OpenBGPD

Claudio Jeker

Past

2003

OpenBGPD started 15 years ago

- First released with OpenBSD 3.5
- Out of frustration with zebra

The Internet in 2003

- Around 125’000 prefixes in the DFZ
- IXP picking up, mostly local and trust based

OpenBGPD and IXP

OpenBGPD ideal as route-server

Special route-server features, e.g.

- Neighbor templates
- Expansion of neighbor-as in e.g. communities
- Later on multiple RIB support

Possible to build route-servers with little to no maintenance tasks

Recent past

2018

OpenBGPD: solid base, all important protocol features present

- Basic design did not change, especially the RIB and filters have not changed

The Internet in 2018

- Over 700’000 IPv4 and 50’000 IPv6 routes in the DFZ
- Trust model no longer works, filtering required especially at IXP

OpenBGPD and IXP

Running a route-server with full filters is too slow

Old RIB design shows troubles with scaling

- Config reload block updates and withdraw until finished

At a small IXP like YYCIX config requires 370’000 filter rules and

- reloading the config takes 1h to finish

As a result most IXP route-server run a bird monoculture

- It is important to make OpenBGPD a viable alternative again

A major effort is required

Solving the issue

Have a full time developer working on OpenBGPD funded by

RIPE NCC
RIPE NETWORK COORDINATION CENTRE

Community Projects Fund

and

DE CIX

NET NOD

amsix

BCIX
Berlin Commercial Internet Exchange

UNIVERSITAS OSLOENSIS
MDCCCXI

LONAP™

Asteroid

NAMEX
ROMA Internet eXchange Point

Present

OpenBGPD in 6.4

- RFC8212 compliance (default deny policy)
  Be careful when updating from 6.3 to 6.4.
- RPKI ROA support (static table, no RTR support)
- Sets for prefixes, ASnum and origins (prefix + source-as)
  Replace large amount of filters with a single fast lookup
- Background soft-reconfiguration on config reload
  On reload new withdraws and updates are now processed
- 154 commits since 6.3 (close to 8% of all commits)

RPKI ROA validation

RFC 6811 BGP Origin Validation through the `roa-set` directive and `ovs` filter.

```
roa-set {
        203.119.88.0/23 maxlen 24 source-as 187
        2001:dd8:7::/48 maxlen 48 source-as 187
        2401:680::/32 maxlen 32 source-as 715
}
deny quick from ebgp ovs invalid
```

`ovs` can be `valid`, `invalid` or `not-found`

Use `bgpctl show rib ovs valid` to show only valid routes.

On my system reloading the roa-set takes a couple of seconds.

Prefix-sets

prefix-set got a lot better, can be used also with network statements

```
prefix-set p4_AS6939 {
        1.0.0.0/24 prefixlen 24 - 32
        1.0.4.0/22 prefixlen 22 - 32
        1.0.16.0/22 prefixlen 22 - 32
}

match from AS 6939 prefix-set p4_AS6939 set { localpref +25 community local-as:102 }
```

Other prefixlen options are or-longer, maxlen X, or nothing.

Prefix-sets, with only prefixes, can use or-longer in the filter

```
deny quick from ebgp prefix-set mynetworks or-longer
```

As-sets

Quick lookup table for large amount of AS numbers

```
as-set asns_AS6939 {
        2 3 4 5 6 8 10 13
        16 17 22 24 25 26 27 31
        32 37 38 42 43 44 45 47
}

match from AS 6939 source-as as-set asns_AS6939 set {
     localpref +50 community local-as:101 }
```

Works for `AS`, `transit-as` and `source-as`

Origin-sets

Similar to `roa-set` can be used when only prefixes matter which are `valid`

```
origin-set "ARINDB" {
        108.160.208.0/20 prefixlen 20 - 32 source-as 10242
        162.219.228.0/22 prefixlen 22 - 32 source-as 10242
        162.222.52.0/22 prefixlen 22 - 32 source-as 10242
}

# community local-as:101 is set when source-as is in IRR as-set
match from group clients community local-as:101 origin-set ARINDB set {
     localpref +25 community local-as:102 }
```

Allows to use ARIN Whois or RPKI ROA as an alternative to IRR DB to validate prefixes based on origin-as.

Results

YYCIX (Calgary Internet Exchange) using arouteserver to generate config

6.3: generated config consists of 370’000 filter rules

6.4: with as-set, prefix-set and origin-set ruleset is now below 6000 rules

Switching to as-set & prefix-set for IRR based filtering reduced initial config to 95’000 rules. Using origin-set for the RPKI and ARIN DB filtering did the rest.

Background soft-reconfiguration no longer blocks update processing during a config reload.

Future

OpenBSD in 6.5 (May 2019)

Better community filtering

- Matching more than one community per filter rule
- Faster set and delete of communities

More filter refinements and tuning

Further RIB refactoring, make multiple RIB support faster

Resurrect portable version and keep it alive (like OpenSSH and LibreSSL)

- No FIB support for now

Future Projects

Multi-thread support in the RDE for filtering and maybe some other tasks

In the portable version add FIB support for other BSDs and Linux

New features (ADD_PATH, BGP multipath, BMP, RTR)

Thanks

RIPE NCC
RIPE NETWORK COORDINATION CENTRE

Community Projects Fund

and

DE CIX

NETNOD

amsix

BCIX
Berlin Commercial Internet Exchange

UNIVERSITAS OSLOENSIS
MDCCCXI

LONAP™

Asteroid

NAMEX
ROMA Internet eXchange Point

also a big thank you goes to Job Snijders for making it possible

Questions?