Lecture 12: Network Verification

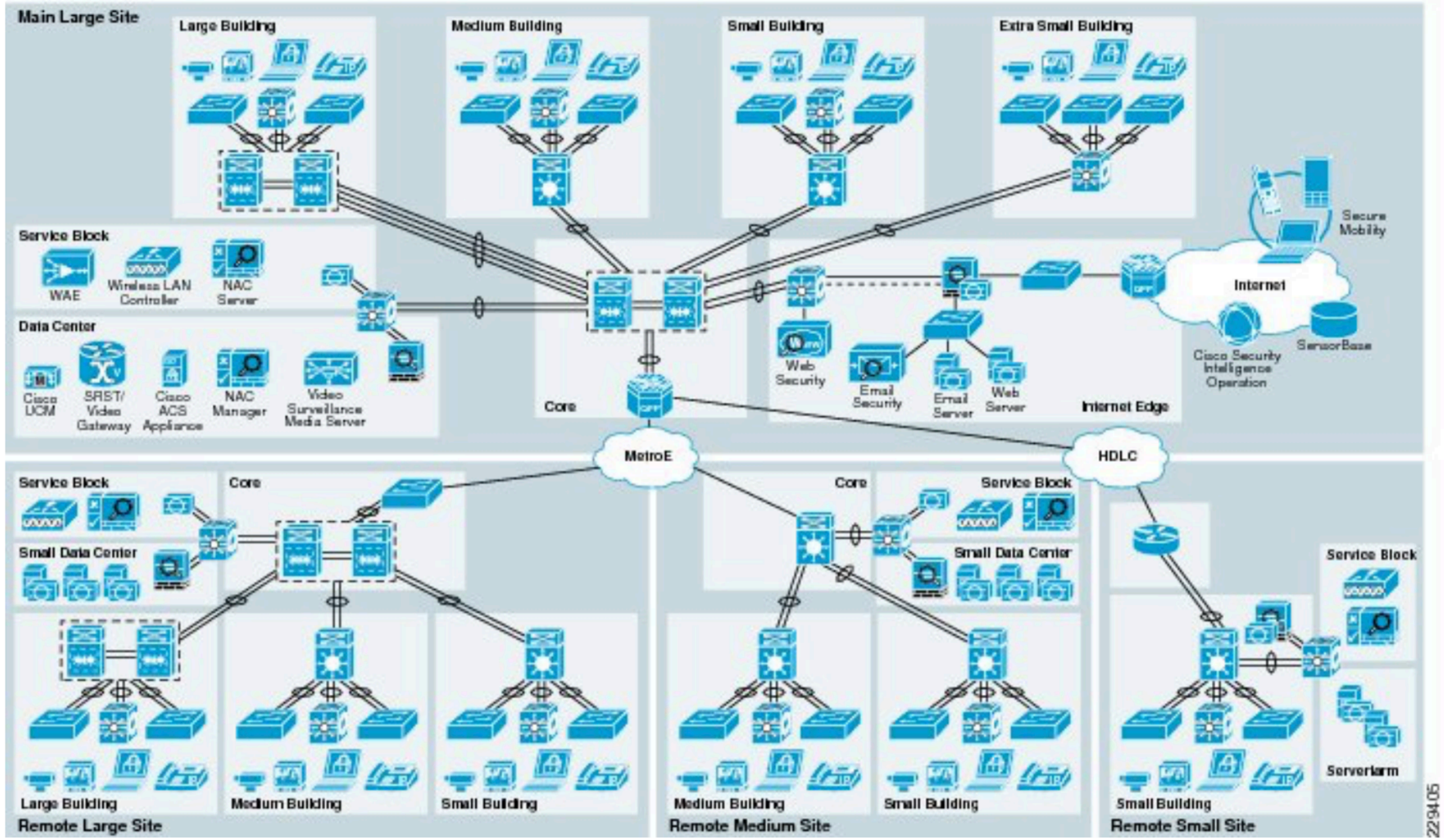
CS 234 / NetSys 210: Advanced Computer Networks

Sangeetha Abdu Jyothi

UCIrvine

This lecture uses material from Veriflow talk and tutorial by Brighten Godfrey and Santhosh Prabhu, and slides by George Varghese

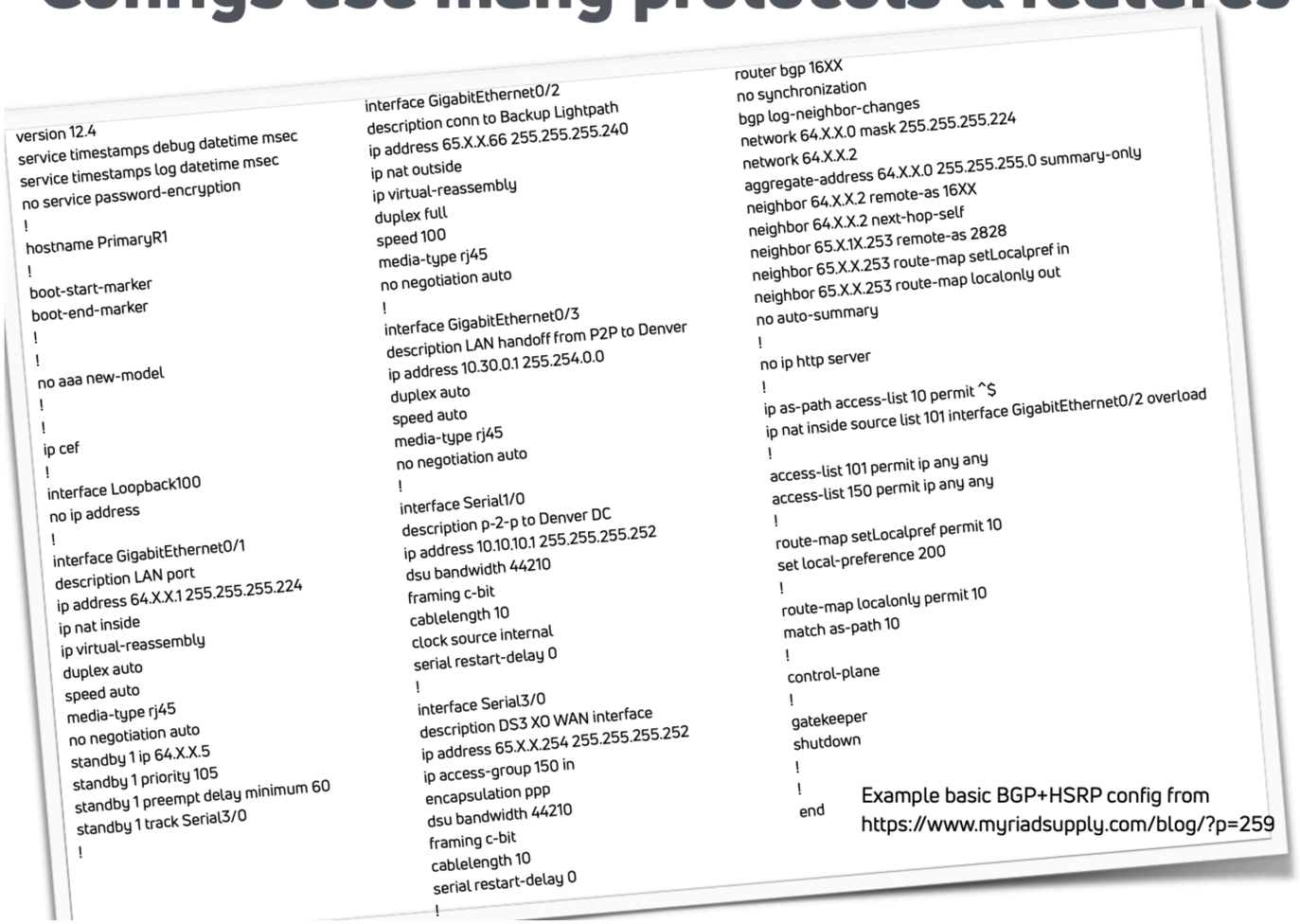
Networks are so complex!



