BIRD Internet Routing Daemon

CZ.NIC
Ondrej Filip / ondrej.filip@nic.cz
Mar 11 2011, Stockholm, Netnod meeting

Project history

- Project started in 1999
- Seminar project – Charles University
  - Prague
- Project slept for a while
- Small reincarnation in 2003 and 2006
- Project fully renewed since Q4 2008 – part of CZ.NIC Labs – http://labs.nic.cz

Project goals

- Opensource routing daemon – alternative to Quagga/Zebra (and GateD that time)
- Fast and efficient
- Portable, modular
- Support current routing protocols
- IPv6 and IPv4 in one source code – dual compilation
- Easy reconfiguration and powerful filtering

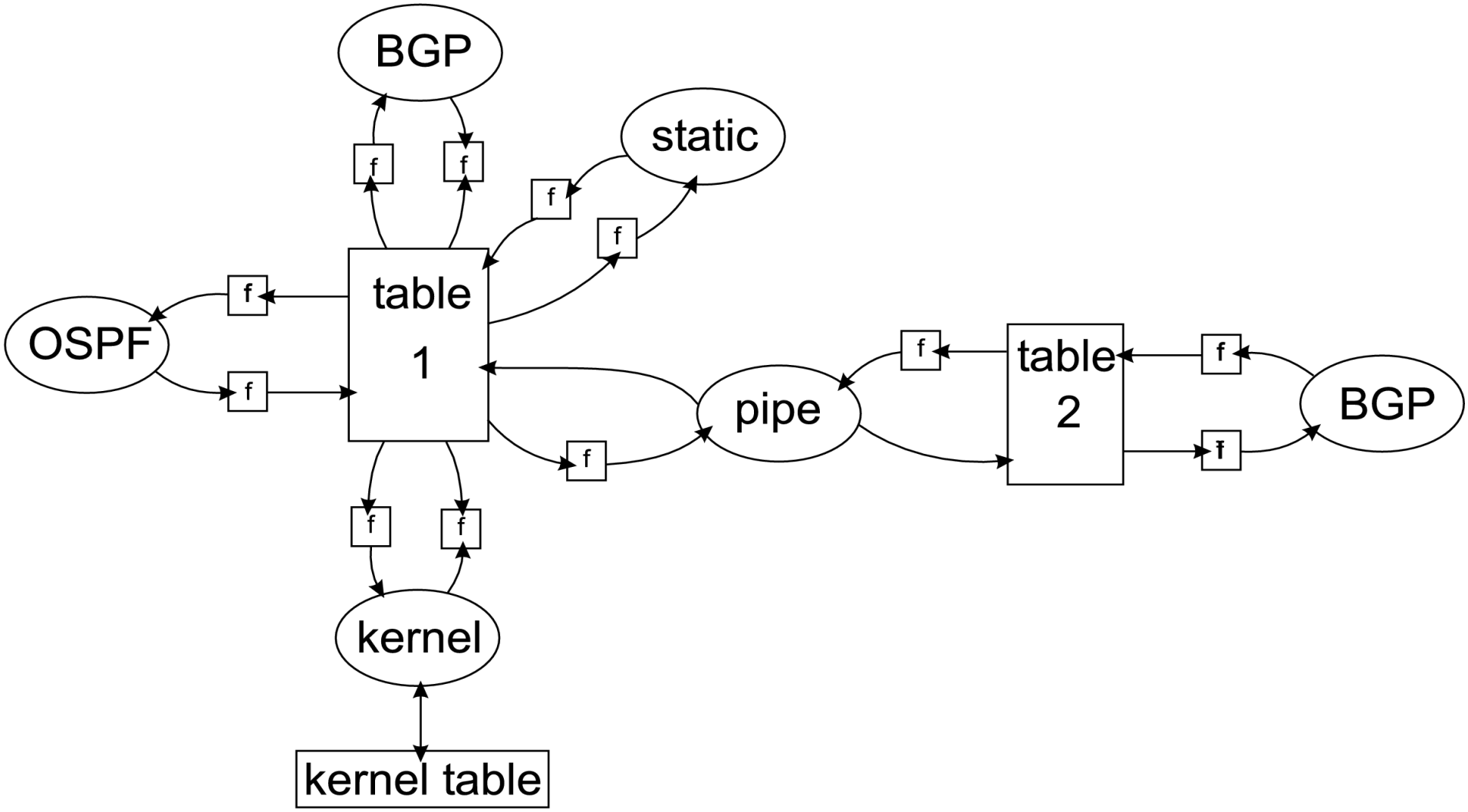
Features

- Portable – Linux, FreeBSD, NetBSD, OpenBSD
- IPv4/IPv6 support
- Static routing
- RIP, RIPv2, RIPng
- OSPFv2, OSPFv3
- BGP (v4 and v6), route reflector
- Route server
- ASN32 (ASPLAIN), MD5
- MRTdump logging

Features

- Multiple routing table - RIBs (internal and also synchronization with OS)
- Protocol PIPE
- Multiple routers, route reflectors on a single system
- Powerful configuration
- Very powerful filtering language
- Command line interface (show, restart, ...)
- Automatic reconfiguration

Design


BGP
static
f
table
1
OSPF
pipe
table
2
BGP
kernel
kernel table

Configuration sample

```
log "/var/log/bird.log" all;

router id 193.51.100.238;

protocol static {
        route 10.0.0.0/8 drop;
        route 172.16.0.0/12 drop;
        route 192.168.0.0/16 drop;
}
filter bgp_out {
        if (net = 192.175.48.0/24 ) &&
                  (source = RTS_DEVICE) then accept;
        else reject;
}
protocol bgp NIX_1 {
        local as 112;
        neighbor 193.51.100.235 as 6981;
        import all;
        export filter bgp_out;
}
```

