Network verification: Lessons learned and outlook

Ratul Mahajan
“At least 41% of all calls that attempted to use T-Mobile’s network during the outage failed, including at least 23,621 failed calls to 911.”

“[An old woman] who has dementia, could not reach [her son] after her car would not start and her roadside-assistance provider could not call her to clarify her location; she was stranded for seven hours”
Anatomy of the outage (illustration)
Anatomy of the outage (illustration)
Anatomy of the outage (illustration)

What if T-Mobile could guarantee that no traffic will transit Denver?

What if T-Mobile could predict the impact of link failure?
Microsoft Says Config. Change Caused Azure Outage

Standard protocol for applying changes was not followed.

Microsoft: Misconfigured Network Device Causing Azure Outage

Microsoft suffers intermittent Azure outage over DNS resolution issues

May 03, 2019  By: Sebastian Moss

With Confidence In AWS Shaken, Who Could Benefit?

Amazon.com, Inc. (NASDAQ: AMZN) faced a setback Tuesday due to an outage at its cloud computing platform — Amazon Web Services, or AWS.

Google cloud is down, affecting numerous applications and services

Google details 'catastrophic' cloud outage events: Promises to do better next time

Data-center automation software was behind what Google describes as a 'catastrophic failure' last Sunday.

Amazon's massive AWS outage was caused by human error

One incorrect command and the whole internet suffers.

By Jason Del Rey  @Delrey  |  Mar 2, 2017, 2:20pm EST
Network verification to the rescue

Guarantee network behavior *

* Some behaviors under some assumptions
How network verification slices the problem

Configure, state
Software (OS, protocols)
Hardware
Verify
Trust
The “haystack” of network behaviors is HUGE

**Large scale**
- $O(10^3)$ devices
- $O(10^6)$ routes
- $O(10^9)$ packets

**Complex interactions**
- Distributed routing
- Protocol redistribution
- Rich route filtering
Batfish: A production-grade network verifier

A General Approach to Network Configuration Analysis

Ari Fogel  Stanley Fung  Luis Pedrosa  Meg Walraed-Sullivan
Ramesh Govindan  Ratul Mahajan  Todd Millstein

University of California, Los Angeles  University of Southern California  Microsoft Research

https://github.com/batfish/batfish
Batfish does proactive network verification

Verify configuration changes *before* they affect the network
Batfish’s original 4-stage pipeline

Configuration → Device model → Data plane → Violating flow → Bug report

- Parsing
- Routing simulation
- Verification
- Explanation
Batfish’s original 4-stage pipeline

- Configuration
- Device model
- Data plane
- Violating flow
- Bug report

New!
- Fidelity
- Parsing
- Routing simulation
- Verification
- Explanation

1500x faster, 400x larger networks
Lessons from the evolution of the Batfish configuration analysis tool

Matt Brown
Intentionet

Ari Fogel
Intentionet

Daniel Halperin
Intentionet

Victor Heorhiadi
Intentionet

Ratul Mahajan
Intentionet
University of Washington

Todd Millstein
Intentionet
UCLA
Lesson 1: Datalog was great for prototyping, but not for production use

Three key challenges:
1. **Expressiveness**
2. **Performance**
3. **Deterministic convergence**

**Solution**: replace Datalog with imperative code
Lesson 2: Model fidelity is hard, but not why you think

**Concern:** “Every software version will have different semantics!”

**Reality:** The real challenge is *undocumented semantics*

**Solution:** New stage to benchmark Batfish against an emulator
Lesson 3: Usability is hard for reasons you think, and then some

**Ambiguity:** “Hosts A can reach hosts B”

- ALL applications can reach SOME DNS server (e.g., in the same AZ)
- SOME SNMP collector can reach infrastructure elements
- ALL service frontends can reach ALL backend VIPs

**Solution:** custom assertions for each use case.
What’s next for network verification?

Make it an *effective* and *universal* practice

Key hurdles

- Lack of network automation
- Lack of expertise
- Lack of precise specifications
Network verification is only as good as its invariants

Network change

invariants

untested network behaviors
Network verification is only as good as its invariants