Source: http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Medium_Enterprise_Design_Profile/MEDP/chap5.html

Complex configurations

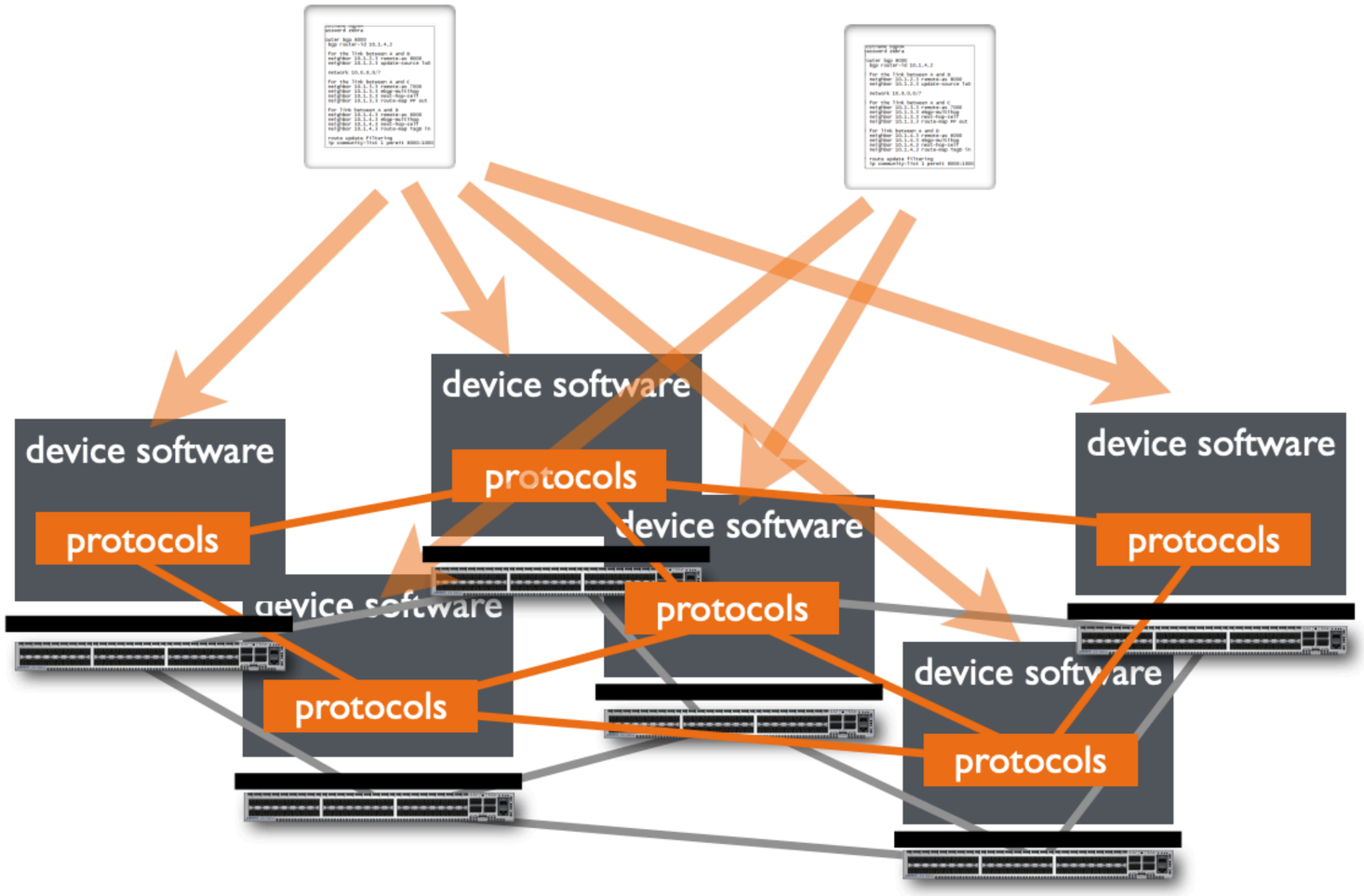
Configs use many protocols & features



```
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname PrimaryR1
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
!
!
ip cef
!
interface Loopback100
no ip address
!
interface GigabitEthernet0/1
description LAN port
ip address 64.X.X.1 255.255.255.224
ip nat inside
ip virtual-reassembly
duplex auto
speed auto
media-type rj45
no negotiation auto
standby 1 ip 64.X.X.5
standby 1 priority 105
standby 1 preempt delay minimum 60
standby 1 track Serial3/0
!
interface GigabitEthernet0/2
description conn to Backup Lightpath
ip address 65.X.X.66 255.255.255.240
ip nat outside
ip virtual-reassembly
duplex full
speed 100
media-type rj45
no negotiation auto
!
interface GigabitEthernet0/3
description LAN handoff from P2P to Denver
ip address 10.30.0.1 255.254.0.0
duplex auto
speed auto
media-type rj45
no negotiation auto
!
interface Serial1/0
description p-2-p to Denver DC
ip address 10.10.10.1 255.255.255.252
dsu bandwidth 44210
framing c-bit
cablelength 10
clock source internal
serial restart-delay 0
!
interface Serial3/0
description DS3 XO WAN interface
ip address 65.X.X.254 255.255.255.252
ip access-group 150 in
encapsulation ppp
dsu bandwidth 44210
framing c-bit
cablelength 10
serial restart-delay 0
!
router bgp 16XX
no synchronization
bgp log-neighbor-changes
network 64.X.X.0 mask 255.255.255.224
network 64.X.X.2
aggregate-address 64.X.X.0 255.255.255.0 summary-only
neighbor 64.X.X.2 remote-as 16XX
neighbor 64.X.X.2 next-hop-self
neighbor 65.X.1X.253 remote-as 2828
neighbor 65.X.X.253 route-map setLocalpref in
neighbor 65.X.X.253 route-map localonly out
no auto-summary
!
no ip http server
!
ip as-path access-list 10 permit ^$
ip nat inside source list 101 interface GigabitEthernet0/2 overload
!
access-list 101 permit ip any any
access-list 150 permit ip any any
!
route-map setLocalpref permit 10
set local-preference 200
!
route-map localonly permit 10
match as-path 10
!
control-plane
!
gatekeeper
shutdown
!
!
end
```

Example basic BGP+HSRP config from
https://www.myriadsupply.com/blog/?p=259

Distributed Route Computation



Networks are Difficult to Change

89%
of operators never sure that config changes are bug-free

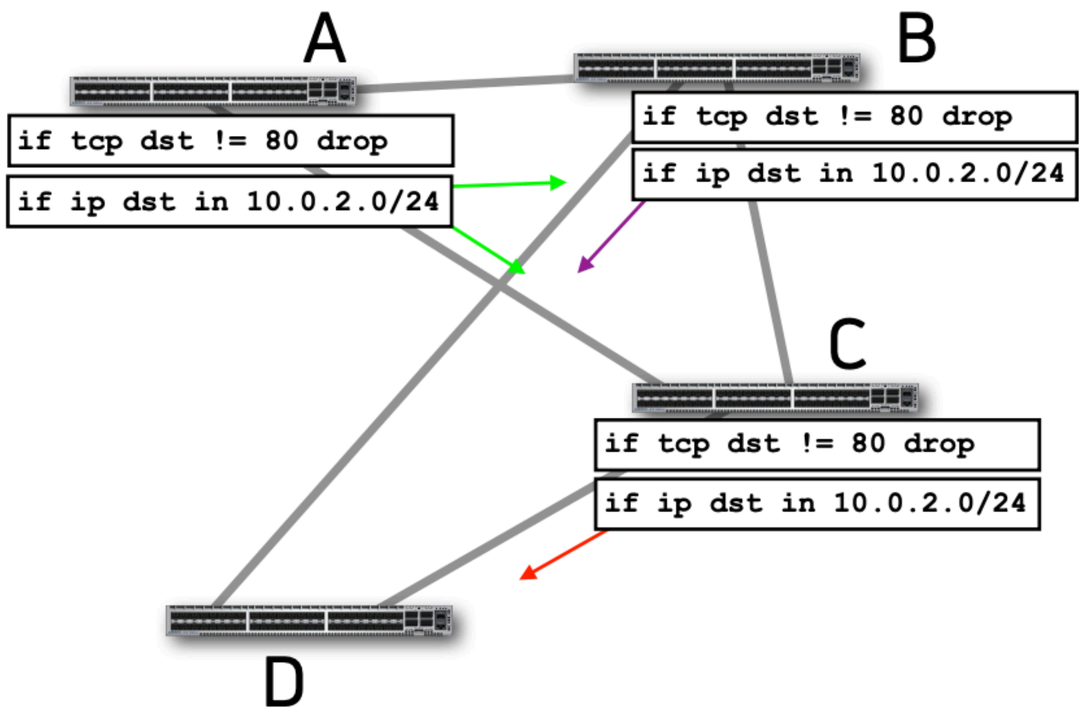
82%
concerned that changes would cause problems with existing functionality

– Survey of network operators
[Kim, Reich, Gupta, Shahbaz, Feamster, Clark, USENIX NSDI 2015]

Simple Questions are Hard to Answer

- Which packets from A can reach B?
- Is Group X provably isolated from Group Y?
- Is the network causing poor performance or the server? Are QoS settings to blame?
- Why is my backbone utilization poor?
- Is my load balancer distributing evenly?
- Where are there mysterious packet losses?

