Testing Challenges in the Open Source Community

(with examples from the FRRouting community)

Martin Winter, NetDEF

Who am I?

- **Martin Winter**
  - Started working on Quagga in 2011 (mostly testing)
  - Co-Founded NetDEF 2013
  - Started FRRouting fork (NetDEF together with Cumulus) in Apr 2017
  - FRR Maintainer & TSC Member
  - Living in Switzerland

Who is NetDEF?

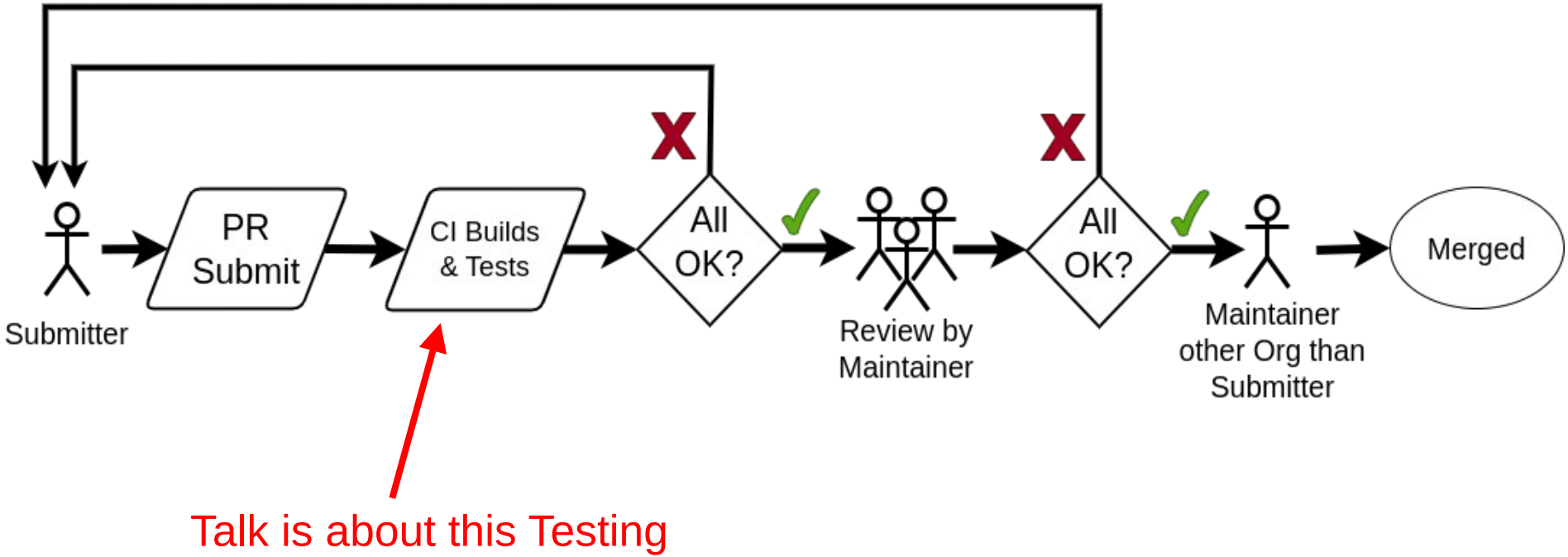
- **Network Device Education Foundation Inc**
  - California based US non-profit (501c3)
    - Living on donations from companies and implementing new FRR features as contracts for companies
  - Dedicated to Open Source & Education
  - Mostly working on FRRouting under the OpenSourceRouting project
  - Running CI System for the FRR community
  - Multiple maintainers

Open Source Routing

netDEF

Example FRRouting Testing

FRRouting Workflow



Test Framework

- CI System
  - We use Atlassian Bamboo (free license for non-profit/open source)
    - Currently ~600 agents (= Test/Build VMs controlled by CI)
  - Test execution all (most) on our own hardware
    - Main CI doesn't use Travis, Github actions etc as our scale doesn't qualify for free plans
    - Test platforms owned and operated by us (Currently 6 racks across 2 colos)

