發出請求以取得物件。若要成功向原始伺服器發出請求，則 CloudFront 在原始伺服器網域上執行 DNS 解決方案。您網域的 DNS 服務遇到問題時，CloudFront 無法解決網域名稱以取得 IP 位址，導致 502 錯誤 (NonS3originDnsError)。若要修正此問題，請聯絡您的 DNS 供應商，或者如果您使用 Amazon Route 53，請參閱[為什麼我無法存取使用 Route 53 DNS 服務的網站？](#)

若要進一步疑難排解此問題，請確保原始伺服器根網域的 [authoritative name servers](#) (授權名稱伺服器) 或 Zone Apex (例如 example.com) 正確運作。您可以使用以下命令來尋找適用於 apex 原始伺服器的名稱，使用 [dig](#) 或 [nslookup](#) 等工具：

```
dig OriginAPEXDomainName NS +short
```

```
nslookup -query=NS OriginAPEXDomainName
```

當您有名稱伺服器的名稱時，請使用下列命令針對他們的原始伺服器的網域名稱做查詢，以確保每個回應都有一個答案：

```
dig OriginDomainName @NameServer
```

```
nslookup OriginDomainName NameServer
```

Important

請務必使用連線至公用網際網路的電腦執行此 DNS 疑難排解。CloudFront 使用網際網路上的公有 DNS 來解析原始網域名稱，因此在類似環境下進行疑難排解非常重要。

如果您的原始網域是一個子網域，其 DNS 授權委派給與根網域不同的名稱伺服器，請確定該子網域的名稱伺服器 (NS) 和授權開始 (SOA) 記錄已正確設定。您可以使用類似上述範例的命令來檢查這些記錄。

如需 DNS 的詳細資訊，請參閱 Amazon Route 53 說明文件中的[網域名稱系統 \(DNS\) 概念](#)。

Application Load Balancer 原始伺服器 502 錯誤

如果您使用 Application Load Balancer 做為原始伺服器並收到 502 錯誤，請參閱「[如何對 Application Load Balancer HTTP 502 錯誤進行疑難排解？](#)」。

API Gateway 原始伺服器 502 錯誤

如果您使用 API Gateway 並收到 502 錯誤，請參閱「[如何透過 Lambda 代理整合解決 API Gateway REST API 的 HTTP 502 錯誤？](#)」。

HTTP 503 狀態碼 (服務無法使用)

HTTP 503 狀態碼 (服務無法使用) 通常表示原始伺服器上的效能問題。在極少數情況下，這表示 CloudFront 因為節點上的資源限制，暫時無法滿足請求。

如果您使用的是 Lambda@Edge 或 CloudFront Functions，問題可能是執行錯誤或 Lambda@Edge 超過限制錯誤。

主題

- [原始伺服器沒有足夠的容量來支援請求率](#)
- [CloudFront 由於節點上的資源限制導致錯誤](#)
- [Lambda@Edge 或 CloudFront Function 執行錯誤](#)
- [超過 Lambda@Edge 限制](#)

原始伺服器沒有足夠的容量來支援請求率

當原始伺服器無法使用或無法提供傳入請求時，它會傳回 HTTP 503 狀態碼 (服務無法使用)。然後 CloudFront 將此錯誤傳回給使用者。若要解決這個問題，請嘗試下列解決方案：

- 如果您使用 Amazon S3 做為原始伺服器：
 - 您可以傳送每個分割的 Amazon S3 字首每秒 3,500 個 PUT/COPY/POST/DELETE 和 5,500 個 GET/HEAD 請求。當 Amazon S3 傳回 503 慢速下降回應時，這通常表示針對特定 Amazon S3 字首的請求率過高。

由於請求率適用於 S3 儲存貯體中的每個字首，因此物件應分散到多個字首。隨著字首上的請求率逐漸增加，Amazon S3 會向上擴展以分別處理每個字首的請求。因此，儲存貯體能處理的整體請求率為字首數的倍數。

- 如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的[最佳化 Amazon S3 效能](#)。
- 如果您使用 ELB 做為原始伺服器：
 - 確定您的後端執行個體可以回應運作狀態檢查。
 - 確定您的負載平衡器和後端執行個體可以處理負載。

如需詳細資訊，請參閱：

- [如何對 Classic Load Balancer 的 503 錯誤進行疑難排解？](#)
- [如何對 Application Load Balancer 的 503 \(服務無法使用\) 錯誤進行疑難排解？](#)
- 如果您使用自訂原始伺服器：
 - 檢查應用程式日誌，以確保原始伺服器有足夠的資源，例如記憶體、CPU 和磁碟大小。
 - 如果使用 Amazon EC2 做為後端，請確保執行個體類型都有適當的資源，以符合傳入的請求。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的[執行個體類型](#)。
- 如果您使用 API Gateway：
 - 此錯誤與當 API Gateway API 無法接收回應時的後端整合有關。後端伺服器可能：
 - 超載超過容量，無法處理新的用戶端請求。
 - 臨時維護中。
 - 若要解決此錯誤，請查看您的 API Gateway 應用程式日誌，以判斷後端容量、整合或其他問題。

CloudFront 由於節點上的資源限制導致錯誤

在極少數的情況下，您會收到此錯誤，CloudFront 無法將請求路由到下一個最佳的可用節點，也因此無法滿足請求。在 CloudFront 分佈中執行負載測試時，常見此錯誤。為協助防止此種情況，請遵循[the section called “負載測試 CloudFront”](#) 準則，以避免 503 (容量超過) 錯誤。

如果您的生產環境中發生這種情況，請聯絡 [支援](#)。

Lambda@Edge 或 CloudFront Function 執行錯誤

如果您使用 Lambda@Edge 或 CloudFront Functions，HTTP 503 狀態碼可能表示您的函數傳回執行錯誤。

如需如何識別和解決 Lambda@Edge 錯誤的詳細資訊，請參閱 [測試和偵錯 Lambda@Edge 函數](#)。

如需測試 CloudFront Functions 的詳細資訊，請參閱 [測試函數](#)。

超過 Lambda@Edge 限制

如果您使用 Lambda@Edge，HTTP 503 狀態碼可能表示 Lambda 傳回了錯誤。此錯誤可能由以下其中一項原因造成：

- 函數執行次數超過 Lambda 設定以調節 AWS 區域（並行執行或調用頻率）執行的其中一個配額。
- 此函數已超過 Lambda 函數逾時配額。

如需 Lambda@Edge 配額的詳細資訊，請參閱 [Lambda@Edge 的配額](#)。如需如何識別和解決 Lambda@Edge 錯誤的詳細資訊，請參閱 [the section called “測試和偵錯”](#)。您也可以參閱《AWS Lambda 開發人員指南》中的 [Lambda 服務配額](#)。

HTTP 504 狀態碼 (閘道逾時)

HTTP 504 狀態碼 (閘道逾時) 代表當 CloudFront 將請求轉送回到原始伺服器時 (因為請求的物件未在邊緣快取中) 發生了下列其中一個事件：

- 原始伺服器將 HTTP 504 狀態碼傳回 CloudFront。
- 原始伺服器未在請求逾期之前回應。

當流量受到防火牆或安全群組封鎖而無法傳入原始伺服器，或是無法從網際網路存取該原始伺服器，則 CloudFront 將傳回 HTTP 504 狀態碼。請先查看這些問題。如果可以正常存取，請探索應用程式延遲和伺服器逾時，以協助您找出問題並進行修正。

主題

- [設定原始伺服器的防火牆來允許 CloudFront 流量](#)
- [設定原始伺服器的安全群組來允許 CloudFront 流量](#)
- [設定您的自訂原始伺服器可從網際網路存取](#)
- [尋找和修正原始伺服器上應用程式的延遲回應](#)

設定原始伺服器的防火牆來允許 CloudFront 流量

如果原始伺服器防火牆封鎖 CloudFront 流量，則 CloudFront 會傳回 HTTP 504 狀態碼，所以最好先確認問題不在於此，再檢查其他可能的問題。

用於判斷防火牆是否有問題的方法，將依原始伺服器使用的系統類型而定：

- 如果是在 Linux 伺服器使用 IPTable 防火牆，則您可以搜尋工具和資訊來協助您處理 IPTable。
- 如果您在 Windows 伺服器上使用 Windows 防火牆，請參閱 Microsoft 文件中的[新增或編輯防火牆規則](#) (英文)。

當您在評估原始伺服器的防火牆組態時，請查看是否有任何因已發布 IP 位址範圍而導致 CloudFront 節點流量遭到封鎖的防火牆或安全規則。如需詳細資訊，請參閱[CloudFront 邊緣伺服器的位置和 IP 位址範圍](#)。

如果允許 CloudFront IP 位址範圍連線至您的原始伺服器，請務必更新伺服器的安全性規則以納入變更。您可以訂閱 Amazon SNS 主題，即可在 IP 位址範圍檔案更新時收到通知。在收到通知後，您可以使用程式碼來擷取檔案及進行剖析，並為您的本機環境進行調整。如需詳細資訊，請參閱新聞部落格上的 AWS [透過 Amazon SNS 訂閱 AWS 公有 IP 地址變更](#)。

設定原始伺服器的安全群組來允許 CloudFront 流量

如果原始伺服器使用 Elastic Load Balancing，請檢閱 [ELB 安全群組](#)，確保安全群組允許從 CloudFront 傳入流量。

您也可以使用 AWS Lambda 自動更新您的安全群組，以允許來自 CloudFront 的傳入流量。

設定您的自訂原始伺服器可從網際網路存取

如果 CloudFront 無法存取自訂原始伺服器是因為此伺服器無法從網際網路公開存取，CloudFront 將傳回 HTTP 504 錯誤。

CloudFront 節點是經由網際網路連接到原始伺服器。如果您的自訂原始伺服器位於私有網路，則 CloudFront 無法與其連線。基於上述原因，您不能使用包括[內部 Classic Load Balancer](#) 等私有伺服器做為搭配 CloudFront 的原始伺服器。

若要檢查網際網路流量是否可以連接到您的原始伺服器，請執行以下命令 (其中的 *OriginDomainName* 是指伺服器的網域名稱)：

檢查 HTTPS 流量時：

- nc -zv *OriginDomainName* 443
- telnet *OriginDomainName* 443

檢查 HTTP 流量時：

- nc -zv *OriginDomainName* 80
- telnet *OriginDomainName* 80

尋找和修正原始伺服器上應用程式的延遲回應

導致伺服器逾時發生的原因通常是等候應用程式回應的時間過長，或是設定的逾時值過短。

直接為分佈設定較高 CloudFront 逾時值，就能快速修正，協助避免 HTTP 504 錯誤的發生。但是，我們建議您先排除任何與應用程式和原始伺服器有關的效能和延遲問題。接著您可以設定合理的逾時值，協助避免 HTTP 504 錯誤的發生，並且正確回應使用者。

以下是找出效能問題並加以更正之步驟的概觀：

1. 測量 Web 應用程式的一般負載和高負載延遲 (回應能力)。
2. 視需要新增額外的資源，例如 CPU 記憶體。採取其他步驟來解決問題，例如調校資料庫查詢以配合高負載情況。
3. 視需要調整 CloudFront 分佈的逾時值。

以下是每個步驟的詳細資訊。

測量一般負載和高負載延遲

若要判斷一台或多台後端 Web 應用程式伺服器是否發生高度延遲，請在每台伺服器上執行下列 Linux curl 命令：

```
curl -w "DNS Lookup Time: %{time_namelookup} \nConnect time: %{time_connect} \nTLS Setup: %{time_appconnect} \nRedirect Time: %{time_redirect} \nTime to first byte: %{time_starttransfer} \nTotal time: %{time_total} \n" -o /dev/null https://www.example.com/yourobject
```

Note

如果伺服器執行 Windows，您可以搜尋和下載適用於 Windows 的 Curl，執行類似的命令。

在測量和評估伺服器中執行應用程式的延遲情況時，請注意以下資訊：

- 延遲值會因應每個應用程式而有不同。不過，相對於幾秒鐘或更久時間，幾毫秒的第一個位元組的時間才算正常。
- 如果在一般負載時測出的應用程式延遲時間正常，這時應注意檢視器在高負載情況下仍然可能遇到逾時問題。當出現高需求量時，伺服器可能會延遲回應或毫無回應。為了協助防止高負載延遲問題，請檢查您的伺服器的資源，例如 CPU、記憶體和磁碟讀取和寫入，確保您的伺服器有足夠容量可因應高負載進行擴展。

您可以執行以下 Linux 命令，檢查 Apache 程序所使用的記憶體：

```
watch -n 1 "echo -n 'Apache Processes: ' && ps -C apache2 --no-headers | wc -l && free -m"
```

- 伺服器的高 CPU 使用率可能大幅降低應用程式的效能。如果後端伺服器使用 Amazon EC2 執行個體，請檢閱伺服器的 CloudWatch 指標以檢查 CPU 使用率。如需詳細資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。若是使用您自己的伺服器，請參閱伺服器說明文件，以取得如何檢查 CPU 使用率的指示。
- 請檢查高負載時的其他潛在問題，例如，若有大量請求時，資料庫查詢的執行速度就會變慢。

新增資源，並調校伺服器和資料庫

在評估過應用程式和伺服器的回應能力後，確保您有足夠的資源可供一般流量和高負載情況使用：

- 如果您有自己的伺服器，請根據您的評估，確定伺服器具備足夠的 CPU、記憶體和磁碟空間來處理檢視器的請求。
- 如果使用 Amazon EC2 執行個體做為後端伺服器，請確保該執行個體類型具備執行傳入請求的適當資源。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 [執行個體類型](#)。

此外，請考慮以下調校步驟，以避免發生逾時：

- 如果 Curl 命令傳回的 Time to First Byte 值看似很高，請採取適當步驟以提升應用程式的效能。提升應用程式回應能力將有助於減少逾時錯誤。
- 調校資料庫查詢以確保其可處理大量請求，且不會降低效能。
- 在您的後端伺服器上設定 [keep-alive \(persistent\)](#) (保持活動 (持續)) 連線。當伺服器必須為後續請求或使用者重新建立連線時，此選項可避免此時發生延遲。
- 如果您使用 ELB 做為原始伺服器，504 錯誤的可能原因如下：

- 負載平衡器無法在連線逾時過期 (10 秒) 之前建立對目標的連線。
- 負載平衡器建立了對目標的連線，但目標未在閒置逾時期間經過之前回應。
- 子網路的網路存取控制清單 (ACL) 不允許從目標到暫時性連接埠 (1024-65535) 上負載平衡器節點的流量。
- 目標傳回的內容長度標頭大於實體主體。負載平衡器等候遺失的位元組時逾時。
- 目標是 Lambda 函數，且 Lambda 未在連線逾時期間經過之前回應。

如需減少延遲的詳細資訊，請參閱[如何對 ELB Classic Load Balancer 上的高延遲進行疑難排解？](#)

- 如果您使用 MediaTailor 做為原始伺服器，504 錯誤的可能原因如下：
 - 如果相對 URL 處理不當，MediaTailor 可能會收到來自播放器的格式不正確的 URL。
 - 如果 MediaPackage 是 MediaTailor 的資訊清單原始伺服器，MediaPackage 404 資訊清單錯誤可能會導致 MediaTailor 傳回 504 錯誤。
 - 對 MediaTailor 原始伺服器的請求需要超過 2 秒才能完成。
- 如果您使用 Amazon API Gateway 做為原始伺服器，下列是 504 錯誤的可能原因：
 - 整合請求所需的時間超過您的 API Gateway REST API 最大整合逾時參數。如需詳細資訊，請參閱[如何對 API Gateway 的 API HTTP 504 逾時錯誤進行疑難排解？](#)

視需要調整 CloudFront 逾時值

如果已評估並解決應用程式效能緩慢問題、原始伺服器容量和其他問題，但檢視器仍然遇到 HTTP 504 錯誤，這時您應該考慮變更在分佈中的原始伺服器回應逾時指定時間。如需詳細資訊，請參閱[the section called “回應逾時”](#)。

負載測試 CloudFront

傳統負載測試方法無法與 CloudFront 正常使用，因為 CloudFront 使用 DNS 去平衡負載分散在各處的節點且每個節點中。當用戶端自 CloudFront 請求內容，用戶端收到包含一組 IP 位址的 DNS 回應。如果您透過傳送請求至 DNS 傳回的唯一一個 IP 位址來測試，您僅測試到在一個 CloudFront 節點中的一小部分，無法準確地表示實際的流量模式。根據請求的資料量，以這種方式測試可能會超載與降低 CloudFront 伺服器的一小部分之效能。

CloudFront 被設計為在多個地理區域有不同用戶端 IP 位址和不同 DNS 解析器的檢視器擴展。要執行能準確評估 CloudFront 效能的負載測試，建議您執行下列事項：

- 傳送來自多個地理區域的用戶端請求。

- 設定您的測試，讓每個用戶端發出獨立的 DNS 請求。然後，每個用戶端都會從 DNS 收到一組不同的 IP 位址。
- 對於每個執行請求的用戶端，分發您的用戶端請求至由 DNS 傳回的 IP 位址。這可確保負載分散在 CloudFront 邊緣節點的多部伺服器。

 備註

- 不允許在具有 Lambda@Edge [檢視器請求或檢視器回應觸發程式](#) 的快取行為上執行負載測試。
- 不允許在啟用了 [Origin Shield](#) 的原始伺服器上執行負載測試。

配額

您可以使用下列其中一個選項來請求提高 CloudFront 配額：

- 您可以使用 Service Quotas 主控台或 AWS Command Line Interface。如需詳細資訊，請參閱下列主題：
 - 《Service Quotas 使用者指南》的[請求增加配額](#)
 - AWS CLI 命令參考資料的 [request-service-quota-increase](#)
- 如果 Service Quotas 中沒有 CloudFront 配額，請使用 AWS Support Center Console 建立服務配額增加案例。<https://support.console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase>

CloudFront 受下列配額的限制。

主題

- [一般配額](#)
- [分佈的一般配額](#)
- [政策的一般配額](#)
- [mTLS 和信任存放區的配額](#)
- [CloudFront Functions 上的配額](#)
- [連線函數的配額](#)
- [鍵值存放區的配額](#)
- [Lambda@Edge 的配額](#)
- [SSL 憑證的配額](#)
- [失效的配額](#)
- [金鑰群組的配額](#)
- [WebSocket 連線的配額](#)
- [欄位層級加密的配額](#)
- [Cookie 的配額 \(舊版快取設定\)](#)
- [查詢字串的配額 \(舊版快取設定\)](#)
- [標頭的配額](#)

- [多租用戶配額](#)
- [相關資訊](#)

一般配額

實體	預設配額
每個分佈的資料傳輸率 (此配額不適用於訂閱 CloudFront 固定費率定價計畫的分佈。如需詳細資訊，請參閱 ??? 。)	150 Gbps 申請提高配額
每個分佈的每秒請求數 (此配額不適用於訂閱 CloudFront 固定費率定價計畫的分佈。如需詳細資訊，請參閱 ??? 。)	250,000 申請提高配額
可新增至分佈的標籤	50 申請提高配額
您可以為每個分佈提供服務的檔案	沒有配額
請求或原始回應的最大長度，包括標頭和查詢字串，但不包括內文內容	20,480 位元組
URL 的最大長度	8,192 位元組
每個的即時存取日誌交付組態數目上限 AWS 帳戶	150
每個 Web ACL 的關聯數目上限	100 申請提高配額

分佈的一般配額

實體	預設配額
每個分佈的備用網域名稱 (CNAME)	100

實體	預設配額
如需更多詳細資訊，請參閱 透過新增備用網域名稱 (CNAME) 使用自訂 URL 。	申請提高配額
每個分佈的快取行為	75 申請提高配額
每個原始伺服器的連線嘗試次數 如需更多詳細資訊，請參閱 連線嘗試 。	1-3
每個原始伺服器的連線逾時 如需更多詳細資訊，請參閱 連線逾時 。	1-10 秒
每個來源的回應逾時 這也稱為原始讀取逾時或原始請求逾時。如需詳細資訊，請參閱 回應逾時 。	1-120 秒 申請提高配額
每個原始伺服器的保持連線逾時 如需詳細資訊，請參閱 保持連線逾時 (僅限自訂與 VPC 原始伺服器) 。	1-120 秒 申請提高配額
每個的分佈 AWS 帳戶 如需詳細資訊，請參閱 建立分發 。	500 申請提高配額
每個原始存取控制的分佈	100 申請提高配額
原始端點請求鏈內的分佈 我們不建議將一個分佈放在另一個分佈前面。超過此配額會導致 403 錯誤。	2
檔案壓縮：CloudFront 壓縮的檔案大小範圍 如需更多詳細資訊，請參閱 提供壓縮檔案 。	1,000 到 10,000,000 位元組

實體	預設配額
每個 HTTP GET 回應的可快取檔案大小上限。 只會快取 HTTP GET 的回應。不會快取 POST 或 PUT 的回應。	50 GB
每個 的原始存取控制 AWS 帳戶	100 申請提高配額
每個 的原始存取身分 AWS 帳戶	100 申請提高配額
每個分佈的來源數	100 申請提高配額
每個分佈的原始伺服器群組數	10 申請提高配額
每個 的預備分佈 AWS 帳戶 如需詳細資訊，請參閱 the section called “使用持續部署以安全地測試變更” 。	20 申請提高配額
與相同 VPC 來源建立關聯的分佈	50
每個 的 VPC 原始伺服器 AWS 帳戶	25 申請提高配額
可與單一 Anycast 靜態 IP 清單建立關聯的分佈數量上限。	100 申請提高配額