Network Correctness (a.k.a Intent)



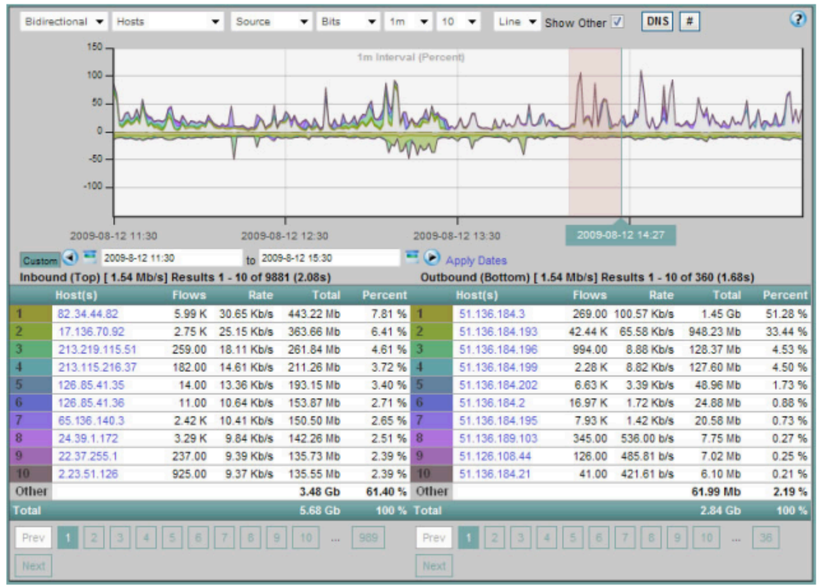
Intent Checklist

- ☑ All packets should follow the shortest paths
- ☑ There should not be any loops
- ☑ There should be multiple paths between A and D
- ☑ HTTP Packets in the subnet 10.0.2.0/24 should reach D
- ☑ External networks attached to A should be isolated from D except via HTTP

Ensuring Correct Operations Before Network Verification

Manual spot-checking (pings, traceroutes)

Monitoring of events & flows

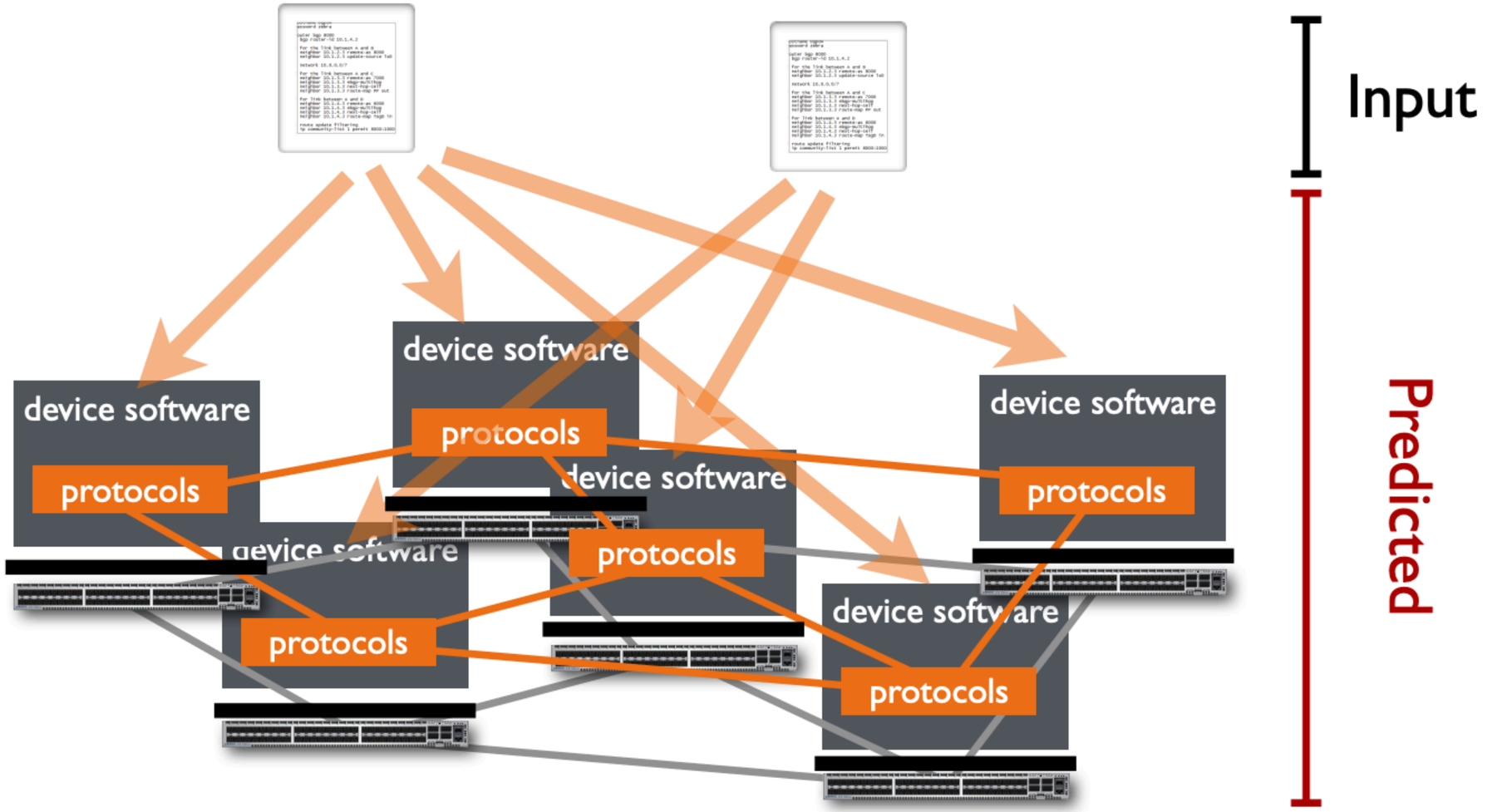


Screenshot from Scrutinizer NetFlow & sFlow analyzer, snmp.co.uk/scrutinizer/

Network Verification

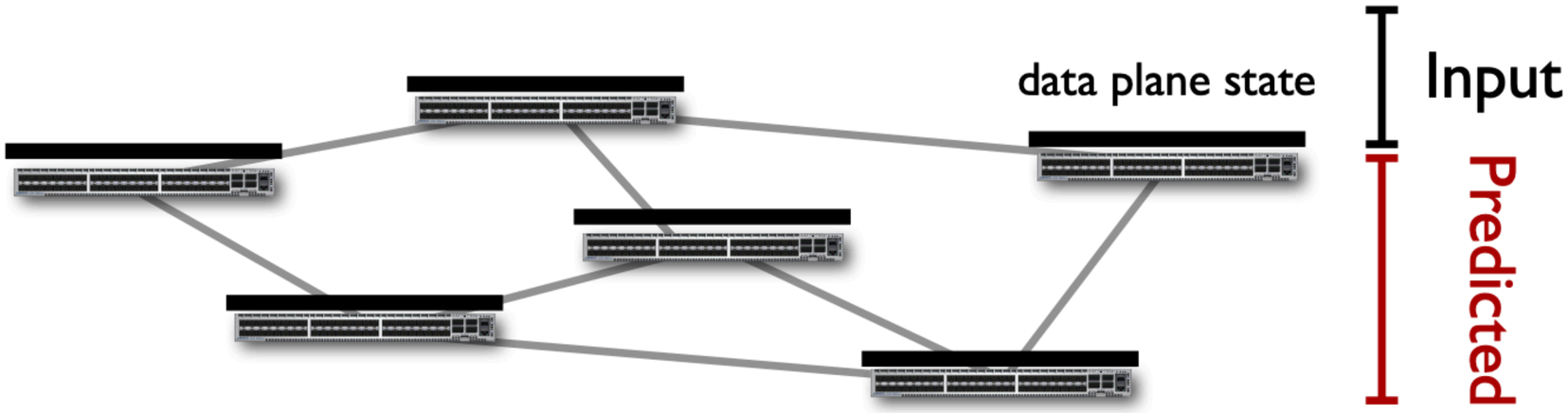
The process of proving whether an **abstraction** of the network satisfies the network-wide intent.