CLI example

```
bird> show protocols
name     proto    table    state since  info
direct1  Direct   master   up    Apr11
kernel1  Kernel   master   up    Apr11
device1  Device   master   up    Apr11
static1  Static   master   up    Apr11
NIX_2    BGP      master   up    Apr11  Established
NIX_1    BGP      master   up    Apr25  Established
ospf1    OSPF     master   up    Apr11  Running
bird>
bird> show status
BIRD 1.1.3
Current server time is 06-08-2009 22:01:06
Last reboot on 11-07-2009 22:54:12
Last reconfiguration on 30-07-2009 06:25:25
Daemon is up and running
bird>
```

CLI example (cont)

```
bird> show route
10.0.0.0/8      via 200.30.10.3 on eth2 [ospf1 13:10] E2 (150/5/1000)
127.0.0.0/8     dev lo [direct1 13:09] (240)
200.30.20.0/24  via 200.30.10.3 on eth2 [ospf1 13:10] I (150/10)
200.30.10.0/24  dev eth2 [direct1 13:09] (240)
                dev eth2 [ospf1 13:10] I (150/5)
200.0.10.0/24   dev eth0 [direct1 13:09] (240)
                dev eth0 [ospf1 13:09] I (150/5)
172.16.0.0/16   via 200.30.10.3 on eth2 [ospf1 13:10] E2 (150/5/1000)
195.47.235.0/24 via 194.50.100.246 on eth1 [NIX2 Apr11] (100)[AS688i]
                via 194.50.100.245 on eth1 [NIX1 Apr25] (100)[AS688i]
bird>
bird> show route protocol ospf1
10.0.0.0/8      via 200.30.10.3 on eth2 [ospf1 13:10] E2 (150/5/1000)
200.30.20.0/24  via 200.30.10.3 on eth2 [ospf1 13:10] I (150/10)
200.30.10.0/24  dev eth2 [ospf1 13:10] I (150/5)
200.0.10.0/24   dev eth0 [ospf1 13:09] I (150/5)
172.16.0.0/16   via 200.30.10.3 on eth2 [ospf1 13:10] E2 (150/5/1000)
```

CLI example (cont)

```
bird> show route for 127.0.0.1
127.0.0.0/8         dev lo [direct1 13:09] (240)

bird> show route filter bgp_out
192.175.48.0/24     dev dummy0 [direct1 Apr1] (240)

bird> show route count
1469 of 1469 routes for 849 networks

bird> show route export NIX_1
192.175.48.0/24     dev dummy0 [direct1 Apr1] (240)

bird> show route where 127.0.0.5 ~ net
0.0.0.0/0      via 195.47.235.1 on eth0 [static1 Apr1](200)
127.0.0.0/8    dev lo [direct1 Apr1] (240)

bird> show route filter {if 127.0.0.5 ~ net then accept;}
0.0.0.0/0      via 195.47.235.1 on eth0 [static1 Apr1](200)
127.0.0.0/8    dev lo [direct1 Apr1] (240)
```

Filters example

```
function avoid_martians()
prefix set martians;
{
  martians = [ 169.254.0.0/16+, 172.16.0.0/12+,
    192.168.0.0/16+, 10.0.0.0/8+, 224.0.0.0/4+,
   240.0.0.0/4+, 0.0.0.0/32-, 0.0.0.0/0{25,32},
   0.0.0.0/0{0,7} ];

  # Avoid RFC1918 networks
  if net ~ martians then return false;

  return true;
}
```

The same filter on Quagga

```
ip prefix-list avmart seq 10 deny 169.254.0.0/16 le 32
ip prefix-list avmart seq 20 deny 172.16.0.0/12 le 32
ip prefix-list avmart seq 30 deny 192.168.0.0/16 le 32
ip prefix-list avmart seq 40 deny 10.0.0.0/8 le 32
ip prefix-list avmart seq 50 deny 224.0.0.0/4 le 32
ip prefix-list avmart seq 60 deny 240.0.0.0/4 le 32
ip prefix-list avmart seq 70 deny 0.0.0.0/0 le 7
ip prefix-list avmart seq 80 deny 0.0.0.0/0 ge 25
ip prefix-list avmart seq 90 permit any
```

And also OpenBGPd

```
# filter bogus networks
allow from any prefixlen 8 – 24
deny from any prefix 0.0.0.0/0
deny from any prefix 10.0.0.0/8 prefixlen >= 8
deny from any prefix 172.16.0.0/12 prefixlen >= 12
deny from any prefix 192.168.0.0/16 prefixlen >= 16
deny from any prefix 169.254.0.0/16 prefixlen >= 16
deny from any prefix 192.0.2.0/24 prefixlen >= 24
deny from any prefix 224.0.0.0/4 prefixlen >= 4
deny from any prefix 240.0.0.0/4 prefixlen >= 4
```

Filters example - AS#

```
function asmatch()
int set asnums;
{
  asnums = [ 11111, 22222, 33333, 44444,  55555,
             66666, 77777, 88888, 99999, 100..200 ];

  # Check originating AS number
  if bgp_path.last ~ asnums then return true;

  return false;
}
```

Filters example - case

```
case bgp_path.last {
        11111: if(prefAS11111()) then accept;
        22222: if(prefAS22222()) then accept;
        33333: if(prefAS33333()) then accept;
        44444: if(prefAS44444()) then accept;
        else:  reject;
};
```

Filter example – route server

- Route server policy - NIX.CZ

Evaluation order	Community	Action
1	0:<peer-as>	Do not advertise to <peer-as>
2	47200:<peer-as>	Advertise to <peer-as>
3	0:47200	Do not advertise to any peer
4	47200:47200	Advertise to all peers

RS policy

- ISP connected to IXP RS does not loose it freedom of defining its own routing policy
- Example 1 – I want to propagate my routes to CZ.NIC only
  - 10.0.0.0/8 community 47200:25192, 0:47200
- Example 2 – I want to propagate my routes to everybody except CZ.NIC
  - 10.0.0.0/8 community 0:25192, 47200:47200