政策的一般配額

實體	預設配額
每個的自訂快取政策 AWS 帳戶 (不適用於 CloudFront 受管快取政策)	20 申請提高配額
與相同快取政策相關聯的分佈	100
每個快取政策的查詢字串	10 申請提高配額
每個快取政策的標頭	10 申請提高配額
每個快取政策的 Cookie	10 申請提高配額
快取政策中所有查詢字串、標頭和 Cookie 名稱的總合長度	1024
每個的自訂原始伺服器請求政策 AWS 帳戶 (不適用於 CloudFront 受管原始請求政策)	20 申請提高配額
與相同原始伺服器請求政策相關聯的分佈	100
每個原始伺服器請求政策的查詢字串	10 申請提高配額
每個原始伺服器請求政策的標頭	10 申請提高配額
根據原始伺服器請求政策	10 申請提高配額

實體	預設配額
原始伺服器請求政策中所有查詢字串、標頭和 Cookie 名稱的總合長度	1024
每個 的自訂回應標頭政策 AWS 帳戶 (不適用於 CloudFront 受管回應標頭政策)	20 申請提高配額
與相同回應標頭政策相關聯的分佈	100 申請提高配額
每個回應標頭政策的自訂標頭	10 申請提高配額
每個 的持續部署政策 AWS 帳戶	20 申請提高配額

mTLS 和信任存放區的配額

實體	預設配額
每個 的信任存放區 AWS 帳戶	20 申請提高配額
每個信任存放區的分佈	25
CA 套件大小	64 KB 申請提高配額
CA 套件中的憑證大小	16384 申請提高配額
CA 套件中的憑證數量	25

實體	預設配額
憑證鏈深度	4

CloudFront Functions 上的配額

實體	預設配額
每個的函數 AWS 帳戶	100
最大函數大小 此配額不可調整。若要存放 CloudFront Functions 的其他資料，請建立鍵值存放區並新增鍵值對。如需詳細資訊，請參閱 Amazon CloudFront KeyValueCollection 。	10 KB
最大函數記憶體	2 MB
與相同函數相關聯的分佈	100

除了這些配額，使用 CloudFront 函數時還有一些其他限制。如需詳細資訊，請參閱[對 CloudFront Functions 的限制](#)。

連線函數的配額

實體	預設配額
每個的連線函數 AWS 帳戶 如需詳細資訊，請參閱 請求提高連線函數配額 。	0
連線函數大小上限 此配額不可調整。若要存放連線函數的其他資料，請建立索引鍵值存放區並新增索引鍵值對。如需詳細資訊，請參閱 Amazon CloudFront KeyValueCollection 。	10 KB

實體	預設配額
最大連線函數記憶體	2 MB
與相同連線函數相關聯的分佈	100

除了這些配額之外，使用 Connection Functions 時還有一些其他限制。如需詳細資訊，請參閱[關聯 CloudFront 連線函數](#)。

鍵值存放區的配額

實體	預設配額
鍵值對中鍵的大小上限	512 個位元組
鍵值對中值的大小上限	1 KB
您可以在單一 API 請求中更新的鍵值對上限	50 個金鑰或 3 MB 承載，以先達到者為準
個別鍵值存放區的大小上限	5 MB
單一鍵值存放區可以關聯的函數數量上限	10
每個函數的鍵值存放區數量上限	1
每個帳戶的鍵值存放區數量上限	50
	申請提高配額

Lambda@Edge 的配額

一般配額

實體	預設配額
可以具有 Lambda@Edge 函數 AWS 帳戶 的 分佈	500

實體	預設配額
	申請提高配額
每個分佈的 Lambda@Edge 函數	100 申請提高配額
並行執行數	1,000 (每個 AWS 區域) 申請提高配額
 備註 • AWS Lambda 管理 Lambda@Edge 的並行配額。AWS 區域中的所有 Lambda 函數都會共用此配額。 • 建議您在預期檢視器請求來自 AWS 區域的所有中檢閱並行執行配額。此外，Lambda@Edge 函數的每個執行個體每秒最多可處理 10 個請求。調用限制總計是並行限制的 10 倍。	
如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的下列主題： • 了解 Lambda 函數擴展 • Lambda API 請求	
與相同函數相關聯的分佈	500
Lambda 函數和任何包含程式庫的最大壓縮大小	50 MB
每秒 Lambda@Edge 請求數 (每個支援的 AWS 區域)。	10,000
如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的 並行配額 。	

因事件類型而異的配額

實體	檢視器請求和檢視器回應事件	原始伺服器請求和原始伺服器回應事件
函數記憶體大小	128 MB	與 Lambda 配額 相同。
函數逾時。此函數可以對 AWS 區域中的 Amazon S3 儲存貯體、DynamoDB 表格或 Amazon EC2 執行個體等資源進行網路呼叫。	30 秒	30 秒
由 Lambda 函數產生之回應的大小，包括標頭和內文	40 KB	1 MB

備註

- 如需可從 Service Quotas 增加的其他 Lambda@Edge 配額清單，請參閱 AWS 一般參考 中的 [Amazon CloudFront 端點和配額](#)。
- 除了這些配額，使用 Lambda@Edge 函數時還有一些其他限制。如需更多詳細資訊，請參閱 [對 Lambda@Edge 的限制](#)。

SSL 憑證的配額

實體	預設配額
使用專用 IP 地址提供 HTTPS 請求 AWS 帳戶 時，每個的 SSL 憑證（使用 SNI 提供 HTTPS 請求時沒有配額）	2 申請提高配額
如需詳細資訊，請參閱 透過 CloudFront 使用 HTTPS 。	
可與 CloudFront 分佈關聯的 SSL 憑證	1

如果您的 SSL 憑證專門用於檢視器和 CloudFront 之間的 HTTPS 通訊，而且如果您使用 AWS Certificate Manager (ACM) 或 IAM 憑證存放區來佈建或匯入憑證，則適用額外的配額。如需詳細資訊，請參閱[搭配 CloudFront 使用 SSL/TLS 憑證的配額 \(僅限檢視器與 CloudFront 之間的 HTTPS\)](#)。

您還可以匯入 AWS Certificate Manager (ACM) 或上傳到 AWS Identity and Access Management (IAM) 的 SSL 憑證數量也有配額。如需詳細資訊，請參閱[增加 SSL/TLS 憑證的配額](#)。

失效的配額

實體	預設配額
檔案失效：作用中失效請求所允許的最大檔案數量，不包括萬用字元失效	3,000
如需更多詳細資訊，請參閱 使檔案失效以移除內容 。	
檔案失效：允許作用中萬用字元失效的最大數量	15
檔案失效：一個萬用字元失效可以處理的檔案數量上限	沒有配額

金鑰群組的配額

實體	預設配額
單一金鑰群組中的公有金鑰	5
	申請提高配額
與單一快取行為相關聯的金鑰群組	4
	申請提高配額
每個的金鑰群組 AWS 帳戶	10
	申請提高配額
與單一金鑰群組相關聯的分佈	100
	申請提高配額

WebSocket 連線的配額

實體	預設配額
原始伺服器回應逾時 (閒置逾時)	10 分鐘
	如果 CloudFront 在過去 10 分鐘內未偵測到從原始伺服器傳送到用戶端的任何位元組，則會將連線假設為閒置並關閉。

欄位層級加密的配額

實體	預設配額
要加密的欄位的最大長度	16 KB
如需更多詳細資訊，請參閱 使用欄位層級加密來協助保護敏感資料 。	
設定欄位層級加密時，請求內文中的欄位數目上限	10
設定欄位層級加密時，請求內文中的最大長度	1 MB
可與一個相關聯的欄位層級加密組態數目上限 AWS 帳戶	10
可與一個相關聯的欄位層級加密設定檔數目上限 AWS 帳戶	10
可新增到一個 AWS 帳戶的最大公有金鑰數量	10
可在一個設定檔中指定的最大加密欄位數	10
可以與欄位層級加密組態關聯的最大 CloudFront 分佈數	20
可包含在欄位層級加密組態中的查詢參數設定檔對應之最大數量	5

Cookie 的配額 (舊版快取設定)

這些配額適用於 CloudFront 的舊版快取設定。建議您使用[快取政策](#)或[原始伺服器請求政策](#)，而不是舊版設定。

實體	預設配額
每個快取行為的 Cookie	10
如需更多詳細資訊，請參閱 根據 Cookie 快取內容 。	申請提高配額
列入 Ccookie 名稱中的總位元組數 (如果配置 CloudFront 以將所有 Cookie 轉送至原始伺服器，則不適用)	512 減去 Cookies 的數量

查詢字串的配額 (舊版快取設定)

這些配額適用於 CloudFront 的舊版快取設定。建議您使用[快取政策](#)或[原始伺服器請求政策](#)，而不是舊版設定。

實體	預設配額
查詢字串中的字元數目上限	128 個字元
相同參數中所有查詢字串的最大總字元數	512 個字元
每個快取行為的查詢字串	10
如需更多詳細資訊，請參閱 根據查詢字串參數快取內容 。	申請提高配額

標頭的配額

實體	預設配額
每個快取行為的標頭 (舊版快取設定)	10
如需詳細資訊，請參閱 the section called “根據請求標頭快取內容” 。	申請提高配額

實體	預設配額
每個快取行為的轉送標頭	25 申請提高配額
自訂標頭：您可以配置 CloudFront 新增到原始伺服器請求的自訂標頭的最大數量 如需詳細資訊，請參閱 the section called “將自訂標頭新增到原始伺服器請求” 。	30 申請提高配額
自訂標頭：您可以新增至回應標頭政策的最大自訂標頭數量	10 申請提高配額
自訂標頭：標頭名稱的最大長度	256 個字元
自訂標頭：標頭值的最大長度	1,783 個字元
自訂標頭：所有標頭值和名稱組合的最大長度	10,240 個字元
Content-Security-Policy 標頭值的長度上限	1,783 個字元 申請提高配額
CORS (Access-Control-Allow-Origin) 標頭值的長度上限	1,783 個字元

多租用戶配額

實體	預設配額
每個的分佈租用戶數量上限 AWS 帳戶	10,000 申請提高配額
每個的多租用戶分佈數量上限 AWS 帳戶	20 申請提高配額

實體	預設配額
每個的連線群組數目上限 AWS 帳戶	100 申請提高配額
每個分佈租用戶的別名數目上限	100 申請提高配額
每個分佈租用戶的參數數目上限	5 申請提高配額
每個多租用戶分佈的參數數目上限	5 申請提高配額
多租用戶分佈中欄位的參數數目上限	2 申請提高配額
每個 Anycast 靜態 IP 清單的連線群組數量上限	5 申請提高配額

如需分佈多租用戶分佈的相關資訊，請參閱 [了解多租用戶分佈的運作方式](#)。

相關資訊

如需詳細資訊，請參閱《AWS 一般參考》中的 [Amazon CloudFront 端點和配額](#)。

CloudFront AWS SDKs的程式碼範例

下列程式碼範例示範如何搭配 AWS 軟體開發套件 (SDK) 使用 CloudFront。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- [CloudFront AWS SDKs的基本範例](#)
 - [CloudFront AWS SDKs的動作](#)
 - [CreateDistribution 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateFunction 搭配 AWS SDK 使用](#)
 - [搭配使用 CreateInvalidation 與 CLI](#)
 - [CreateKeyGroup 搭配 AWS SDK 使用](#)
 - [CreatePublicKey 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteDistribution 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 GetCloudFrontOriginAccessIdentity 與 CLI](#)
 - [搭配使用 GetCloudFrontOriginAccessIdentityConfig 與 CLI](#)
 - [搭配使用 GetDistribution 與 CLI](#)
 - [GetDistributionConfig 搭配 AWS SDK 或 CLI 使用](#)
 - [搭配使用 ListCloudFrontOriginAccessIdentities 與 CLI](#)
 - [ListDistributions 搭配 AWS SDK 或 CLI 使用](#)
 - [UpdateDistribution 搭配 AWS SDK 或 CLI 使用](#)
 - [CloudFront AWS SDKs的案例](#)
 - [建立 SaaS 管理員資源 AWS SDK](#)
 - [使用 AWS SDK 刪除 CloudFront 簽署資源](#)
 - [使用 CLI 開始使用基本 CloudFront 分佈](#)
 - [使用 AWS SDK 建立簽章URLs 和 Cookie](#)

- [CloudFront 的 CloudFront Functions 範例](#)

- [將 HTTP 安全標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將 CORS 標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將快取控制標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將真正的用戶端 IP 標頭新增至 CloudFront Functions 檢視器請求事件](#)
- [將原始伺服器標頭新增至 CloudFront Functions 檢視器請求事件](#)
- [在 CloudFront Functions 檢視器請求事件中，將 index.html 新增至沒有檔案名稱的請求 URL](#)
- [標準化 CloudFront Functions 檢視器請求中的查詢字串參數](#)
- [重新導向至 CloudFront Functions 檢視器請求事件中的新 URL](#)
- [根據 CloudFront Functions 檢視器請求事件的 KeyValueStore 組態重寫請求 URI](#)
- [在 CloudFront Functions 檢視器請求事件中，將請求路由到更接近檢視器的原始伺服器](#)
- [在 CloudFront Functions 檢視器請求中使用鍵/值對](#)
- [在 CloudFront Functions 檢視器請求中驗證簡單字符](#)

CloudFront AWS SDKs的基本範例

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon CloudFront 基本功能。

範例

- [CloudFront AWS SDKs的動作](#)

- [CreateDistribution 搭配 AWS SDK 或 CLI 使用](#)
- [CreateFunction 搭配 AWS SDK 使用](#)
- [搭配使用 CreateInvalidation 與 CLI](#)
- [CreateKeyGroup 搭配 AWS SDK 使用](#)
- [CreatePublicKey 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteDistribution 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetCloudFrontOriginAccessIdentity 與 CLI](#)
- [搭配使用 GetCloudFrontOriginAccessIdentityConfig 與 CLI](#)
- [搭配使用 GetDistribution 與 CLI](#)
- [GetDistributionConfig 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListCloudFrontOriginAccessIdentities 與 CLI](#)

- [ListDistributions 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateDistribution 搭配 AWS SDK 或 CLI 使用](#)

CloudFront AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 CloudFront 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 CloudFront API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 [CloudFront AWS SDKs的案例](#) 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《[Amazon CloudFront API 參考](#)》。

範例

- [CreateDistribution 搭配 AWS SDK 或 CLI 使用](#)
- [CreateFunction 搭配 AWS SDK 使用](#)
- [搭配使用 CreateInvalidation 與 CLI](#)
- [CreateKeyGroup 搭配 AWS SDK 使用](#)
- [CreatePublicKey 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteDistribution 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 GetCloudFrontOriginAccessIdentity 與 CLI](#)
- [搭配使用 GetCloudFrontOriginAccessIdentityConfig 與 CLI](#)
- [搭配使用 GetDistribution 與 CLI](#)
- [GetDistributionConfig 搭配 AWS SDK 或 CLI 使用](#)
- [搭配使用 ListCloudFrontOriginAccessIdentities 與 CLI](#)
- [ListDistributions 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateDistribution 搭配 AWS SDK 或 CLI 使用](#)

CreateDistribution 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateDistribution。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [建立多租用戶分佈和分佈租用戶](#)
- [開始使用 CloudFront](#)

CLI

AWS CLI

範例 1：建立 CloudFront 分佈

下列 create-distribution 範例會使用命令列引數為名為 amzn-s3-demo-bucket 的 S3 儲存貯體建立分佈，並將 index.html 指定為預設根物件。

```
aws cloudfront create-distribution \  
  --origin-domain-name amzn-s3-demo-bucket.s3.amazonaws.com \  
  --default-root-object index.html
```

輸出：

```
{  
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/  
EMLARXS9EXAMPLE",  
  "ETag": "E9LHASXEXAMPLE",  
  "Distribution": {  
    "Id": "EMLARXS9EXAMPLE",  
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",  
    "Status": "InProgress",  
    "LastModifiedTime": "2019-11-22T00:55:15.705Z",  
    "InProgressInvalidationBatches": 0,  
    "DomainName": "d1111111abcdef8.cloudfront.net",  
    "ActiveTrustedSigners": {  
      "Enabled": false,  
      "Quantity": 0  
    },  
    "DistributionConfig": {  
      "CallerReference": "cli-example",  
      "Aliases": {  
        "Quantity": 0  
      },  
      "DefaultRootObject": "index.html",  
      "Origins": {  
        "Quantity": 1,  
        "Items": [  

```

```

        {
            "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
            "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
            "OriginPath": "",
            "CustomHeaders": {
                "Quantity": 0
            },
            "S3OriginConfig": {
                "OriginAccessIdentity": ""
            }
        }
    ],
    "OriginGroups": {
        "Quantity": 0
    },
    "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
        "ForwardedValues": {
            "QueryString": false,
            "Cookies": {
                "Forward": "none"
            },
            "Headers": {
                "Quantity": 0
            },
            "QueryStringCacheKeys": {
                "Quantity": 0
            }
        },
        "TrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "ViewerProtocolPolicy": "allow-all",
        "MinTTL": 0,
        "AllowedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ],
            "CachedMethods": {

```

```
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ]
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
        "Quantity": 0
    },
    "FieldLevelEncryptionId": "",
},
"CacheBehaviors": {
    "Quantity": 0
},
"CustomErrorResponses": {
    "Quantity": 0
},
"Comment": "",
"Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
},
"PriceClass": "PriceClass_All",
"Enabled": true,
"ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
},
"Restrictions": {
    "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
    }
},
"WebACLId": "",
"HttpVersion": "http2",
```



```
        "IsIPv6Enabled": true
      }
    }
  }
}
```

範例 2：使用 JSON 檔案建立 CloudFront 分佈

下列 create-distribution 範例會使用 JSON 檔案為名為 amzn-s3-demo-bucket 的 S3 儲存貯體建立分佈，並將 index.html 指定為預設根物件。

```
aws cloudfront create-distribution \  
  --distribution-config file://dist-config.json
```

dist-config.json 的內容：

```
{  
  "CallerReference": "cli-example",  
  "Aliases": {  
    "Quantity": 0  
  },  
  "DefaultRootObject": "index.html",  
  "Origins": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",  
        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",  
        "OriginPath": "",  
        "CustomHeaders": {  
          "Quantity": 0  
        },  
        "S3OriginConfig": {  
          "OriginAccessIdentity": ""  
        }  
      }  
    ]  
  },  
  "OriginGroups": {  
    "Quantity": 0  
  },  
  "DefaultCacheBehavior": {  
    "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",  
    "ForwardedValues": {
```

```
    "QueryString": false,
    "Cookies": {
      "Forward": "none"
    },
    "Headers": {
      "Quantity": 0
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "MinTTL": 0,
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
    "CachedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ]
    }
  },
  "SmoothStreaming": false,
  "DefaultTTL": 86400,
  "MaxTTL": 31536000,
  "Compress": false,
  "LambdaFunctionAssociations": {
    "Quantity": 0
  },
  "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
```

```
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": true,
  "ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
  },
  "Restrictions": {
    "GeoRestriction": {
      "RestrictionType": "none",
      "Quantity": 0
    }
  },
  "WebACLId": "",
  "HttpVersion": "http2",
  "IsIPV6Enabled": true
}
```

如需範例輸出，請參閱範例 1。

範例 3：使用憑證建立 CloudFront 多租用戶分佈

下列 create-distribution 範例會建立具有多租用戶支援的 CloudFront 分佈，並指定 TLS 憑證。

```
aws cloudfront create-distribution \
  --distribution-config file://dist-config.json
```

dist-config.json 的內容：

```
{
  "CallerReference": "cli-example-with-cert",
  "Comment": "CLI example distribution",
  "DefaultRootObject": "index.html",
```

```

"Origins": {
  "Quantity": 1,
  "Items": [
    {
      "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
      "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
      "OriginPath": "/{{tenantName}}",
      "CustomHeaders": {
        "Quantity": 0
      },
      "S3OriginConfig": {
        "OriginAccessIdentity": ""
      }
    }
  ]
},
"DefaultCacheBehavior": {
  "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
  "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
  "ViewerProtocolPolicy": "allow-all",
  "AllowedMethods": {
    "Quantity": 2,
    "Items": ["HEAD", "GET"],
    "CachedMethods": {
      "Quantity": 2,
      "Items": ["HEAD", "GET"]
    }
  }
},
"Enabled": true,
"ViewerCertificate": {
  "ACMCertificateArn": "arn:aws:acm:us-east-1:123456789012:certificate/191306a1-db01-49ca-90ef-fc414ee5dabc",
  "SSLSupportMethod": "sni-only"
},
"HttpVersion": "http2",
"ConnectionMode": "tenant-only",
"TenantConfig": {
  "ParameterDefinitions": [
    {
      "Name": "tenantName",
      "Definition": {
        "StringSchema": {
          "Comment": "tenantName parameter",

```

```

        "DefaultValue": "root",
        "Required": false
      }
    }
  ]
}

```

輸出：

```

{
  "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E1HVIAU7UABC",
  "ETag": "E20LT7R1BABC",
  "Distribution": {
    "Id": "E1HVIAU7U12ABC",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/E1HVIAU7U12ABC",
    "Status": "InProgress",
    "LastModifiedTime": "2025-07-10T20:33:31.117000+00:00",
    "InProgressInvalidationBatches": 0,
    "DomainName": "example.com",
    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ActiveTrustedKeyGroups": {
      "Enabled": false,
      "Quantity": 0
    },
    "DistributionConfig": {
      "CallerReference": "cli-example-with-cert",
      "DefaultRootObject": "index.html",
      "Origins": {
        "Quantity": 1,
        "Items": [
          {
            "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
            "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
            "OriginPath": "/{{tenantName}}",
            "CustomHeaders": {
              "Quantity": 0
            }
          }
        ]
      }
    }
  }
}