Facebook outage triggered by BGP configuration issue as services fail for 6 billion

WAN router IP address change blamed for global Microsoft 365 outage

Command line not vetted using full qualification process, says Redmond. We think it involved chewing gum somewhere

Paul Kurnec
Mon 30 Jan 2023 13:30 UTC
Inspiration from code coverage

<table>
<thead>
<tr>
<th>Files</th>
<th>Complexity</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>allinone/src/main/java/org/batfish/allinone</td>
<td>52.38%</td>
<td>62.91%</td>
</tr>
<tr>
<td>batfish-client/src/main/java/org/batfish/client</td>
<td>61.10%</td>
<td>64.57%</td>
</tr>
<tr>
<td>batfish-common-protocol/src/main/java/org/batfish</td>
<td>+12.00% 70.02%</td>
<td>-0.02% 78.02%</td>
</tr>
<tr>
<td>batfish/src/main/java/org/batfish</td>
<td>+4.00% 62.26%</td>
<td>-0.01% 70.63%</td>
</tr>
<tr>
<td>coordinator/src/main/java/org/batfish/coordinator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>minesweeper/src/main/java/org/batfish/minesweeper</td>
<td></td>
<td></td>
</tr>
<tr>
<td>question/src/main/java/org/batfish/question</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NetCov: Coverage for network configurations

Current view: top level - config
Test: internet2.init
Date: 2022-09-20

<table>
<thead>
<tr>
<th>Filename</th>
<th>Total</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>atla.conf</td>
<td>1211</td>
<td>26.1%</td>
</tr>
<tr>
<td>chic.conf</td>
<td>4376</td>
<td></td>
</tr>
<tr>
<td>clev.conf</td>
<td>1156</td>
<td></td>
</tr>
<tr>
<td>hous.conf</td>
<td>1196</td>
<td></td>
</tr>
<tr>
<td>kans.conf</td>
<td>1235</td>
<td></td>
</tr>
<tr>
<td>losa.conf</td>
<td>1832</td>
<td></td>
</tr>
<tr>
<td>newy32aoa.conf</td>
<td>770</td>
<td></td>
</tr>
<tr>
<td>salt.conf</td>
<td>568</td>
<td></td>
</tr>
<tr>
<td>seat.conf</td>
<td>1845</td>
<td></td>
</tr>
<tr>
<td>wash.conf</td>
<td>2723</td>
<td></td>
</tr>
</tbody>
</table>

https://github.com/UWNetworksLab/netcov
NetCov maps tested data plane state to covered config lines

RIB entry @R2
20.0.0.0/8 -> R1 (BGP)

BGP announcement
R1 → R2, 20.0.0.0/8

BGP session
R1 ↔ R2

Import policy @R2
match tag 74 permit

RIB entry @R1
20.0.0.0/8 -> ISP (BGP)

BGP peer config @R1
bgp peer R2

BGP peer config @R2
bgp peer R1
NetCov maps tested data plane state to covered config lines

Test Coverage for Network Configurations

Xieyang Xu\textsuperscript{1} Weixin Deng\textsuperscript{1} Ryan Beckett\textsuperscript{2} Ratul Mahajan\textsuperscript{1} David Walker\textsuperscript{3}

\textsuperscript{1}University of Washington \textsuperscript{2}Microsoft \textsuperscript{3}Princeton University

20.0.0.0/8 $\rightarrow$ ISP (BGP)

bgp peer R2

bgp peer R1

...
Toward “specification-less” verification

**Insight:** Network’s spec may not be known but a change’s intent is

- The change should have no impact on reachability
- The change should make the new subnet reachable from here
- The change should make traffic on *path1* take *path2*

**Solution:** Differential network verification

- A relational language for network changes
- A evaluation procedure based on finite state transducers

w/ Xieyang Xu, Zak Kincaid, Arvind Krishnamurthy, David Walker, and Yifei Yuan
Summary

Network verification is key to high network availability

First generation of tools have taught us a lot about what (does not) work

Must focus now on making network verification an effective and universal practice