RS policy – Quagga (per ISP!)

```
ip community-list standard C-0-10001 permit 0:10001
ip community-list standard C-0-47200 permit 0:47200
ip community-list standard C-47200-10001 permit 47200:10001

route-map Policy10001 deny 10
 match community C-0-10001
!
route-map Policy10001 permit 20
 match community C-47200-10001
!
route-map Policy10001 deny 30
 match community C-0-47200
!
route-map Policy10001 permit 40
!
```

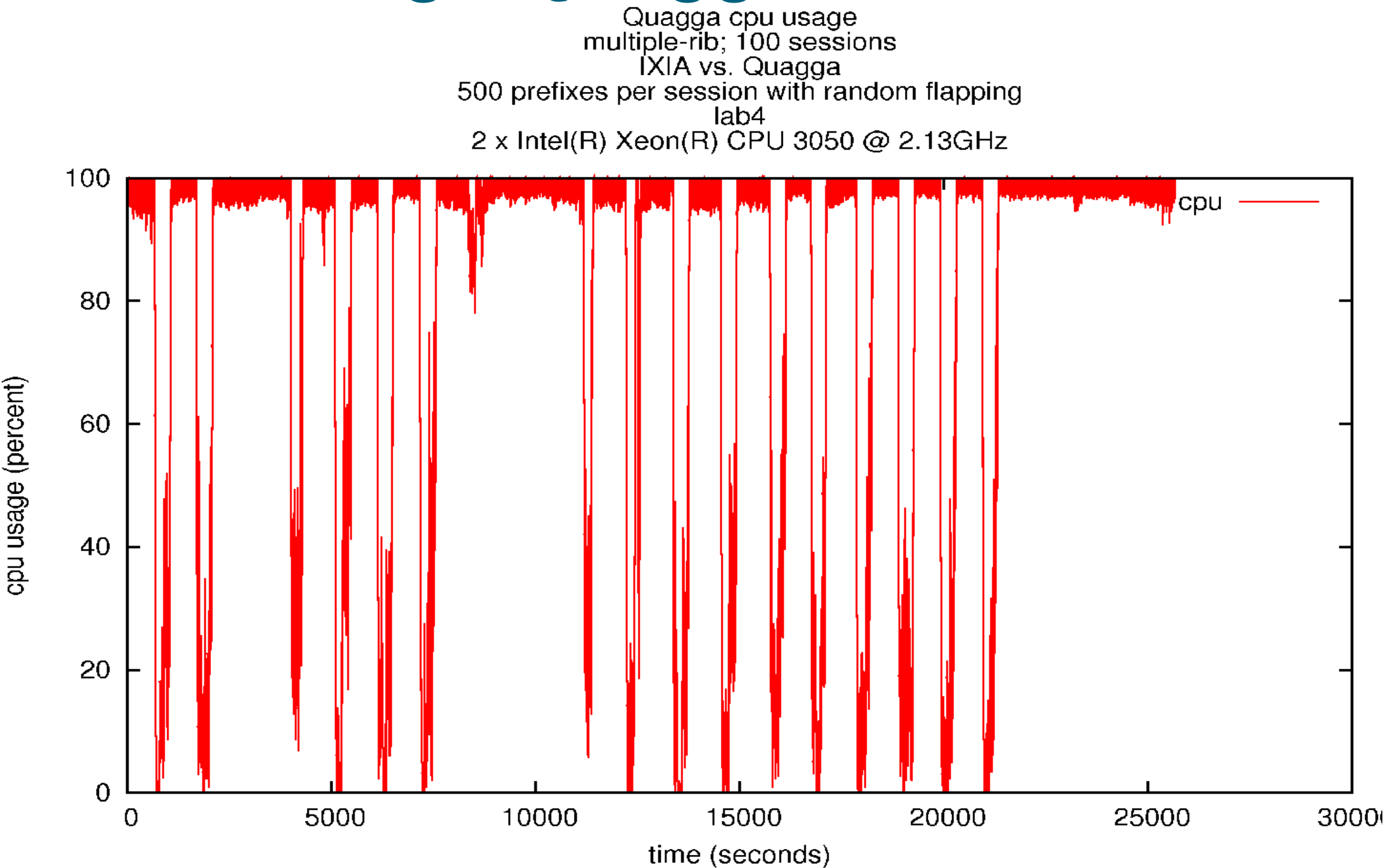
RS policy - BIRD

```
define myas = 47200;

function bgp_out(int peeras)
{
  if ! (source = RTS_BGP ) then return false;
  if (0,peeras) ~ bgp_community then return false;
  if (myas,peeras) ~ bgp_community then return true;
  if (0, myas) ~ bgp_community then return false;
  return true;
}

protocol bgp R25192x1 {
  local as myas;
  neighbor 194.50.100.13 as 25192;
  import where bgp_in(25192);
  export where bgp_out(25192);
  rs client;
}
```