```

```

        },
        "S3OriginConfig": {
            "OriginAccessIdentity": ""
        },
        "ConnectionAttempts": 3,
        "ConnectionTimeout": 10,
        "OriginShield": {
            "Enabled": false
        },
        "OriginAccessControlId": ""
    }
]
},
"OriginGroups": {
    "Quantity": 0
},
"DefaultCacheBehavior": {
    "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
    "TrustedKeyGroups": {
        "Enabled": false,
        "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "AllowedMethods": {
        "Quantity": 2,
        "Items": ["HEAD", "GET"],
        "CachedMethods": {
            "Quantity": 2,
            "Items": ["HEAD", "GET"]
        }
    },
    "Compress": false,
    "LambdaFunctionAssociations": {
        "Quantity": 0
    },
    "FunctionAssociations": {
        "Quantity": 0
    },
    "FieldLevelEncryptionId": "",
    "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
    "GrpcConfig": {
        "Enabled": false
    }
}

```

```

    },
    "CacheBehaviors": {
      "Quantity": 0
    },
    "CustomErrorResponses": {
      "Quantity": 0
    },
    "Comment": "CLI example distribution",
    "Logging": {
      "Enabled": false,
      "IncludeCookies": false,
      "Bucket": "",
      "Prefix": ""
    },
    "Enabled": true,
    "ViewerCertificate": {
      "CloudFrontDefaultCertificate": false,
      "ACMCertificateArn": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
      "SSLSupportMethod": "sni-only",
      "MinimumProtocolVersion": "TLSv1.2_2021",
      "Certificate": "arn:aws:acm:us-
east-1:123456789012:certificate/1954f095-11b6-4daf-9952-0c308a00abc",
      "CertificateSource": "acm"
    },
    "Restrictions": {
      "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
      }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "TenantConfig": {
      "ParameterDefinitions": [
        {
          "Name": "tenantName",
          "Definition": {
            "StringSchema": {
              "Comment": "tenantName parameter",
              "DefaultValue": "root",
              "Required": false
            }
          }
        }
      ]
    }
  }

```

```

        }
      ]
    },
    "ConnectionMode": "tenant-only"
  }
}
}

```

如需詳細資訊，請參閱《Amazon CloudFront 開發人員指南》中的[使用分佈](#)。

範例 4：建立不含憑證的 CloudFront 多租用戶分佈

下列 create-distribution 範例會建立具有多租用戶支援、但不含 TLS 憑證的 CloudFront 分佈。

```
aws cloudfront create-distribution \
  --distribution-config file://dist-config.json
```

dist-config.json 的內容：

```

{
  "CallerReference": "cli-example",
  "Comment": "CLI example distribution",
  "DefaultRootObject": "index.html",
  "Origins": {
    "Quantity": 1,
    "Items": [
      {
        "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
        "OriginPath": "/{{tenantName}}",
        "CustomHeaders": {
          "Quantity": 0
        },
        "S3OriginConfig": {
          "OriginAccessIdentity": ""
        }
      }
    ]
  },
  "DefaultCacheBehavior": {
    "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
    "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",

```



```

    "ViewerProtocolPolicy": "allow-all",
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "Enabled": true,
    "HttpVersion": "http2",
    "ConnectionMode": "tenant-only",
    "TenantConfig": {
      "ParameterDefinitions": [
        {
          "Name": "tenantName",
          "Definition": {
            "StringSchema": {
              "Comment": "tenantName parameter",
              "DefaultValue": "root",
              "Required": false
            }
          }
        }
      ]
    }
  }
}

```

輸出：

```

{
  "Location": "https://cloudfront.amazonaws.com/2020-05-31/distribution/
E2GJ5J9QN12ABC",
  "ETag": "E37YLVVQIABC",
  "Distribution": {
    "Id": "E2GJ5J9QNABC",

```

```

"ARN": "arn:aws:cloudfront::123456789012:distribution/E2GJ5J9QN12ABC",
>Status": "InProgress",
>LastModifiedTime": "2025-07-10T20:35:20.565000+00:00",
>InProgressInvalidationBatches": 0,
>DomainName": "example.com",
>ActiveTrustedSigners": {
>    "Enabled": false,
>    "Quantity": 0
>},
>ActiveTrustedKeyGroups": {
>    "Enabled": false,
>    "Quantity": 0
>},
>DistributionConfig": {
>    "CallerReference": "cli-example-no-cert",
>    "DefaultRootObject": "index.html",
>    "Origins": {
>        "Quantity": 1,
>        "Items": [
>            {
>                "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
>                "DomainName": "amzn-s3-demo-bucket.s3.us-
east-1.amazonaws.com",
>                "OriginPath": "/{{tenantName}}",
>                "CustomHeaders": {
>                    "Quantity": 0
>                },
>                "S3OriginConfig": {
>                    "OriginAccessIdentity": ""
>                },
>                "ConnectionAttempts": 3,
>                "ConnectionTimeout": 10,
>                "OriginShield": {
>                    "Enabled": false
>                },
>                "OriginAccessControlId": ""
>            }
>        ]
>    },
>    "OriginGroups": {
>        "Quantity": 0
>    },
>    "DefaultCacheBehavior": {

```

```
    "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
    "TrustedKeyGroups": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": "",
    "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e5ABC",
    "GrpcConfig": {
      "Enabled": false
    }
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "CLI example distribution",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
```

```

        "Bucket": "",
        "Prefix": ""
    },
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "SSLSupportMethod": "sni-only",
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "TenantConfig": {
        "ParameterDefinitions": [
            {
                "Name": "tenantName",
                "Definition": {
                    "StringSchema": {
                        "Comment": "tenantName parameter",
                        "DefaultValue": "root",
                        "Required": false
                    }
                }
            }
        ]
    },
    "ConnectionMode": "tenant-only"
}

```

如需詳細資訊，請參閱《Amazon CloudFront 開發人員指南》中的[設定分佈](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateDistribution](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

以下範例使用 Amazon Simple Storage Service (Amazon S3) 儲存貯體做為內容來源。

建立分佈之後，程式碼會建立 [CloudFrontWaiter](#) 以等待部署分佈，然後再傳回分佈。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.ItemSelection;
import software.amazon.awssdk.services.cloudfront.model.Method;
import software.amazon.awssdk.services.cloudfront.model.ViewerProtocolPolicy;
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;
import software.amazon.awssdk.services.s3.S3Client;

import java.time.Instant;

public class CreateDistribution {

    private static final Logger logger =
        LoggerFactory.getLogger(CreateDistribution.class);

    public static Distribution createDistribution(CloudFrontClient
        cloudFrontClient, S3Client s3Client,
            final String bucketName, final String keyGroupId, final
        String originAccessControlId) {

        final String region = s3Client.headBucket(b ->
            b.bucket(bucketName)).sdkHttpResponse().headers()
```

```

        .get("x-amz-bucket-region").get(0);
        final String originDomain = bucketName + ".s3." + region +
".amazonaws.com";
        String originId = originDomain; // Use the originDomain value for
the originId.

        // The service API requires some deprecated methods, such as
        // DefaultCacheBehavior.Builder#minTTL and #forwardedValue.
        CreateDistributionResponse createDistResponse =
cloudFrontClient.createDistribution(builder -> builder
        .distributionConfig(b1 -> b1
            .origins(b2 -> b2
                .quantity(1)
                .items(b3 -> b3

.domainName(originDomain)

.id(originId)

.s3OriginConfig(builder4 -> builder4

        .originAccessIdentity(

            ""))

        .originAccessControlId(

            originAccessControlId)))

        .defaultCacheBehavior(b2 -> b2

.viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)

.targetOriginId(originId)

        .minTTL(200L)

.forwardedValues(b5 -> b5

.cookies(cp -> cp

        .forward(ItemSelection.NONE))

.queryString(true))

.trustedKeyGroups(b3 -> b3

```

```

    .quantity(1)

    .items(keyGroupId)

    .enabled(true))

    .allowedMethods(b4 -> b4

    .quantity(2)

    .items(Method.HEAD, Method.GET)

    .cachedMethods(b5 -> b5

        .quantity(2)

        .items(Method.HEAD,

            Method.GET))))

    .cacheBehaviors(b -> b

        .quantity(1)

        .items(b2 -> b2

    .pathPattern("/index.html")

    .viewerProtocolPolicy(

        ViewerProtocolPolicy.ALLOW_ALL)

    .targetOriginId(originId)

    .trustedKeyGroups(b3 -> b3

        .quantity(1)

        .items(keyGroupId)

        .enabled(true))

    .minTTL(200L)

    .forwardedValues(b4 -> b4

```

```

        .cookies(cp -> cp

                .forward(ItemSelection.NONE))

        .queryString(true))

.allowedMethods(b5 -> b5.quantity(2)

        .items(Method.HEAD,

                Method.GET)

        .cachedMethods(b6 -> b6

                .quantity(2)

                .items(Method.HEAD,

                        Method.GET))))

                .enabled(true)

                .comment("Distribution built with
java")

.callerReference(Instant.now().toString())));

        final Distribution distribution =
createDistResponse.distribution();
        logger.info("Distribution created. DomainName: [{}] Id: [{}]",
distribution.domainName(),

                distribution.id());
        logger.info("Waiting for distribution to be deployed ...");
        try (CloudFrontWaiter cfWaiter =
CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
                ResponseOrException<GetDistributionResponse>
responseOrException = cfWaiter

                        .waitUntilDistributionDeployed(builder ->
builder.id(distribution.id()))

                        .matched();
                responseOrException.response()

                        .orElseThrow(() -> new
RuntimeException("Distribution not created"));
                logger.info("Distribution deployed. DomainName: [{}] Id:
[{}]", distribution.domainName(),

```



```

        distribution.id());
    }
    return distribution;
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateDistribution](#)。

PowerShell

Tools for PowerShell V4

範例 1：建立以日誌記錄和快取設定的基本 CloudFront 分佈。

```

$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3OriginConfig = New-Object Amazon.CloudFront.Model.S3OriginConfig
$origin.S3OriginConfig.OriginAccessIdentity = ""
New-CFDistribution `
    -DistributionConfig_Enabled $true `
    -DistributionConfig_Comment "Test distribution" `
    -Origins_Item $origin `
    -Origins_Quantity 1 `
    -Logging_Enabled $true `
    -Logging_IncludeCookie $true `
    -Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
    -Logging_Prefix "help/" `
    -DistributionConfig_CallerReference Client1 `
    -DistributionConfig_DefaultRootObject index.html `
    -DefaultCacheBehavior_TargetOriginId $origin.Id `
    -ForwardedValues_QueryString $true `
    -Cookies_Forward all `
    -WhitelistedNames_Quantity 0 `
    -TrustedSigners_Enabled $false `
    -TrustedSigners_Quantity 0 `
    -DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
    -DefaultCacheBehavior_MinTTL 1000 `
    -DistributionConfig_PriceClass "PriceClass_All" `
    -CacheBehaviors_Quantity 0 `
    -Aliases_Quantity 0

```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [CreateDistribution](#)。

Tools for PowerShell V5

範例 1：建立以日誌記錄和快取設定的基本 CloudFront 分佈。

```
$origin = New-Object Amazon.CloudFront.Model.Origin
$origin.DomainName = "amzn-s3-demo-bucket.s3.amazonaws.com"
$origin.Id = "UniqueOrigin1"
$origin.S3OriginConfig = New-Object Amazon.CloudFront.Model.S3OriginConfig
$origin.S3OriginConfig.OriginAccessIdentity = ""
New-CFDistribution `
    -DistributionConfig_Enabled $true `
    -DistributionConfig_Comment "Test distribution" `
    -Origins_Item $origin `
    -Origins_Quantity 1 `
    -Logging_Enabled $true `
    -Logging_IncludeCookie $true `
    -Logging_Bucket amzn-s3-demo-logging-bucket.s3.amazonaws.com `
    -Logging_Prefix "help/" `
    -DistributionConfig_CallerReference Client1 `
    -DistributionConfig_DefaultRootObject index.html `
    -DefaultCacheBehavior_TargetOriginId $origin.Id `
    -ForwardedValues_QueryString $true `
    -Cookies_Forward all `
    -WhitelistedNames_Quantity 0 `
    -TrustedSigners_Enabled $false `
    -TrustedSigners_Quantity 0 `
    -DefaultCacheBehavior_ViewerProtocolPolicy allow-all `
    -DefaultCacheBehavior_MinTTL 1000 `
    -DistributionConfig_PriceClass "PriceClass_All" `
    -CacheBehaviors_Quantity 0 `
    -Aliases_Quantity 0
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [CreateDistribution](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateFunction 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateFunction。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.CloudFrontException;
import software.amazon.awssdk.services.cloudfront.model.CreateFunctionRequest;
import software.amazon.awssdk.services.cloudfront.model.CreateFunctionResponse;
import software.amazon.awssdk.services.cloudfront.model.FunctionConfig;
import software.amazon.awssdk.services.cloudfront.model.FunctionRuntime;
import java.io.InputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CreateFunction {

    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <functionName> <filePath>

            Where:
```

```

        functionName - The name of the function to create.\s
        filePath - The path to a file that contains the application
logic for the function.\s
        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String functionName = args[0];
    String filePath = args[1];
    CloudFrontClient cloudFrontClient = CloudFrontClient.builder()
        .region(Region.AWS_GLOBAL)
        .build();

    String funArn = createNewFunction(cloudFrontClient, functionName,
filePath);
    System.out.println("The function ARN is " + funArn);
    cloudFrontClient.close();
}

public static String createNewFunction(CloudFrontClient cloudFrontClient,
String functionName, String filePath) {
    try {
        InputStream fileIs =
CreateFunction.class.getClassLoader().getResourceAsStream(filePath);
        SdkBytes functionCode = SdkBytes.fromInputStream(fileIs);

        FunctionConfig config = FunctionConfig.builder()
            .comment("Created by using the CloudFront Java API")
            .runtime(FunctionRuntime.CLOUDFRONT_JS_1_0)
            .build();

        CreateFunctionRequest functionRequest =
CreateFunctionRequest.builder()
            .name(functionName)
            .functionCode(functionCode)
            .functionConfig(config)
            .build();

        CreateFunctionResponse response =
cloudFrontClient.createFunction(functionRequest);
        return response.functionSummary().functionMetadata().functionARN();
    }
}

```

```
        } catch (CloudFrontException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的「[CreateFunction](#)」。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreateInvalidation** 與 CLI

下列程式碼範例示範如何使用 CreateInvalidation。

CLI

AWS CLI

為 CloudFront 分佈建立失效證明

下列 create-invalidation 範例會為指定 CloudFront 分佈中的指定檔案建立失效證明：

```
aws cloudfront create-invalidation \
  --distribution-id EDFDVBD6EXAMPLE \
  --paths "/example-path/example-file.jpg" "/example-path/example-file2.png"
```

輸出：

```
{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/EDFDVBD6EXAMPLE/invalidation/I1JLWSDAP8FU89",
  "Invalidation": {
    "Id": "I1JLWSDAP8FU89",
    "Status": "InProgress",
    "CreateTime": "2019-12-05T18:24:51.407Z",
    "InvalidationBatch": {
      "Paths": {
```

```

        "Quantity": 2,
        "Items": [
            "/example-path/example-file2.png",
            "/example-path/example-file.jpg"
        ]
    },
    "CallerReference": "cli-1575570291-670203"
}
}
}

```

在先前的範例中，CLI AWS 會自動產生隨機的 CallerReference。若要指定您自己的 CallerReference，或避免傳遞失效參數做為命令列引數，您可以使用 JSON 檔案。下列範例會藉由在名為 `inv-batch.json` 的 JSON 檔案中提供失效參數，為兩個檔案建立失效證明：

```

aws cloudfront create-invalidation \
  --distribution-id EDFDVBD6EXAMPLE \
  --invalidation-batch file://inv-batch.json

```

`inv-batch.json` 的內容：

```

{
  "Paths": {
    "Quantity": 2,
    "Items": [
      "/example-path/example-file.jpg",
      "/example-path/example-file2.png"
    ]
  },
  "CallerReference": "cli-example"
}

```

輸出：

```

{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/distribution/EDFDVBD6EXAMPLE/invalidation/I2J0I21PCUY0IK",
  "Invalidation": {
    "Id": "I2J0I21PCUY0IK",
    "Status": "InProgress",

```

```

    "CreateTime": "2019-12-05T18:40:49.413Z",
    "InvalidationBatch": {
      "Paths": {
        "Quantity": 2,
        "Items": [
          "/example-path/example-file.jpg",
          "/example-path/example-file2.png"
        ]
      },
      "CallerReference": "cli-example"
    }
  }
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateInvalidation](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例會在 ID 為 EXAMPLNSTXAXE 的分佈上建立新的失效證明。CallerReference 是使用者選擇的唯一 ID；在此情況下，會使用代表 2019 年 5 月 15 日上午 9:00 的時間戳記。\$Paths 變數存放了使用者不想要當作分佈快取一部分的映像和媒體檔案的三個路徑。- Paths_Quantity 參數值是 -Paths_Item 參數中指定的路徑總數。

```

$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLNSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -
Paths_Quantity 3

```

輸出：

Invalidation	Location
-----	-----
Amazon.CloudFront.Model.Invalidation	https://cloudfront.amazonaws.com/2018-11-05/distribution/EXAMPLNSTXAXE/invalidation/EXAMPLE8N0K9H

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [CreateInvalidation](#)。

Tools for PowerShell V5

範例 1：此範例會在 ID 為 EXAMPLNSTXAXE 的分佈上建立新的失效證明。CallerReference 是使用者選擇的唯一 ID；在此情況下，會使用代表 2019 年 5 月 15 日上午 9:00 的時間戳記。\$Paths 變數存放了使用者不想要當作分佈快取一部分的映像和媒體檔案的三個路徑。-Paths_Quantity 參數值是 -Paths_Item 參數中指定的路徑總數。

```
$Paths = "/images/*.gif", "/images/image1.jpg", "/videos/*.mp4"
New-CFInvalidation -DistributionId "EXAMPLNSTXAXE" -
InvalidationBatch_CallerReference 20190515090000 -Paths_Item $Paths -
Paths_Quantity 3
```

輸出：

Invalidation	Location
-----	-----
Amazon.CloudFront.Model.Invalidation	https://cloudfront.amazonaws.com/2018-11-05/distribution/EXAMPLNSTXAXE/invalidation/EXAMPLE8N0K9H

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [CreateInvalidation](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateKeyGroup 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 CreateKeyGroup。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

金鑰群組需要至少一個用於驗證已簽署 URL 或 Cookie 的公有金鑰。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;

import java.util.UUID;

public class CreateKeyGroup {
    private static final Logger logger =
        LoggerFactory.getLogger(CreateKeyGroup.class);

    public static String createKeyGroup(CloudFrontClient cloudFrontClient, String
        publicKeyId) {
        String keyGroupId = cloudFrontClient.createKeyGroup(b ->
        b.keyGroupConfig(c -> c
            .items(publicKeyId)
            .name("JavaKeyGroup" + UUID.randomUUID()))
            .keyGroup().id());
        logger.info("KeyGroup created with ID: [{}]", keyGroupId);
        return keyGroupId;
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateKeyGroup](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreatePublicKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreatePublicKey。

CLI

AWS CLI

建立 CloudFront 公有金鑰

下列範例會藉由在名為 `pub-key-config.json` 的 JSON 檔案中提供參數，來建立 CloudFront 公有金鑰。您必須具有 PEM 編碼的公有金鑰，才能使用此命令。如需詳細資訊，請參閱《Amazon CloudFront 開發人員指南》中的[建立 RSA 金鑰對](#)。

```
aws cloudfront create-public-key \
  --public-key-config file://pub-key-config.json
```

檔案 `pub-key-config.json` 是目前資料夾中的 JSON 文件，包含下列項目。請注意，公有金鑰會以 PEM 格式編碼。

```
{
  "CallerReference": "cli-example",
  "Name": "ExampleKey",
  "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks01nd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBaz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPNhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
+kGZ2NQ0FyIyT2eiLK0X5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nnrwIDAQAB\n-----
END PUBLIC KEY-----\n",
  "Comment": "example public key"
}
```

