Data Plane Verification and Anteater

Brighten Godfrey
University of Illinois

Work with Haohui Mai, Ahmed Khurshid, Rachit Agarwal, Matthew Caesar, and Sam King

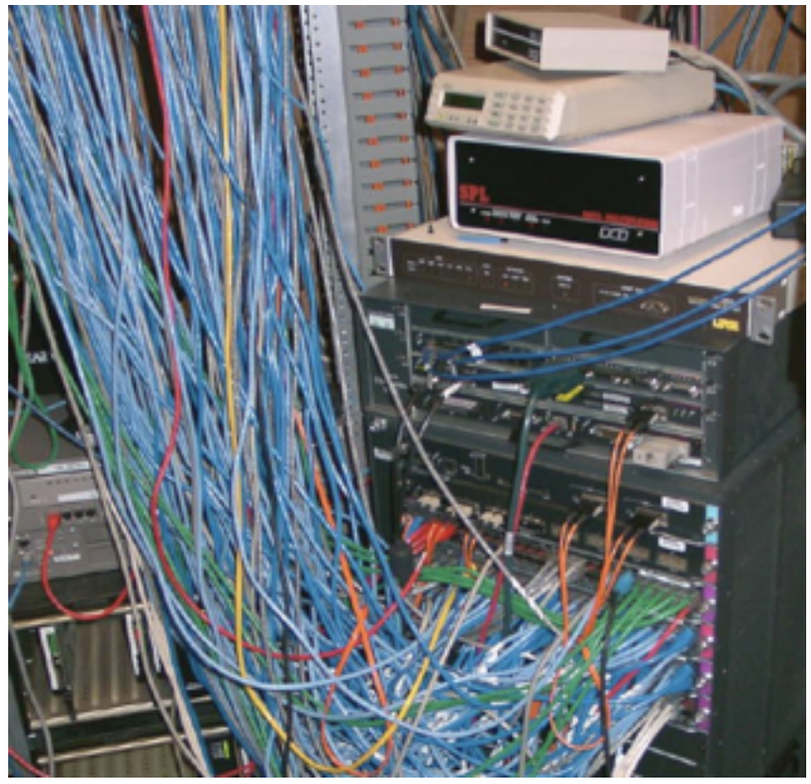
Summer School on Formal Methods and Networks
Cornell University, June 11, 2013

Data Plane Verification

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

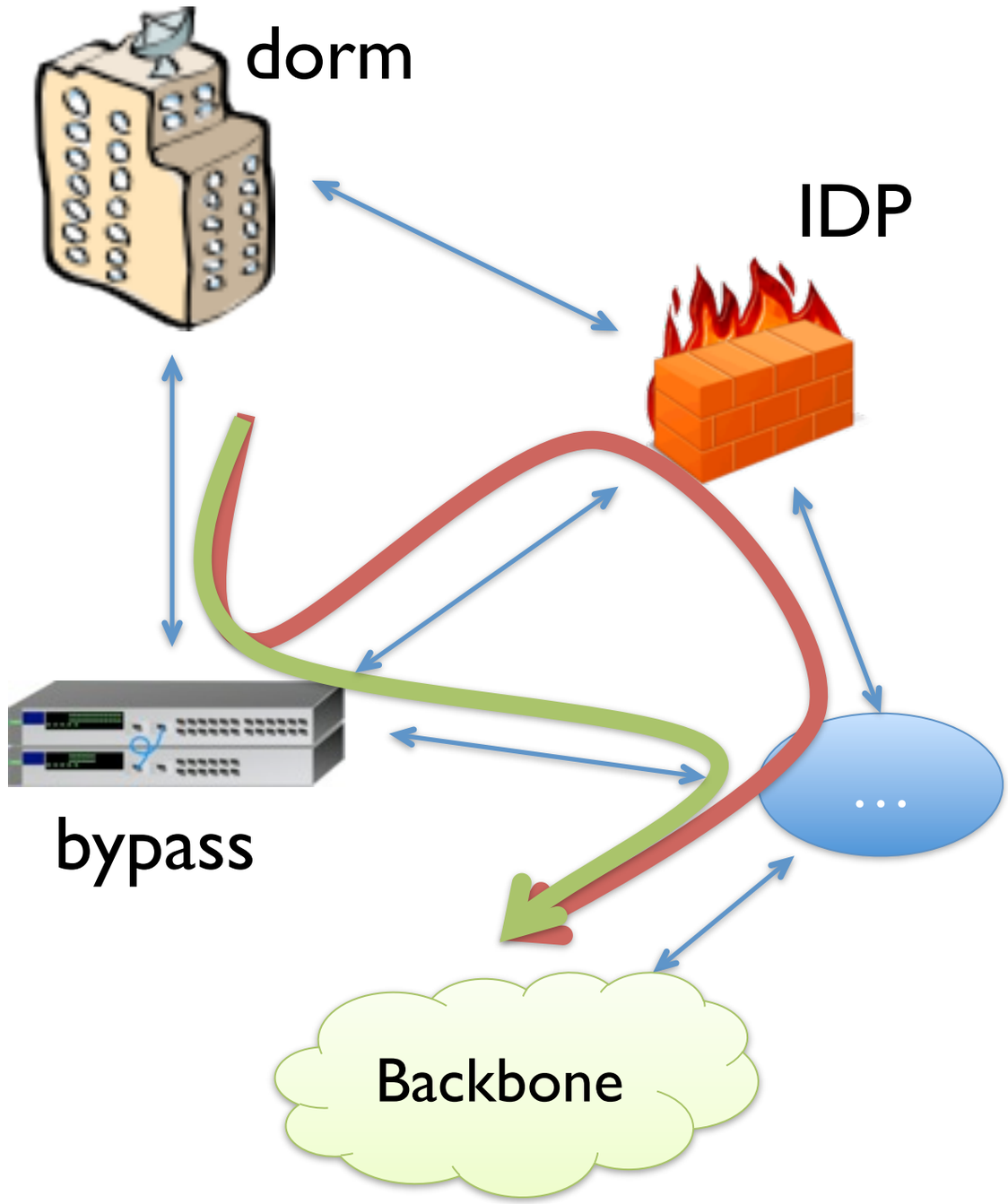
A real example from UIUC

Previously, an intrusion detection and prevention (IDP) device inspected all traffic to/from dorms