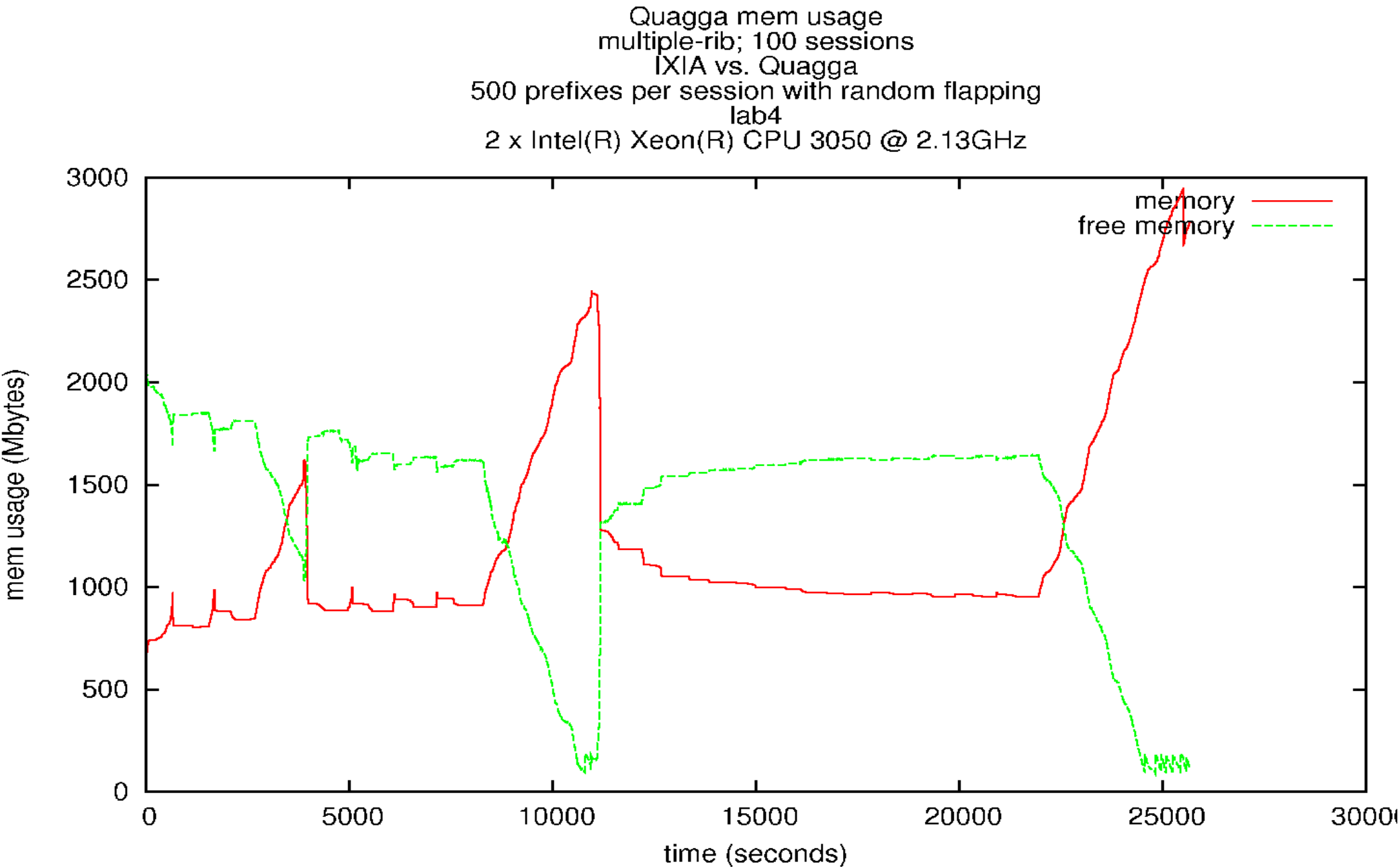
Route Server Testing

- Euro-IX WG - Andy Davidson – LONAP, Chris Malayter – Switch&Data, Elisa Jasinska – AMS-IX, Mo Shivji – LINX, Robert Wozny – PL-IX, Sebastian Spies – DE-CIX, Wolfgang Hennerbichler – VIX
- 100 session – IXIA testing device
- 500 or 1000 prefixes per session
- Random flapping
- Memory and CPU