輸出：

```
{
  "Location": "https://cloudfront.amazonaws.com/2019-03-26/public-key/
KDFB19YGCR002",
  "ETag": "E2QWRUHEXAMPLE",
  "PublicKey": {
    "Id": "KDFB19YGCR002",
    "CreatedTime": "2019-12-05T18:51:43.781Z",
    "PublicKeyConfig": {
      "CallerReference": "cli-example",
      "Name": "ExampleKey",
      "EncodedKey": "-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxPMbCA2Ks01nd7IR+3pw
\nwd3H/7jPGwj8bLUmore7bX+oeGpZ6QmLAe/1U0WcmZX2u70dYcSIzB1ofZtcn4cJ
\nenHBaz03ohBY/L1tQGJfS2A+omnN6H16VZE1JCK8XSJyfze7MDLcUyHZETdxuvRb
\nA9X343/vMAuQPNhinFJ8Wdy8YBXSPpy7r95ylUQd9LfYTBzVZYG2tSesplc0kjM3\n2Uu
+oMWxQAw1NINnSLPinMVsutJy6Zq1V3McWNWe4T+STGtWhrPNqJEn45sIcCx4\nnq
```

```
+kGZ2NQ0FyIyT2eiLK0X5Rgb/a36E/aMk4VoDsaenBQgG7WLTnstb9sr7MIhS6A\nrwIDAQAB\n-----
END PUBLIC KEY-----\n",
    "Comment": "example public key"
  }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreatePublicKey](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下列程式碼範例讀取公有金鑰，並將其上傳至 Amazon CloudFront。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.CreatePublicKeyResponse;
import software.amazon.awssdk.utils.IoUtils;

import java.io.IOException;
import java.io.InputStream;
import java.util.UUID;

public class CreatePublicKey {
    private static final Logger logger =
        LoggerFactory.getLogger(CreatePublicKey.class);

    public static String createPublicKey(CloudFrontClient cloudFrontClient,
        String publicKeyFileName) {
        try (InputStream is =
            CreatePublicKey.class.getClassLoader().getResourceAsStream(publicKeyFileName)) {
            String publicKeyString = IoUtils.toUtf8String(is);
            CreatePublicKeyResponse createPublicKeyResponse = cloudFrontClient
                .createPublicKey(b -> b.publicKeyConfig(c -> c
```

```
        .name("JavaCreatedPublicKey" + UUID.randomUUID())
        .encodedKey(publicKeyString)
        .callerReference(UUID.randomUUID().toString()));
String createdPublicKeyId = createPublicKeyResponse.publicKey().id();
logger.info("Public key created with id: [{}]", createdPublicKeyId);
return createdPublicKeyId;

    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreatePublicKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteDistribution 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteDistribution。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [開始使用 CloudFront](#)

CLI

AWS CLI

刪除 CloudFront 分佈

下列範例會刪除 ID 為 EDFDVBD6EXAMPLE 的 CloudFront 分佈。您必須先停用分佈，才能加以刪除。若要停用分佈，請使用 update-distribution 命令。如需詳細資訊，請參閱 update-distribution 範例。

分佈停用時，可以刪除。若要刪除分佈，您必須使用 --if-match 選項提供分佈的 ETag。若要取得 ETag，請使用 get-distribution 或 get-distribution-config 命令。

```
aws cloudfront delete-distribution \
```

```
--id EDFDVBDGEXAMPLE \  
--if-match E2QWRUHEXAMPLE
```

此命令成功時不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteDistribution](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下列程式碼範例將分佈更新為已停用、使用等待部署變更的等待程式，然後刪除分佈。

```
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;  
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;  
import  
    software.amazon.awssdk.services.cloudfront.model.DeleteDistributionResponse;  
import software.amazon.awssdk.services.cloudfront.model.DistributionConfig;  
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;  
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;  
  
public class DeleteDistribution {  
    private static final Logger logger =  
        LoggerFactory.getLogger(DeleteDistribution.class);  
  
    public static void deleteDistribution(final CloudFrontClient  
cloudFrontClient, final String distributionId) {  
        // First, disable the distribution by updating it.  
        GetDistributionResponse response =  
cloudFrontClient.getDistribution(b -> b  
                                .id(distributionId));  
        String etag = response.eTag();  
        DistributionConfig distConfig =  
response.distribution().distributionConfig();
```

```

        cloudFrontClient.updateDistribution(builder -> builder
            .id(distributionId)
            .distributionConfig(builder1 -> builder1

                .cacheBehaviors(distConfig.cacheBehaviors())

                .defaultCacheBehavior(distConfig.defaultCacheBehavior())
                    .enabled(false)
                    .origins(distConfig.origins())
                    .comment(distConfig.comment())

                .callerReference(distConfig.callerReference())

                .defaultCacheBehavior(distConfig.defaultCacheBehavior())

                .priceClass(distConfig.priceClass())
                    .aliases(distConfig.aliases())
                    .logging(distConfig.logging())

                .defaultRootObject(distConfig.defaultRootObject())

                .customErrorResponses(distConfig.customErrorResponses())

                .httpVersion(distConfig.httpVersion())

                .isIPV6Enabled(distConfig.isIPV6Enabled())

                .restrictions(distConfig.restrictions())

                .viewerCertificate(distConfig.viewerCertificate())
                    .webACLId(distConfig.webACLId())

                .originGroups(distConfig.originGroups())
                    .ifMatch(etag));

        logger.info("Distribution [{}] is DISABLED, waiting for
deployment before deleting ...",
                    distributionId);
        GetDistributionResponse distributionResponse;
        try (CloudFrontWaiter cfWaiter =
CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
            ResponseOrException<GetDistributionResponse>
responseOrException = cfWaiter

```

```

        .waitUntilDistributionDeployed(builder ->
builder.id(distributionId)).matched();
        distributionResponse = responseOrException.response()
            .orElseThrow(() -> new
RuntimeException("Could not disable distribution"));
    }

    DeleteDistributionResponse deleteDistributionResponse =
cloudFrontClient
        .deleteDistribution(builder -> builder
            .id(distributionId)

        .ifMatch(distributionResponse.eTag()));
    if (deleteDistributionResponse.sdkHttpResponse().isSuccessful())
    {
        logger.info("Distribution [{}] DELETED", distributionId);
    }
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteDistribution](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetCloudFrontOriginAccessIdentity** 與 CLI

下列程式碼範例示範如何使用 `GetCloudFrontOriginAccessIdentity`。

CLI

AWS CLI

取得 CloudFront 原始存取身分

下列範例會取得 ID 為 E74FTE3AEXAMPLE 的 CloudFront 原始存取身分 (OAI)，包括其 ETag 和相關聯的 S3 正式 ID。OAI ID 會在 `create-cloud-front-origin-access-identity` 和 `list-cloud-front-origin-access-identities` 命令的輸出中傳回。

```
aws cloudfront get-cloud-front-origin-access-identity --id E74FTE3AEXAMPLE
```

輸出：

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "CloudFrontOriginAccessIdentity": {
    "Id": "E74FTE3AEXAMPLE",
    "S3CanonicalUserId":
"cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
    "CloudFrontOriginAccessIdentityConfig": {
      "CallerReference": "cli-example",
      "Comment": "Example OAI"
    }
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetCloudFrontOriginAccessIdentity](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例傳回由 -Id 參數指定的特定 Amazon CloudFront 原始存取身分。雖然不需要 -Id 參數，但如果沒有指定該參數，就不會傳回任何結果。

```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

輸出：

```
CloudFrontOriginAccessIdentityConfig    Id
S3CanonicalUserId
-----
Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
4b6e...
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetCloudFrontOriginAccessIdentity](#)。

Tools for PowerShell V5

範例 1：此範例傳回由 `-Id` 參數指定的特定 Amazon CloudFront 原始存取身分。雖然不需要 `-Id` 參數，但如果沒有指定該參數，就不會傳回任何結果。

```
Get-CFCloudFrontOriginAccessIdentity -Id E3XXXXXXXXXXRT
```

輸出：

```
CloudFrontOriginAccessIdentityConfig    Id
S3CanonicalUserId
-----
Amazon.CloudFront.Model.CloudFrontOr... E3XXXXXXXXXXRT
4b6e...
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetCloudFrontOriginAccessIdentity](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 `GetCloudFrontOriginAccessIdentityConfig` 與 CLI

下列程式碼範例示範如何使用 `GetCloudFrontOriginAccessIdentityConfig`。

CLI

AWS CLI

取得 CloudFront 原始存取身分組態

下列範例會取得 ID 為 `E74FTE3AEXAMPLE` 的 CloudFront 原始存取身分 (OAI) 的相關中繼資料，包括其 ETag。OAI ID 會在 `create-cloud-front-origin-access-identity` 和 `list-cloud-front-origin-access-identities` 命令的輸出中傳回。

```
aws cloudfront get-cloud-front-origin-access-identity-config --id E74FTE3AEXAMPLE
```

輸出：

```
{
```

```

    "ETag": "E2QWRUHEXAMPLE",
    "CloudFrontOriginAccessIdentityConfig": {
        "CallerReference": "cli-example",
        "Comment": "Example OAI"
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetCloudFrontOriginAccessIdentityConfig](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例傳回 -Id 參數所指定之單一 Amazon CloudFront 原始存取身分的組態資訊。如果未指定 -Id 參數，則會發生錯誤。

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXXXRT
```

輸出：

CallerReference	Comment
-----	-----
mycallerreference: 2/1/2011 1:16:32 PM	Caller
reference: 2/1/2011 1:16:32 PM	

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetCloudFrontOriginAccessIdentityConfig](#)。

Tools for PowerShell V5

範例 1：此範例傳回 -Id 參數所指定之單一 Amazon CloudFront 原始存取身分的組態資訊。如果未指定 -Id 參數，則會發生錯誤。

```
Get-CFCloudFrontOriginAccessIdentityConfig -Id E3XXXXXXXXXXXXRT
```

輸出：

CallerReference	Comment
-----	-----

```
mycallerreference: 2/1/2011 1:16:32 PM  
reference: 2/1/2011 1:16:32 PM
```

Caller

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetCloudFrontOriginAccessIdentityConfig](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetDistribution** 與 CLI

下列程式碼範例示範如何使用 GetDistribution。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [開始使用 CloudFront](#)

CLI

AWS CLI

取得 CloudFront 分佈

下列 get-distribution 範例會取得 ID 為 EDFDVBD6EXAMPLE 的 CloudFront 分佈，包括其 ETag。分佈 ID 會在 create-distribution 和 list-distributions 命令中傳回。

```
aws cloudfront get-distribution \  
--id EDFDVBD6EXAMPLE
```

輸出：

```
{  
  "ETag": "E2QWRUHEXAMPLE",  
  "Distribution": {  
    "Id": "EDFDVBD6EXAMPLE",  
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",  
    "Status": "Deployed",  
    "LastModifiedTime": "2019-12-04T23:35:41.433Z",  
    "InProgressInvalidationBatches": 0,  
    "DomainName": "d1111111abcdef8.cloudfront.net",
```

```

    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "DistributionConfig": {
      "CallerReference": "cli-example",
      "Aliases": {
        "Quantity": 0
      },
      "DefaultRootObject": "index.html",
      "Origins": {
        "Quantity": 1,
        "Items": [
          {
            "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
            "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
            "OriginPath": "",
            "CustomHeaders": {
              "Quantity": 0
            },
            "S3OriginConfig": {
              "OriginAccessIdentity": ""
            }
          }
        ]
      },
      "OriginGroups": {
        "Quantity": 0
      },
      "DefaultCacheBehavior": {
        "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-
example",
        "ForwardedValues": {
          "QueryString": false,
          "Cookies": {
            "Forward": "none"
          },
          "Headers": {
            "Quantity": 0
          },
          "QueryStringCacheKeys": {
            "Quantity": 0
          }
        }
      }
    },

```

```
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": true,
```

```
    "ViewerCertificate": {
      "CloudFrontDefaultCertificate": true,
      "MinimumProtocolVersion": "TLSv1",
      "CertificateSource": "cloudfront"
    },
    "Restrictions": {
      "GeoRestriction": {
        "RestrictionType": "none",
        "Quantity": 0
      }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetDistribution](#)。

PowerShell

Tools for PowerShell V4

範例 1：擷取特定分佈的資訊。

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetDistribution](#)。

Tools for PowerShell V5

範例 1：擷取特定分佈的資訊。

```
Get-CFDistribution -Id EXAMPLE0000ID
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetDistribution](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetDistributionConfig 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetDistributionConfig。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [開始使用 CloudFront](#)

CLI

AWS CLI

取得 CloudFront 分佈組態

下列範例會取得 ID 為 EDFDVBD6EXAMPLE 的 CloudFront 分佈的相關中繼資料，包括其 ETag。分佈 ID 會在 create-distribution 和 list-distributions 命令中傳回。

```
aws cloudfront get-distribution-config \  
--id EDFDVBD6EXAMPLE
```

輸出：

```
{  
  "ETag": "E2QWRUHEXAMPLE",  
  "DistributionConfig": {  
    "CallerReference": "cli-example",  
    "Aliases": {  
      "Quantity": 0  
    },  
    "DefaultRootObject": "index.html",  
    "Origins": {  
      "Quantity": 1,  
      "Items": [  
        {  
          "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",  
          "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",  
          "OriginPath": "",  
          "CustomHeaders": {  
            "Quantity": 0  
          },  
          "S3OriginConfig": {  

```

```

        "OriginAccessIdentity": ""
    }
}
],
},
"OriginGroups": {
    "Quantity": 0
},
"DefaultCacheBehavior": {
    "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-cli-example",
    "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
            "Forward": "none"
        },
        "Headers": {
            "Quantity": 0
        },
        "QueryStringCacheKeys": {
            "Quantity": 0
        }
    },
    "TrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ],
        "CachedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ]
        }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,

```



```
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
            "Quantity": 0
        },
        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetDistributionConfig](#)。

PowerShell

Tools for PowerShell V4

範例 1：擷取特定分佈的組態。

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [GetDistributionConfig](#)。

Tools for PowerShell V5

範例 1：擷取特定分佈的組態。

```
Get-CFDistributionConfig -Id EXAMPLE0000ID
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [GetDistributionConfig](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client
```

```
def update_distribution(self):
    distribution_id = input(
        "This script updates the comment for a CloudFront distribution.\n"
        "Enter a CloudFront distribution ID: "
    )

    distribution_config_response =
self.cloudfront_client.get_distribution_config(
    Id=distribution_id
)
    distribution_config = distribution_config_response["DistributionConfig"]
    distribution_etag = distribution_config_response["ETag"]

    distribution_config["Comment"] = input(
        f"\nThe current comment for distribution {distribution_id} is "
        f"'{distribution_config['Comment']}'.\n"
        f"Enter a new comment: "
    )
    self.cloudfront_client.update_distribution(
        DistributionConfig=distribution_config,
        Id=distribution_id,
        IfMatch=distribution_etag,
    )
    print("Done!")
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetDistributionConfig](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListCloudFrontOriginAccessIdentities** 與 CLI

下列程式碼範例示範如何使用 `ListCloudFrontOriginAccessIdentities`。

CLI

AWS CLI

列出 CloudFront 原始存取身分

下列範例會取得您 AWS 帳戶中 CloudFront 原始存取身分 (OAIs) 的清單：

```
aws cloudfront list-cloud-front-origin-access-identities
```

輸出：

```
{
  "CloudFrontOriginAccessIdentityList": {
    "Items": [
      {
        "Id": "E74FTE3AEXAMPLE",
        "S3CanonicalUserId":
        "cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE",
        "Comment": "Example OAI"
      },
      {
        "Id": "EH1HDMBEXAMPLE",
        "S3CanonicalUserId":
        "1489f6f2e6faacaae7ff64c4c3e6956c24f78788abfc1718c3527c263bf7a17EXAMPLE",
        "Comment": "Test OAI"
      },
      {
        "Id": "E2X2C9TEXAMPLE",
        "S3CanonicalUserId":
        "cbfeebb915a64749f9be546a45b3fcfd3a31c779673c13c4dd460911ae402c2EXAMPLE",
        "Comment": "Example OAI #2"
      }
    ]
  }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListCloudFrontOriginAccessIdentities](#)。

PowerShell

Tools for PowerShell V4

範例 1：此範例傳回 Amazon CloudFront 原始存取身分的清單。由於 `-MaxItem` 參數指定值 2，因此結果包含兩個身分。

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

輸出：

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YWXXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXXX9B
Quantity    : 2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListCloudFrontOriginAccessIdentities](#)。

Tools for PowerShell V5

範例 1：此範例傳回 Amazon CloudFront 原始存取身分的清單。由於 -MaxItem 參數指定值 2，因此結果包含兩個身分。

```
Get-CFCloudFrontOriginAccessIdentityList -MaxItem 2
```

輸出：

```
IsTruncated : True
Items       : {E326XXXXXXXXXT, E1YWXXXXXXXX9B}
Marker      :
MaxItems    : 2
NextMarker  : E1YXXXXXXXXX9B
Quantity    : 2
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListCloudFrontOriginAccessIdentities](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListDistributions 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListDistributions。

CLI

AWS CLI

列出 CloudFront 分佈

下列範例會取得您 AWS 帳戶中 CloudFront 分佈的清單。

```
aws cloudfront list-distributions
```

輸出：

```
{
  "DistributionList": {
    "Items": [
      {
        "Id": "E23YS80EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/E23YS80EXAMPLE",
        "Status": "Deployed",
        "LastModifiedTime": "2024-08-05T18:23:40.375000+00:00",
        "DomainName": "abcdefgh12ijk.cloudfront.net",
        "Aliases": {
          "Quantity": 0
        },
        "Origins": {
          "Quantity": 1,
          "Items": [
            {
              "Id": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
              "DomainName": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
              "OriginPath": "",
              "CustomHeaders": {
                "Quantity": 0
              },
              "S3OriginConfig": {
                "OriginAccessIdentity": ""
              },
              "ConnectionAttempts": 3,
              "ConnectionTimeout": 10,
              "OriginShield": {
```

```

        "Enabled": false
      },
      "OriginAccessControlId": "EIAP8PEXAMPLE"
    }
  ],
},
"OriginGroups": {
  "Quantity": 0
},
"DefaultCacheBehavior": {
  "TargetOriginId": "amzn-s3-demo-bucket.s3.us-east-1.amazonaws.com",
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "TrustedKeyGroups": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
  },
  "CachedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ]
  }
},
"SmoothStreaming": false,
"Compress": true,
"LambdaFunctionAssociations": {
  "Quantity": 0
},
"FunctionAssociations": {
  "Quantity": 0
},
"FieldLevelEncryptionId": "",

```

```
        "CachePolicyId": "658327ea-f89d-4fab-a63d-7e886EXAMPLE"
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "SSLSupportMethod": "vip",
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "HTTP2",
    "IsIPV6Enabled": true,
    "Staging": false
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListDistributions](#)。

PowerShell

Tools for PowerShell V4

範例 1：傳回分佈。

```
Get-CFDistributionList
```


- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 [ListDistributions](#)。

Tools for PowerShell V5

範例 1：傳回分佈。

```
Get-CFDistributionList
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 [ListDistributions](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client

    def list_distributions(self):
        print("CloudFront distributions:\n")
        distributions = self.cloudfront_client.list_distributions()
        if distributions["DistributionList"]["Quantity"] > 0:
            for distribution in distributions["DistributionList"]["Items"]:
                print(f"Domain: {distribution['DomainName']}")
                print(f"Distribution Id: {distribution['Id']}")
                print(
                    f"Certificate Source: "
```

```
        f"{distribution['ViewerCertificate']['CertificateSource']}"
    )
    if distribution["ViewerCertificate"]["CertificateSource"] ==
"acm":
        print(
            f"Certificate: {distribution['ViewerCertificate']
['Certificate']}"
        )
        print("")
    else:
        print("No CloudFront distributions detected.")
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListDistributions](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UpdateDistribution 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UpdateDistribution。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [開始使用 CloudFront](#)

CLI

AWS CLI

範例 1：更新 CloudFront 分佈的預設根物件

下列範例會將 ID 為 EDFDVBD6EXAMPLE 的 CloudFront 分佈的預設根物件更新為 index.html。

```
aws cloudfront update-distribution \
    --id EDFDVBD6EXAMPLE \
    --default-root-object index.html
```

輸出：

```
{
  "ETag": "E2QWRUHEXAMPLE",
  "Distribution": {
    "Id": "EDFDVBD6EXAMPLE",
    "ARN": "arn:aws:cloudfront::123456789012:distribution/EDFDVBD6EXAMPLE",
    "Status": "InProgress",
    "LastModifiedTime": "2019-12-06T18:55:39.870Z",
    "InProgressInvalidationBatches": 0,
    "DomainName": "d1111111abcdef8.cloudfront.net",
    "ActiveTrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
  },
  "DistributionConfig": {
    "CallerReference": "6b10378d-49be-4c4b-a642-419ccaf8f3b5",
    "Aliases": {
      "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
      "Quantity": 1,
      "Items": [
        {
          "Id": "example-website",
          "DomainName": "www.example.com",
          "OriginPath": "",
          "CustomHeaders": {
            "Quantity": 0
          },
        },
      ],
      "CustomOriginConfig": {
        "HTTPPort": 80,
        "HTTPSPort": 443,
        "OriginProtocolPolicy": "match-viewer",
        "OriginSslProtocols": {
          "Quantity": 2,
          "Items": [
            "SSLv3",
            "TLSv1"
          ]
        },
      },
      "OriginReadTimeout": 30,
      "OriginKeepaliveTimeout": 5
    }
  }
}
```

```

        }
    }
]
},
"OriginGroups": {
    "Quantity": 0
},
"DefaultCacheBehavior": {
    "TargetOriginId": "example-website",
    "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
            "Forward": "none"
        },
        "Headers": {
            "Quantity": 1,
            "Items": [
                "*"
            ]
        },
        "QueryStringCacheKeys": {
            "Quantity": 0
        }
    },
    "TrustedSigners": {
        "Enabled": false,
        "Quantity": 0
    },
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
        "Quantity": 2,
        "Items": [
            "HEAD",
            "GET"
        ],
        "CachedMethods": {
            "Quantity": 2,
            "Items": [
                "HEAD",
                "GET"
            ]
        }
    }
},

```

```

        "SmoothStreaming": false,
        "DefaultTTL": 86400,
        "MaxTTL": 31536000,
        "Compress": false,
        "LambdaFunctionAssociations": {
            "Quantity": 0
        },
        "FieldLevelEncryptionId": ""
    },
    "CacheBehaviors": {
        "Quantity": 0
    },
    "CustomErrorResponses": {
        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": true,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http1.1",
    "IsIPV6Enabled": true
}
}
}

```

範例 2：更新 CloudFront 分佈

下列範例會藉由在名為 `dist-config-disable.json` 的 JSON 檔案中提供分佈組態，來停用 ID 為 `EMLARXS9EXAMPLE` 的 CloudFront 分佈。若要更新分佈，您必須使用 `--if-match` 選項提供分佈的 ETag。若要取得 ETag，請使用 `get-distribution` 或 `get-distribution-config` 命令。請注意，JSON 檔案中的 `Enabled` 欄位設定為 `false`。

使用下列範例停用分佈後，您可以使用 `delete-distribution` 命令將其刪除。