Compiling... (on VMs)
All on x86 (64bit) arch except where noted

ubuntu

Ubuntu 18.04 (+ i386 + ARM 32bit + ARM 64bit + PPC64)

Ubuntu 20.04

Ubuntu 22.04

Release packages built additionally for:
Ubuntu 20.04: + ARM 32bit + ARM 64bit
Ubuntu 22.04: + ARM 32 bit + ARM 64bit

CentOS

CentOS 7

Red Hat

RedHat 8
RedHat 9

debian

Debian 10 (+ ARM 64bit)

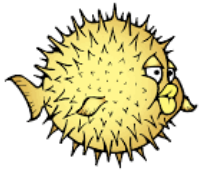
Debian 11 (+ ARM 64bit)

Debian 12

Release packages built additionally for:
Debian 10: + ARM 32bit + i386
Debian 11: + ARM 32bit + i386
Debian 12: + ARM 64bit + ARM 32bit + i386

FreeBSD®

FreeBSD 11
FreeBSD 12



OpenBSD

OpenBSD 7

Testing: Topotests

- FRR community developed Test Framework
- Based on pytest / Mininet
  - Collection of ~280 scripts with ~1600 tests total
- Linux Network Namespaces to build topologies
- Currently executed on Ubuntu i386, x86_64, arm8 and Debian x86_64
- More details?
  - http://docs.frrouting.org/projects/dev-guide/en/latest/topotests.html

Testing: Unittests

- Limited amount of Unittests
- Written by the developers

```
lib/test_typelist.py::TestTypelist::test_RBTREE_NONUNIQ_end PASSED       [ 97%]
lib/test_typelist.py::TestTypelist::test_ATOMSORT_UNIQ_end PASSED        [ 97%]
lib/test_typelist.py::TestTypelist::test_ATOMSORT_NONUNIQ_end PASSED     [ 97%]
lib/test_versioncmp.py::TestVersionCmp::test_exit_cleanly PASSED         [ 98%]
lib/test_xref.py::TestXref::test_exit_cleanly PASSED                     [ 98%]
lib/test_zmq.py::TestZMQ::test_refout SKIPPED (ZEROMQ not enabled)       [ 98%]
lib/cli/test_cli.py::TestCli::test_refout PASSED                         [ 98%]
lib/cli/test_commands.py::TestCommands::test_refout SKIPPED (QUAGGA_...) [ 99%]
lib/northbound/test_oper_data.py::TestNbOperData::test_refout PASSED     [ 99%]
ospf6d/test_lsdb.py::TestLSDB::test_refout PASSED                        [ 99%]
ospfd/test_ospf_spf.py::TestOspfSPF::test_refout PASSED                  [ 99%]
zebra/test_lm_plugin.py::TestLmplugin::test_refout PASSED                [100%]

----- generated xml file: /home/ci/cibuild.822/frr-source/tests/tests.xml ------
======================== 441 passed, 3 skipped in 13.06s ========================
```

- Executed as part of the builds (“make check”)
- Not much development in the community, mostly historic leftovers

Testing: Other Testing

- IXIA IxANVL (Commercial RFC Compliance Test Tool)
- Fuzzing (Coverity & Google OSS-Fuzz)
- Static Analysis (Clang)
- Address Sanitizer
  - Executing all of Topotests one more time with Address Sanitizer
- Installation Tests
  - Installing/Uninstalling RPM/DEB packages