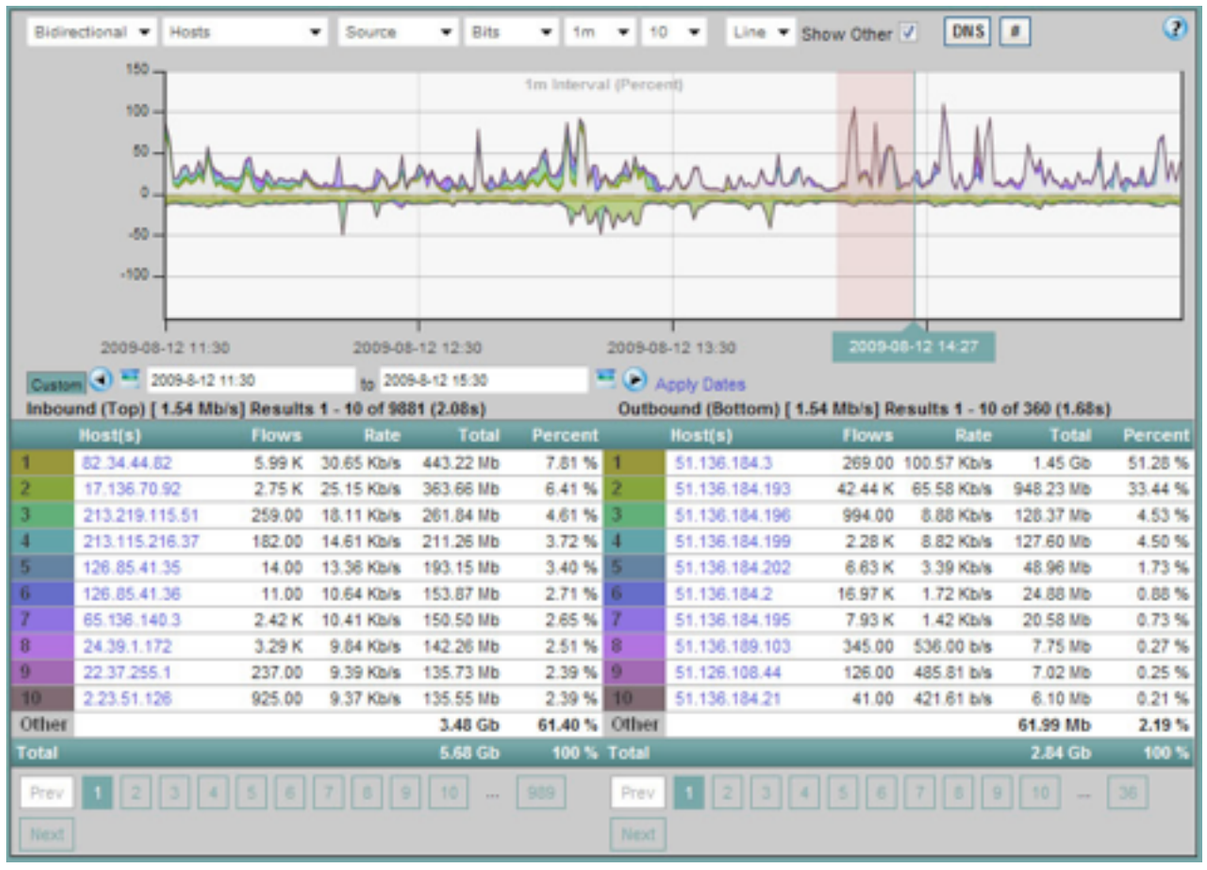
IDP couldn't handle load; added bypass

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

How do you know if it worked?



Understanding your network



	Host(s)	Flows	Rate	Total	Percent
1	82.34.44.82	5.99 K	30.65 Kb/s	443.22 Mb	7.81 %
2	17.136.70.92	2.75 K	25.15 Kb/s	363.66 Mb	6.41 %
3	213.219.115.51	259.00	18.11 Kb/s	261.84 Mb	4.61 %
4	213.115.216.37	182.00	14.61 Kb/s	211.26 Mb	3.72 %
5	126.85.41.35	14.00	13.36 Kb/s	193.15 Mb	3.40 %
6	126.85.41.36	11.00	10.64 Kb/s	153.87 Mb	2.71 %
7	65.136.140.3	2.42 K	10.41 Kb/s	150.50 Mb	2.65 %
8	24.39.1.172	3.29 K	9.84 Kb/s	142.26 Mb	2.51 %
9	22.37.255.1	237.00	9.39 Kb/s	135.73 Mb	2.39 %
10	2.23.51.126	925.00	9.37 Kb/s	135.55 Mb	2.39 %
Other				3.48 Gb	61.40 %
Total				5.68 Gb	100 %

	Host(s)	Flows	Rate	Total	Percent
1	51.136.184.3	269.00	100.57 Kb/s	1.45 Gb	51.28 %
2	51.136.184.193	42.44 K	65.58 Kb/s	948.23 Mb	33.44 %
3	51.136.184.196	994.00	8.88 Kb/s	128.37 Mb	4.53 %
4	51.136.184.199	2.28 K	8.82 Kb/s	127.60 Mb	4.50 %
5	51.136.184.202	6.63 K	3.39 Kb/s	48.96 Mb	1.73 %
6	51.136.184.2	16.97 K	1.72 Kb/s	24.88 Mb	0.88 %
7	51.136.184.195	7.93 K	1.42 Kb/s	20.58 Mb	0.73 %
8	51.136.189.103	345.00	536.00 b/s	7.75 Mb	0.27 %
9	51.126.108.44	126.00	485.81 b/s	7.02 Mb	0.25 %
10	51.136.184.21	41.00	421.61 b/s	6.10 Mb	0.21 %
Other				61.99 Mb	2.19 %
Total				2.84 Gb	100 %

Flow monitoring

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

```
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
!
```

