Simulating networks in your laptop

no clouds, just bird flocks

Maria Matejka

CZ DOMAIN REGISTRY

Maria

- working for CZ.NIC
- team leader of BIRD Internet Routing Daemon

Maria

- working for CZ.NIC
- team leader of BIRD Internet Routing Daemon
- IPv6 maximalist
- complaining expert

Maria

- working for CZ.NIC
- team leader of BIRD Internet Routing Daemon
- IPv6 maximalist
- complaining expert
- cook at ProTab

ProTab, summer camp for young programmers

ProTab, summer camp for young programmers

- age limit 19 (pre-university)
- we spend a week (partially) in the nature
- they already know the basics
- half of the schedule is teaching

https://protab.cz/

Teaching routing at ProTab

Hardware simulation is greedy

Containerlab is greedy

Uplink is unstable

On-site cloud is expensive

Simulation primary requirements

- run offline
- run without `sudo`
- acceptable user experience
- least resources consumed

BIRD's Netlab

- written and maintained by Ondřej Zajíček of BIRD Team
- uses Linux kernel network namespaces

BIRD's Netlab

- written and maintained by Ondřej Zajíček of BIRD Team
- uses Linux kernel network namespaces
- minimized startup times
- minimal memory / CPU overhead (just the software running in)
- runnable in your laptop

BIRD Netlab (cont.)

- needs local `root` access
- no other isolation than network namespaces
- CLI-only tool with config files
- expects BIRD to run in every node
- single cluster on one machine
- mixes Python and Bash
- too many Netlabs

BIRD's testing needs

- full automation
- run offline
- multiple clusters on the same machine
- control everything possible
- repeatable tests

Introducing Flock

Introducing Flock

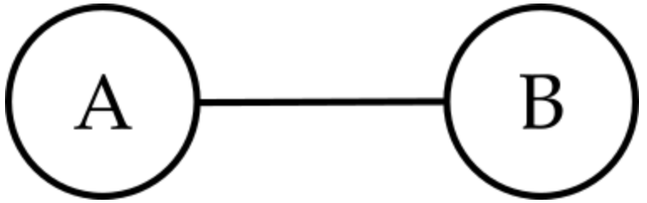
Too many Flocks already, maybe?

Flock

- makes half-containers
- no `root` required, no `subuid` and `subgid` needed
- by default runs nothing (may run BIRD, FRR, whatever)
- better environment isolation

Two machines, one link (the most basic setup)

```
$ flock-sim create bastu
$ flock-sim start bastu A
$ flock-sim start bastu B
$ flock-sim ptp bastu A B B A
```



Login to a machine

```
$ ./flock.py shell bastu A
root@A:/#
```

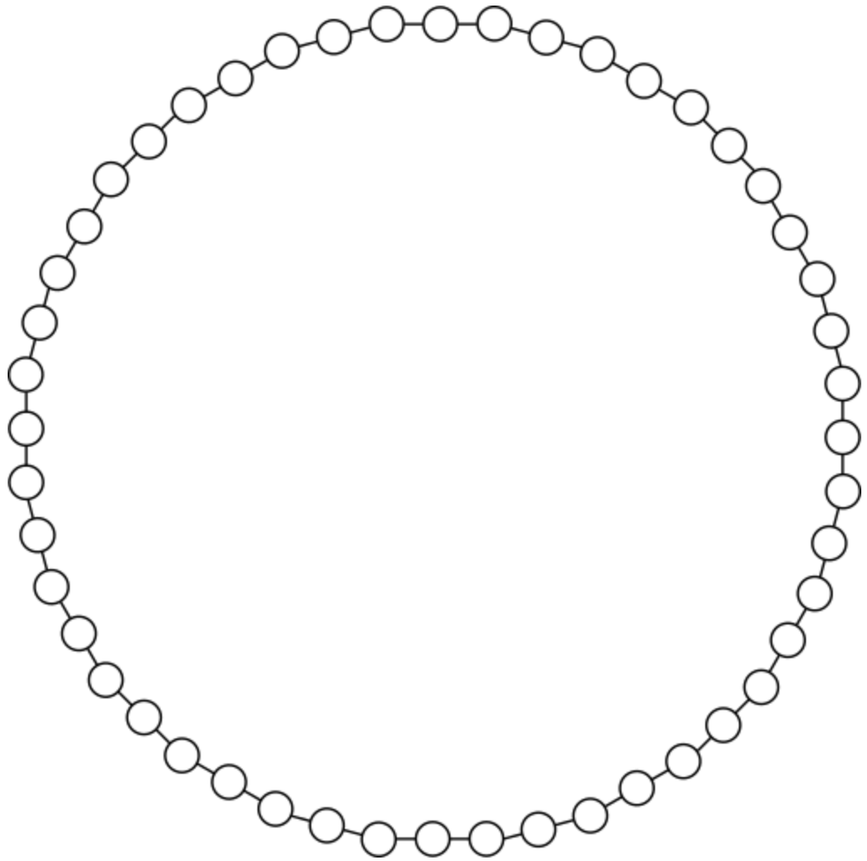
Fixing common problems

- forwarding may be disabled by default

```
root@B:~# sysctl -w net.ipv6.conf.all.forwarding=1
root@B:~# sysctl -w net.ipv4.conf.all.forwarding=1
```