```
aws cloudfront update-distribution \  
  --id EMLARXS9EXAMPLE \  
  --if-match E2QWRUHEXAMPLE \  
  --distribution-config file://dist-config-disable.json
```

`dist-config-disable.json` 的內容：

```
{  
  "CallerReference": "cli-1574382155-496510",  
  "Aliases": {  
    "Quantity": 0  
  },  
  "DefaultRootObject": "index.html",  
  "Origins": {  
    "Quantity": 1,  
    "Items": [  
      {  
        "Id": "amzn-s3-demo-bucket.s3.amazonaws.com-1574382155-273939",  
        "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",  
        "OriginPath": "",  
        "CustomHeaders": {  
          "Quantity": 0  
        },  
        "S3OriginConfig": {  
          "OriginAccessIdentity": ""  
        }  
      }  
    ]  
  },  
  "OriginGroups": {  
    "Quantity": 0  
  },  
  "DefaultCacheBehavior": {  
    "TargetOriginId": "amzn-s3-demo-bucket.s3.amazonaws.com-1574382155-273939",  
    "ForwardedValues": {
```

```
    "QueryString": false,
    "Cookies": {
      "Forward": "none"
    },
    "Headers": {
      "Quantity": 0
    },
    "QueryStringCacheKeys": {
      "Quantity": 0
    }
  },
  "TrustedSigners": {
    "Enabled": false,
    "Quantity": 0
  },
  "ViewerProtocolPolicy": "allow-all",
  "MinTTL": 0,
  "AllowedMethods": {
    "Quantity": 2,
    "Items": [
      "HEAD",
      "GET"
    ],
    "CachedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ]
    }
  },
  "SmoothStreaming": false,
  "DefaultTTL": 86400,
  "MaxTTL": 31536000,
  "Compress": false,
  "LambdaFunctionAssociations": {
    "Quantity": 0
  },
  "FieldLevelEncryptionId": ""
},
"CacheBehaviors": {
  "Quantity": 0
},
"CustomErrorResponses": {
```

```

        "Quantity": 0
    },
    "Comment": "",
    "Logging": {
        "Enabled": false,
        "IncludeCookies": false,
        "Bucket": "",
        "Prefix": ""
    },
    "PriceClass": "PriceClass_All",
    "Enabled": false,
    "ViewerCertificate": {
        "CloudFrontDefaultCertificate": true,
        "MinimumProtocolVersion": "TLSv1",
        "CertificateSource": "cloudfront"
    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}

```

輸出：

```

{
    "ETag": "E9LHASXEXAMPLE",
    "Distribution": {
        "Id": "EMLARXS9EXAMPLE",
        "ARN": "arn:aws:cloudfront::123456789012:distribution/EMLARXS9EXAMPLE",
        "Status": "InProgress",
        "LastModifiedTime": "2019-12-06T18:32:35.553Z",
        "InProgressInvalidationBatches": 0,
        "DomainName": "d111111abcdef8.cloudfront.net",
        "ActiveTrustedSigners": {
            "Enabled": false,
            "Quantity": 0
        },
        "DistributionConfig": {

```



```

    "CallerReference": "cli-1574382155-496510",
    "Aliases": {
      "Quantity": 0
    },
    "DefaultRootObject": "index.html",
    "Origins": {
      "Quantity": 1,
      "Items": [
        {
          "Id": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
          "DomainName": "amzn-s3-demo-bucket.s3.amazonaws.com",
          "OriginPath": "",
          "CustomHeaders": {
            "Quantity": 0
          },
          "S3OriginConfig": {
            "OriginAccessIdentity": ""
          }
        }
      ]
    },
    "OriginGroups": {
      "Quantity": 0
    },
    "DefaultCacheBehavior": {
      "TargetOriginId": "amzn-s3-demo-
bucket.s3.amazonaws.com-1574382155-273939",
      "ForwardedValues": {
        "QueryString": false,
        "Cookies": {
          "Forward": "none"
        }
      },
      "Headers": {
        "Quantity": 0
      },
      "QueryStringCacheKeys": {
        "Quantity": 0
      }
    },
    "TrustedSigners": {
      "Enabled": false,
      "Quantity": 0
    },
  },

```

```
    "ViewerProtocolPolicy": "allow-all",
    "MinTTL": 0,
    "AllowedMethods": {
      "Quantity": 2,
      "Items": [
        "HEAD",
        "GET"
      ],
      "CachedMethods": {
        "Quantity": 2,
        "Items": [
          "HEAD",
          "GET"
        ]
      }
    },
    "SmoothStreaming": false,
    "DefaultTTL": 86400,
    "MaxTTL": 31536000,
    "Compress": false,
    "LambdaFunctionAssociations": {
      "Quantity": 0
    },
    "FieldLevelEncryptionId": ""
  },
  "CacheBehaviors": {
    "Quantity": 0
  },
  "CustomErrorResponses": {
    "Quantity": 0
  },
  "Comment": "",
  "Logging": {
    "Enabled": false,
    "IncludeCookies": false,
    "Bucket": "",
    "Prefix": ""
  },
  "PriceClass": "PriceClass_All",
  "Enabled": false,
  "ViewerCertificate": {
    "CloudFrontDefaultCertificate": true,
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "cloudfront"
```

```

    },
    "Restrictions": {
        "GeoRestriction": {
            "RestrictionType": "none",
            "Quantity": 0
        }
    },
    "WebACLId": "",
    "HttpVersion": "http2",
    "IsIPV6Enabled": true
}
}
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UpdateDistribution](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionRequest;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.DistributionConfig;
import
    software.amazon.awssdk.services.cloudfront.model.UpdateDistributionRequest;
import software.amazon.awssdk.services.cloudfront.model.CloudFrontException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 */

```

```
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class ModifyDistribution {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <id>\s

            Where:
                id - the id value of the distribution.\s
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String id = args[0];
        CloudFrontClient cloudFrontClient = CloudFrontClient.builder()
            .region(Region.AWS_GLOBAL)
            .build();

        modDistribution(cloudFrontClient, id);
        cloudFrontClient.close();
    }

    public static void modDistribution(CloudFrontClient cloudFrontClient, String
idVal) {
        try {
            // Get the Distribution to modify.
            GetDistributionRequest disRequest = GetDistributionRequest.builder()
                .id(idVal)
                .build();

            GetDistributionResponse response =
cloudFrontClient.getDistribution(disRequest);
            Distribution disObject = response.distribution();
            DistributionConfig config = disObject.distributionConfig();

            // Create a new DistributionConfig object and add new values to
comment and
            // aliases
```

```
DistributionConfig config1 = DistributionConfig.builder()
    .aliases(config.aliases()) // You can pass in new values here
    .comment("New Comment")
    .cacheBehaviors(config.cacheBehaviors())
    .priceClass(config.priceClass())
    .defaultCacheBehavior(config.defaultCacheBehavior())
    .enabled(config.enabled())
    .callerReference(config.callerReference())
    .logging(config.logging())
    .originGroups(config.originGroups())
    .origins(config.origins())
    .restrictions(config.restrictions())
    .defaultRootObject(config.defaultRootObject())
    .webACLId(config.webACLId())
    .httpVersion(config.httpVersion())
    .viewerCertificate(config.viewerCertificate())
    .customErrorResponses(config.customErrorResponses())
    .build();

UpdateDistributionRequest updateDistributionRequest =
UpdateDistributionRequest.builder()
    .distributionConfig(config1)
    .id(disObject.id())
    .ifMatch(response.eTag())
    .build();

cloudFrontClient.updateDistribution(updateDistributionRequest);

} catch (CloudFrontException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [UpdateDistribution](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class CloudFrontWrapper:
    """Encapsulates Amazon CloudFront operations."""

    def __init__(self, cloudfront_client):
        """
        :param cloudfront_client: A Boto3 CloudFront client
        """
        self.cloudfront_client = cloudfront_client

    def update_distribution(self):
        distribution_id = input(
            "This script updates the comment for a CloudFront distribution.\n"
            "Enter a CloudFront distribution ID: "
        )

        distribution_config_response =
self.cloudfront_client.get_distribution_config(
            Id=distribution_id
        )
        distribution_config = distribution_config_response["DistributionConfig"]
        distribution_etag = distribution_config_response["ETag"]

        distribution_config["Comment"] = input(
            f"\nThe current comment for distribution {distribution_id} is "
            f"'{distribution_config['Comment']}'.\n"
            f"Enter a new comment: "
        )
        self.cloudfront_client.update_distribution(
            DistributionConfig=distribution_config,
            Id=distribution_id,
            IfMatch=distribution_etag,
```

```
)  
print("Done!")
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [UpdateDistribution](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CloudFront AWS SDKs的案例

下列程式碼範例示範如何在 CloudFront AWS SDKs中實作常見案例。這些案例向您展示如何透過呼叫 CloudFront 中的多個函數或與其他 AWS 服務組合，完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- [建立 SaaS 管理員資源 AWS SDK](#)
- [使用 AWS SDK 刪除 CloudFront 簽署資源](#)
- [使用 CLI 開始使用基本 CloudFront 分佈](#)
- [使用 AWS SDK 建立簽章URLs 和 Cookie](#)

建立 SaaS 管理員資源 AWS SDK

下列程式碼範例示範如何使用各種組態，建立多租用戶分佈和分佈租用戶。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

下列範例示範如何使用參數和萬用字元憑證，建立多租用戶分佈。

```
import software.amazon.awssdk.core.internal.waiters.ResponseOrException;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.ConnectionMode;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.Distribution;
import software.amazon.awssdk.services.cloudfront.model.GetDistributionResponse;
import software.amazon.awssdk.services.cloudfront.model.HttpVersion;
import software.amazon.awssdk.services.cloudfront.model.Method;
import software.amazon.awssdk.services.cloudfront.model.SSLSupportMethod;
import software.amazon.awssdk.services.cloudfront.model.ViewerProtocolPolicy;
import software.amazon.awssdk.services.cloudfront.waiters.CloudFrontWaiter;
import software.amazon.awssdk.services.s3.S3Client;

import java.time.Instant;

public class CreateMultiTenantDistribution {
    public static Distribution
    CreateMultiTenantDistributionWithCert(CloudFrontClient cloudFrontClient,
                                          S3Client
    s3Client,
                                          final String
    bucketName,
                                          final String
    certificateArn) {
        // fetch the origin info if necessary
        final String region = s3Client.headBucket(b ->
        b.bucket(bucketName)).sdkHttpResponse().headers()
            .get("x-amz-bucket-region").get(0);
```



```

        final String originDomain = bucketName + ".s3." + region +
".amazonaws.com";
        String originId = originDomain; // Use the originDomain value for the
originId.

        CreateDistributionResponse createDistResponse =
cloudFrontClient.createDistribution(builder -> builder
            .distributionConfig(b1 -> b1
                .httpVersion(HttpVersion.HTTP2)
                .enabled(true)
                .comment("Template Distribution with cert built with
java")

                .connectionMode(ConnectionMode.TENANT_ONLY)
                .callerReference(Instant.now().toString())
                .viewerCertificate(certBuilder -> certBuilder
                    .acmCertificateArn(certificateArn)
                    .sslSupportMethod(SSLSupportMethod.SNI_ONLY))
                .origins(b2 -> b2
                    .quantity(1)
                    .items(b3 -> b3
                        .domainName(originDomain)
                        .id(originId)
                        .originPath("/{tenantName}")
                        .s3OriginConfig(builder4 -> builder4
                            .originAccessIdentity(
                                ""))))
                .tenantConfig(b5 -> b5
                    .parameterDefinitions(b6 -> b6
                        .name("tenantName")
                        .definition(b7 -> b7
                            .stringSchema(b8 -> b8
                                .comment("tenantName
value")

                                .defaultValue("root")
                                .required(false))))
                    .defaultCacheBehavior(b2 -> b2

                        .viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)
                        .targetOriginId(originId)
                        .cachePolicyId("658327ea-f89d-4fab-
a63d-7e88639e58f6") // CachingOptimized Policy
                        .allowedMethods(b4 -> b4
                            .quantity(2)
                            .items(Method.HEAD, Method.GET)))

```

```

        ));

        final Distribution distribution = createDistResponse.distribution();
        try (CloudFrontWaiter cfWaiter =
CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
            ResponseOrException<GetDistributionResponse> responseOrException =
cfWaiter
                .waitUntilDistributionDeployed(builder ->
builder.id(distribution.id()))
                .matched();
            responseOrException.response()
                .orElseThrow(() -> new RuntimeException("Distribution not
created"));
        }
        return distribution;
    }

    public static Distribution
CreateMultiTenantDistributionNoCert(CloudFrontClient cloudFrontClient,
                                    S3Client s3Client,
                                    final String
bucketName) {
        // fetch the origin info if necessary
        final String region = s3Client.headBucket(b ->
b.bucket(bucketName)).sdkHttpResponse().headers()
            .get("x-amz-bucket-region").get(0);
        final String originDomain = bucketName + ".s3." + region +
".amazonaws.com";
        String originId = originDomain; // Use the originDomain value for the
originId.

        CreateDistributionResponse createDistResponse =
cloudFrontClient.createDistribution(builder -> builder
            .distributionConfig(b1 -> b1
                .httpVersion(HttpVersion.HTTP2)
                .enabled(true)
                .comment("Template Distribution with cert built with
java")

                .connectionMode(ConnectionMode.TENANT_ONLY)
                .callerReference(Instant.now().toString())
                .origins(b2 -> b2
                    .quantity(1)
                    .items(b3 -> b3
                        .domainName(originDomain)

```

```

        .id(originId)
        .originPath("/{tenantName}}")
        .s3OriginConfig(builder4 -> builder4
            .originAccessIdentity(
                ""))))
        .tenantConfig(b5 -> b5
            .parameterDefinitions(b6 -> b6
                .name("tenantName")
                .definition(b7 -> b7
                    .stringSchema(b8 -> b8
                        .comment("tenantName
value")
                        .defaultValue("root")
                        .required(false))))))
        .defaultCacheBehavior(b2 -> b2

        .viewerProtocolPolicy(ViewerProtocolPolicy.ALLOW_ALL)
        .targetOriginId(originId)
        .cachePolicyId("658327ea-f89d-4fab-
a63d-7e88639e58f6") // CachingOptimized Policy
        .allowedMethods(b4 -> b4
            .quantity(2)
            .items(Method.HEAD, Method.GET)))
    ));

    final Distribution distribution = createDistResponse.distribution();
    try (CloudFrontWaiter cfWaiter =
CloudFrontWaiter.builder().client(cloudFrontClient).build()) {
        ResponseOrException<GetDistributionResponse> responseOrException =
cfWaiter
            .waitUntilDistributionDeployed(builder ->
builder.id(distribution.id()))
            .matched();
        responseOrException.response()
            .orElseThrow(() -> new RuntimeException("Distribution not
created"));
    }
    return distribution;
}
}

```

下列範例示範如何建立與該範本相關聯的分佈租用戶，包括使用上述宣告的參數。請注意，我們不需要在此處新增憑證資訊，因為父系範本已涵蓋我們的網域。

```
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
    software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
    software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
    createDistributionTenantNoCert(CloudFrontClient cloudFrontClient,
                                   Route53Client
                                   route53Client,
                                   String
                                   distributionId,
                                   String
                                   domain,
                                   String
                                   hostedZoneId) {
        CreateDistributionTenantResponse createResponse =
        cloudFrontClient.createDistributionTenant(builder -> builder
            .distributionId(distributionId)
            .domains(b1 -> b1
                .domain(domain))
            .parameters(b2 -> b2
                .name("tenantName")
                .value("myTenant"))
            .enabled(false)
            .name("no-cert-tenant")
        );
    }
}
```

```

        final DistributionTenant distributionTenant =
            createResponse.distributionTenant();

        // Then update the Route53 hosted zone to point your domain at the
        // distribution tenant
        // We fetch the RoutingEndpoint to point to via the default connection
        // group that was created for your tenant
        final GetConnectionGroupResponse fetchedConnectionGroup =
            cloudFrontClient.getConnectionGroup(builder -> builder
                .identifier(distributionTenant.connectionGroupId()));

        route53Client.changeResourceRecordSets(builder -> builder
            .hostedZoneId(hostedZoneId)
            .changeBatch(b1 -> b1
                .comment("ChangeBatch comment")
                .changes(b2 -> b2
                    .resourceRecordSet(b3 -> b3
                        .name(domain)
                        .type("CNAME")
                        .ttl(300L)
                        .resourceRecords(b4 -> b4
                            .value(fetchedConnectionGroup.connectionGroup().routingEndpoint()))
                        .action("CREATE"))
                    ));
        return distributionTenant;
    }
}

```

如果從上層範本省略檢視器憑證，則需要改為在與其相關聯的租用戶上新增憑證資訊。下列範例示範如何透過涵蓋租用戶必要網域的 ACM 憑證 ARN 來執行此操作。

```

import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
    software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
    software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;

```

```

import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
    createDistributionTenantWithCert(CloudFrontClient cloudFrontClient,

    Route53Client route53Client,                                     String
    distributionId,                                                String
    domain,                                                         String
    hostedZoneId,                                                  String
    certificateArn) {
        CreateDistributionTenantResponse createResponse =
        cloudFrontClient.createDistributionTenant(builder -> builder
            .distributionId(distributionId)
            .domains(b1 -> b1
                .domain(domain))
            .enabled(false)
            .name("tenant-with-cert")
            .parameters(b2 -> b2
                .name("tenantName")
                .value("myTenant"))
            .customizations(b3 -> b3
                .certificate(b4 -> b4
                    .arn(certificateArn))) // NOTE: Cert must be in
        Us-East-1 and cover the domain provided in this request

        );

        final DistributionTenant distributionTenant =
        createResponse.distributionTenant();

        // Then update the Route53 hosted zone to point your domain at the
        distribution tenant
        // We fetch the RoutingEndpoint to point to via the default connection
        group that was created for your tenant

```

```

        final GetConnectionGroupResponse fetchedConnectionGroup =
cloudFrontClient.getConnectionGroup(builder -> builder
            .identifier(distributionTenant.connectionGroupId()));

        route53Client.changeResourceRecordSets(builder -> builder
            .hostedZoneId(hostedZoneId)
            .changeBatch(b1 -> b1
                .comment("ChangeBatch comment")
                .changes(b2 -> b2
                    .resourceRecordSet(b3 -> b3
                        .name(domain)
                        .type("CNAME")
                        .ttl(300L)
                        .resourceRecords(b4 -> b4
                            .value(fetchedConnectionGroup.connectionGroup().routingEndpoint()))
                        .action("CREATE"))
                    ));
        return distributionTenant;
    }
}

```

下列範例示範如何使用 CloudFront 託管的受管憑證請求，以執行此操作。如果您還沒有流向您網域的流量，這是理想的選擇。在此情況下，我們會建立 ConnectionGroup 來產生 RoutingEndpoint。然後，我們使用 RoutingEndpoint 建立 DNS 記錄，以驗證網域擁有權，並指向 CloudFront。CloudFront 接著自動提供字符，驗證網域擁有權並建立受管憑證。

```

import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
    software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
    software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

```

```

public class CreateDistributionTenant {

    public static DistributionTenant
    createDistributionTenantCfHosted(CloudFrontClient cloudFrontClient,

    Route53Client route53Client,                                     String
    distributionId,                                                String
    domain,                                                         String
    hostedZoneId) throws InterruptedException {
        CreateConnectionGroupResponse createConnectionGroupResponse =
        cloudFrontClient.createConnectionGroup(builder -> builder
            .ipv6Enabled(true)
            .name("cf-hosted-connection-group")
            .enabled(true));

        route53Client.changeResourceRecordSets(builder -> builder
            .hostedZoneId(hostedZoneId)
            .changeBatch(b1 -> b1
                .comment("cf-hosted domain validation record")
                .changes(b2 -> b2
                    .resourceRecordSet(b3 -> b3
                        .name(domain)
                        .type(RRType.CNAME)
                        .ttl(300L)
                        .resourceRecords(b4 -> b4

        .value(createConnectionGroupResponse.connectionGroup().routingEndpoint()))
            .action("CREATE"))
        ));

        // Give the R53 record time to propagate, if it isn't being returned by
        servers yet, the following call will fail
        Thread.sleep(60000);

        CreateDistributionTenantResponse createResponse =
        cloudFrontClient.createDistributionTenant(builder -> builder
            .distributionId(distributionId)
            .domains(b1 -> b1
                .domain(domain))

```



```

        .connectionGroupId(createConnectionGroupResponse.connectionGroup().id())
        .enabled(false)
        .name("cf-hosted-tenant")
        .parameters(b2 -> b2
            .name("tenantName")
            .value("myTenant"))
        .managedCertificateRequest(b3 -> b3
            .validationTokenHost(ValidationTokenHost.CLOUDFRONT)
        )
    );

    return createResponse.distributionTenant();
}
}

