The work behind an Open Source Routing stack

SWINOG - Oct 30, 2018

Martin Winter
NetDEF / OpenSourceRouting
About me...

- **FRRouting Maintainer**
- **FRRouting TSC Member**
- **Runs CI System for community**
- **Co-Founder NetDEF (a US non-profit)**
  - (OpenSourceRouting is a project by NetDEF)
- **Past:**
  - Quagga
  - Cisco (Engineering)
  - Large ISP (Network Architecture)
What is FRRouting
Quick Introduction
What is FRR?
(for the not so technical People)

- Open Source (GPLv2+) **Routing Stack**
- Implements RIP, RIPng, OSPF (v2&v3), ISIS, BGP, PIM, LDP, NHRP, Babel, EIGRP, BFD, ...
- Fork of Quagga
- Works on Linux and most BSD based systems
- For use in many Clouds as virtual routers, white box vendors and network providers (full routing stack)
FRR - Why a fork?

Community Led and Driven

Fast & Open Development

Open Community Model
FRR - Who is behind the Fork?
FRR Major features added since Fork

- **2.0 (April 2017)**
  - RFC 5549 Unnumbered Support (most protocols)
  - LDP IPv4/IPv6
  - VRF lite
  - JSON output support

- **3.0 (Oct 2017)**
  - BGP EVPN base
  - PIM Sparse Mode
  - NHRP

- **4.0 (March 2018)**
  - BGP RPKI
  - BGP EVPN Type 3 & 4
  - BGP MPLS Ethernet VPN & Multicast
  - BABEL
  - EIGRP
FRR Major features added since Fork

- [continue] 4.0 (March 2018)
  - Static VRF route leaking
  - OSPFv2 Segment Routing

- 5.0 (July 2018)
  - PIM Multicast Trace [draft-ietf-idmr-traceroute-ipm]
  - IS-IS 3-way Handshake [RFC5303]
  - BGP VPN-VRF route leaking per [RFC4364]
  - BGP VRF with NETNS backend
  - New Policy Based Routing Daemon

- 6.0 (Oct 2018)
  - BFD daemon
  - Static Routes moved from zebra to it’s own daemon
  - IS-IS Src-Dest Routing [draft-ietf-isis-ipv6-dst-src-routing]
FRR Major features added since Fork

- **Current development branch**
  - OpenFabric Draft [draft-white-openfabric]
  - YANG Northbound (= Configuration) API
  - Work on Southbound (= Kernel) API/Interface

JOIN US on future development

https://github.com/frrouting
The life of a PR (Git pull request)

What happens when someone pushes a pull request
Key Points on Testing:

• We can’t afford time for manual testing. Automate. Everything.

• Automated responses should be clear to first time users. We can’t afford time to explain test failures.
Step 1

Github Pull Request is pushed
Step 2

CI System is triggered by Github
Step 3 – First CI Phase
CI System Builds on all supported Distros
**Compiling... (on VMs)**

All build on x86 (64bit) architecture except where noted

<table>
<thead>
<tr>
<th>Ubuntu</th>
<th>CentOS</th>
<th>Debian</th>
<th>FreeBSD</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Ubuntu Logo" /></td>
<td><img src="image" alt="CentOS Logo" /></td>
<td><img src="image" alt="Debian Logo" /></td>
<td><img src="image" alt="FreeBSD Logo" /></td>
</tr>
<tr>
<td>Ubuntu 14.04</td>
<td>CentOS 6</td>
<td>Debian 8</td>
<td>FreeBSD 10</td>
</tr>
<tr>
<td>Ubuntu 16.04 (x86_64 + i386 + ARMv7 + ARMv8)</td>
<td>CentOS 7</td>
<td>Debian 9 (x86_64 + ARMv7 + ARMv8)</td>
<td>FreeBSD 11</td>
</tr>
<tr>
<td>Ubuntu 18.04</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NetBSD</th>
<th>OpenBSD</th>
<th>OmniOS</th>
<th>Alpine (in testing)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="NetBSD Logo" /></td>
<td><img src="image" alt="OpenBSD Logo" /></td>
<td><img src="image" alt="OmniOS Logo" /></td>
<td><img src="image" alt="Alpine Logo" /></td>
</tr>
<tr>
<td>NetBSD 6</td>
<td>OpenBSD 6</td>
<td>OmniOS</td>
<td>Alpine 3.8</td>
</tr>
<tr>
<td>NetBSD 7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