Coverity

freerangerouting/frr

Overview | Project Settings | Analysis Settings | Members

coverity | passed 6 new defects

Analysis Metrics

Version: master\ build

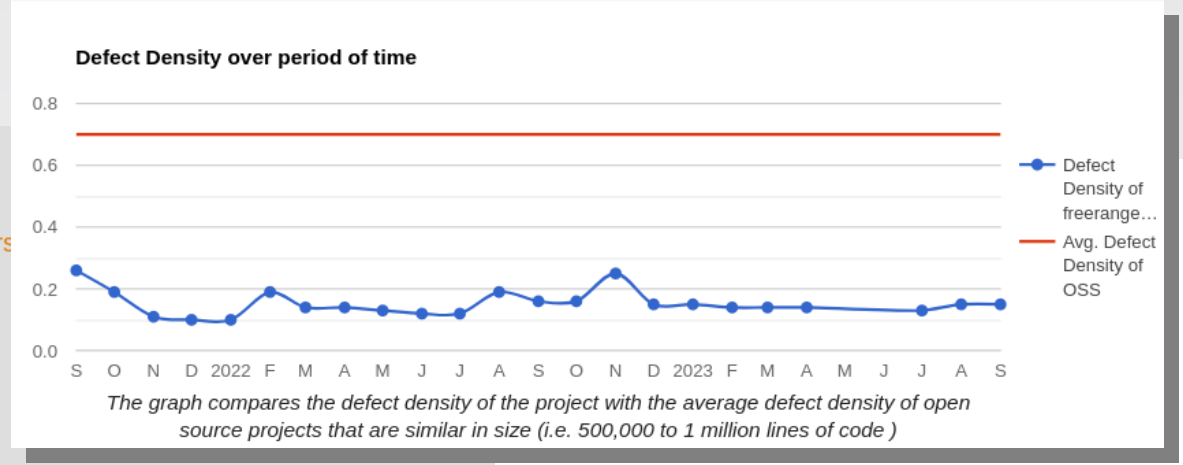
Sep 13, 2023	910,330	0.13
Last Analyzed	Lines of Code Analyzed	Defect Density

Defect changes since previous build dated Sep 08, 2023

6	26
Newly detected	Eliminated

Defects by status for current build

1,553	120	63	1,370
Total defects	Outstanding	Dismissed	Fixed



The graph compares the defect density of the project with the average defect density of open source projects that are similar in size (i.e. 500,000 to 1 million lines of code)

Topotato (soon)

log messages ☑ ☑ err ☑ warn ☑ notify ☑ info ☐ debug ☐ startup/??? | CLI ☐ | ☐ repeats with identical output (↟)

test_mld_basic.py::MLDBasic

- Download network diagram (as shown to the right, SVG)
- Download captured packets (pcap-ng)

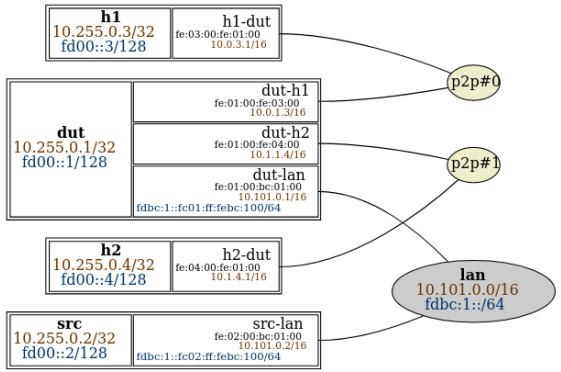
dut | zebra.conf | pim6d.conf

::startup
::prepare:#73:dut/pim6d/log
::prepare:#80:h1_dut/packet
::test_ssm:#89:h1/h1-dut/multicast-join[fdbc:1::fc02:ff:febc:100,ff05::2345]
::test_ssm:#91:dut/pim6d/log
::test_ssm:#92:dut/pim6d/vtysh[debug show mld interface dut-h1]
::test_ssm:#96:src/scapy[src-lan/IPv6/UDP]
::test_ssm:#106:h1_dut/packet
::test_asm:#110:h1/h1-dut/multicast-join[*,ff05::1234]
::test_asm:#112:dut/pim6d/log
::test_asm:#113:dut/pim6d/vtysh[debug show mld interface dut-h1]
::test_no_rtralert:#123:h1/scapy[h1-dut/IPv6/ICMPv6MLReport2]
::test_no_rtralert:#128:dut/pim6d/log
::test_invalid_group:#139:h1/scapy[h1-dut/IPv6/IPv6ExtHdrHopByHop/ICMPv6MLReport2]
::test_invalid_group:#144:dut/pim6d/log
::shutdown