Configuration Verification



Data Plane Verification

Verify the network
as close as possible
to its actual behavior

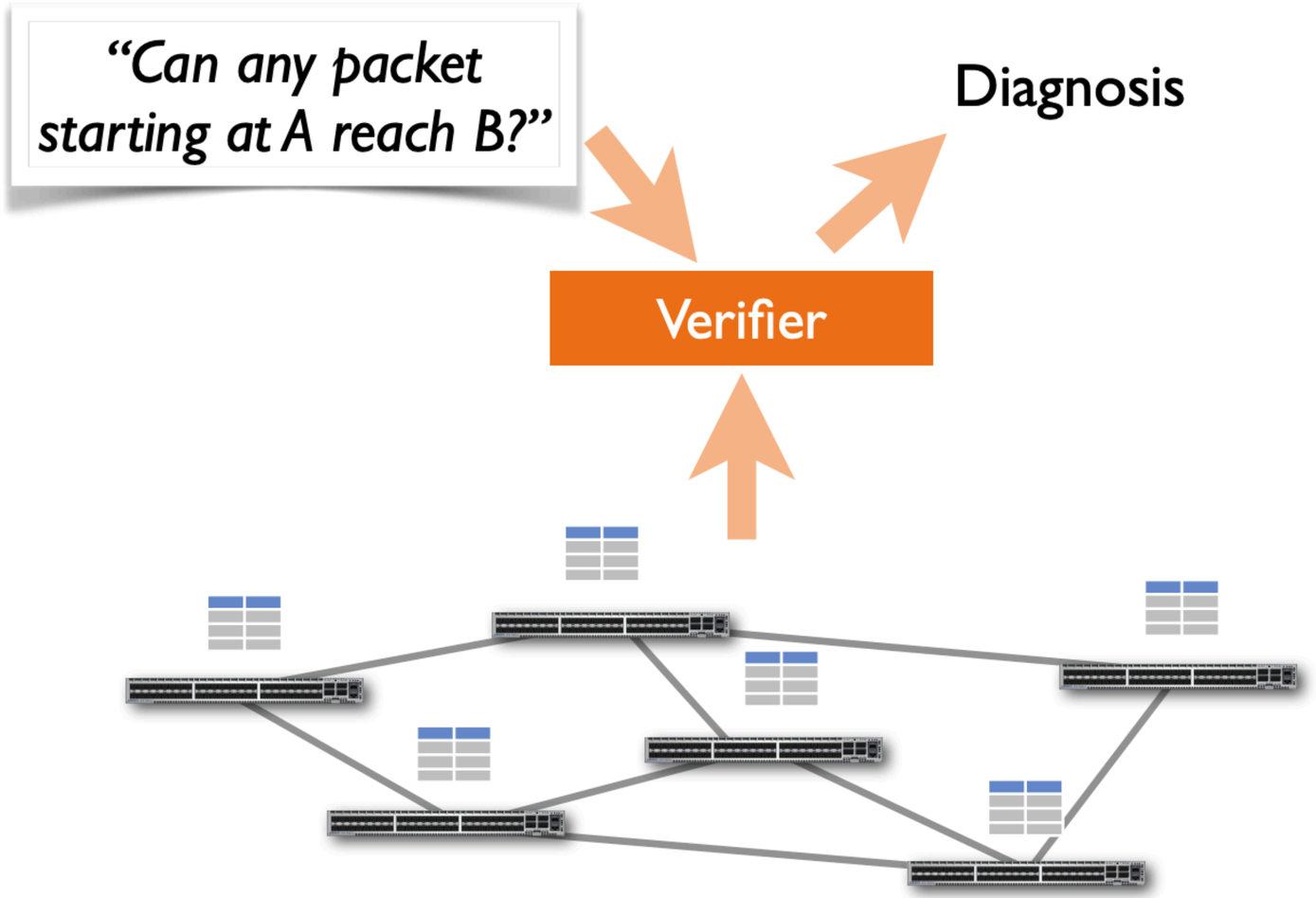


Data Plane Verification

Data Plane Verification Benefits

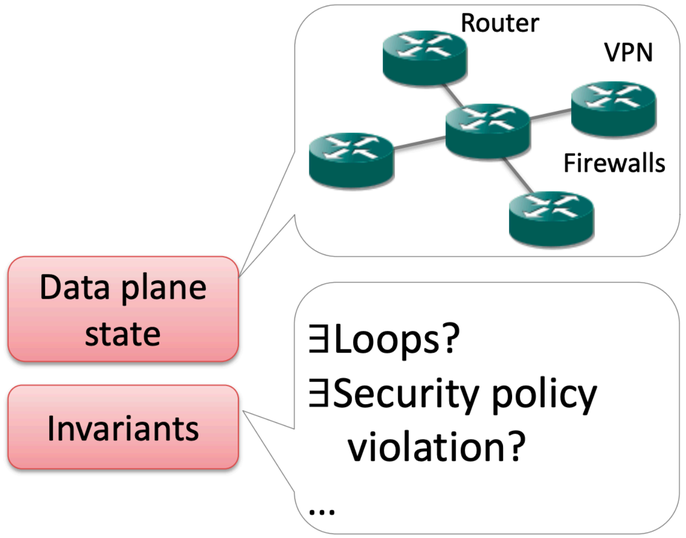
- Insensitive to control protocols
- Accurate model
- Checks current snapshot

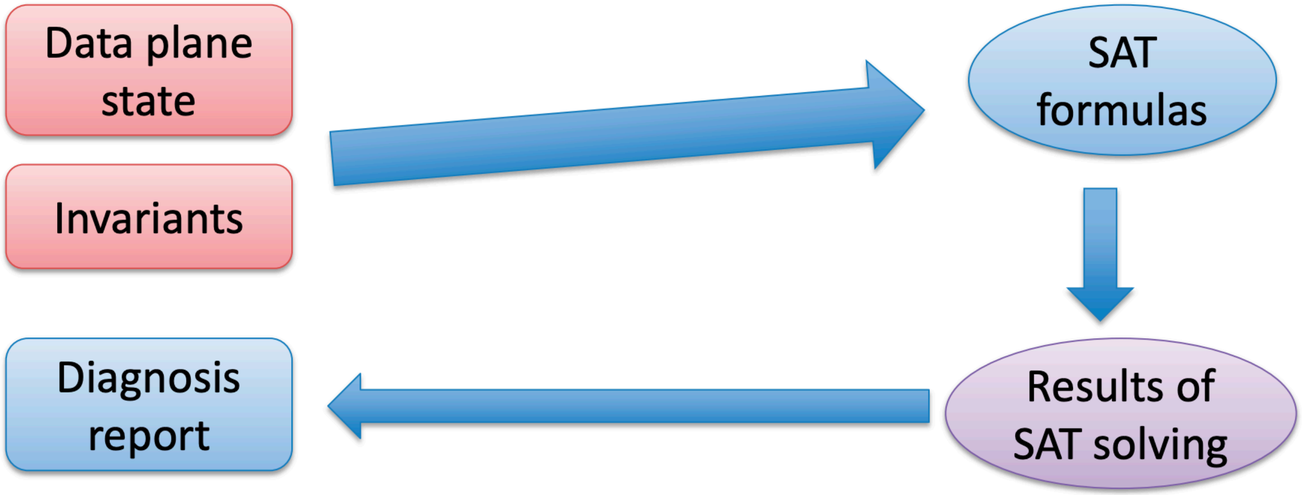
Data Plane Verification Architecture



Debugging the Data Plane with Anteater [SIGCOMM'11]

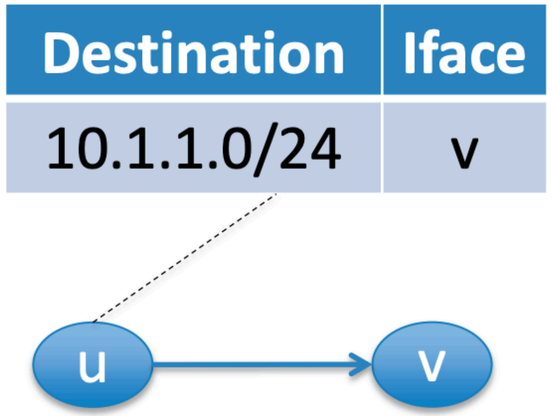
- Express data plane and invariants as SAT
- Check with off-the-shelf SAT solver (Boolector)





Data plane as boolean functions

- Define P(u, v) as the policy function for packets traveling from u to v
  - A packet can flow over (u, v) *if and only if* it satisfies P(u, v)