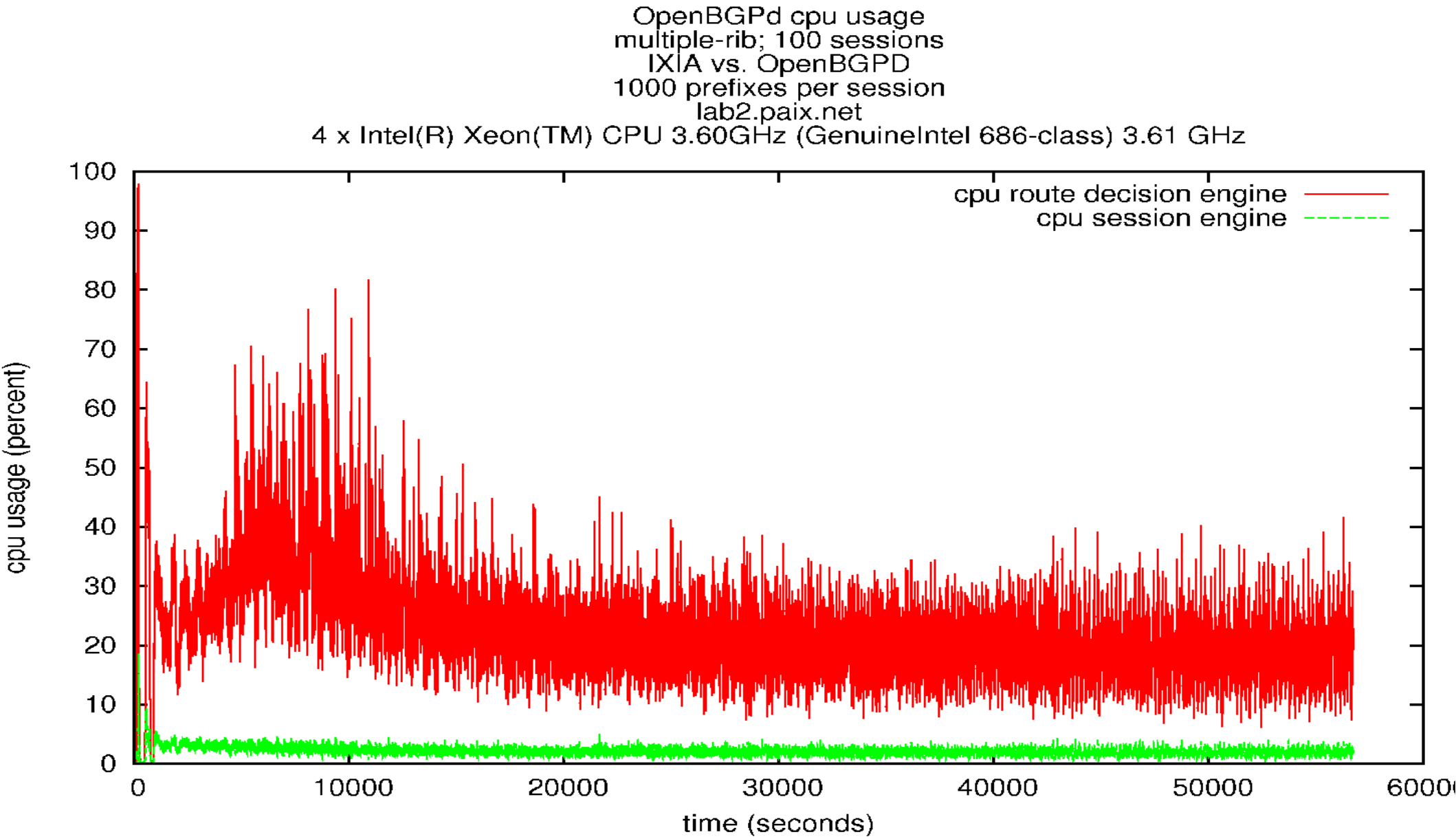
RS testing - Quagga



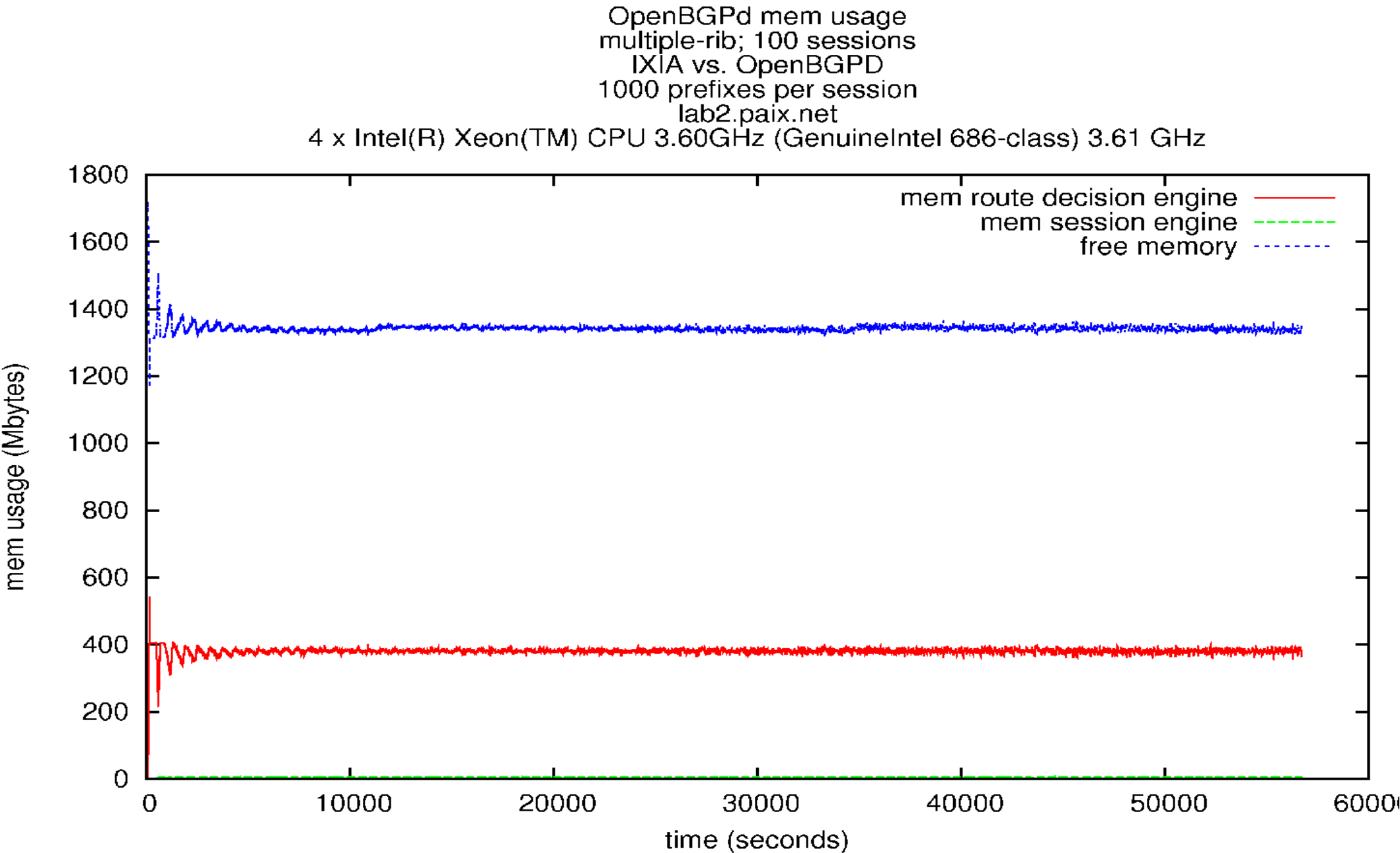
RS testing - Quagga



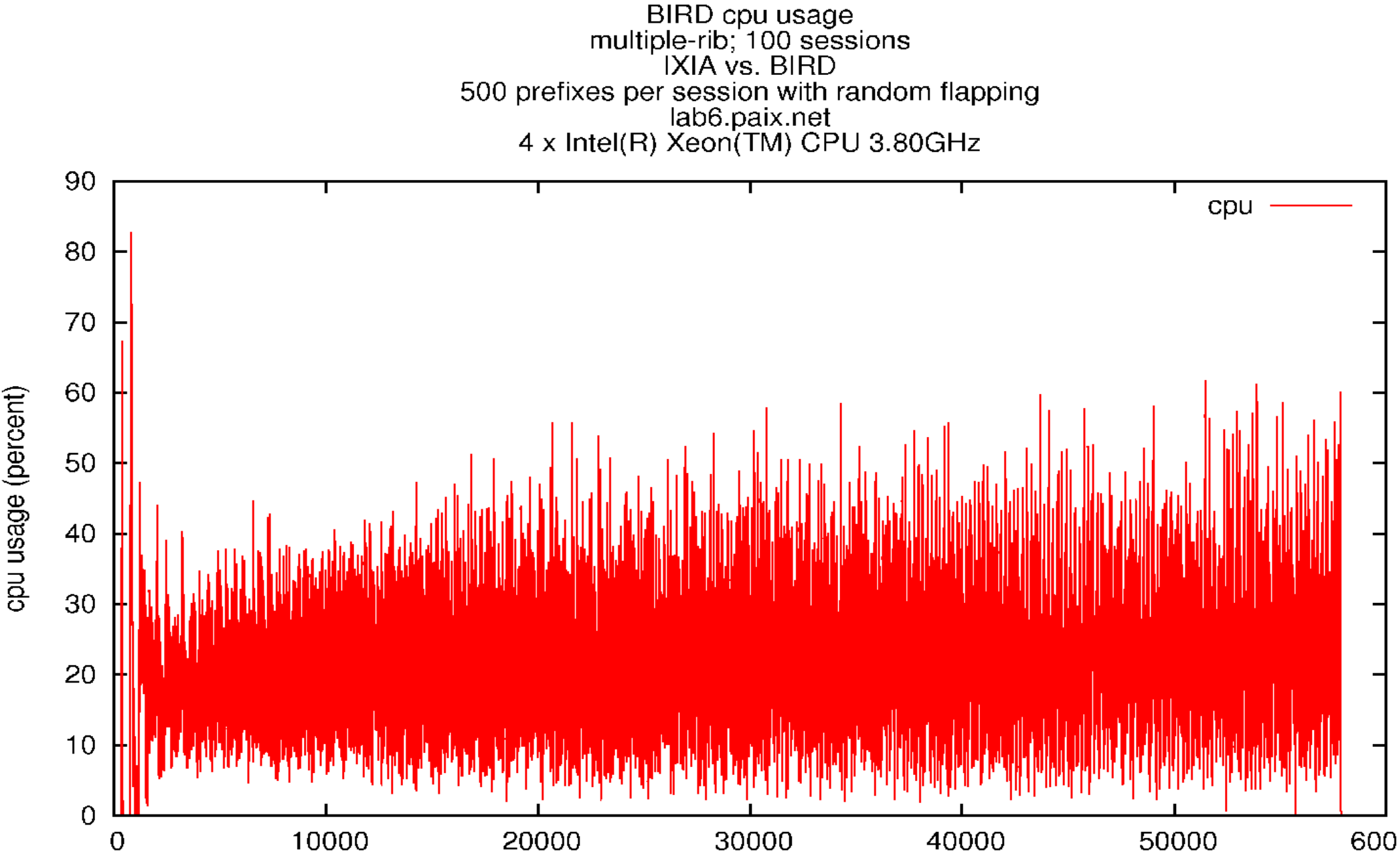
RS testing - OpenBGPd