```

下列範例示範如何使用自我託管的受管憑證請求，以執行此操作。如果您有流向您網域的流量，且無法容忍移轉期間出現的停機時間，則這會是理想的選擇。在此範例結束時，將會在等待網域驗證和 DNS 設定的狀態下建立租用戶。當您準備好移轉流量時，請依照步驟 [此處](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/managed-cloudfront-certificates.html#complete-domain-ownership) 完成設定。

```

import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import
    software.amazon.awssdk.services.cloudfront.model.CreateConnectionGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.CreateDistributionTenantResponse;
import software.amazon.awssdk.services.cloudfront.model.DistributionTenant;
import
    software.amazon.awssdk.services.cloudfront.model.GetConnectionGroupResponse;
import software.amazon.awssdk.services.cloudfront.model.ValidationTokenHost;
import software.amazon.awssdk.services.route53.Route53Client;
import software.amazon.awssdk.services.route53.model.RRType;

import java.time.Instant;

public class CreateDistributionTenant {

    public static DistributionTenant
        createDistributionTenantSelfHosted(CloudFrontClient cloudFrontClient,

```

```

distributionId,
domain) {
    CreateDistributionTenantResponse createResponse =
cloudFrontClient.createDistributionTenant(builder -> builder
    .distributionId(distributionId)
    .domains(b1 -> b1
        .domain(domain))
    .parameters(b2 -> b2
        .name("tenantName")
        .value("myTenant"))
    .enabled(false)
    .name("self-hosted-tenant")
    .managedCertificateRequest(b3 -> b3
        .validationTokenHost(ValidationTokenHost.SELF_HOSTED)
        .primaryDomainName(domain)
    )
);

    return createResponse.distributionTenant();
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [CreateDistribution](#)
- [CreateDistributionTenant](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 刪除 CloudFront 簽署資源

下列程式碼範例示範如何刪除用來存取 Amazon Simple Storage Service (Amazon S3) 儲存貯體中受限內容的資源。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontClient;
import software.amazon.awssdk.services.cloudfront.model.DeleteKeyGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.DeleteOriginAccessControlResponse;
import software.amazon.awssdk.services.cloudfront.model.DeletePublicKeyResponse;
import software.amazon.awssdk.services.cloudfront.model.GetKeyGroupResponse;
import
    software.amazon.awssdk.services.cloudfront.model.GetOriginAccessControlResponse;
import software.amazon.awssdk.services.cloudfront.model.GetPublicKeyResponse;

public class DeleteSigningResources {
    private static final Logger logger =
        LoggerFactory.getLogger(DeleteSigningResources.class);

    public static void deleteOriginAccessControl(final CloudFrontClient
        cloudFrontClient,
        final String originAccessControlId) {
        GetOriginAccessControlResponse getResponse = cloudFrontClient
            .getOriginAccessControl(b -> b.id(originAccessControlId));
        DeleteOriginAccessControlResponse deleteResponse =
            cloudFrontClient.deleteOriginAccessControl(builder -> builder
                .id(originAccessControlId)
                .ifMatch(getResponse.eTag()));
        if (deleteResponse.sdkHttpResponse().isSuccessful()) {
            logger.info("Successfully deleted Origin Access Control [{}]",
                originAccessControlId);
        }
    }
}
```

```
public static void deleteKeyGroup(final CloudFrontClient cloudFrontClient,
final String keyGroupId) {

    GetKeyGroupResponse getResponse = cloudFrontClient.getKeyGroup(b ->
b.id(keyGroupId));
    DeleteKeyGroupResponse deleteResponse =
cloudFrontClient.deleteKeyGroup(builder -> builder
        .id(keyGroupId)
        .ifMatch(getResponse.eTag()));
    if (deleteResponse.sdkHttpResponse().isSuccessful()) {
        logger.info("Successfully deleted Key Group [{}]", keyGroupId);
    }
}

public static void deletePublicKey(final CloudFrontClient cloudFrontClient,
final String publicKeyId) {
    GetPublicKeyResponse getResponse = cloudFrontClient.getPublicKey(b ->
b.id(publicKeyId));

    DeletePublicKeyResponse deleteResponse =
cloudFrontClient.deletePublicKey(builder -> builder
        .id(publicKeyId)
        .ifMatch(getResponse.eTag()));

    if (deleteResponse.sdkHttpResponse().isSuccessful()) {
        logger.info("Successfully deleted Public Key [{}]", publicKeyId);
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
- [DeleteKeyGroup](#)
- [DeleteOriginAccessControl](#)
- [DeletePublicKey](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 CLI 開始使用基本 CloudFront 分佈

以下程式碼範例顯示做法：

- 建立用來儲存內容的 Amazon S3 儲存貯體
- 將範例內容上傳至 S3 儲存貯體
- 建立原始存取控制 (OAC) 以進行安全的 S3 存取
- 使用 S3 原始伺服器建立 CloudFront 分佈
- 更新 S3 儲存貯體政策以允許 CloudFront 存取
- 等待分佈部署和測試內容存取
- 清除資源，包括分佈、OAC 和 S3 儲存貯體

Bash

AWS CLI 搭配 Bash 指令碼

Note

GitHub 上提供更多範例。在[範例開發人員教學課程](#)儲存庫中尋找完整範例，並了解如何加以設定和執行。

```
#!/bin/bash

# CloudFront Getting Started Tutorial Script
# This script creates an S3 bucket, uploads sample content, creates a CloudFront
# distribution with OAC,
# and demonstrates how to access content through CloudFront.

# Set up logging
LOG_FILE="cloudfront-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting CloudFront Getting Started Tutorial at $(date)"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created before error:"
```

```

    if [ -n "$BUCKET_NAME" ]; then
        echo "- S3 Bucket: $BUCKET_NAME"
    fi
    if [ -n "$OAC_ID" ]; then
        echo "- CloudFront Origin Access Control: $OAC_ID"
    fi
    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "- CloudFront Distribution: $DISTRIBUTION_ID"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "Cleaning up resources..."

    if [ -n "$DISTRIBUTION_ID" ]; then
        echo "Disabling CloudFront distribution $DISTRIBUTION_ID..."

        # Get the current configuration and ETag
        ETAG=$(aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" --
query 'ETag' --output text)
        if [ $? -ne 0 ]; then
            echo "Failed to get distribution config. Continuing with cleanup..."
        else
            # Create a modified configuration with Enabled=false
            aws cloudfront get-distribution-config --id "$DISTRIBUTION_ID" | \
jq '.DistributionConfig.Enabled = false' > temp_disabled_config.json

            # Update the distribution to disable it
            aws cloudfront update-distribution \
                --id "$DISTRIBUTION_ID" \
                --distribution-config file://<(jq '.DistributionConfig'
temp_disabled_config.json) \
                --if-match "$ETAG"

            if [ $? -ne 0 ]; then
                echo "Failed to disable distribution. Continuing with cleanup..."
            else
                echo "Waiting for distribution to be disabled (this may take
several minutes)..."
            fi
        fi
    fi
}

```

```

        aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"

        # Delete the distribution
        ETAG=$(aws cloudfront get-distribution-config --id
"$DISTRIBUTION_ID" --query 'ETag' --output text)
        aws cloudfront delete-distribution --id "$DISTRIBUTION_ID" --if-
match "$ETAG"

        if [ $? -ne 0 ]; then
            echo "Failed to delete distribution. You may need to delete
it manually."
        else
            echo "CloudFront distribution deleted."
        fi
    fi
fi

if [ -n "$OAC_ID" ]; then
    echo "Deleting Origin Access Control $OAC_ID..."
    OAC_ETAG=$(aws cloudfront get-origin-access-control --id "$OAC_ID" --
query 'ETag' --output text 2>/dev/null)
    if [ $? -ne 0 ]; then
        echo "Failed to get Origin Access Control ETag. You may need to
delete it manually."
    else
        aws cloudfront delete-origin-access-control --id "$OAC_ID" --if-match
"$OAC_ETAG"
        if [ $? -ne 0 ]; then
            echo "Failed to delete Origin Access Control. You may need to
delete it manually."
        else
            echo "Origin Access Control deleted."
        fi
    fi
fi

if [ -n "$BUCKET_NAME" ]; then
    echo "Deleting S3 bucket $BUCKET_NAME and its contents..."
    aws s3 rm "s3://$BUCKET_NAME" --recursive
    if [ $? -ne 0 ]; then
        echo "Failed to remove bucket contents. Continuing with bucket
deletion..."
    fi
fi

```

```
aws s3 rb "s3://$BUCKET_NAME"
if [ $? -ne 0 ]; then
    echo "Failed to delete bucket. You may need to delete it manually."
else
    echo "S3 bucket deleted."
fi
fi

# Clean up temporary files
rm -f temp_disabled_config.json
rm -rf temp_content
}

# Generate a random identifier for the bucket name
RANDOM_ID=$(openssl rand -hex 6)
BUCKET_NAME="cloudfront-${RANDOM_ID}"
echo "Using bucket name: $BUCKET_NAME"

# Create a temporary directory for content
TEMP_DIR="temp_content"
mkdir -p "$TEMP_DIR/css"
if [ $? -ne 0 ]; then
    handle_error "Failed to create temporary directory"
fi

# Step 1: Create an S3 bucket
echo "Creating S3 bucket: $BUCKET_NAME"
aws s3 mb "s3://$BUCKET_NAME"
if [ $? -ne 0 ]; then
    handle_error "Failed to create S3 bucket"
fi

# Step 2: Create sample content
echo "Creating sample content..."
cat > "$TEMP_DIR/index.html" << 'EOF'
<!DOCTYPE html>
<html>
<head>
    <title>Hello World</title>
    <link rel="stylesheet" type="text/css" href="css/styles.css">
</head>
<body>
    <h1>Hello world!</h1>
</body>
```



```
</html>
EOF

cat > "$TEMP_DIR/css/styles.css" << 'EOF'
body {
    font-family: Arial, sans-serif;
    margin: 40px;
    background-color: #f5f5f5;
}
h1 {
    color: #333;
    text-align: center;
}
EOF

# Step 3: Upload content to the S3 bucket
echo "Uploading content to S3 bucket..."
aws s3 cp "$TEMP_DIR/" "s3://$BUCKET_NAME/" --recursive
if [ $? -ne 0 ]; then
    handle_error "Failed to upload content to S3 bucket"
fi

# Step 4: Create Origin Access Control
echo "Creating Origin Access Control..."
OAC_RESPONSE=$(aws cloudfront create-origin-access-control \
    --origin-access-control-config Name="oac-for-
$BUCKET_NAME",SigningProtocol=sigv4,SigningBehavior=always,OriginAccessControlOriginType=

if [ $? -ne 0 ]; then
    handle_error "Failed to create Origin Access Control"
fi

OAC_ID=$(echo "$OAC_RESPONSE" | jq -r '.OriginAccessControl.Id')
echo "Created Origin Access Control with ID: $OAC_ID"

# Step 5: Create CloudFront distribution
echo "Creating CloudFront distribution..."

# Get AWS account ID for bucket policy
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)
if [ $? -ne 0 ]; then
    handle_error "Failed to get AWS account ID"
fi
```

```
# Create distribution configuration
cat > distribution-config.json << EOF
{
  "CallerReference": "cli-tutorial-$(date +%s)",
  "Origins": {
    "Quantity": 1,
    "Items": [
      {
        "Id": "S3-$BUCKET_NAME",
        "DomainName": "$BUCKET_NAME.s3.amazonaws.com",
        "S3OriginConfig": {
          "OriginAccessIdentity": ""
        },
        "OriginAccessControlId": "$OAC_ID"
      }
    ]
  },
  "DefaultCacheBehavior": {
    "TargetOriginId": "S3-$BUCKET_NAME",
    "ViewerProtocolPolicy": "redirect-to-https",
    "AllowedMethods": {
      "Quantity": 2,
      "Items": ["GET", "HEAD"],
      "CachedMethods": {
        "Quantity": 2,
        "Items": ["GET", "HEAD"]
      }
    },
    "DefaultTTL": 86400,
    "MinTTL": 0,
    "MaxTTL": 31536000,
    "Compress": true,
    "ForwardedValues": {
      "QueryString": false,
      "Cookies": {
        "Forward": "none"
      }
    }
  },
  "Comment": "CloudFront distribution for tutorial",
  "Enabled": true,
  "WebACLId": ""
}
EOF
```

```

DIST_RESPONSE=$(aws cloudfront create-distribution --distribution-config file://
distribution-config.json)
if [ $? -ne 0 ]; then
    handle_error "Failed to create CloudFront distribution"
fi

DISTRIBUTION_ID=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.Id')
DOMAIN_NAME=$(echo "$DIST_RESPONSE" | jq -r '.Distribution.DomainName')

echo "Created CloudFront distribution with ID: $DISTRIBUTION_ID"
echo "CloudFront domain name: $DOMAIN_NAME"

# Step 6: Update S3 bucket policy
echo "Updating S3 bucket policy..."
cat > bucket-policy.json << EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::$BUCKET_NAME/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::
$ACCOUNT_ID:distribution/$DISTRIBUTION_ID"
                }
            }
        }
    ]
}
EOF

aws s3api put-bucket-policy --bucket "$BUCKET_NAME" --policy file://bucket-
policy.json
if [ $? -ne 0 ]; then
    handle_error "Failed to update S3 bucket policy"
fi

```

```
# Step 7: Wait for distribution to deploy
echo "Waiting for CloudFront distribution to deploy (this may take 5-10
minutes)..."
aws cloudfront wait distribution-deployed --id "$DISTRIBUTION_ID"
if [ $? -ne 0 ]; then
    echo "Warning: Distribution deployment wait timed out. The distribution may
still be deploying."
else
    echo "CloudFront distribution is now deployed."
fi

# Step 8: Display access information
echo ""
echo "==== CloudFront Distribution Setup Complete ====="
echo "You can access your content at: https://$DOMAIN_NAME/index.html"
echo ""
echo "Resources created:"
echo "- S3 Bucket: $BUCKET_NAME"
echo "- CloudFront Origin Access Control: $OAC_ID"
echo "- CloudFront Distribution: $DISTRIBUTION_ID"
echo ""

# Ask user if they want to clean up resources
read -p "Do you want to clean up all resources created by this script? (y/n): "
CLEANUP_RESPONSE
if [[ "$CLEANUP_RESPONSE" =~ ^[Yy] ]]; then
    cleanup
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually delete them later."
    echo "To access your content, visit: https://$DOMAIN_NAME/index.html"
fi

echo "Tutorial completed at $(date)"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。

- [CreateDistribution](#)
- [CreateOriginAccessControl](#)
- [DeleteDistribution](#)
- [DeleteOriginAccessControl](#)

- [GetDistribution](#)
- [GetDistributionConfig](#)
- [GetOriginAccessControl](#)
- [UpdateDistribution](#)
- [WaitDistributionDeployed](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 建立簽章URLs 和 Cookie

下列程式碼範例示範如何建立已簽署的 URL 和 Cookie，以允許存取受限的資源。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

使用 [CannedSignerRequest](#) 類別簽署具有標準政策的 URL 或 Cookie。

```
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;

import java.net.URL;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.temporal.ChronoUnit;

public class CreateCannedPolicyRequest {

    public static CannedSignerRequest createRequestForCannedPolicy(String
distributionDomainName,
        String fileNameToUpload,
        String privateKeyFullPath, String publicKeyId) throws Exception {
```

```
String protocol = "https";
String resourcePath = "/" + fileNameToUpload;

String cloudFrontUrl = new URL(protocol, distributionDomainName,
resourcePath).toString();
Instant expirationDate = Instant.now().plus(7, ChronoUnit.DAYS);
Path path = Paths.get(privateKeyFullPath);

return CannedSignerRequest.builder()
    .resourceUrl(cloudFrontUrl)
    .privateKey(path)
    .keyPairId(publicKeyId)
    .expirationDate(expirationDate)
    .build();
}
}
```

使用 [CustomSignerRequest](#) 類別簽署具有自訂政策的 URL 或 Cookie。activeDate 和 ipRange 是選用方法。

```
import software.amazon.awssdk.services.cloudfront.model.CustomSignerRequest;

import java.net.URL;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.temporal.ChronoUnit;

public class CreateCustomPolicyRequest {

    public static CustomSignerRequest createRequestForCustomPolicy(String
distributionDomainName,
        String fileNameToUpload,
        String privateKeyFullPath, String publicKeyId) throws Exception {
        String protocol = "https";
        String resourcePath = "/" + fileNameToUpload;

        String cloudFrontUrl = new URL(protocol, distributionDomainName,
resourcePath).toString();
        Instant expireDate = Instant.now().plus(7, ChronoUnit.DAYS);
        // URL will be accessible tomorrow using the signed URL.
        Instant activeDate = Instant.now().plus(1, ChronoUnit.DAYS);
```

```

        Path path = Paths.get(privateKeyFullPath);

        return CustomSignerRequest.builder()
            .resourceUrl(cloudFrontUrl)
            // .resourceUrlPattern("https://*.example.com/*") // Optional.
            .privateKey(path)
            .keyPairId(publicKeyId)
            .expirationDate(expireDate)
            .activeDate(activeDate) // Optional.
            // .ipRange("192.168.0.1/24") // Optional.
            .build();
    }
}

```

下列範例示範如何使用 [CloudFrontUtilities](#) 類別來產生已簽署的 URL。在 GitHub 上[檢視](#)此程式碼範例。

```

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.cloudfront.CloudFrontUtilities;
import software.amazon.awssdk.services.cloudfront.cookie.CookiesForCannedPolicy;
import software.amazon.awssdk.services.cloudfront.cookie.CookiesForCustomPolicy;
import software.amazon.awssdk.services.cloudfront.model.CannedSignerRequest;
import software.amazon.awssdk.services.cloudfront.model.CustomSignerRequest;
import software.amazon.awssdk.services.cloudfront.url.SignedUrl;

public class SigningUtilities {
    private static final Logger logger =
        LoggerFactory.getLogger(SigningUtilities.class);
    private static final CloudFrontUtilities cloudFrontUtilities =
        CloudFrontUtilities.create();

    public static SignedUrl signUrlForCannedPolicy(CannedSignerRequest
        cannedSignerRequest) {
        SignedUrl signedUrl =
            cloudFrontUtilities.getSignedUrlWithCannedPolicy(cannedSignerRequest);
        logger.info("Signed URL: [{}]", signedUrl.url());
        return signedUrl;
    }

    public static SignedUrl signUrlForCustomPolicy(CustomSignerRequest
        customSignerRequest) {

```

```

        SignedUrl signedUrl =
cloudFrontUtilities.getSignedUrlWithCustomPolicy(customSignerRequest);
        logger.info("Signed URL: [{}]", signedUrl.url());
        return signedUrl;
    }

    public static CookiesForCannedPolicy
getCookiesForCannedPolicy(CannedSignerRequest cannedSignerRequest) {
        CookiesForCannedPolicy cookiesForCannedPolicy = cloudFrontUtilities
            .getCookiesForCannedPolicy(cannedSignerRequest);
        logger.info("Cookie EXPIRES header [{}]",
cookiesForCannedPolicy.expiresHeaderValue());
        logger.info("Cookie KEYPAIR header [{}]",
cookiesForCannedPolicy.keyPairIdHeaderValue());
        logger.info("Cookie SIGNATURE header [{}]",
cookiesForCannedPolicy.signatureHeaderValue());
        return cookiesForCannedPolicy;
    }

    public static CookiesForCustomPolicy
getCookiesForCustomPolicy(CustomSignerRequest customSignerRequest) {
        CookiesForCustomPolicy cookiesForCustomPolicy = cloudFrontUtilities
            .getCookiesForCustomPolicy(customSignerRequest);
        logger.info("Cookie POLICY header [{}]",
cookiesForCustomPolicy.policyHeaderValue());
        logger.info("Cookie KEYPAIR header [{}]",
cookiesForCustomPolicy.keyPairIdHeaderValue());
        logger.info("Cookie SIGNATURE header [{}]",
cookiesForCustomPolicy.signatureHeaderValue());
        return cookiesForCustomPolicy;
    }
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CloudFrontUtilities](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CloudFront 的 CloudFront Functions 範例

下列程式碼範例示範如何使用 CloudFront 搭配 AWS SDKs。

範例

- [將 HTTP 安全標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將 CORS 標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將快取控制標頭新增至 CloudFront Functions 檢視器回應事件](#)
- [將真正的用戶端 IP 標頭新增至 CloudFront Functions 檢視器請求事件](#)
- [將原始伺服器標頭新增至 CloudFront Functions 檢視器請求事件](#)
- [在 CloudFront Functions 檢視器請求事件中，將 index.html 新增至沒有檔案名稱的請求 URL](#)
- [標準化 CloudFront Functions 檢視器請求中的查詢字串參數](#)
- [重新導向至 CloudFront Functions 檢視器請求事件中的新 URL](#)
- [根據 CloudFront Functions 檢視器請求事件的 KeyValueStore 組態重寫請求 URI](#)
- [在 CloudFront Functions 檢視器請求事件中，將請求路由到更接近檢視器的原始伺服器](#)
- [在 CloudFront Functions 檢視器請求中使用鍵/值對](#)
- [在 CloudFront Functions 檢視器請求中驗證簡單字符](#)

將 HTTP 安全標頭新增至 CloudFront Functions 檢視器回應事件

下列程式碼範例示範如何將 HTTP 安全標頭新增至 CloudFront Functions 檢視器回應事件。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
async function handler(event) {
    var response = event.response;
    var headers = response.headers;

    // Set HTTP security headers
    // Since JavaScript doesn't allow for hyphens in variable names, we use the
    dict["key"] notation
```

```
headers['strict-transport-security'] = { value: 'max-age=63072000;  
includeSubdomains; preload'};  
headers['content-security-policy'] = { value: "default-src 'none'; img-src  
'self'; script-src 'self'; style-src 'self'; object-src 'none'; frame-ancestors  
'none'"};  
headers['x-content-type-options'] = { value: 'nosniff'};  
headers['x-frame-options'] = {value: 'DENY'};  
headers['x-xss-protection'] = {value: '1; mode=block'};  
headers['referrer-policy'] = {value: 'same-origin'};  
  
// Return the response to viewers  
return response;  
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將 CORS 標頭新增至 CloudFront Functions 檢視器回應事件

下列程式碼範例示範如何將 CORS 標頭新增至 CloudFront Functions 檢視器回應事件。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#) 中設定和執行。