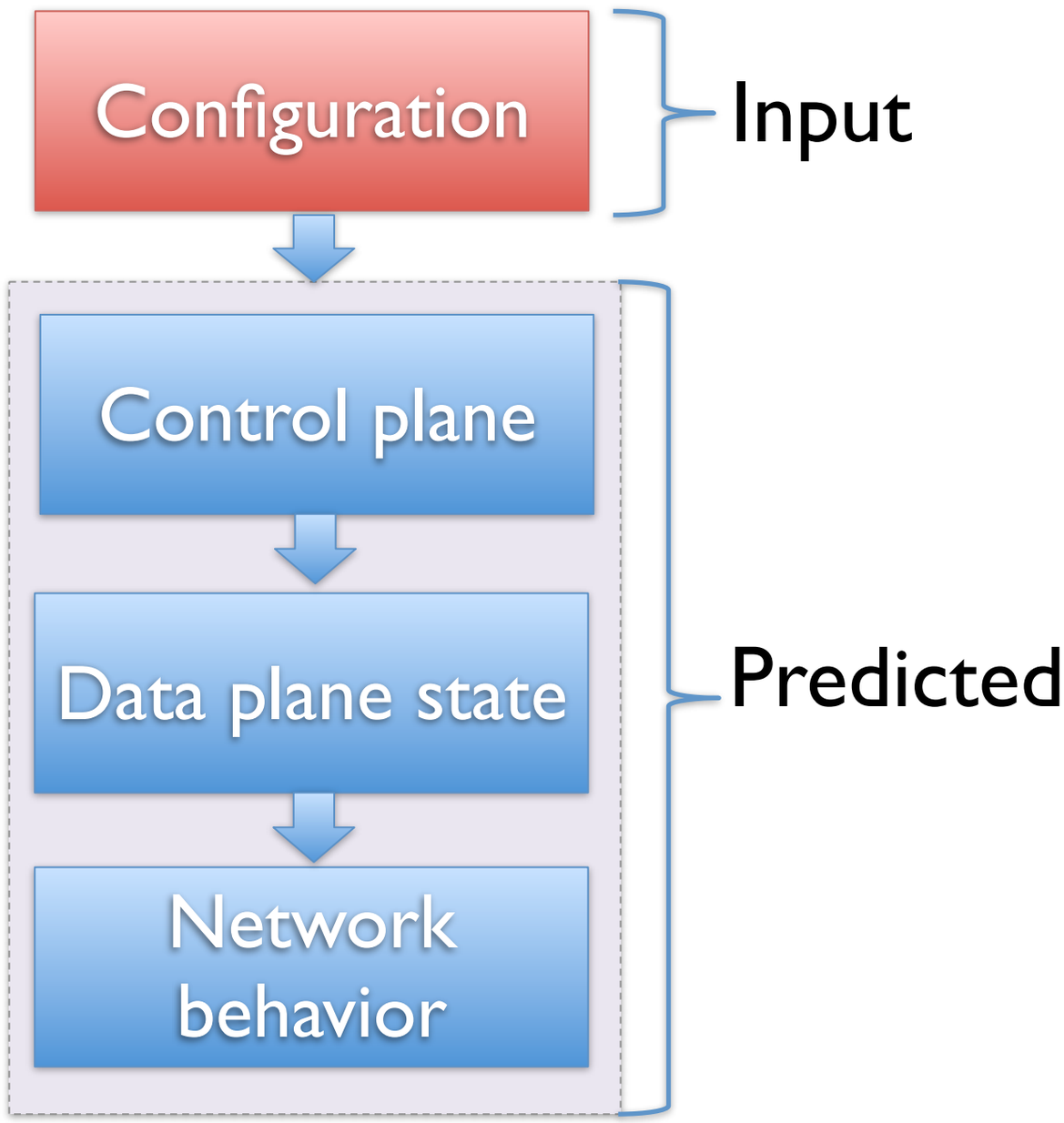
Configuration verification

Past approach: Config. verification

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

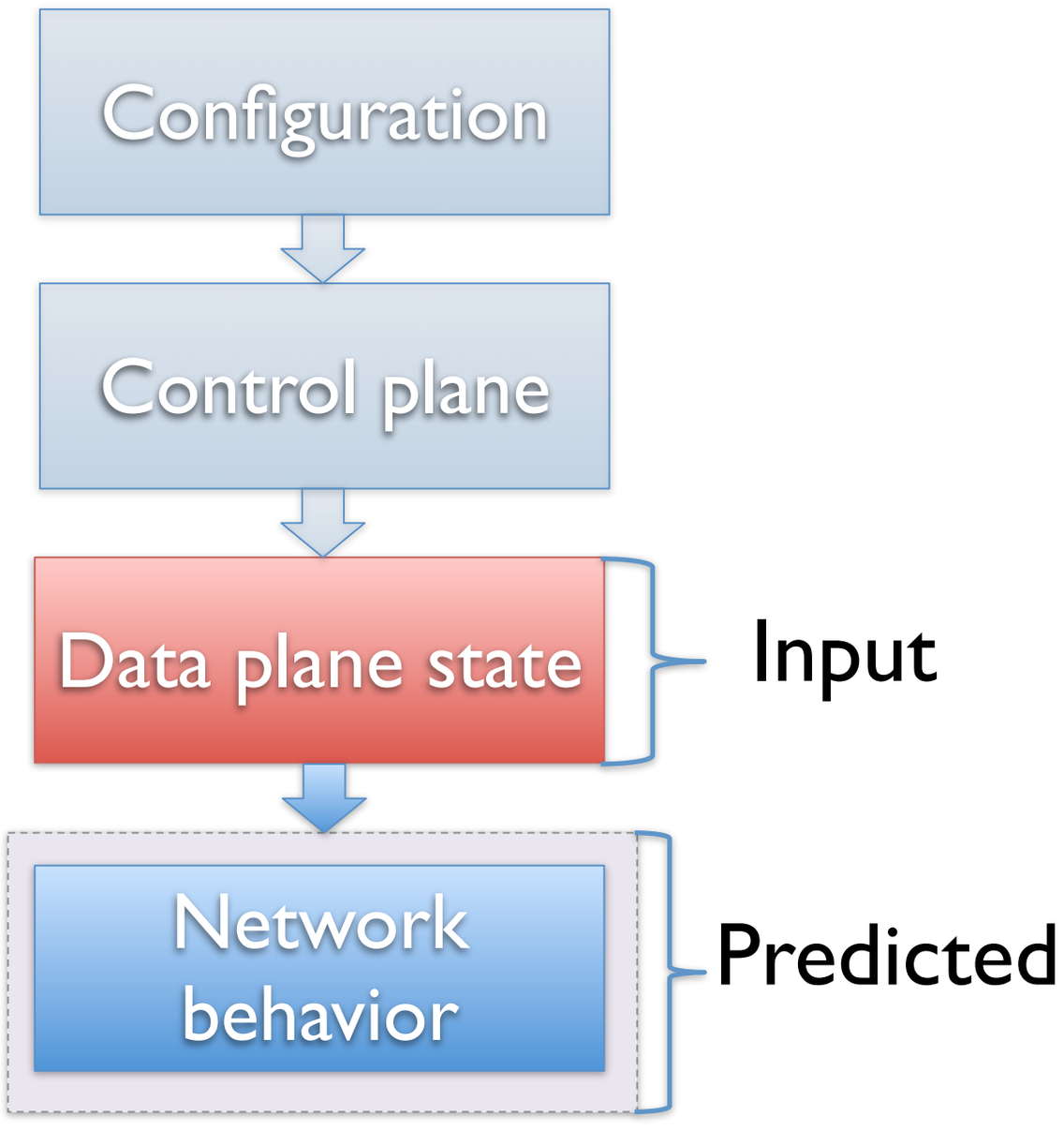
Margrave for firewalls [Nelson, Barratt, Dougherty, Fisler, Krishnamurthi, LISA'10]

UCLA+MSR [in progress...]