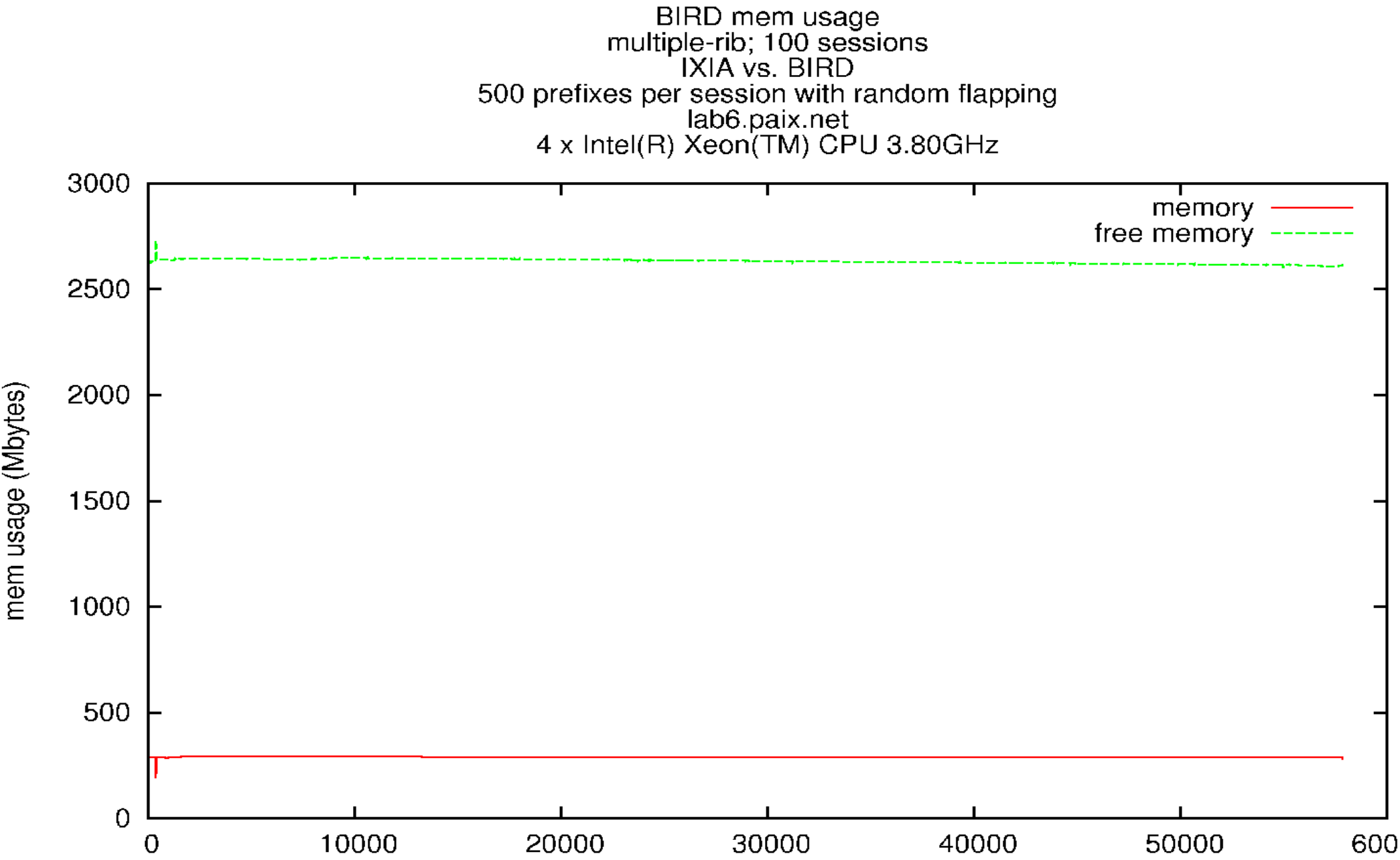
RS testing - OpenBGPd



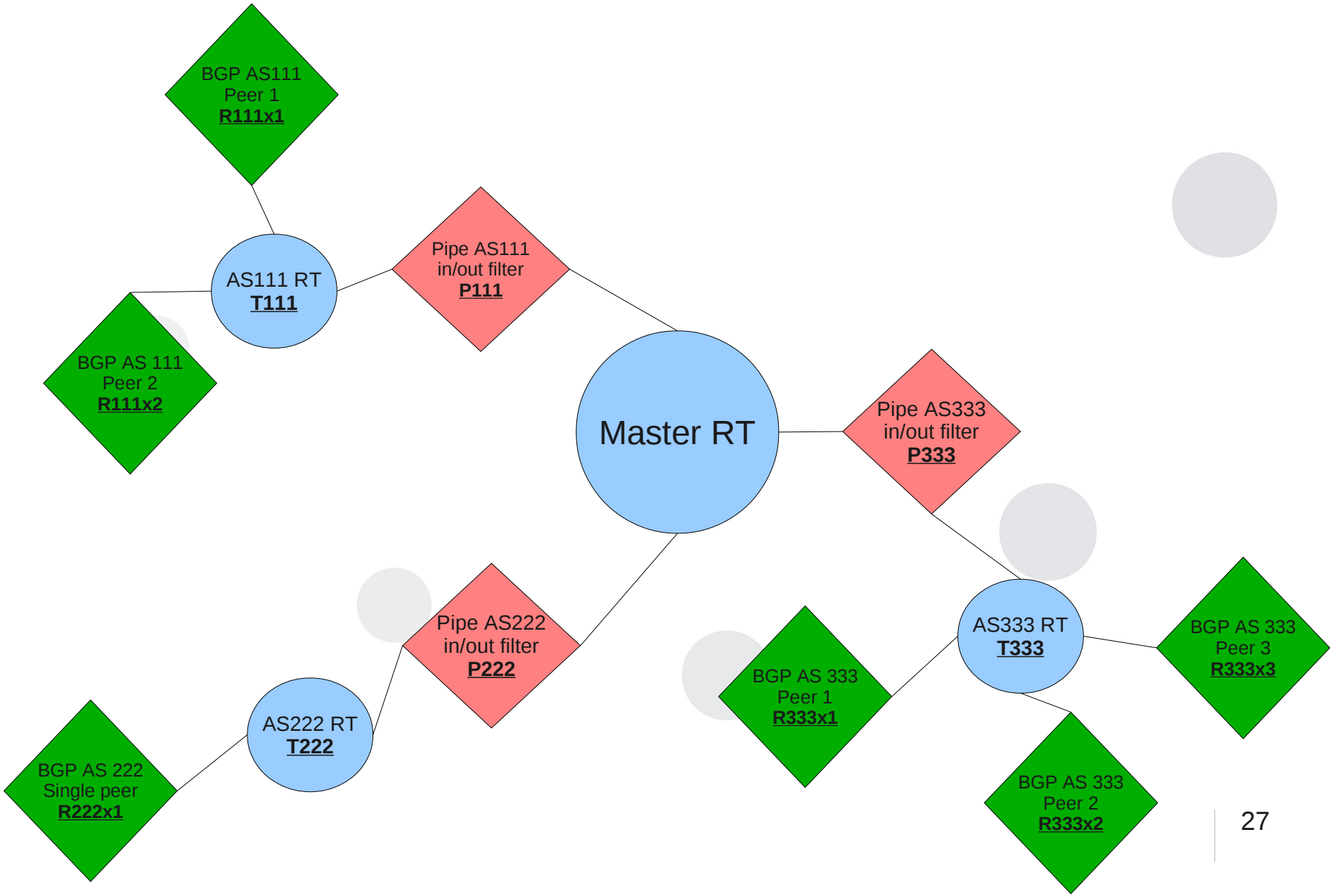
RS testing - BIRD



RS testing - BIRD



BIRD as a route server



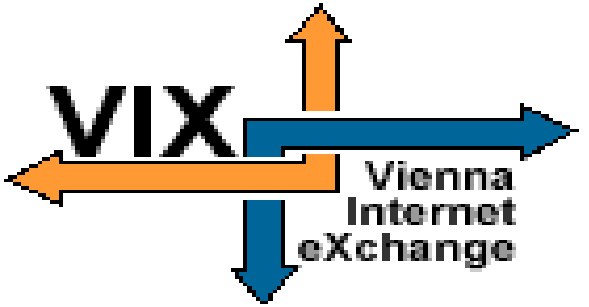
BIRD at LoNAP

- First BIRD RS implementation
- Two route-servers
- BIRD and OpenBGPd
- BIRD in multiple RIBs setup
- Thanks for help with debugging!
- ~100 IPv4 sessions, ~3000 prefixes
- ~50 IPv4 sessions, ~100 prefixes