And building again... Packages

- Ubuntu 14.04 for 64bit Intel
- Ubuntu 16.04 for 32/64bit Intel, ARM7, ARM8
- Debian 8 for 64bit Intel
- Debian 9 for 64bit Intel, ARM7, ARM8
- CentOS 6 / 7 for 64bit Intel
- Snap Package for 32/64bit Intel, ARM7, ARM8

Linux only (to save time)

Testing uses packages - whenever possible
Step 4

**Testing Phase**

**Code commits**

<table>
<thead>
<tr>
<th>Author</th>
<th>Commit</th>
<th>Message</th>
<th>Commit date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quentin Young</td>
<td>dd5e280...</td>
<td>Merge pull request #3247 from opensourcerouting/spalchk *: spalchk</td>
<td>1 hour ago</td>
</tr>
</tbody>
</table>

**Shared artifacts**

<table>
<thead>
<tr>
<th>Artifact</th>
<th>File size</th>
</tr>
</thead>
<tbody>
<tr>
<td>CentOS 6 x86_64 RPMs</td>
<td>12 MB</td>
</tr>
<tr>
<td>CentOS 7 x86_64 RPMs</td>
<td>13 MB</td>
</tr>
<tr>
<td>Debian 8 (Jessie) x86_64 Packages</td>
<td>0 bytes</td>
</tr>
<tr>
<td>Debian 9 (Stretch) x86_64 Packages</td>
<td>0 bytes</td>
</tr>
<tr>
<td>Fedora 24 x86_64 RPMs</td>
<td>13 MB</td>
</tr>
<tr>
<td>Ubuntu 12.04 x86_64 Packages</td>
<td>0 bytes</td>
</tr>
<tr>
<td>Ubuntu 14.04 x86_64 Packages</td>
<td>0 bytes</td>
</tr>
<tr>
<td>Ubuntu 16.04 x86_64 Snap Package</td>
<td>11 MB</td>
</tr>
</tbody>
</table>

4 more shared artifacts...

**Live activity log for AddressSanitizer TopoTest**

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
</table>

Level up your DevOps kung fu with Bamboo, the Continuous Delivery tool for Jira teams. (Free community license for Network Device Education Foundation)
Packages Test: Install/Uninstall

- Install package with no errors/warnings
- Start all routing daemons
- Kill each routing daemon and verify automatic restart
- Uninstall package with no errors/warnings

Run on Intel 64-bit:
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.04
- CentOS 6
- CentOS 7
- Debian 8
- Debian 9
Basic Protocol functionality

- Install package on Ubuntu 16.04
- Verify basic protocol functionality (2-5 basic tests each protocol):
  - RIP (IPv4)
  - RIPng (IPv6)
  - OSPFv2 (IPv4)
  - OSPFv3 (IPv6)
  - ISIS (IPv4 & IPv6)
  - BGP (IPv4 & IPv6)
  - LDP (with MPLS) (IPv4)

Tests use commercial Ixia IxANVL RFC Compliance Tests Suites
Comparing to previous build for Pull Requests to find new potential issues

Note: This output is from FRR 2.0
The current FRR 6.0 has no warnings or errors found by Clang SA
CI System: Topology Tests

FRR test framework for specific (complex) Networks

- Testing features which are specific to FRR or lack existing test tools
- Should be easy for everyone in community to run
- Tests exist for various protocols and features
- Allows anyone in community to add test which are executed by CI system (CI executes all the topology tests)
- Topology Simulations with Mininet on Ubuntu 16.04 on x86_64, i386, ARMv7 and ARMv8
- Uses pytest framework

https://github.com/frrouting/topotests
CI System: Address Sanitizer

- **Rebuilds and runs TopoTests with Address Sanitizer**
  - Similar to Valgrind – Finds:
    - Buffer Overflows
    - Memory leaks
    - Use after free/return
  - Better performance than Valgrind
  - (so far) no false positives!

https://github.com/google/sanitizers/wiki/AddressSanitizer
**Continuous Integration Result: FAILED**

See below for issues.