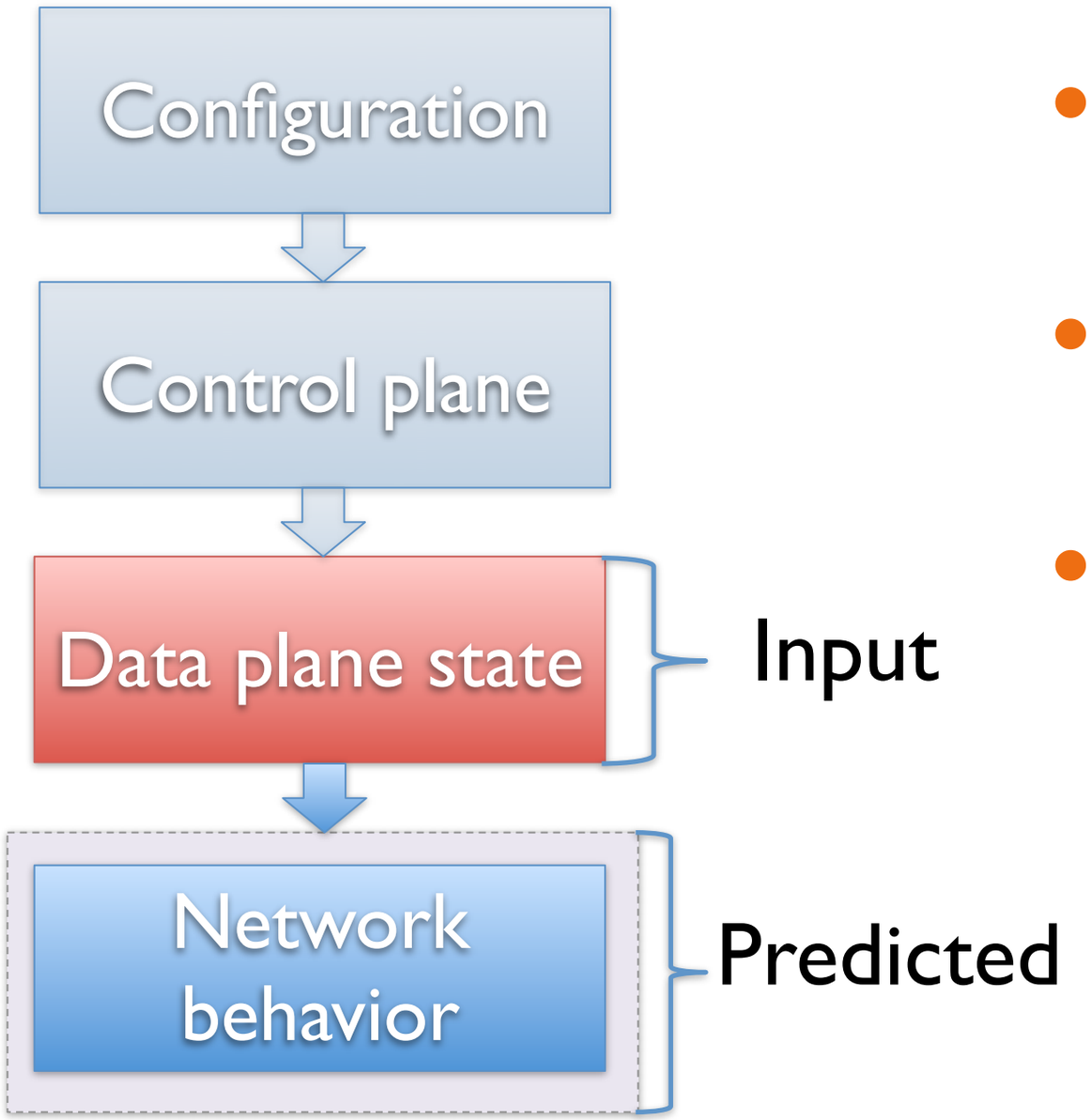
Data plane verification

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



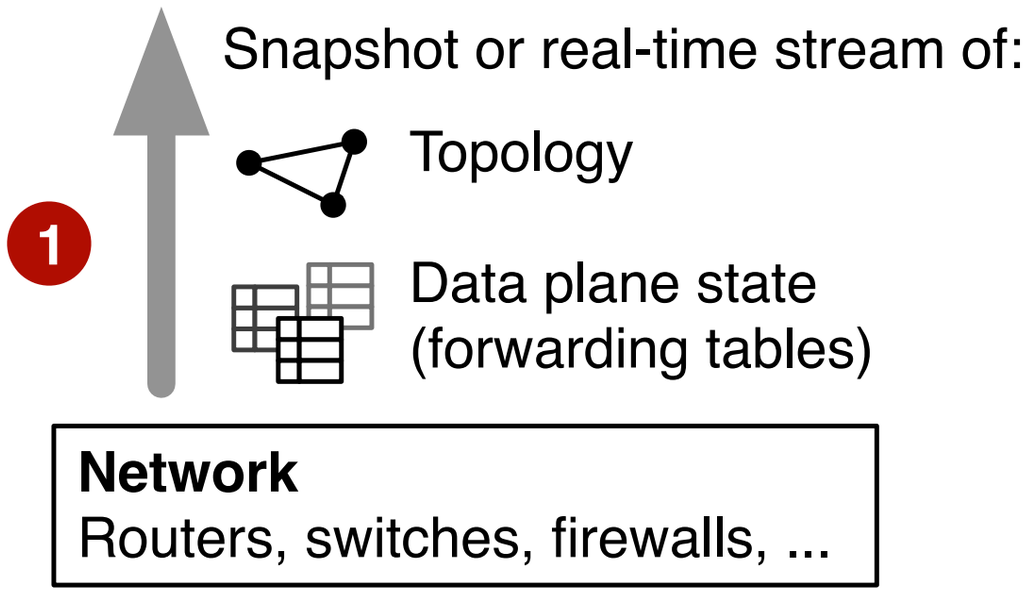
Data plane verification

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

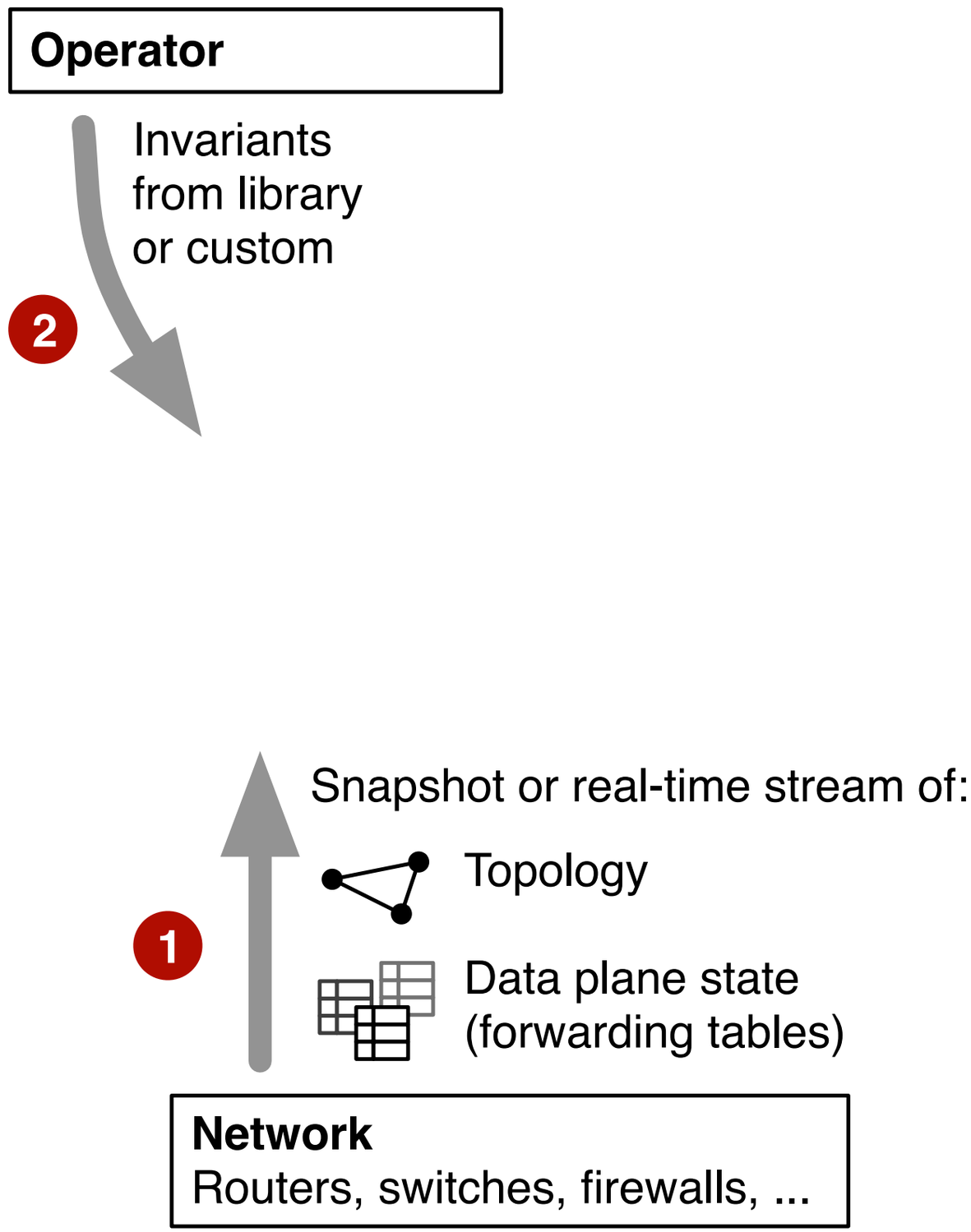


- Simpler, unified analysis across control protocols
- Catch bugs in control software
- Checks current snapshot