P(u, v) = dst_ip ∈10.1.1.0/24

Some Examples

Destination	Iface
10.1.1.0/24	v
Drop port 80 to v	

P(u, v) = dst_ip ∈10.1.1.0/24
∧ dst_port ≠ 80

Packet filtering

Destination	Iface
10.1.1.0/24	v
10.1.1.128/25	v’
10.1.2.0/24	v

P(u, v) = (dst_ip ∈10.1.1.0/24
∧ dst_ip ∉ 10.1.1.128/25)
∨ dst_ip ∈10.1.2.0/24

Longest prefix matching

Reachability as SAT solving

- Goal: reachability from u to w

  $C = (P(u, v) \wedge P(v,w))$ is satisfiable

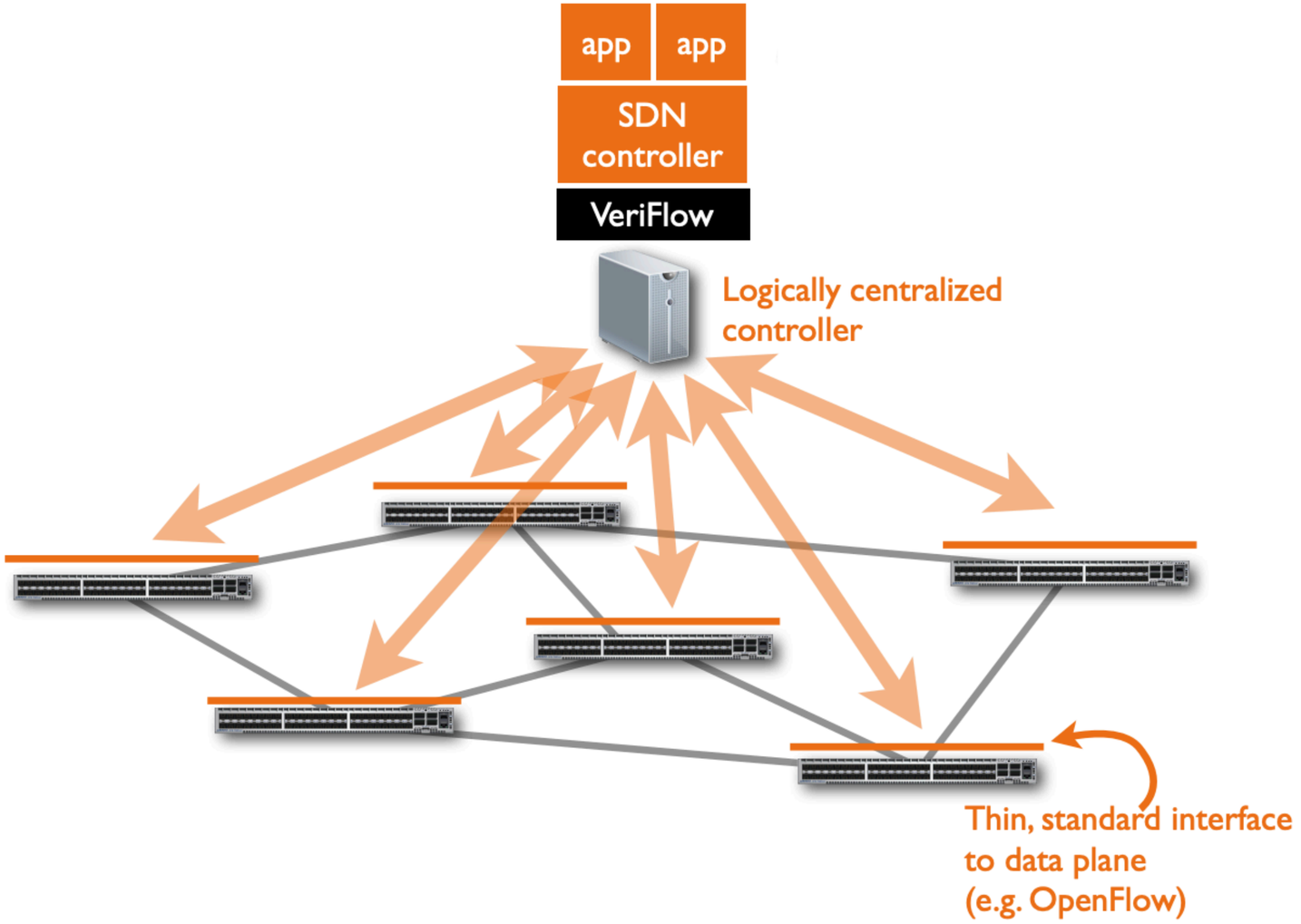
- SAT solver determines the satisfiability of C

Anteater challenges

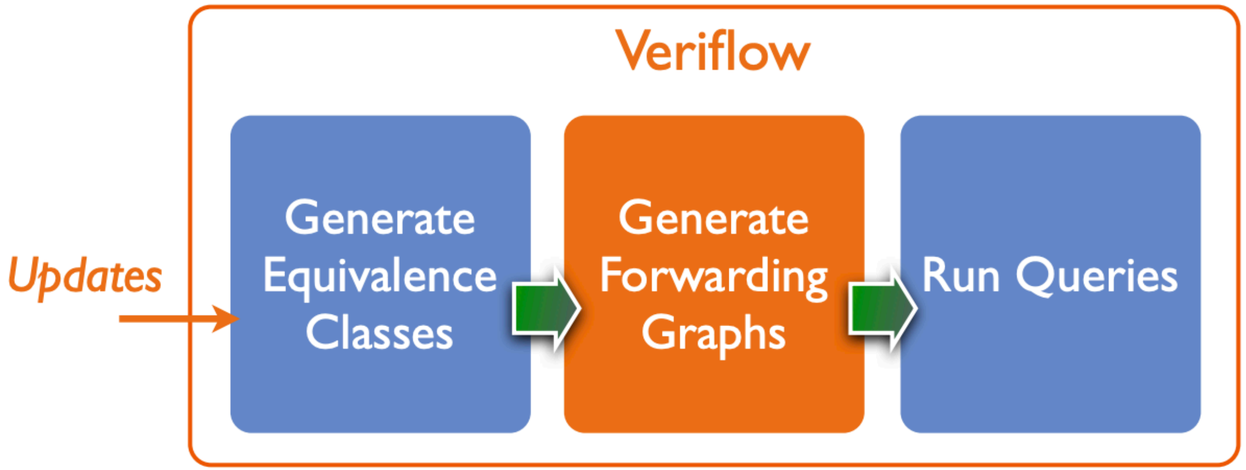
- Challenge #1: Obtaining real time view