CI System Testrun URL: [https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/](https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/)

This is a comment from an EXPERIMENTAL automated CI system.
For questions and feedback in regards to this CI system, please feel free to email Martin Winter -- mwinter (at) opensourcerouting.org.

**Get source and apply patch from patchwork: Successful**

**Building Stage: Failed**

Ubuntu1204 amd64 build: Successful

CentOS6 amd64 build: Failed

DejaGNU Unitests (make check) failed for CentOS6 amd64 build
see PyTest log at [https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1006BUILD/ErrorLog/log_pytests.txt](https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1006BUILD/ErrorLog/log_pytests.txt)
CentOS6 amd64 build: config.status output from configure script can be found at [https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1006BUILD/config.status/config.status](https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1006BUILD/config.status/config.status)

FreeBSD9 amd64 build: Failed

DejaGNU Unitests (make check) failed for FreeBSD9 amd64 build
FreeBSD9 amd64 build: config.status output from configure script can be found at [https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1004BUILD/config.status/config.status](https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1004BUILD/config.status/config.status)

Debian9 amd64 build: Failed

DejaGNU Unitests (make check) failed for Debian9 amd64 build
Debian9 amd64 build: config.status output from configure script can be found at [https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1021BUILD/config.status/config.status](https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5360/artifact/C1021BUILD/config.status/config.status)
Step 5 (better...)
CI Results

Continuous Integration Result: SUCCESSFUL

Congratulations, this patch passed basic tests

Tested-by: NetDEF / OpenSourceRouting.org CI System

CI System Testrun URL: https://ci1.netdef.org/browse/FRR-FRRPULLREQ-5478/

This is a comment from an EXPERIMENTAL automated CI system. For questions and feedback in regards to this CI system, please feel free to email Martin Winter – mwinter (at) opensourcerouting.org.

CLANG Static Analyzer Summary

- Github Pull Request 3051, comparing to Git base SHA 9b00962

Fixed warnings:
- Memory error: Use-after-free in ldpd/ld.c, function lde_address_list_free, line 1624
- Memory error: Use-after-free in lib/msg-buffer.c, function msgbuf_clear, line 199
- Memory error: Use-after-free in lib/msg.c, function imsg_get_fd, line 305
- Logic error: Assigned value is garbage or undefined in lib/skiplist.c, function skiplist_insert, line 224

Static Analysis warning summary compared to base:
- Fixed warnings: 4
- New warnings: 0
Step 6
Manual Code Review

- **First manual step**
  - Manual review done by one or more of the maintainers

- **Back to submitter if questions or issues**
  - Any comments / disagreement need to be resolved in code or comments before a merge

- **Merge has to be done by a maintainer working for a different organization**

- **Missing / slow reviews or unresolved disagreements are addressed in weekly open meeting**
  - If still no agreement, then TSC has final decision
Step 7

Merge (finally!!)

*BUT - WAIT...*

*not yet done!*

*More testing after the merge of the updated branch*
Testing the merged code

- **Same tests as for PR plus:**
  - Build with more setups (incl slower ARM7 builds)
  - Build packages for all platforms and publish on CI
  - Build RPKI version of FRR with packages (and test them)
  - Run full RFC compliance test
    - Ubuntu 16.04 only
    - Just single pass (time constrains – 4 VMs run for 18hrs)
  - Coverity Scan (Commercial Static Analysis)
Coverity Scan

https://scan.coverity.com/projects/freerangerouting-frr

freerangerouting/frr

Overview  Project Settings  Analysis Settings  Members  Invite

coverity passed

Analysis Metrics

Version: FRR-master_2018-10-27_git.7508...

Oct 27, 2018  436,807  0.11
Last Analyzed  Lines of Code Analyzed  Defect Density

Defect changes since previous build dated Oct 27, 2018

0  0
Newly detected  Eliminated

Defects by status for current build

873  46  68  759
Total defects  Outstanding  Dismissed  Fixed
DONE.
Pull Request completed

We do on average 4 merges per day
(incl weekends and holidays)
And a few less frequent tests...
## RFC Compliance Test: Ixia IxANVL