- Debian 12 `tcpdump` build didn’t work
  (upgrade to Trixie fixed this)

Running 200 looped machines with OSPF



```
N=200

# Start all the machines
flock-sim create bada
for m in $(seq $N); do
  flock-sim start bada M$m
done
```

Running 200 looped machines with OSPF

```
# Prepare common configuration
cat <<EOF > bada-common.conf
protocol device {}
protocol kernel { ipv6 { export all; };}
protocol ospf v3 { ipv6 { export all; };
  area 0 { interface "M*" { hello 2; };
        interface "loc" { stub; };};}
EOF
```

Running 200 looped machines with OSPF

```
# Prepare specific configurations
for m in $(seq $N); do
  cp bada-common.conf bada/M$m/overlay/common.conf
  cat <<EOF > bada/M$m/overlay/bird.conf
router id $m;
include "common.conf";
EOF
done
```

Running 200 looped machines with OSPF

```
# Run BIRD on all the machines
for m in $(seq $N); do (
  cat <<EOF
sysctl -w net.ipv6.conf.all.forwarding=1
bird -l
ip link add name loc type dummy
ip link set loc up
ip a add 2001:db8::$(printf '%x' $m)/128 dev loc
EOF
  sleep 0.3) | flock-sim shell bada M$m
done
```

Running 200 looped machines with OSPF

```
# Link all the machines together
flock-sim ptp bada M$N M1 M1 M$N
for m in $(seq $((N-1))); do
  flock-sim ptp bada M$m M$((m+1)) M$((m+1)) M$m
  echo "ip link set M$((m+1)) up" | flock-sim shell bada M$m
  echo "ip link set M$m up" | flock-sim shell bada M$((m+1))
done
```

Running 200 looped machines with OSPF

```
# Show routes
root@M1:~# ip -6 r
2001:db8::62 via fe80::c25:2bff:fe92:58dc dev L1-2 proto bird [...]
2001:db8::63 via fe80::c25:2bff:fe92:58dc dev L1-2 proto bird [...]
2001:db8::64 via fe80::c25:2bff:fe92:58dc dev L1-2 proto bird [...]
2001:db8::65 proto bird [...]
    nexthop via fe80::c25:2bff:fe92:58dc dev L1-2 weight 1
    nexthop via fe80::9cc2:4ff:feb9:2c44 dev L200-1 weight 1
2001:db8::66 via fe80::9cc2:4ff:feb9:2c44 dev L200-1 proto bird [...]
2001:db8::67 via fe80::9cc2:4ff:feb9:2c44 dev L200-1 proto bird [...]
2001:db8::68 via fe80::9cc2:4ff:feb9:2c44 dev L200-1 proto bird [...]
```

Running 200 looped machines with OSPF

```
# Check reachability
root@M1:~# ping -c1 2001:db8::20
[...]
64 bytes from 2001:db8::20: icmp_seq=1 ttl=34 time=2.04 ms

root@M1:~# ping -c1 2001:db8::40
[...]
64 bytes from 2001:db8::40: icmp_seq=1 ttl=2 time=2.11 ms

root@M1:~# ping -c1 2001:db8::64
[...]
From 2001:db8::41 icmp_seq=1 Time exceeded: Hop limit
```

Flock today

- runs BIRD, FRR, OpenBGPd (and possibly others)
- written in C and Python
- CLI interface
- rudimentary JSON interface
- dynamic topology updates

To be fixed

- Weird problems with overlayfs
- Shell broker terminal setting
- Documentation
- Installation

In development

- automation as BIRD testing tool

BIRD 3 has been released

Now flying in multiple threads

In development

- automation as BIRD testing tool
- TAP / VXLAN link
- Docker image support
- using actual physical interfaces and machines
- GUI to show status and define networks by mouse clicking

Flock tryout

- available at https://gitlab.nic.cz/labs/flock
- still under development
- contributions welcome

Contacts

Maria Matejka
BIRD Team Leader at CZ.NIC
maria.matejka@nic.cz
Mastodon: @marenamat@slon.jmq.cz

BIRD Users Mailing list: bird-users@network.cz
BIRD Support: https://bird.nic.cz/

Questions, Explanations, Discussion

Maria Matejka

CZ DOMAIN REGISTRY