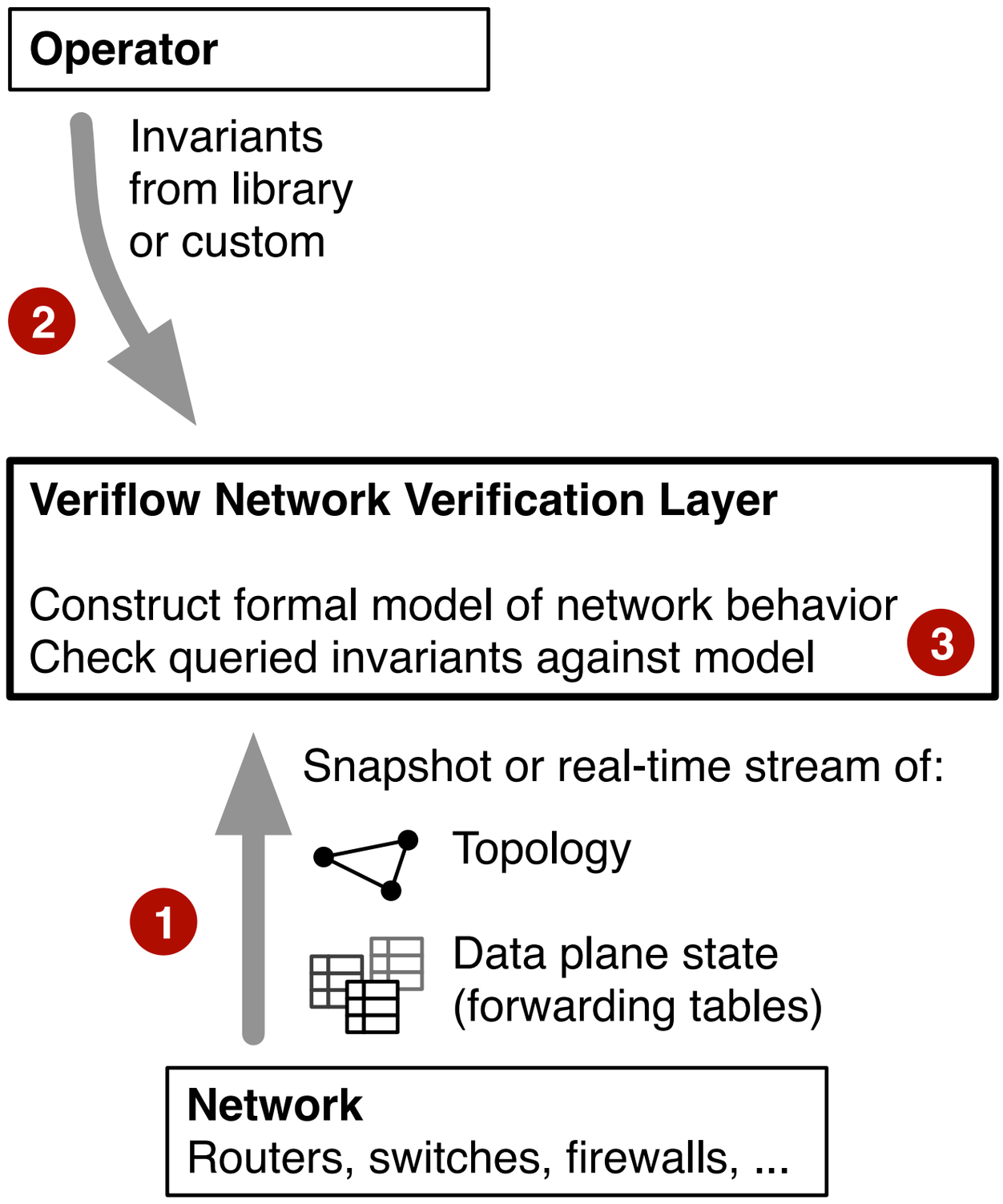
Architecture overview



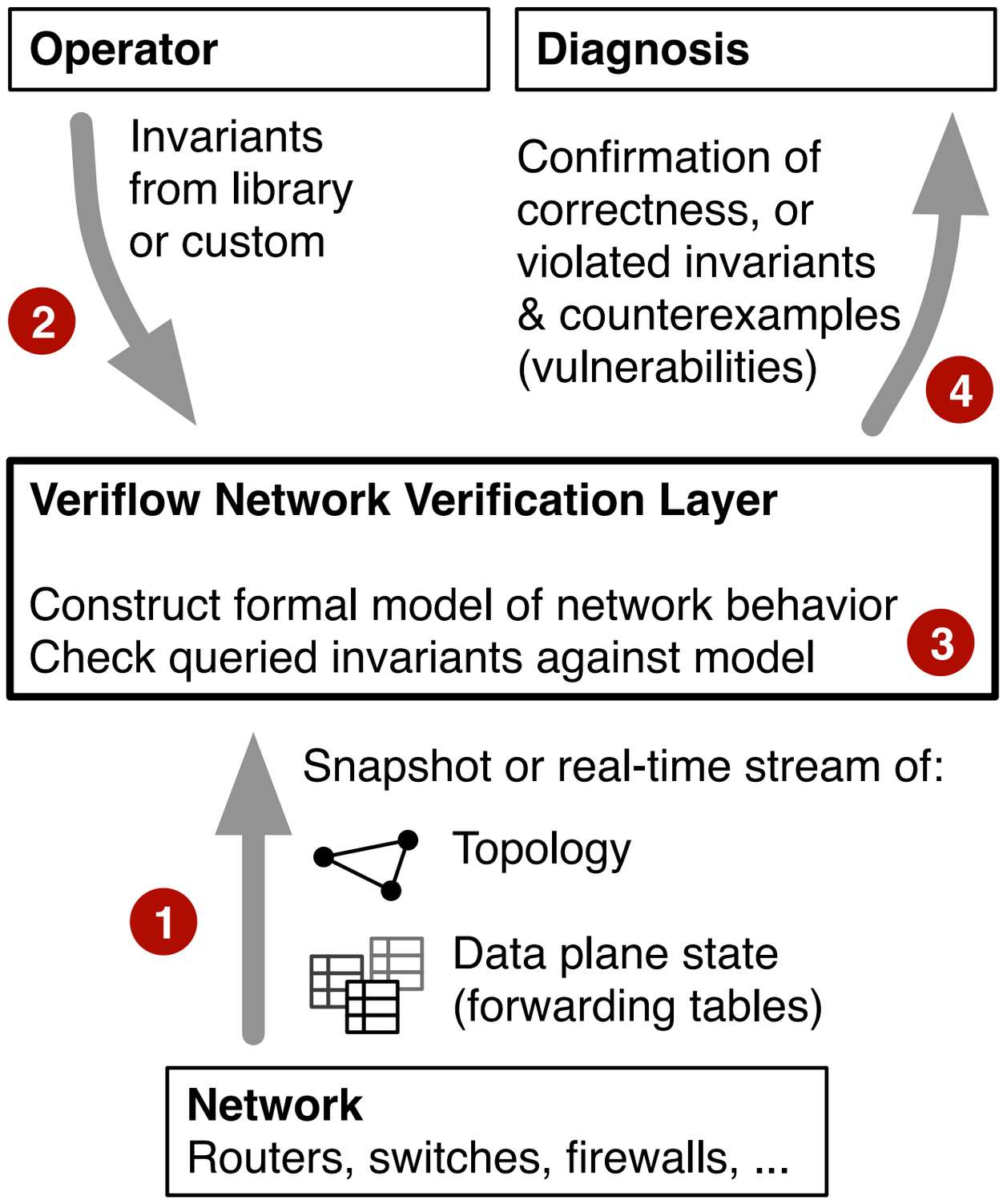
Architecture overview



Architecture overview



Architecture overview



Control software bugs

78 bugs sampled randomly from Bugzilla repository of Quagga (open source software router)

67 could cause data plane effect

- Under heavy load, Quagga 0.96.5 fails to update Linux kernel's routing tables
- In Quagga 0.99.5, a BGP session could remain active after it has been shut down

11 would not affect data plane

- Mgmt. terminal hangs in Quagga 0.96.4 on "show ip bgp"