BIRD at NIX.CZ

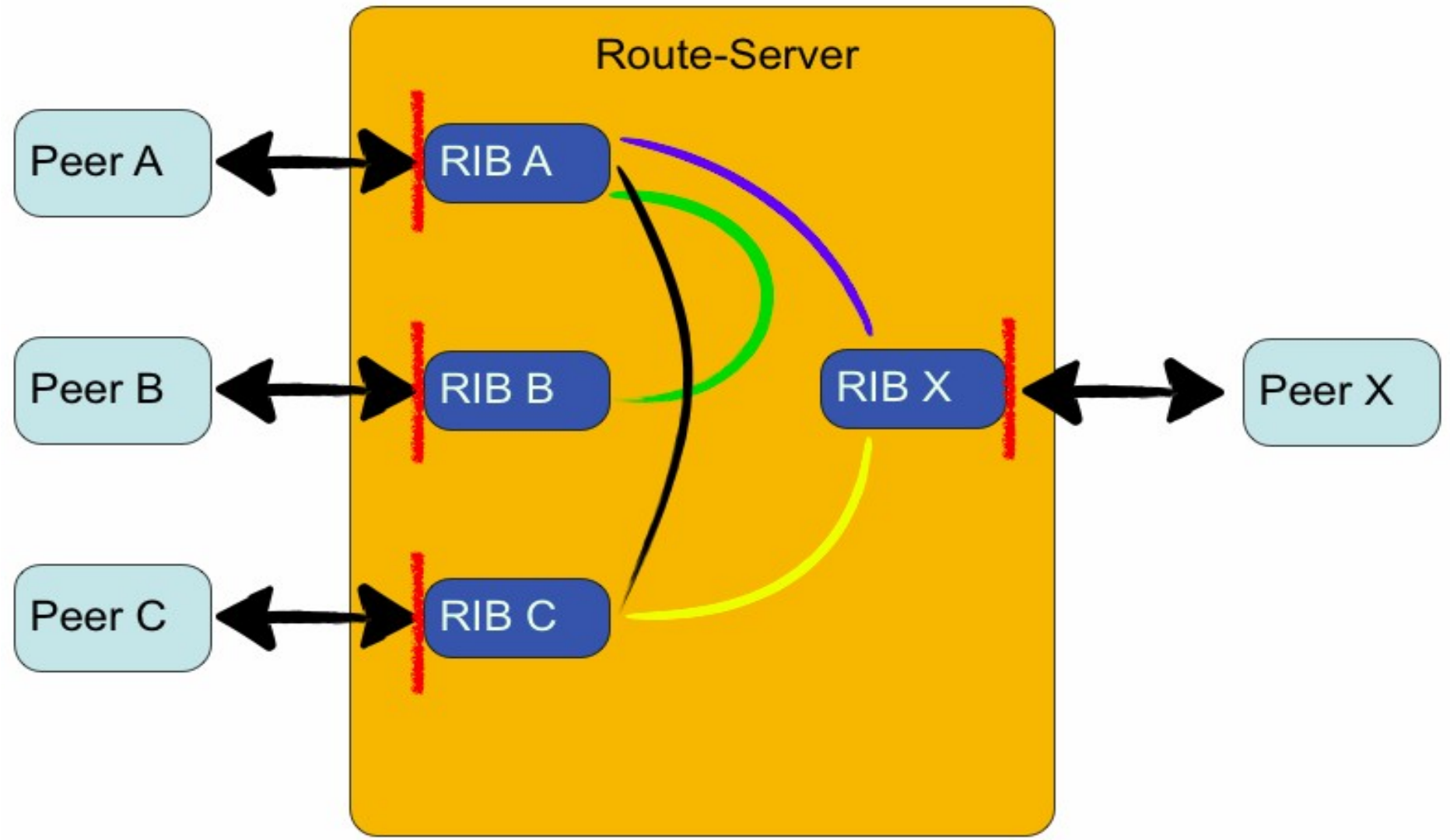
- Classical RS configuraton
- Dual route servers
- BIRD on Linux and Quagga (still) on FreeBSD
- Multiple RIBs setup (BIRD only)
- No BIRD's crash since implementation (1.1.3)
- Approx 120 IPv4 sessions (70 IPv6)
- Approx 11000 IPv4 prefixes (250 Ipv6)
- Filtering based on routing db (RIPEdb mirror)
- Reconfiguration every hour

BIRD at VIX

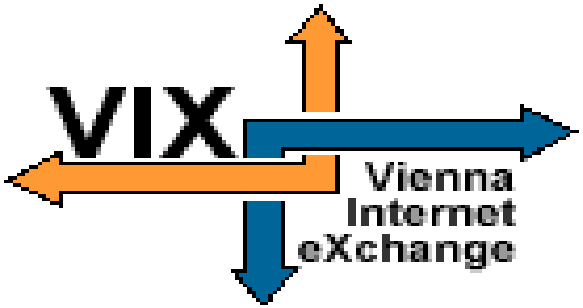


- Based only on BIRD
- Different approach – P2P PIPE protocols
- Unusual setup
- Configuration taken from intranet

BIRD at VIX



BIRD implementations



PAIX – multiple locations

-PAO, SEA, CHI, DAL, NYC, ATL, TOR, TYS

Some smaller – MINAP, ECIP, NapAfrica

LINX Award 2010

Other applications

- AS112 server at NIX.CZ
- BGP/OSPF router for smaller ISPs
- Router for some CZ.NIC's anycast nodes
- Used is small embedded system – part of firmware of some WiFI Aps (OpenWRT)
- BGP table analyser
- Implementations at some other IXPs in progress

Future development

- Last version - 1.2.5
- Monthly releases (except now – working on 1.3.0)
- New web pages!!!
- IPv6 route advertisements
- Lightweight CLI (for small systems without readline lib)
- Route flap dampening
- ...
- Depends on user demand

Summary

PROS

- Powerful configuration and filter language
- Lightweight, efficient
- Used in heaviest environment (largest IXPs)
- Active development

CONS

- Unusual configuration language
- IPv4/IPv6 separation

¿Questions?

http://bird.network.cz
<ondrej.filip@nic.cz>