- Challenge #2: Verify quickly

Real-Time Data Plane Verification: Veriflow

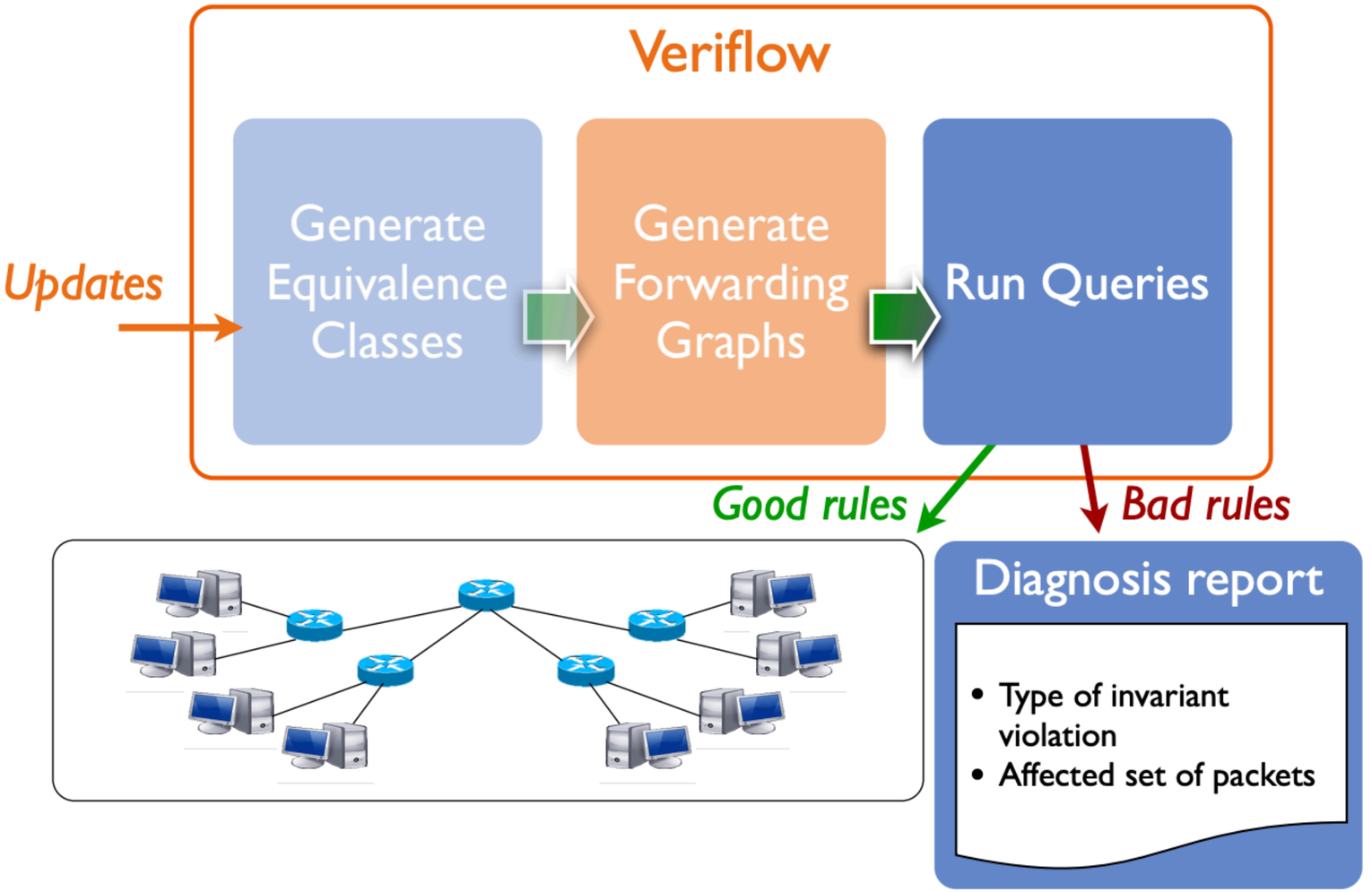


Verifying Invariants Quickly

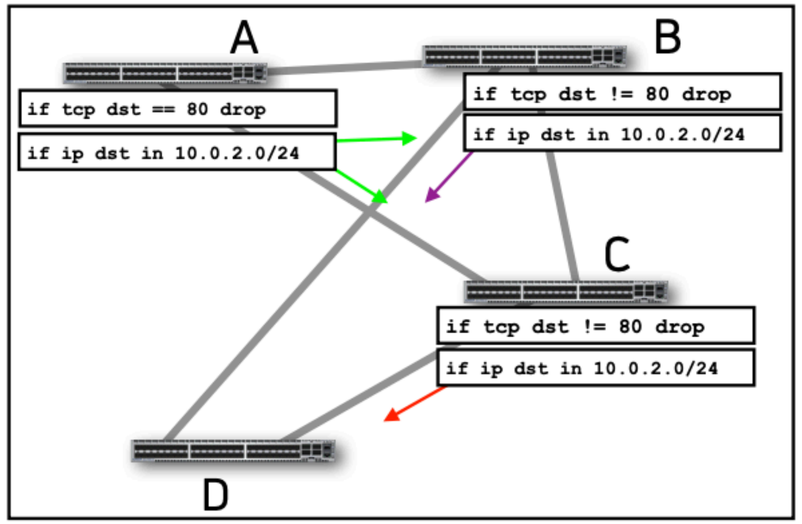


- Limit the search space
- Represent forwarding behaviors using graphs
- Run Light-weight graph-based algorithm to check

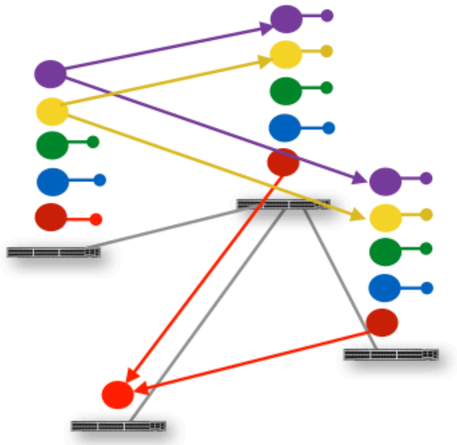
Real-time Verification with Veriflow



Forwarding Graphs



ip dst [0, 167772671], tcp dst [0, 65535]
ip dst [167772672, 167772927], tcp dst [80, 80]
ip dst [167772928, 4294967295], tcp dst [0, 65535]
ip dst [167772672, 167772927], tcp dst [0, 79]
ip dst [167772672, 167772927], tcp dst [81, 65535]

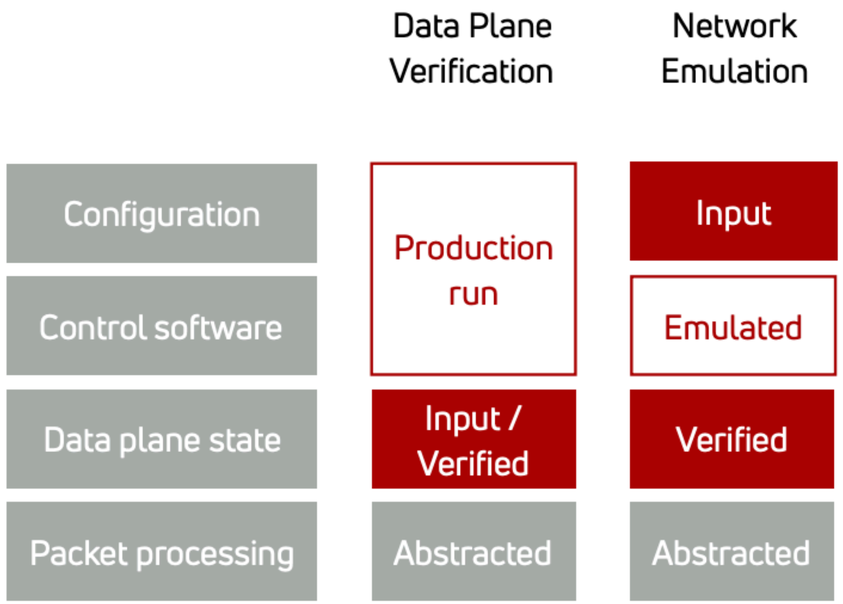


Intent API

- What intents can you check?
  - Anything within data plane state (forwarding rules)...
  - ...that can be verified incrementally
- Veriflow's API enables custom query algorithms
  - Full access to forwarding graph
  - For incremental verification: Gives access to the "diff": forwarding subgraph affected by an update from the SDN controller
  - Verification becomes a standard graph traversal algorithm