Q: Where does SDN fit in?

Unified data plane interface

- Helpful, but not absolutely necessary

Centralized control of network

- Critical for real time verification

Our Two Tools

Anteater

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

Veriflow

- [Khurshid, Zhou, Caesar, Godfrey, HotSDN 2012]
- [Khurshid, Zou, Zhou, Caesar, Godfrey, NSDI 2013]
- Online real-time verification of data plane
- Interoperates with OpenFlow controller

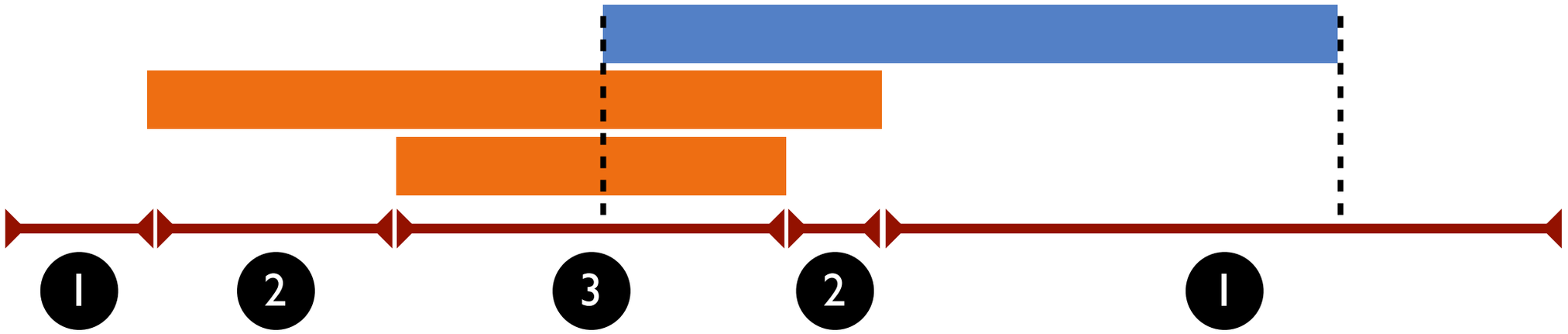
Anteater

Modeling the network is nontrivial

What if only longest prefix match rules on one field?