::startup

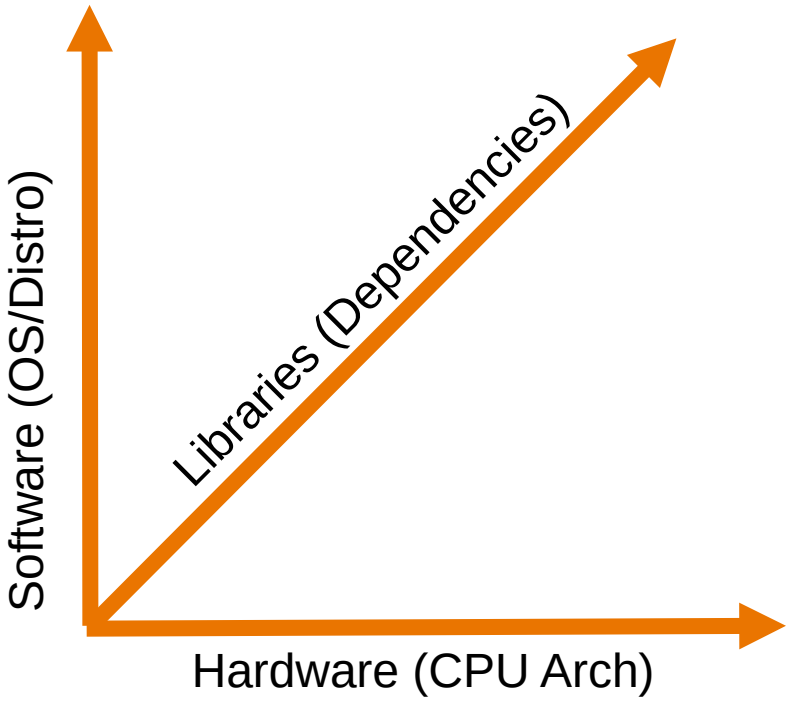
▸ passed after 1.60s

-1.116	dut	zebra	X25ZY-TGTPJ◇	info	yang model directory "/usr/lib/frr/share/yang" does not exist
-1.113	dut	zebra	X25ZY-TGTPJ◇	info	yang model directory "/usr/lib/frr/share/yang" does not exist
-1.080	dut	zebra	X0AM3-MRSP8◇	notif	can't setsockopt NETLINK_ADD_MEMBERSHIP for group RTNLGRP_TUNNEL(34), this linux kernel does not support it: Invalid argument(22)
-1.080	dut	zebra	PV4NQ-1GG1Z◇	notif	Registration for RTNLGRP_BRVLAN Membership failed : 22 Invalid argument
-1.043	dut	zebra	T83RR-8SM5G◇	notif	zebra 9.1-dev--git. starting: vty@2601
-1.001	dut	zebra			⊡ enable
-0.999	dut	zebra			⊡ configure
-0.999	dut	zebra			⊡ log file /tmp/tmphyhk3pgl/dut/zebra.log



How to know what platforms and features are used without spying ?

Complexity in Testing



- Do we cover all essential Distros? (no spying accepted in OS)
- What about Hardware? Which CPUs are relevant now or soon?
- What potential combination on libraries are used?