```
async function handler(event) {  
  var request = event.request;  
  var response = event.response;  
  
  // If Access-Control-Allow-Origin CORS header is missing, add it.  
  // Since JavaScript doesn't allow for hyphens in variable names, we use the  
  dict["key"] notation.  
  if (!response.headers['access-control-allow-origin'] &&  
    request.headers['origin']) {
```

```
    response.headers['access-control-allow-origin'] = {value:
request.headers['origin'].value};
    console.log("Access-Control-Allow-Origin was missing, adding it now.");
  }

  return response;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將快取控制標頭新增至 CloudFront Functions 檢視器回應事件

下列程式碼範例示範如何將快取控制標頭新增至 CloudFront Functions 檢視器回應事件。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
async function handler(event) {
  var response = event.response;
  var headers = response.headers;

  if (response.statusCode >= 200 && response.statusCode < 400) {
    // Set the cache-control header
    headers['cache-control'] = {value: 'public, max-age=63072000'};
  }

  // Return response to viewers
  return response;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將真正的用戶端 IP 標頭新增至 CloudFront Functions 檢視器請求事件

下列程式碼範例示範如何將真正的用戶端 IP 標頭新增至 CloudFront Functions 檢視器請求事件。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#) 中設定和執行。

```
async function handler(event) {
  var request = event.request;
  var clientIP = event.viewer.ip;

  //Add the true-client-ip header to the incoming request
  request.headers['true-client-ip'] = {value: clientIP};

  return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

將原始伺服器標頭新增至 CloudFront Functions 檢視器請求事件

下列程式碼範例示範如何將原始伺服器標頭新增至 CloudFront Functions 檢視器請求事件。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
async function handler(event) {
  var request = event.request;
  var headers = request.headers;
  var host = request.headers.host.value;

  // If origin header is missing, set it equal to the host header.
  if (!headers.origin)
    headers.origin = {value: `https://${host}`};

  return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在 CloudFront Functions 檢視器請求事件中，將 index.html 新增至沒有檔案名稱的請求 URL

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求事件中，將 index.html 新增至沒有檔案名稱的請求 URL。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#) 中設定和執行。

```
async function handler(event) {
  var request = event.request;
  var uri = request.uri;

  // Check whether the URI is missing a file name.
  if (uri.endsWith('/')) {
    request.uri += 'index.html';
  }
  // Check whether the URI is missing a file extension.
  else if (!uri.includes('.')) {
    request.uri += '/index.html';
  }

  return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

標準化 CloudFront Functions 檢視器請求中的查詢字串參數

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求中標準化查詢字串參數。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
function handler(event) {
    var qs=[];
    for (var key in event.request.querystring) {
        if (event.request.querystring[key].multiValue) {
            event.request.querystring[key].multiValue.forEach((mv) =>
            {qs.push(key + "=" + mv.value)});
        } else {
            qs.push(key + "=" + event.request.querystring[key].value);
        }
    };

    event.request.querystring = qs.sort().join('&');

    return event.request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

重新導向至 CloudFront Functions 檢視器請求事件中的新 URL

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求事件中重新導向至新的 URL。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#) 中設定和執行。

```
async function handler(event) {
    var request = event.request;
    var headers = request.headers;
    var host = request.headers.host.value;
    var country = 'DE' // Choose a country code
    var newurl = `https://${host}/de/index.html`; // Change the redirect URL to
    your choice

    if (headers['cloudfront-viewer-country']) {
        var countryCode = headers['cloudfront-viewer-country'].value;
        if (countryCode === country) {
            var response = {
                statusCode: 302,
                statusDescription: 'Found',
                headers:
                    { "location": { "value": newurl } }
            }

            return response;
        }
    }
    return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

根據 CloudFront Functions 檢視器請求事件的 KeyValueStore 組態重寫請求 URI

下列程式碼範例示範如何根據 CloudFront Functions 檢視器請求事件的 KeyValueStore 組態重寫請求 URI。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#) 中設定和執行。

```
import cf from 'cloudfront';

// (Optional) Replace KVS_ID with actual KVS ID
const kvsId = "KVS_ID";
// enable stickiness by setting a cookie from origin or using another edge
function
const stickinessCookieName = "appversion";
// set to true to enable console logging
const loggingEnabled = false;

// function rewrites the request uri based on configuration in KVS
// example config in KVS in key:value format
// "latest": {"a_weightage": .8, "a_url": "v1", "b_url": "v2"}
// given above key and value in KVS the request uri will be rewritten
// for example http(s)://domain/latest/something/else will be rewritten as
// http(s)://domain/v1/something/else or http(s)://domain/v2/something/else
// depending on weightage
// if no configuration is found, then the request is returned as is
async function handler(event) {
    // NOTE: This example function is for a viewer request event trigger.
    // Choose viewer request for event trigger when you associate this function
    // with a distribution.
    const request = event.request;
    const pathSegments = request.uri.split('/');
    const key = pathSegments[1];
```

```

    // if empty path segment or if there is valid stickiness cookie
    // then skip call to KVS and let the request continue.
    if (!key || hasValidStickinessCookie(request.cookies[stickinessCookieName],
key)) {
        return event.request;
    }

    try {
        // get the prefix replacement from KVS
        const replacement = await getPathPrefixByWeightage(key);
        if (!replacement) {
            return event.request;
        }
        //Replace the first path with the replacement
        pathSegments[1] = replacement;
        log(`using prefix ${pathSegments[1]}`)
        const newUri = pathSegments.join('/');
        log(`${request.uri} -> ${newUri}`);
        request.uri = newUri;

        return request;
    } catch (err) {
        // No change to the path if the key is not found or any other error
        log(`request uri: ${request.uri}, error: ${err}`);
    }
    // no change to path - return request
    return event.request;
}

// function to get the prefix from KVS
async function getPathPrefixByWeightage(key) {
    const kvsHandle = cf.kvs(kvsId);
    // get the weightage config from KVS
    const kvsResponse = await kvsHandle.get(key);
    const weightageConfig = JSON.parse(kvsResponse);
    // no configuration - return null
    if (!weightageConfig || !isFinite(weightageConfig.a_weightage)) {
        return null;
    }
    // return the url based on weightage
    // return null if no url is configured
    if (Math.random() <= weightageConfig.a_weightage) {
        return weightageConfig.a_url ? weightageConfig.a_url: null;
    }
}

```

```
    } else {
        return weightageConfig.b_url ? weightageConfig.b_url : null;
    }
}

// function to check if the stickiness cookie is valid
function hasValidStickinessCookie(stickinessCookie, pathSegment) {
    // if the value exists and it matches pathSegment
    return (stickinessCookie && stickinessCookie.value === pathSegment)
}

function log(message) {
    if (loggingEnabled) {
        console.log(message);
    }
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在 CloudFront Functions 檢視器請求事件中，將請求路由到更接近檢視器的原始伺服器

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求事件中，將請求路由至更接近檢視器的原始伺服器。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例儲存庫](#)中設定和執行。

```
import cf from 'cloudfront';

function handler(event) {
```

```
const request = event.request;
const headers = request.headers;
const country = headers['cloudfront-viewer-country'] &&
    headers['cloudfront-viewer-country'].value;

//List of Regions with S3 buckets containing content
const countryToRegion = {
    'DE': 'eu-central-1',
    'IE': 'eu-west-1',
    'GB': 'eu-west-2',
    'FR': 'eu-west-3',
    'JP': 'ap-northeast-1',
    'IN': 'ap-south-1'
};

const DEFAULT_REGION = 'us-east-1';

const selectedRegion = (country && countryToRegion[country]) ||
    DEFAULT_REGION;

const domainName =
    `cloudfront-functions-demo-bucket-in-${selectedRegion}.s3.
    ${selectedRegion}.amazonaws.com`;

cf.updateRequestOrigin({
    "domainName": domainName,
    "originAccessControlConfig": {
        "enabled": true,
        "region": selectedRegion,
        "signingBehavior": "always",
        "signingProtocol": "sigv4",
        "originType": "s3"
    },
});

return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在 CloudFront Functions 檢視器請求中使用鍵/值對

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求中使用鍵/值對。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
import cf from 'cloudfront';

// This fails if there is no key value store associated with the function
const kvsHandle = cf.kvs();

// Remember to associate the KVS with your function before referencing KVS in
// your code.
// https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/kvs-with-
// functions-associate.html
async function handler(event) {
  const request = event.request;
  // Use the first segment of the pathname as key
  // For example http(s)://domain/<key>/something/else
  const pathSegments = request.uri.split('/')
  const key = pathSegments[1]
  try {
    // Replace the first path of the pathname with the value of the key
    // For example http(s)://domain/<value>/something/else
    pathSegments[1] = await kvsHandle.get(key);
    const newUri = pathSegments.join('/');
    console.log(`${request.uri} -> ${newUri}`)
    request.uri = newUri;
  } catch (err) {
    // No change to the pathname if the key is not found
    console.log(`${request.uri} | ${err}`);
  }
  return request;
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在 CloudFront Functions 檢視器請求中驗證簡單字符

下列程式碼範例示範如何在 CloudFront Functions 檢視器請求中驗證簡易字符。

JavaScript

適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [CloudFront Functions 範例](#) 儲存庫中設定和執行。

```
import crypto from 'crypto';
import cf from 'cloudfront';

//Response when JWT is not valid.
const response401 = {
  statusCode: 401,
  statusDescription: 'Unauthorized'
};

// Remember to associate the KVS with your function before calling the const
kvsKey = 'jwt.secret'.
// https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/kvs-with-
functions-associate.html
const kvsKey = 'jwt.secret';
// set to true to enable console logging
const loggingEnabled = false;

function jwt_decode(token, key, noVerify, algorithm) {
  // check token
  if (!token) {
    throw new Error('No token supplied');
```

```
}
// check segments
const segments = token.split('.');
if (segments.length !== 3) {
    throw new Error('Not enough or too many segments');
}

// All segment should be base64
const headerSeg = segments[0];
const payloadSeg = segments[1];
const signatureSeg = segments[2];

// base64 decode and parse JSON
const payload = JSON.parse(_base64urlDecode(payloadSeg));

if (!noVerify) {
    const signingMethod = 'sha256';
    const signingType = 'hmac';

    // Verify signature. `sign` will return base64 string.
    const signingInput = [headerSeg, payloadSeg].join('.');

    if (!_verify(signingInput, key, signingMethod, signingType,
signatureSeg)) {
        throw new Error('Signature verification failed');
    }

    // Support for nbf and exp claims.
    // According to the RFC, they should be in seconds.
    if (payload.nbf && Date.now() < payload.nbf*1000) {
        throw new Error('Token not yet active');
    }

    if (payload.exp && Date.now() > payload.exp*1000) {
        throw new Error('Token expired');
    }
}

return payload;
}

//Function to ensure a constant time comparison to prevent
//timing side channels.
function _constantTimeEquals(a, b) {
```

```
    if (a.length !== b.length) {
      return false;
    }

    let xor = 0;
    for (let i = 0; i < a.length; i++) {
      xor |= (a.charCodeAt(i) ^ b.charCodeAt(i));
    }

    return 0 === xor;
  }

function _verify(input, key, method, type, signature) {
  if(type === "hmac") {
    return _constantTimeEquals(signature, _sign(input, key, method));
  }
  else {
    throw new Error('Algorithm type not recognized');
  }
}

function _sign(input, key, method) {
  return crypto.createHmac(method, key).update(input).digest('base64url');
}

function _base64urlDecode(str) {
  return Buffer.from(str, 'base64url')
}

async function handler(event) {
  let request = event.request;

  //Secret key used to verify JWT token.
  //Update with your own key.
  const secret_key = await getSecret()

  if(!secret_key) {
    return response401;
  }

  // If no JWT token, then generate HTTP redirect 401 response.
  if(!request.querystring.jwt) {
    log("Error: No JWT in the querystring");
    return response401;
  }
}
```



```
    }

    const jwtToken = request.querystring.jwt.value;

    try{
        jwt_decode(jwtToken, secret_key);
    }
    catch(e) {
        log(e);
        return response401;
    }

    //Remove the JWT from the query string if valid and return.
    delete request.querystring.jwt;
    log("Valid JWT token");
    return request;
}

// get secret from key value store
async function getSecret() {
    // initialize cloudfront kv store and get the key value
    try {
        const kvsHandle = cf.kvs();
        return await kvsHandle.get(kvsKey);
    } catch (err) {
        log(`Error reading value for key: ${kvsKey}, error: ${err}`);
        return null;
    }
}