Configuration Analysis

Landscape of Approaches



Emulating Production Networks

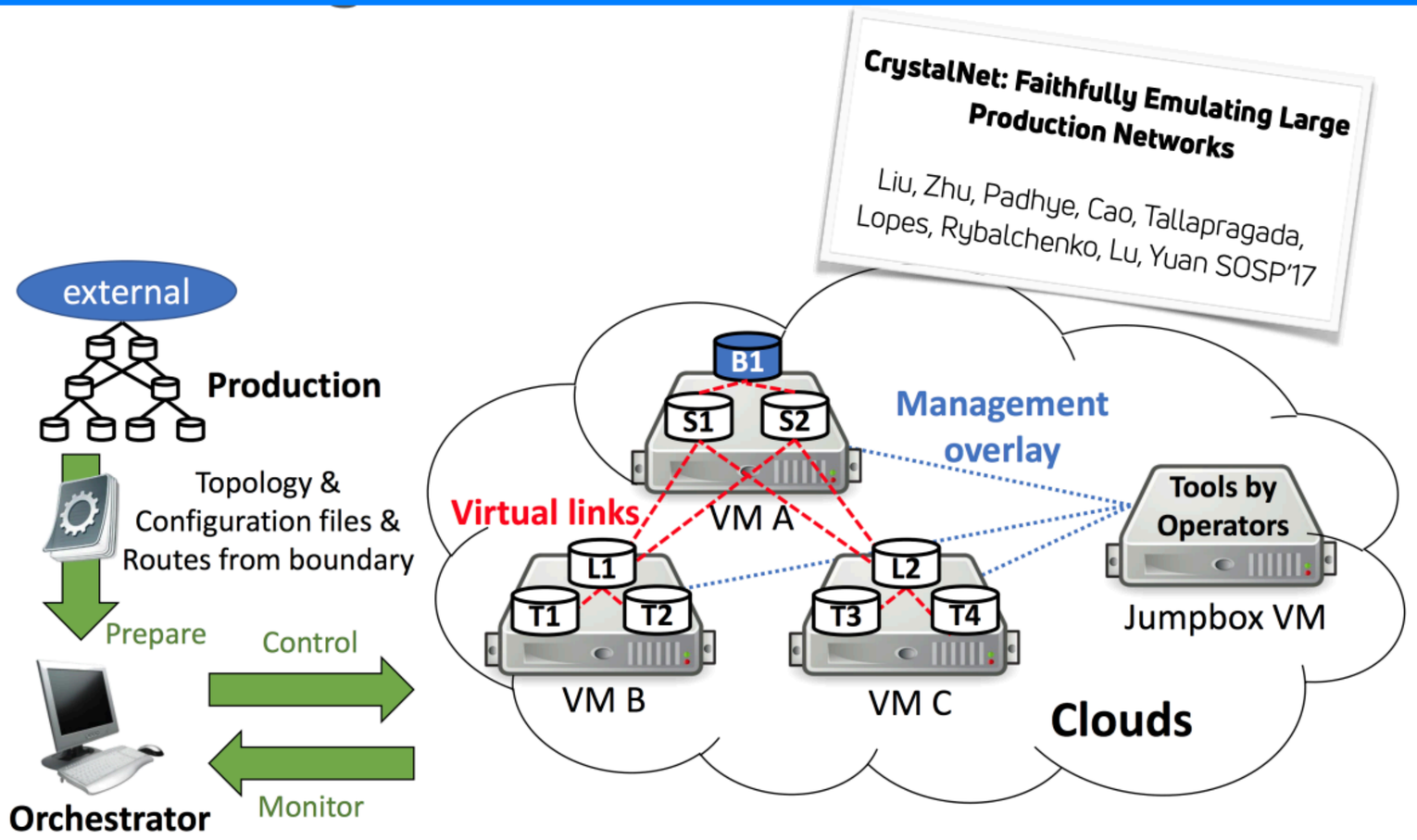
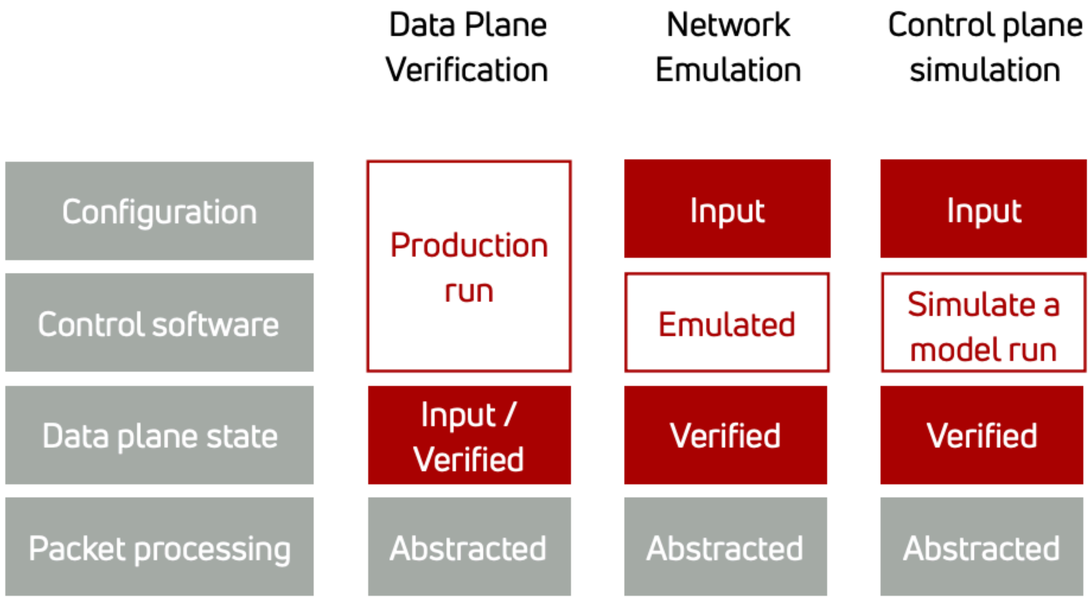


Image Reproduced from the authors' SOSP '17 paper

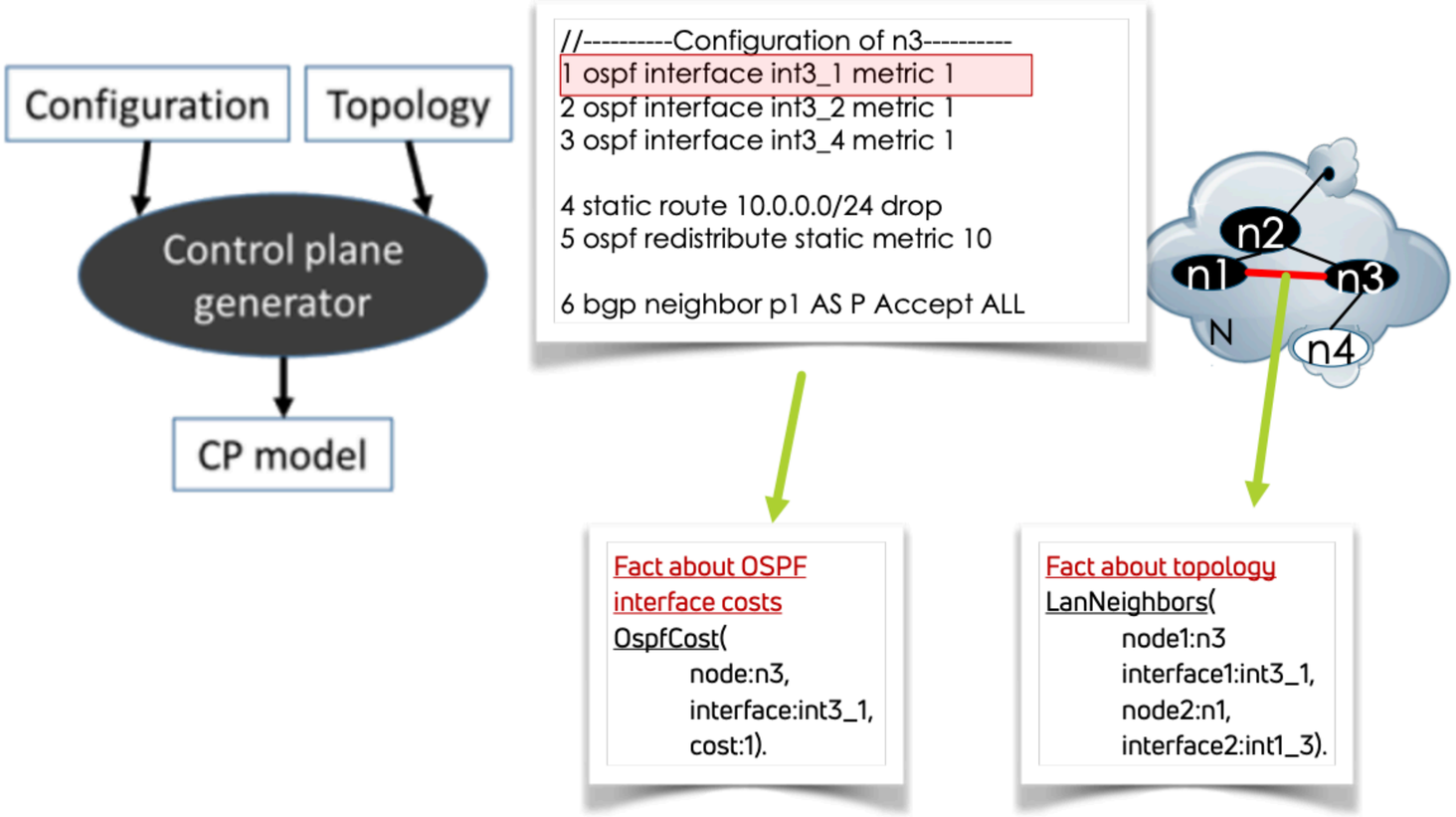
Landscape of Approaches



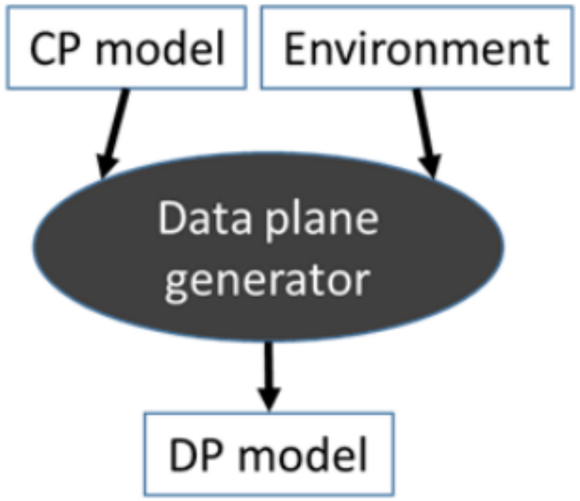
Control Plane Simulation

- Challenges in faithfully deriving the data plane
- Batfish [NSDI'15]
    - Approach: High-fidelity declarative model of control plane
        - Set of relations that expresses the network's control plane computation
        - Provides queryability and provenance for free