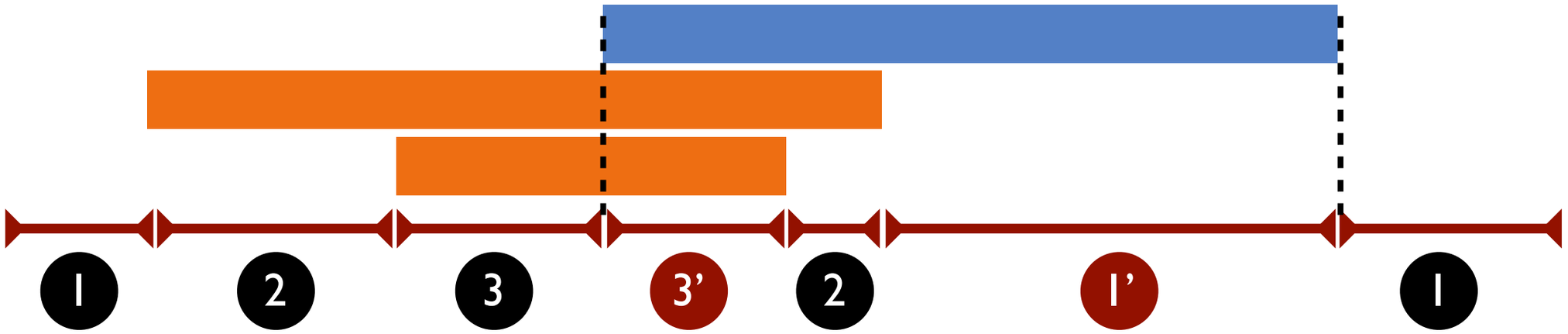
Modeling the network is nontrivial

What if only longest prefix match rules on one field?



Modeling the network is nontrivial

What if only longest prefix match rules on one field?



equivalence classes ≤ 2 • #rules

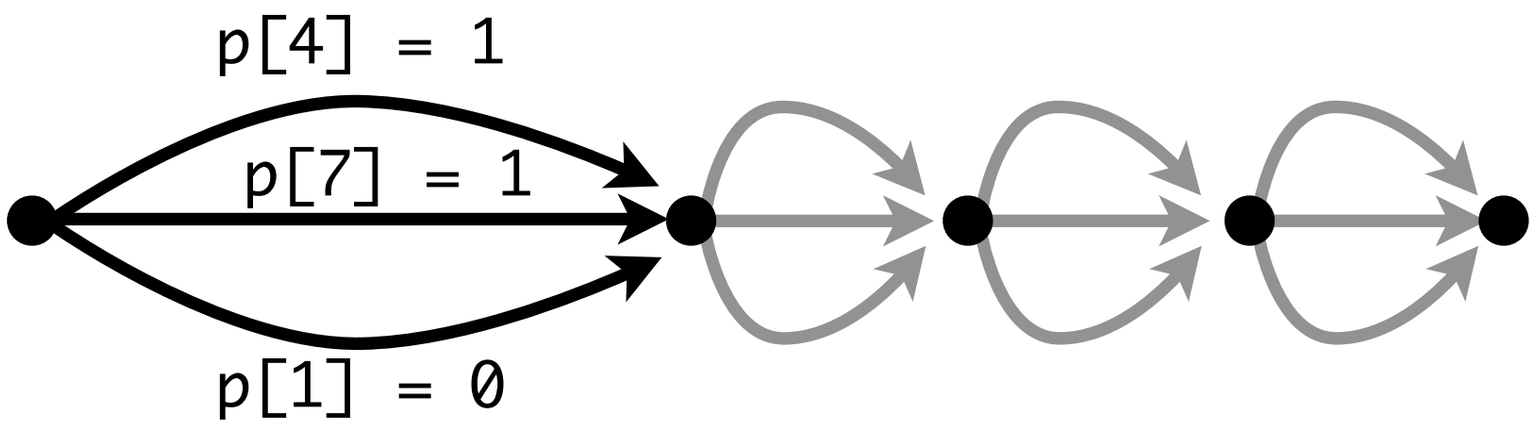
Modeling the network is nontrivial

What if only longest prefix match rules on one field?

- easy: reachability is polynomial time

Add one-bit packet filters: `"if p[43] = 0 then drop"`

- reachability is NP-complete



$$(x_4 \vee x_7 \vee \bar{x_1}) \wedge (\ldots) \wedge (\ldots) \wedge (\ldots)$$

Modeling the network is nontrivial

What if only longest prefix match rules on one field?

- easy: reachability is polynomial time

Add one-bit packet filters: `"if p[43] = 0 then drop"`

- reachability is NP-complete

Add packet header transformations...

- even harder (depends on assumptions, e.g. packet header length bound)

Anteater's solution

Express data plane and invariants as SAT

- ...up to some max # hops

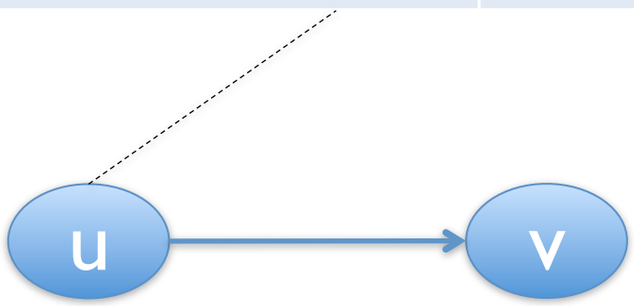
Check with off-the-shelf SAT solver (Boolector)

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)

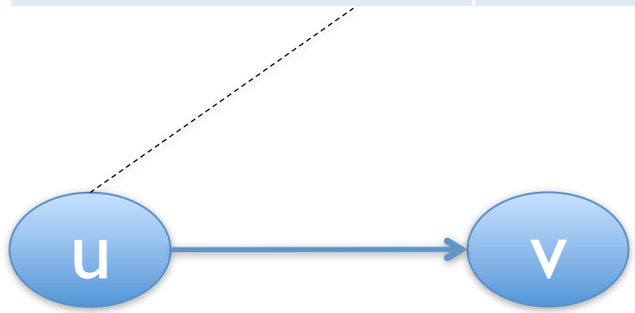
Destination	Iface
10.1.1.0/24	v



P(u, v) = dst_ip ∈10.1.1.0/24

Simpler example

Destination	Iface
0.0.0.0/0	v

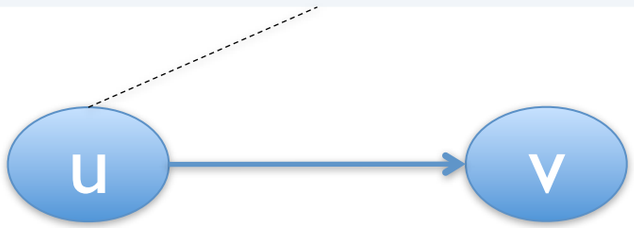


P(u, v) = true

Default routing

Some more examples

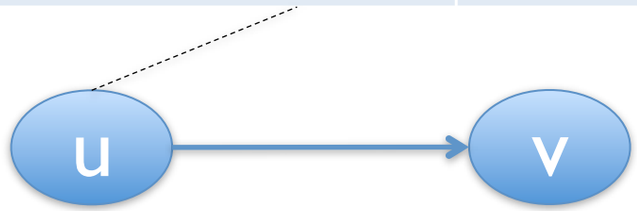
Destination	Iface
10.1.1.0/24	v
Drop port 80 to v	



