Managing networks is challenging

Production networks are complex

- Security policies
- Traffic engineering
- Legacy devices
- Protocol inter-dependencies
- ...

- Even well-managed networks have downtime & security vulnerabilities
- Few good tools to ensure all networking components working together correctly
Previously, an intrusion detection and prevention (IDP) device inspected all traffic to/from dorms.

IDP couldn’t handle load; added bypass:

- IDP only inspected traffic between dorm and campus
- Seemingly simple changes

How do you know if it worked?
Understanding your network

Flow monitoring
Screenshot from Scrutinizer
NetFlow & sFlow analyzer,
snmp.co.uk/scrutinizer/

Configuration verification
hostname bgpdA
password zebra
router bgp 8000
bgp router-id 10.1.4.2

for the link between A and B
neighbor 10.1.2.3 remote-as 8000
neighbor 10.1.2.3 update-source lo0
network 10.0.0.0/7

for the link between A and C
neighbor 10.1.3.3 remote-as 7000
neighbor 10.1.3.3 ebgp-multihop
neighbor 10.1.3.3 next-hop-self
neighbor 10.1.3.3 route-map PP out

for link between A and D
neighbor 10.1.4.3 remote-as 6000
neighbor 10.1.4.3 ebgp-multihop
neighbor 10.1.4.3 next-hop-self
neighbor 10.1.4.3 route-map TagD in

route update filtering
ip community-list 1 permit 8000:1000
Past approach: Config. verification

e.g.: RCC for BGP [Feamster & Balakrishnan, NSDI’05]

Margrave for firewalls [Nelson, Barratt, Dougherty, Fisler, Krishnamurthi, LISA’10]
Our approach: Verify the network as close as possible to its actual behavior.
Data plane verification

Our approach: Verify the network as close as possible to its actual behavior

- Unified analysis across control software
- Catch bugs in control software
- Checks current snapshot

Diagram:
- Configuration
- Control plane
- Data plane state
  - Input
  - Predicted
- Network behavior
Architecture overview

Veriflow Network Verification Layer

1. Snapshot or real-time stream of:
   - Topology
   - Data plane state (forwarding tables)

2. Operator
   - Invariants from library or custom

3. Diagnosis
   - Check queried invariants against model

4. Confirmation of correctness, or violated invariants & counterexamples (vulnerabilities)

Network
Routers, switches, firewalls, ...
Our Two Tools

Anteater

- [Mai, Khurshid, Agarwal, Caesar, Godfrey, King, SIGCOMM 2011]
- Offline verification of data plane

VeriFlow

- [Khurshid, Zhou, Caesar, Godfrey, HotSDN 2012 (best paper)]
- [Khurshid, Zou, Zhou, Caesar, Godfrey, NSDI 2013]
- Online real-time verification of data plane
- Interoperates with OpenFlow controller
Is it possible to check network-wide invariants in real time as the network evolves?
Challenge #1: Obtaining real time view of network

- Solution: interpose between Software Defined Networking (SDN) controller and routers/switches

Challenge #2: Verification speed

- Past tools too slow and/or not incremental
- Solution: Algorithms :-)

Not so simple
VeriFlow architecture

- Logically centralized controller
- Thin, standard interface to data plane (e.g. OpenFlow)

software abstractions

app

app
VeriFlow architecture

- Logically centralized controller
- Thin, standard interface to data plane (e.g., OpenFlow)
- Software abstractions

VeriFlow
Verifying invariants quickly

VeriFlow

Generate Equivalence Classes

Updates

Fwd’ing rules
Equiv classes

0.0.0.0/1 64.0.0.0/3

Find only equivalence classes affected by the update via a multidimensional trie data structure
Verifying invariants quickly

VeriFlow

Generate Equivalence Classes

Generate Forwarding Graphs

Updates

All the info to answer queries!
Verifying invariants quickly

VeriFlow

Generate Equivalence Classes ➔ Generate Forwarding Graphs ➔ Run Queries

Updates

Good rules ➔ Bad rules

Diagnosis report

- Type of invariant violation
- Affected set of packets
Simulated network

- Real-world BGP routing tables (RIBs) from RouteViews totaling 5 million RIB entries
- Injected into 172-router network (AS 1755 topology)

Measure time to process each forwarding change

- 90,000 updates from RouteViews
- Check for loops and black holes
97.8% of updates verified within 1 ms
Deployment
Deployed Anteater and VeriFlow in University of Illinois campus backbone

- 244 routers, serving 70,000+ machines
- Predominantly OSPF, BGP, and static routing
- State collected via vty scripts
IDP was overloaded, operator introduced bypass

- IDP only inspected traffic for campus

bypass routed campus traffic to IDP through static routes

Introduced 9 loops
Errors discovered

Loops in internal network

Externally-exploitable DoS vulnerability

Packet loss due to ‘stale’ configs

Inconsistent security policy: over-exposure of router management interface

Duplicate IP addresses on router interfaces

Router vendor software error: faulty config output
Related work

Configuration verification


Data plane verification

• Static reachability in IP networks [Bush’03, Xie’05]
• FlowChecker [Al-Shaer, Al-Haj, SafeConfig ’10]
• ConfigChecker [Al-Shaer, Al-Saleh, SafeConfig ’11]
• Header Space Analysis [Kazemian, Varghese, and McKeown, NSDI ’12]
• NetPlumber [Kazemian, Chang, Zeng, Varghese, McKeown, Whyte, NSDI ’13]
What we’ve seen

Data plane verification is valuable

• Unified network-wide analysis across protocols
• Demonstrated effectiveness in large campus network

Real-time verification is feasible

• millisecond timescales enabled by SDN + algorithms
Thanks!