Stage 1: Extract control plane model



Stage 2: Compute data plane

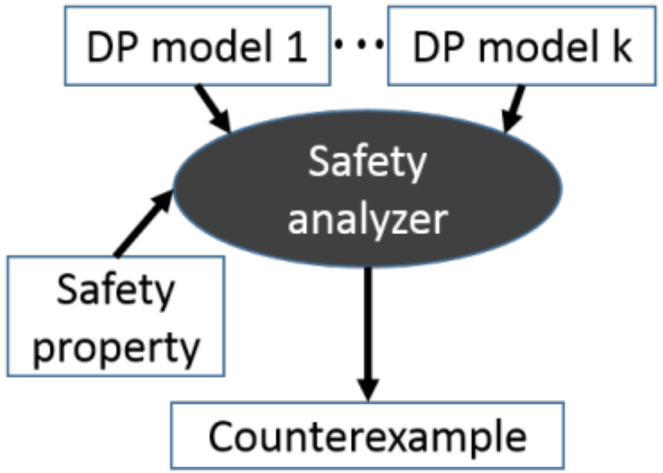



```
OspfExport(
        node=n2,
        network=10.0.0.0/24,
        cost=10,
        type=ospfE2).
```

```
InstalledRoute(route={
        node=n1,
        network=10.0.0.0/24,
        nextHop=n2
        administrativeCost=110,
        protocolCost=10,
        protocol=ospfE2}).
```

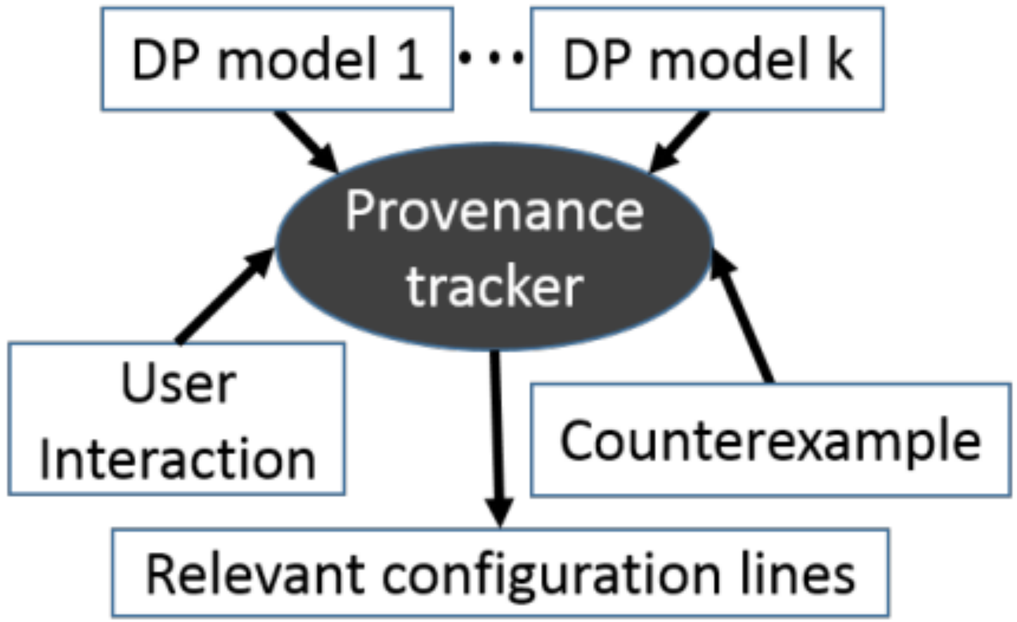
```
Fib(
        node=n1,
        network=10.0.0.0/24,
        egressInterface=int1_2).
```

Stage 3: Data plane analysis



Counterexample of multipath consistency
{
IngressNode=n1,
SrcIp=0.0.0.0,
DstIp=10.0.0.2,
IpProtocol=0
}

Stage 4: Helping Repair



The Research Landscape

Configuration Verification

Configuration

Control software

Data plane state

Packet processing

Configuration verification

- RCC (Detecting BGP config faults w/static analysis) [Feamster & Balakrishnan, USENIX '05]
- ConfigAssure [Narain et al, '08]
- ConfigChecker [Al-Shaer, Marrero, El-Atawy, ICNP '09]
- Batfish [Fogel, Fung, Pedrosa, Walraed-Sullivan, Govindan, Mahajan, Millstein, NSDI'15]
- Bagpipe [Weitz, Woos, Torlak, Ernst, Krishnamurthy, Tatlock, NetPL'16 & OOPSLA'16]
- ARC [Gember-Jacobson, Viswanathan, Akella, Mahajan SIGCOMM'16]
- ERA [Fayaz, Sharma, Fogel, Mahajan, Millstein, Sekar, Varghese, OSDI'16]
- Minesweeper [Beckett, Gupta, Mahajan, Walker SIGCOMM'17]
- Plankton [Prabhu, Kheradmand, Godfrey, Caesar APNet'17]
- CrystalNet [Liu, Zhu, Padhye, Cao, Tallapragada, Lopes, Rybalchenko, Lu, Yuan SOSP'17]

Control Software Verification

Configuration

Control software

Data plane state

Packet processing

Verifiable controllers & control languages

- NICE [Canini, Venzano, Perešíni, Kostić, Rexford, NSDI'12]
- NetKAT [Anderson, Foster, Guha, Jeannin, Kozen, Schlesinger, Walker, POPL'14]
- Kinetic: Verifiable Dynamic Network Control [Kim, Gupta, Shahbaz, Reich, Feamster, Clark, NSDI'15]

Data Plane Verification

Configuration

Control software

Data plane state

Packet processing

Static

- On static reachability in IP networks [Xie, Zhan, Maltz, Zhang, Greenberg, Hjalmtysson, Rexford, INFOCOM '05]
  - Computed reachable sets with IP forwarding rules
- FlowChecker [Al-Shaer, Al-Haj, SafeConfig '10]
- Anteater [Mai, Khurshid, Agarwal, Caesar, G., King, SIGCOMM'11]
- Header Space Analysis [Kazemian, Varghese, and McKeown, NSDI '12]
- Network-Optimized Datalog (NoD) [Lopes, Bjørner, Godefroid, Jayaraman, Varghese, NSDI 2015]

Real time (incremental)

- VeriFlow [Khurshid, Zou, Zhou, Caesar, G., HotSDN'12, NSDI'13]
- NetPlumber [Kazemian, Chang, Zeng, Varghese, McKeown, Whyte, NSDI '13]
- CCG [Zhou, Jin, Croft, Caesar, G., NSDI'15]
- DeltaNet [Horn, Kheradmand, Prasad] NSDI'17]

Optimizations

- Libra: Divide and Conquer to Verify Forwarding Tables in Huge Networks [Zeng, Zhang, Ye, Google, Jeyakumar, Ju, Liu, McKeown, Vahdat, NSDI'14]
- Atomic Predicates [Yang, Lam, ToN'16]
- ddNF [Bjorner, Juniwal, Mahajan, Seshia, Varghese, HVC'16]

Richer data plane models

- SymNet [Stoenescu, Popovici, Negreanu, Raiciu, SIGCOMM'16]
- Mutable datapaths [Panda, Lahav, Argyraki, Sagiv, Shenker, NSDI'17]

Packet Processing Verification

Configuration

Control software

Data plane state

Packet processing

Verification of data plane software and data plane languages

- Software Dataplane Verification [Dobrescu, Argyraki, NSDI'14]
- Executable Formal Semantic of P4 and Applications [Kheradmand, Rosu, P4 Workshop'17]
- A Formally Verified NAT [Zaostrovnykh, Pirelli, Pedrosa, Argyraki, Candea, SIGCOMM'17]

Thanks!