function log(message) {
    if (loggingEnabled) {
        console.log(message);
    }
}
}
```

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [以 AWS SDK 使用 CloudFront](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

文件歷史紀錄

以下表格說明 CloudFront 文件中的重要變更。如需獲取更新通知，您可以[訂閱 RSS 摘要](#)。

變更	描述	日期
新增交互 TLS (檢視器)	CloudFront 支援交互 TLS (檢視器)。	2025 年 11 月 24 日
新增存取日誌的日誌欄位	新增存取日誌 (標準日誌) 和即時存取日誌 connection-id 的欄位。	2025 年 11 月 24 日
新增連線日誌	新增連線日誌做為交互 TLS (檢視器) 的新記錄功能。	2025 年 11 月 24 日
新增連線函數	CloudFront 支援交互 TLS (檢視器) 的連線函數。	2025 年 11 月 24 日
新增使用 IPAM 將您自己的 IP 帶入 CloudFront 的支援	CloudFront 支援透過 IPAM 的 BYOIP 為全域服務帶來您自己的 IPv4 地址。	2025 年 11 月 24 日
AWS 受管政策更新	CloudFrontReadOnlyAccess 和 CloudFrontFullAccess IAM 政策現在支援 ec2:DescribeIpamPools 和 ec2:GetIpamPoolCidrs 動作。	2025 年 11 月 24 日
CloudFront Functions 的更新	此版本新增 CloudFront Functions 中原始伺服器修改協助程式方法的參數。您可以使用 hostHeader、sni、allowedCertificateNames 和 originOverride 參數。	2025 年 11 月 20 日

CloudFront Functions 的更新	新增 CloudFront Functions 的 CBOR Web Token (CWT) 支援。	2025 年 11 月 20 日
CloudFront Functions 的更新	新增 CloudFront Functions 的一般協助程式方法。	2025 年 11 月 20 日
更新 AWS 受管政策	更新 CloudFrontFullAccess 以允許許可建立 AWS WAF Web ACL 資源，以及建立、更新、刪除和讀取 AWS Pricing Plan Manager。	2025 年 11 月 18 日
更新 AWS 受管政策	更新 CloudFrontReadOnlyAccess 以允許對 AWS Pricing Plan Manager 的唯讀許可。	2025 年 11 月 18 日
更新 AWS 受管政策	更新 CloudFrontFullAccess 以允許許可建立 AWS WAF Web ACL 資源，以及建立、更新、刪除和讀取 AWS Pricing Plan Manager。	2025 年 11 月 18 日
更新 AWS 受管政策	更新 CloudFrontReadOnlyAccess 以允許對 AWS Pricing Plan Manager 的唯讀許可。	2025 年 11 月 18 日
CloudFront 支援固定費率定價計劃	您現在可以訂閱 CloudFront 固定費率定價計劃的分佈。	2025 年 11 月 18 日
Anycast 靜態 IP	您現在可以選擇僅 IPv4 地址或 IPv4 和 IPv6 地址（雙堆疊）。	2025 年 11 月 5 日

[新增跨 共用 VPC 原始伺服器的支援 AWS 帳戶](#)

您可以建立資源共享，並將 VPC 原始伺服器新增至其中。這可讓您將 VPC 原始伺服器和 CloudFront 分佈分開 AWS 帳戶，讓您的組織能夠維持多帳戶需求。其他 AWS 帳戶 可以為其 CloudFront 分佈關聯共用 VPC 原始伺服器。

2025 年 11 月 5 日

[已新增檢視器安全政策](#)

已新增 TLSv1.2_2025 安全政策。

2025 年 10 月 10 日

- [檢視器和 CloudFront 之間支援的通訊協定和密碼](#)
- [安全政策 \(最低 SSL/TLS 版本\)](#)

[已新增後量子 \(PQ\) 金鑰交換演算法](#)

已將兩個新的 PQ 金鑰交換演算法新增至 CloudFront TLS 政策。

2025 年 9 月 4 日

[IPv6 原始伺服器請求](#)

當您使用自訂原始伺服器 (不含 Amazon S3 和 VPC 原始伺服器) 時，您可以自訂分佈的原始伺服器設定，以選擇 CloudFront 如何使用 IPv4 或 IPv6 位址連接到原始伺服器。

2025 年 9 月 3 日

[已新增新的受管原始伺服器請求政策](#)

已新增新的受管原始伺服器請求政策 HostHeaderOnly 。

2025 年 8 月 29 日

[CloudFront 公有端點現在支援 IPv6](#)

請參閱《Amazon VPC 使用者指南》中的 AWS 一般參考和[支援 IPv6 的 AWS 服務](#) 中的 [Amazon CloudFront 端點和配額](#)。

2025 年 8 月 21 日

已新增檢視器安全政策	已新增 TLSv1.3_2025，這是僅限 TLS 1.3 的新安全政策。	2025 年 8 月 7 日
已新增新的原始伺服器逾時設定	已新增所有原始伺服器的回應完成逾時，並新增 S3 原始伺服器的回應逾時 (原始伺服器讀取逾時)。	2025 年 7 月 30 日
已新增預先設定的標準分佈設定	已新增標準分佈的預先設定。	2025 年 6 月 17 日
已新增適用於標準分佈網域設定的新主控台工作流程。	已新增適用於標準分佈網域設定的新主控台工作流程。	2025 年 6 月 17 日
已新增範例參數	已新增在分佈租用戶中使用參數搭配網域名稱和原始伺服器路徑的範例。	2025 年 6 月 17 日
已新增對 CloudFront SaaS Manager 的 CloudFront Functions 支援	已新增協助程式函數和 context 物件的 endpoint 欄位。	2025 年 5 月 2 日
標準記錄 (v2) 的更新	已新增 {distributionid} 分割區變數，以支援將存取日誌傳送至 AWS Glue。	2025 年 5 月 1 日
CloudFront 受管政策的更新	已將 ACM 許可新增至 CloudFrontReadOnlyAccess 和 CloudFrontFullAccess 受管政策。	2025 年 4 月 28 日

[已新增對多租用戶分佈和分佈租用戶的支援](#)

您可以建立多租用戶分佈，以根據您的原始伺服器類型設定常見的分佈設定。然後，您可以重複使用多租用戶分佈來建立多個共用這些設定的分佈租用戶。然後，您可以在新增其他網站或應用程式時自訂特定的分佈租用戶。

2025 年 4 月 28 日

[Lambda@Edge 函數的更新](#)

Lambda@Edge 函數現在支援進階記錄控制和自訂 CloudWatch 日誌群組名稱。

2025 年 4 月 7 日

[Anycast 靜態 IP](#)

您可以使用 Anycast 靜態 IP 位址，讓 apex 網域直接路由到您的 CloudFront 分佈。

2025 年 4 月 4 日

[已新增用於原始伺服器修改的其他協助程式方法](#)

已新增 `selectRequestOriginById()` 和 `createRequestOriginGroup()` CloudFront Functions 協助程式方法。

2025 年 4 月 2 日

[標準記錄 \(v2\) 的更新](#)

已新增 `{accountid}` 分割區變數和範例尾碼路徑，以將存取日誌交付至 Amazon S3。

2025 年 2 月 14 日

[新增標準記錄 \(v2\) 的其他即時存取日誌欄位](#)

您可以在啟用標準記錄 (v2) 時指定 `c-country` 和 `cache-behavior-path-pattern` 即時存取日誌欄位。

2025 年 1 月 31 日

[Lambda@Edge 支援較新的執行時間版本](#)

Lambda@Edge 現在支援具有 Node.js 22 執行時期的 Lambda 函數。

2024 年 11 月 22 日

CloudFront 的媒體品質感知彈性支援	您可以使用媒體品質感知彈性 (MQAR) 功能，讓 CloudFront 自動選擇媒體品質分數最高的原始伺服器群組中的原始伺服器。	2024 年 11 月 21 日
原始伺服器修改的協助程式方法	已新增用於原始伺服器修改的新 CloudFront Functions 協助程式方法。	2024 年 11 月 21 日
VPC 原始伺服器	使用 CloudFront VPC 原始伺服器來限制對 Application Load Balancer、Network Load Balancer 或 EC2 執行個體原始伺服器的存取。	2024 年 11 月 20 日
受管政策的更新項目	更新了 CloudFrontFullAccess 受管政策。	2024 年 11 月 20 日
Anycast 靜態 IP	您可以從 CloudFront 請求 Anycast 靜態 IP 以搭配您的分佈使用。	2024 年 11 月 20 日
已新增對標準記錄的支援	CloudFront 支援標準記錄 (v2) 並將日誌傳送至 Amazon CloudWatch Logs、Amazon Data Firehose 和 Amazon Simple Storage Service (Amazon S3)。	2024 年 11 月 20 日
已新增對 gRPC 的支援	CloudFront 現在支援分佈的 gRPC 請求。	2024 年 11 月 20 日
已新增 VPC 原始伺服器的新受管政策	已新增新的受管政策 AWSCloudFrontVPCOriginServiceRolePolicy。	2024 年 11 月 20 日

Lambda@Edge 支援較新的執行時間版本	Lambda@Edge 現在支援具有 Python 3.13 執行時間的 Lambda 函數。	2024 年 11 月 13 日
使用 AWS Config 規則評估	使用 AWS Config 規則評估您的 CloudFront 組態。	2024 年 9 月 20 日
已新增更多疑難排解內容	已針對 HTTP 4xx 和 5xx 錯誤回應狀態碼新增更多疑難排解內容。	2024 年 8 月 26 日
已新增新的受管快取政策	已新增新的受管快取政策 UseOriginCacheControlHeaders 和 UseOriginCacheControlHeaders-QueryString 。	2024 年 5 月 24 日
已新增原始存取控制支援	您現在可以建立 for AWS Elemental MediaPackage V2 和 AWS Lambda 函數 URL 的原始存取控制 (OAC)。	2024 年 4 月 11 日
CMCD 的即時存取日誌欄位	新增 18 個適用於即時存取日誌的常見媒體用戶端資料 (CMCD) 欄位。	2024 年 4 月 9 日
開始使用 CloudFront 標準分佈	已更新使用 Amazon S3 原始伺服器搭配原始存取控制 (OAC) 的標準分佈教學課程。	2024 年 3 月 18 日
CloudFront AWS SDKs 的程式碼範例	新增程式碼範例，示範如何搭配 AWS 軟體開發套件 (SDK) 使用 CloudFront。這些範例分為程式碼摘錄 (示範如何呼叫個別服務函數) 和範例 (示範如何透過呼叫相同服務中的多個函數來完成特定任務)。	2024 年 2 月 16 日

AWS 受管政策	CloudFrontReadOnly Access 和 CloudFrontFullAccess IAM 政策現在支援 KeyValueStore 操作。	2023 年 12 月 19 日
JavaScript 執行時間 2.0	已新增適用於 CloudFront Functions 的 JavaScript 執行時期 2.0 功能。	2023 年 11 月 21 日
CloudFront KeyValueStore	Amazon CloudFront 現在支援 CloudFront KeyValueStore。此功能是安全、全域、低延遲的鍵值資料儲存，可從 CloudFront Functions 內部進行讀取存取。您可以在 CloudFront 邊緣節點啟用進階可自訂邏輯。	2023 年 11 月 21 日
Lambda@Edge 支援較新的執行時間版本	Lambda@Edge 現在支援具有 Node.js 20 執行時期的 Lambda 函數。	2023 年 11 月 15 日
安全性儀表板	當您建立分佈時，CloudFront 會建立一個安全性儀表板。啟用 AWS WAF、管理地理限制，以及檢視請求、機器人和日誌的高階資料。	2023 年 11 月 8 日
排序函數中的查詢字串	CloudFront 現在支援使用 CloudFront Functions 進行查詢字串排序。	2023 年 10 月 3 日
AWS WAF 安全建議	Amazon CloudFront 現在會在 CloudFront 主控台上顯示 AWS WAF 安全建議。	2023 年 9 月 26 日

提供過時 (過期) 快取內容的支援	CloudFront 支援 Stale-While-Revalidate 和 Stale-If-Error 快取控制指令。	2023 年 5 月 15 日
一鍵啟用 AWS WAF 保護	將 AWS WAF 安全性保護新增至 CloudFront 分佈的簡化方法。	2023 年 5 月 10 日
為用於標準日誌的新 S3 儲存貯體啟用 ACL	已新增附註和連結，以解決新 S3 儲存貯體的預設 ACL 設定。	2023 年 4 月 11 日
使用 Amazon S3 Object Lambda 建立原始伺服器	您可以使用 Amazon S3 Object Lambda 存取點別名作為您發佈的原始伺服器。	2023 年 3 月 31 日
使用 CloudFront Functions 自訂 HTTP 狀態和本文	您可以使用 CloudFront Functions 來更新檢視器回應狀態碼，以及取代或移除回應本文。	2023 年 3 月 29 日
已為連接埠新增 CORS 標頭萬用字元選項	您現在可以在 CORS 存取控制標頭中包含連接埠的萬用字元組態。	2023 年 3 月 20 日
新增 AWS Security Hub CSPM 使用者指南的新連結	已更新語言，並新增 AWS Security Hub CSPM 使用者指南中重新組織的 Amazon CloudFront 控制項連結。	2023 年 3 月 9 日
CloudFront 現在支援原始伺服器請求政策中的封鎖清單 (「全部，除了」)	使用原始伺服器請求政策中的封鎖清單，將所有查詢字串、HTTP 標頭或 Cookie (指定的字串除外) 納入 CloudFront 傳送至原始伺服器的請求中。	2023 年 2 月 22 日

CloudFront 新增新的受管原始伺服器請求政策，以轉寄主機標頭以外的所有檢視器標頭	使用 CloudFront 的新受管原始伺服器請求政策，將檢視器請求中的所有標頭 (Host 標頭除外) 納入 CloudFront 傳送至原始伺服器的請求中。	2023 年 2 月 22 日
更新對 Lambda@Edge 的限制	Lambda@Edge 支援設為 Auto (自動) 的 Lambda 執行時期管理組態。	2023 年 2 月 16 日
更新適用於 CloudFront 的 IAM 指導方針	更新了指南以符合 IAM 最佳實務。如需更多詳細資訊，請參閱 IAM 中的安全最佳實務 。	2023 年 2 月 15 日
擁有增強型安全的原始伺服器存取控制	您現在可以僅允許存取指定的 CloudFront 分佈，來確保 MediaStore 原始伺服器的安全。	2023 年 2 月 9 日
用於判斷檢視者標頭結構的新標頭	您現在可以新增標頭順序和數量，協助根據檢視者傳送的標頭來識別檢視者。	2023 年 1 月 13 日
Lambda@Edge 支援較新的執行時間版本	Lambda@Edge 現在支援具有 Node.js 18 執行時間的 Lambda 函數。	2023 年 1 月 12 日
使用回應標頭政策移除回應標頭	您現在可以使用 CloudFront 回應標頭政策，從原始伺服器移除 CloudFront 在回應中收到的標頭。指定的標頭不包含在 CloudFront 傳送給檢視者的回應中。	2023 年 1 月 3 日
持續部署以安全測試組態變更	您現在可以測試生產流量子集，以將變更部署到 CDN 組態。	2022 年 11 月 18 日

發行 CloudFront-Viewer-JA3-Fingerprint 標頭	您現在可以使用 JA3 指紋，協助判斷請求是否來自已知用戶端。	2022 年 11 月 16 日
已新增 CORS 標頭萬用字元選項	您現在可以在部份 CORS 存取控制標頭中使用各種萬用字元組態。	2022 年 11 月 11 日
CloudFront 分佈的其他指標	在 CloudFront API 和 CloudFormation 中支援 MonitoringSubscription 。	2022 年 10 月 3 日
擁有增強型安全的原始伺服器存取控制	您現在可以僅允許存取指定的 CloudFront 分佈，來確保 Amazon S3 原始伺服器的安全。	2022 年 8 月 24 日
支援 CloudFront 分佈的 HTTP/3	您現在可以為您的 CloudFront 分佈選擇 HTTP/3。	2022 年 8 月 15 日
將交握詳細資訊新增至 CloudFront-Viewer-TLS 標頭	您可以檢視所用 SSL/TLS 交握的資訊。	2022 年 6 月 27 日
Server-Timing 標頭中的新指標	已將新 cdn-downstream-fb1 指標新增至 Server-Timing 標頭。	2022 年 6 月 13 日
用於獲取有關 TLS 版本和密碼資訊的新標頭	您現在可以使用 CloudFront-Viewer-TLS 標頭來獲取有關 TLS (或 SSL) 版本以及用於檢視工具和 CloudFront 之間連線密碼的資訊。	2022 年 5 月 23 日
CloudFront Functions 的新 FunctionThrottles 指標	使用 Amazon CloudWatch，您現在可以監控在特定期間內限流 CloudFront Function 的次數。	2022 年 5 月 4 日

[CloudFront 支援 Lambda 函數 URL](#)

使用含有函數 URL 的 Lambda 函數建置無伺服器 Web 應用程式，可讓 CloudFront 享有一系列優勢。

2022 年 4 月 6 日

[HTTP 回應中的 Server-Timing 標頭](#)

您現在可以為 CloudFront 傳送的 HTTP 回應啟用 Server-Timing 標頭，以查看可幫助您洞察 CloudFront 行為和效能的指標。

2022 年 3 月 30 日

[使用 AWS 受管字首清單來限制傳入流量](#)

您現在可以限制連至原始伺服器的傳入 HTTP 和 HTTPS 流量，使其僅限從屬於 CloudFront 面對原始伺服器的伺服器之 IP 位址發出。

2022 年 2 月 7 日

[新功能](#)

CloudFront 新增對回應標頭政策的支援，該政策可讓您指定 CloudFront 新增至其傳送給檢視者 (Web 瀏覽器或其他用戶端) 之 HTTP 回應的 HTTP 標頭。您可以指定所需的標頭 (及其值)，而不需對原始伺服器進行任何變更或撰寫任何程式碼。如需詳細資訊，請參閱 [CloudFront 回應中的新增或移除 HTTP 標頭](#)。

2021 年 11 月 2 日

[新的 CloudFront-Viewer-Address 請求標頭](#)

CloudFront 新增了對新標頭 CloudFront-Viewer-Address 的支援，其中包含將 HTTP 請求傳送到 CloudFront 的檢視者的 IP 位址。如需詳細資訊，請參閱 [新增 CloudFront 請求標頭](#)。

2021 年 10 月 25 日

[Lambda@Edge 支援新的執行時間版本](#)

Lambda@Edge 現在支援具有 Python 3.9 執行時間的 Lambda 函數。如需更多資訊，請參閱[支援的執行時期](#)。

2021 年 9 月 22 日

[AWS 受管政策更新](#)

CloudFront 已更新 CloudFrontReadOnlyAccess 政策。如需詳細資訊，請參閱[CloudFront AWS 受管政策的更新](#)。

2021 年 9 月 8 日

[新功能](#)

CloudFront 現在支援 ECDSA 憑證 (適用於面向檢視器的 HTTPS 連線)。如需詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)，以及[搭配 CloudFront 使用 SSL/TLS 憑證的要求](#)。

2021 年 7 月 14 日

[新功能](#)

CloudFront 現在支援更多方式將替代網域名稱從一個分佈移至另一個分佈，而無需聯絡支援。如需詳細資訊，請參閱[將替代網域名稱移至不同的分佈](#)。

2021 年 7 月 7 日

[新的安全政策](#)

CloudFront 現在能使用少數可用的加密方法，支援新的安全政策 TLSv1.2_2021。如需詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

2021 年 6 月 23 日

新功能

Amazon CloudFront 現在支援 CloudFront Functions，CloudFront 的原生功能可讓您在 JavaScript 中撰寫輕量型函數，以進行高擴展性、延遲敏感的 CDN 自訂。如需詳細資訊，請參閱[使用 CloudFront Functions 在邊緣進行自訂](#)。

2021 年 5 月 3 日

Lambda@Edge 支援較新的執行時間版本

Lambda@Edge 現在支援具有 Node.js 14 執行時間的 Lambda 函數。如需更多資訊，請參閱[支援的執行時期](#)。

2021 年 4 月 29 日

移除 RTMP 分佈的文件

[Amazon CloudFront 已在 2020 年 12 月 31 日停用即時簡訊協定 \(RTMP\) 分佈](#)。RTMP 分佈的文件現在已從 Amazon CloudFront 開發人員指南中移除。

2021 年 2 月 10 日

新的定價選項

Amazon CloudFront 推出簡單易用的 CloudFront 安全節省服務包，可讓您的 AWS 帳單節省高達 30% 的 CloudFront 費用。如需詳細資訊，請參閱優惠組合[常見問答集](#)。

2021 年 2 月 5 日

新的教學課程

Amazon CloudFront 開發人員指南現在包含使用 Amazon CloudFront 限制存取 ELB 中 Application Load Balancer 的教學課程。如需詳細資訊，請參閱[限制對 Application Load Balancer 的存取](#)。

2020 年 12 月 18 日

公有金鑰管理的新選項

CloudFront 現在支援透過 CloudFront 主控台和 API 管理已簽署 URLs 和已簽署 Cookie 的公有金鑰，而無需存取 AWS 帳戶 根使用者。如需詳細資訊，請參閱[指定可以建立已簽署 URL 和已簽署 Cookie 的簽署者](#)。

2020 年 10 月 22 日

新功能 – Origin Shield

CloudFront 現在支援 CloudFront Origin Shield，這是 CloudFront 快取基礎設施中的額外層，可協助您將原始伺服器的負載降至最低、改善其可用性，並降低營運成本。如需詳細資訊，請參閱[使用 Amazon CloudFront Origin Shield](#)。

2020 年 10 月 20 日

新的壓縮格式

現在，當您設定 CloudFront 壓縮位在 CloudFront 節點的物件時，CloudFront 可以支援 Brotli 壓縮格式。您也可以使用標準化的 Accept-Encoding 標頭設定 CloudFront 快取 Brotli 物件。如需詳細資訊，請參閱[提供壓縮檔案](#)和[壓縮支援](#)。

2020 年 9 月 14 日

新的 TLS 通訊協定

CloudFront 現在支援 TLS 1.3 協定，供檢視器和 CloudFront 發佈間的 HTTPS 連線使用。根據預設，在所有 CloudFront 安全政策中都會啟用 TLS 1.3。如需詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

2020 年 9 月 3 日

[新的即時存取日誌](#)

CloudFront 現在支援可設定的即時存取日誌。透過即時存取日誌，您可以即時取得對分佈提出之請求的相關資訊。您可以使用即時存取日誌，根據內容交付效能來監控、分析和採取行動。如需詳細資訊，請參閱[即時日誌](#)。

2020 年 8 月 31 日

[API 支援其他指標](#)

CloudFront 現在支援以 CloudFront API 啟用八個額外的即時指標。如需詳細資訊，請參閱[開啟其他指標](#)。

2020 年 8 月 28 日

[新的 CloudFront HTTP 標頭](#)

CloudFront 新增的額外 HTTP 標頭，可判斷裝置類型、地理位置等等檢視器的相關資訊。如需詳細資訊，請參閱[新增 CloudFront 請求標頭](#)。

2020 年 7 月 23 日

[新功能](#)

CloudFront 現在支援快取政策和原始伺服器請求政策，讓您更精細地控制 CloudFront 發佈的快取金鑰和原始伺服器請求。如需詳細資訊，請參閱[控制快取金鑰](#)和[控制原始伺服器請求](#)。

2020 年 7 月 22 日

[新的安全政策](#)

CloudFront 現在可使用一組較小的支援加密，支援新的安全政策 TLSv1.2_2019。如需詳細資訊，請參閱[檢視器和 CloudFront 之間支援的通訊協定和密碼](#)。

2020 年 7 月 8 日

[控制原始伺服器逾時和嘗試的新設定](#)

CloudFront 新增了控制原始伺服器逾時和嘗試次數的新設定。如需詳細資訊，請參閱[控制原始伺服器逾時和嘗試](#)。

2020 年 6 月 5 日

[建立安全靜態網站以開始使用 CloudFront 的新文件](#)

使用 Amazon S3、CloudFront、Lambda@Edge 等等建立安全的靜態網站，藉以開始使用 CloudFront，全部都是使用 CloudFormation 部署。如需詳細資訊，請參閱[安全靜態網站入門](#)。

2020 年 6 月 2 日

[Lambda@Edge 支援較新的執行時間版本](#)

Lambda@Edge 現在支援具有 Node.js 12 和 Python 3.8 執行時間的 Lambda 函數。如需更多資訊，請參閱[支援的執行時期](#)。

2020 年 2 月 27 日

[CloudWatch 中的新即時指標](#)

現在，Amazon CloudFront 在 Amazon CloudWatch 中新增了八個額外的即時指標。如需詳細資訊，請參閱[開啟其他 CloudFront 分佈指標](#)。

2019 年 12 月 19 日

[存取日誌中的新欄位](#)

CloudFront 在存取日誌中新增七個新欄位。如需詳細資訊，請參閱[標準日誌檔案欄位](#)。

2019 年 12 月 12 日

[AWS WordPress 外掛程式](#)

您可以使用 AWS WordPress 外掛程式，為 WordPress 網站的訪客提供使用 CloudFront 的加速檢視體驗。(更新：自 2022 年 9 月 30 日起，WordPress 外掛程式的 AWS 已棄用。)

2019 年 10 月 30 日

[標籤型和資源層級 IAM 許可政策](#)

CloudFront 現在支援其他兩種指定 IAM 許可政策的方式：標籤型和資源層級的政策許可。如需詳細資訊，請參閱[管理資源存取](#)。

2019 年 8 月 8 日

[支援 Python 程式設計語言](#)

除了 Node.js 之外，您現在可以使用 Python 程式設計語言來開發 Lambda@Edge 中的函數。如需涵蓋各種案例的範例函數，請參閱 [Lambda@Edge 範例函數](#)。

2019 年 8 月 1 日

[更新監控圖表](#)

內容更新以描述直接從 CloudFront 主控台監控與 CloudFront 分佈關聯的 Lambda 函數的新方法，以便更輕鬆地追蹤和除錯錯誤。如需詳細資訊，請參閱[監控 CloudFront](#)。

2019 年 6 月 20 日

[合併的安全內容](#)

全新的安全性章節整合了有關 CloudFront 的功能和資料保護實作、IAM、記錄、合規等資訊。如需詳細資訊，請參閱[安全性](#)。

2019 年 5 月 24 日

[現在需要網域驗證](#)

CloudFront 現在需要您使用 SSL 憑證來驗證您有搭配分發使用替代網域名稱的許可。如需詳細資訊，請參閱[使用備用網域名稱與 HTTPS](#)。

2019 年 4 月 9 日

[更新的 PDF 檔案名稱](#)

《Amazon CloudFront 開發人員指南》的新檔案名稱是：AmazonCloudFront_DevGuide。先前檔名是：cf-dg。

2019 年 1 月 7 日

新功能

CloudFront 現在支援 WebSocket，當您在用戶端與伺服器之間需要長時間的連線時，這是一種很實用的 TCP 型協定。對於需要高可用性的案例，您現在也可以將 CloudFront 設定為具有原始伺服器容錯移轉。如需詳細資訊，請參閱[使用 WebSocket 與 CloudFront 發佈](#)以及[使用 CloudFront 原始伺服器容錯移轉最佳化高可用性](#)。

2018 年 11 月 20 日

新功能

CloudFront 現在支援執行 Lambda 函數之 HTTP 請求的詳細錯誤記錄日誌。當您的函數會傳回無效的回應時，您可以將日誌存放在 CloudWatch 中，用以故障診斷 HTTP 5xx 錯誤。如需詳細資訊，請參閱[適用於 Lambda 函數的 CloudWatch 指標和 CloudWatch Logs](#)。

2018 年 10 月 8 日

新功能

您現在可以選擇讓 Lambda@Edge 在可寫入的 HTTP 方法 (POST、PUT 和 DELETE 等) 中公開請求的內文，如此您就能在 Lambda 函數中存取該內文。您可以選擇唯讀存取或指定您將替換內文。如需詳細資訊，請參閱[選擇 Include Body \(包含內文\) 選項以存取請求內文](#)。

2018 年 8 月 14 日

新功能

CloudFront 現可支援用 brotli 或 gzip 之外的其他壓縮演算法提供壓縮內容。如需詳細資訊，請參閱[提供壓縮檔案](#)。

2018 年 7 月 25 日

重組

《Amazon CloudFront 開發人員指南》經過重組，簡化了尋找相關內容的方式並改進了可瀏覽性和導覽。

2018 年 6 月 28 日

新功能

Lambda@Edge 現允許您在面對原始伺服器的事件內存取其他標頭 (包括自訂標頭)，以讓您進一步自訂 Amazon S3 儲存貯體所存放內容的交付方式。如需詳細資訊，請參閱下列根據[瀏覽者位置](#)和[瀏覽者裝置類型](#)顯示個人化內容的範例。

2018 年 3 月 20 日

新功能

您現在可以使用 Amazon CloudFront 來交涉採用橢圓曲線數位簽章演算法(ECDSA)的原始伺服器 HTTPS 連線。ECDSA 使用的金鑰更小更快，但與舊版 RSA 演算法一樣安全。如需詳細資訊，請參閱[CloudFront 與原始伺服器通訊支援的 SSL/TLS 協定和密碼](#)以及[關於 RSA 和 ECDSA 加密](#)。

2018 年 3 月 15 日

[新功能](#)

Lambda@Edge 允許您執行 Lambda 函數以回應 Amazon CloudFront 從原始伺服器收到的 HTTP 錯誤，讓您自訂來自原始伺服器的錯誤回應。如需詳細資訊，請參閱下列顯示[重新導向至其他地點](#)以及[用狀態代碼 200 \(OK\) 產生回應](#)的範例。

2017 年 12 月 21 日

[新功能](#)

CloudFront 的新功能，欄位層級資料加密，可協助您進一步提升機密資料的安全性，例如信用卡號或身分證號碼等個人身分資訊 (PII)。如需詳細資訊，請參閱[使用欄位層級加密來協助保護敏感資料](#)。

2017 年 12 月 14 日

[文件歷史記錄已封存](#)

舊文件歷史記錄已存檔。

2017 年 12 月 1 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。