### BGP4 Results

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Commit ID</td>
<td>1a664f5</td>
<td>3e71b5d</td>
<td>3d7746c</td>
<td>b84ccd4</td>
<td>731e65</td>
<td>bade23d</td>
<td>15a732</td>
<td>R2B3b</td>
</tr>
<tr>
<td>Commit Date</td>
<td>2017-03-08</td>
<td>2017-04-25</td>
<td>2017-05-16</td>
<td>2017-05-24</td>
<td>2017-06-02</td>
<td>2017-06-29</td>
<td>2017-07-01</td>
<td>2017-08-09</td>
</tr>
</tbody>
</table>

### ANVL-BGP4-1.1 MUST
- ANVL, setup verification
- DUT listens on TCP port 179 for BGP4 connection

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

### ANVL-BGP4-1.2 MUST
- ANVL, setup verification
- Establish BGP4 connection to the DUT and transit to Established state

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

### ANVL-BGP4-1.3 MUST
- ANVL, setup verification
- Router adds routes contained in the newly received Update Message to its routing table

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

### ANVL-BGP4-1.4 MUST
- ANVL, setup verification
- Router forwards new Update routes

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

### ANVL-BGP4-2.1 MUST
- RFC4271, Sect. 4. p11, Message Formats
- The maximum message size is 4096 octets. All implementations are required to support this maximum message size.

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

### ANVL-BGP4-3.1 MUST
- RFC4271, Sect. 4.2, page 13, OPEN message format
- OPEN Message Format
- After a TCP connection is established, the first message sent by each side is an OPEN message.

|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|

---

Page 31 of 37
Protocol Performance & Scale

▪ Executed on physical Linux PCs
  • DUT Hardware needs consistency to compare
  • 4 x PCs (Quad-Core, 8GB RAM, 6 Ethernet [Lanner FW-7525D])

▪ Test Equipment Ixia Chassis X16 (IxNetwork / IxAutomate)
  • 40+ Gbit/s ports

▪ Not yet automated
  • Test mostly automated
  • Needs result parsing (Pass/Fail criteria?)
Protocol Fuzzer: Spirent SPS-8000

BGP OPEN Message Format (RFC 4271)

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Version | My Autonomous System | Hold Time |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| BGP Identifier |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Opt Parm Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Optional Parameters (variable) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

- **Basic Function of the Fuzzer:**
  - attacks the System with malformed packets.
  - Protocol Fuzzer knows packet and doesn’t just random changes.
    - Fields missing, too long, too short
    - Value in field just 1 too high or too low
    - Negative values (or what might be negative if someone accidently uses a signed integer)
  - Verifies recovery of system between each test (or multiple tests)
It all adds up to a lot of work...
FRRouting activity since the fork

Code size doubled, > 2000 Pull Requests, > 9000 commits
# Testing Executed

Total Runtime for Tests executed Jan until October 2018

<table>
<thead>
<tr>
<th>Pull Request Create/Update</th>
<th>Pull Request Merge</th>
<th>Bi-Weekly &amp; Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Github triggered Pull Request Testing</td>
<td>Github triggered extended tests (Merges)</td>
<td>Manual triggered in-depth testing</td>
</tr>
<tr>
<td>5614 Runs</td>
<td>1610 Runs main</td>
<td>1 214 000 single RFC Compliance tests</td>
</tr>
<tr>
<td>Average runtime 79min</td>
<td>Average runtime 131min</td>
<td>Average runtime 3min</td>
</tr>
<tr>
<td>25 nodes in parallel required</td>
<td>30 nodes in parallel</td>
<td>2 nodes in parallel required</td>
</tr>
</tbody>
</table>

- **21 years VM runtime**
- **13 years VM runtime**

**Pull Request Merge**

- 530 Runs RPKI
- Average runtime 58min
- 11 nodes in parallel

**Bi-Weekly & Release**

- 1 214 000 single RFC Compliance tests
- Average runtime 3min
- 2 nodes in parallel required

**21 years VM runtime**

**13 years VM runtime**

(plus other manual tests)
Of the Pull Requests CI runs discover an error (and fail)

Giving instant feedback to contributor about commit without waiting for another community member to review
Challenges

‣ Other package dependencies
  • Testing quality?
  • Available for all our platforms?
  • We may need to run testing for them too…

‣ More contributions
  • → More testing work

‣ Keep up Documentation

‣ Keep up with new tests
JOIN US:
https://frrouting.org
Github.com/frrouting
Mailing lists: lists.frrouting.org