- **Which features to test ?**
- **Which compile options to test ?**

How to decide what to test

Hardware

Intel X86_64

MIPS

PowerPC 32bit

Intel i386

PowerPC 64EL

ARMv8 64bit

Big Endian vs Little Endian

Low Memory

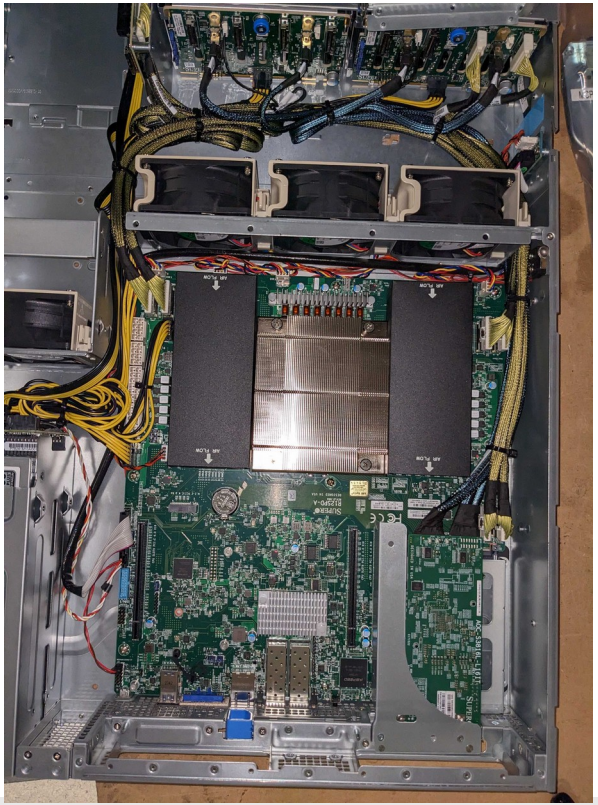
High-Performance vs Low-Performance

Challenge: Get the testing Hardware

2016
Hosted ARM Servers

2020
60 Raspi4's

2023
Supermicro 128-core ARM servers



Challenge: Big-Endian vs Little-Endian

- Network vs Host Byte Order
- Most current systems are little endian:
  - Intel i386, x86_64, ARM, PPC64, ...
- Very few Big-Endian systems remain:
  - MIPS (some), PPC (older 32bit
  - systems, some 64bit)
- Using Freescale T4240-QDS Development System
  - 24core e6500 PowerPC CPU (64bit)

How to decide what to test

Software

Develop your own
vs Third-Party Open Source
vs Commercial Test Frameworks

Scaling

Performance

Test Quality

Test Framework Selection

- Build your own framework?
  - FRR Example: Topotests
- Pick Open Source Framework (which can be expanded)
- Or commercial test framework/Service? (some might be free, others $$$)
  - Github
  - Atlassian Bamboo (CI)
  - Ixia IxANVL (Compliance Tests)

Long Term vs Short Term Benefits?

Challenge: Scaling Testbeds (Topotests)

- 4 Platforms:
  - x86_64 Ubuntu 18.04, i386 Ubuntu 18.04, x86_64 Debian 10, arm64 Ubuntu 18.04
  - Each platform runs for total of about 12 hrs
  - Initially all on standard VM's
- Changed to LXC
  - Initially better scale, more flexible, but all on same kernel and kernel locking for network operations

Challenge: Scaling Testbeds (cont)

- Impact LXC Kernel Locks on 128-core ARM64 server
  - 24 agents (= 24 parallel execution) running topology tests
  - 5% CPU load
  - 100GB RAM (out of 512GB)
  - → 2..10s delay on any network config
- Working to move to MicroVMs
  - Multiple kernels → No delays
  - Different kernels possible per testrun
  - Planning to play with actual time speedup in future

Challenge: Bad Test Output on Failure

Challenge: Stability

Trust in the Testing (CI) ?

- **CI Stability**
  - 600+ Build & Test Systems
  - X86_64, i386, arm8, arm7, ppc
  - Lots of different distros, no unique setup
  - Need to test different distros / kernels / architectures
  - Changes in dependencies
  - Constant add of new features and new tests

Challenge: Budget

- **Running a large CI costs $$$$**
  - Headcount and Infrastructure
- **Everyone loves some good CI and testing**
  - Not many (companies) are willing to pay (donate) for it
  - Easier to get money for
    - Feature development contracts
    - Support contracts

→ Limits on how much and how fast can be tested

The GOOD side of Open Source Testing

- **Honesty**
  - Ok to publish results with failed tests
- **Acceptance**
  - Community is more willing to accept testing and potential impact on release delay
- **Freedom**
  - Tests can be picked based on what is seen as useful
- **Running CI as a non-profit**
  - Discounts, Independent from any commercial Interests

Discussion

- How do you do it?
- Good/Bad Experiences
- Interested in sharing
  - Ideas?
  - Know-How?
  - Resources?

Questions / Feedback ?

mwinter at netdef dot org

Or post to RIPE opensource-wg list with your ideas / comments