$P(u, v) = dst\_ip \in 10.1.1.0/24 \wedge dst\_port \neq 80$

Packet filtering

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



$P(u, v) = (dst\_ip \in 10.1.1.0/24 \wedge dst\_ip \notin 10.1.1.128/25) \vee dst\_ip \in 10.1.2.0/24$

Longest prefix matching

Reachability as SAT solving

Goal: reachability from u to w

C = (P(u, v) ∧ P(v,w)) is satisfiable

⇔∃A packet that makes P(u,v) ∧ P(v,w) true

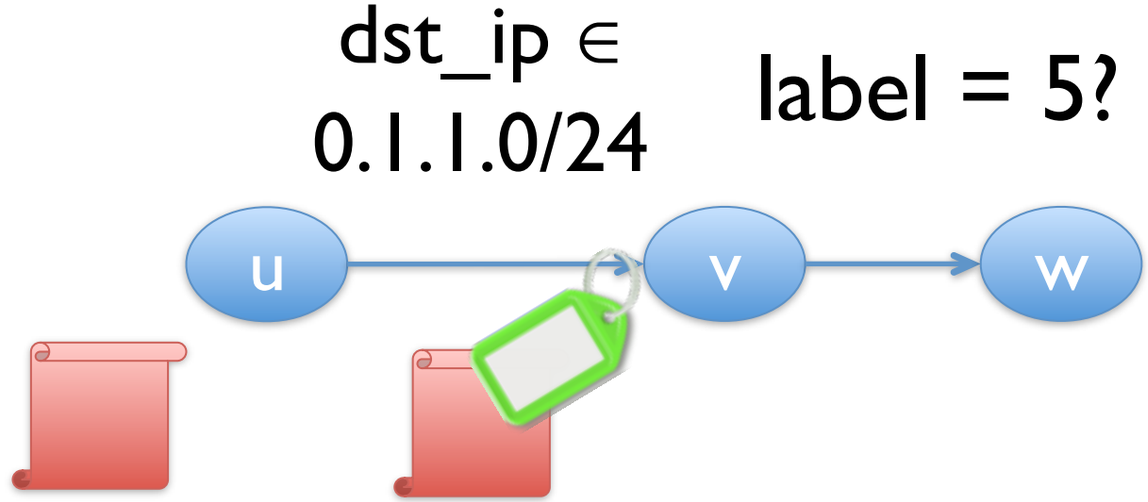
⇔∃A packet that can flow over (u, v) and (v,w)

⇔ u can reach w

- SAT solver determines the satisfiability of C
- Problem: exponentially many paths
  - Solution: Dynamic programming (a.k.a. loop unrolling)
  - Intermediate variables: "Can reach *x* in *k* hops?"
  - Similar to [Xie, Zhan, Maltz, Zhang, Greenberg, Hjalmtysson, Rexford, INFOCOM'05]

Packet transformation

Essential to model
MPLS, QoS, NAT, etc.



- Model the history of packets: vector over time
- Packet transformation ⇒ boolean constraints

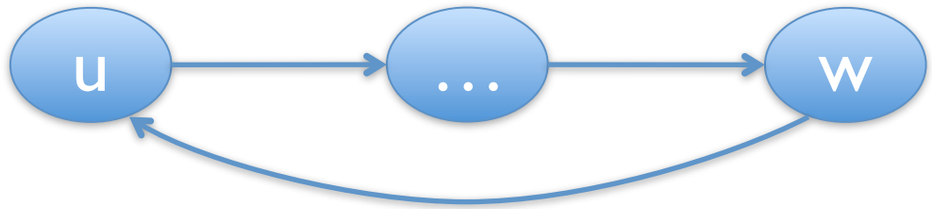
over adjacent packet versions

$$(p_i.dst\_ip \in 0.1.1.0/24) \wedge (p_{i+1}.label = 5)$$

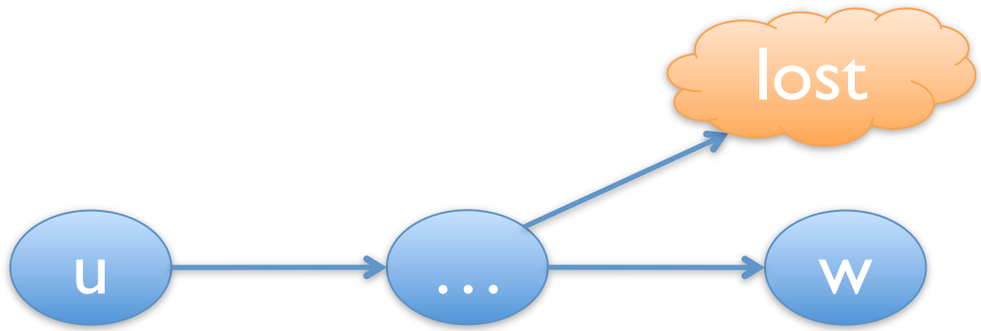
More generally: $p_{i+1} = f(p_i)$

Invariants

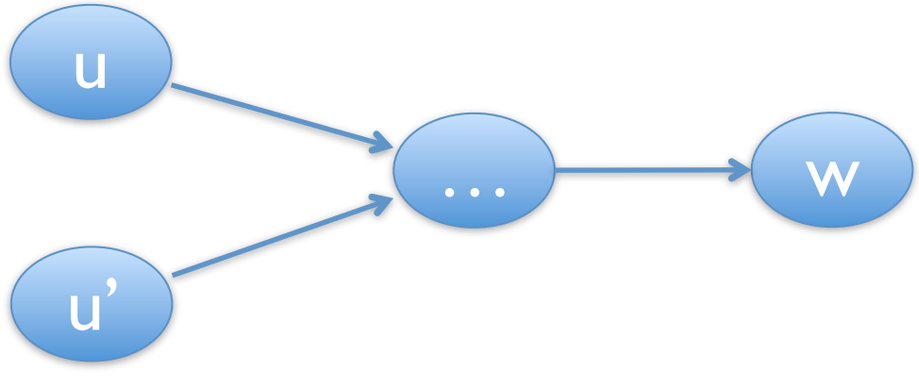
Loop detection



Packet loss (black holes)



Consistency



Experience with the UIUC Network

Experiences with UIUC network

Evaluated Anteater with UIUC campus network

- ~178 routers supporting >70,000 machines
- Predominantly OSPF, also uses BGP and static routing
- 1,627 FIB entries per router (mean)
- State collected using operator's SNMP scripts

Revealed 23 bugs with 3 invariants in 2 hours

	Loop	Packet loss	Consistency
Being fixed	9	0	0
Stale config.	0	13	1
False pos.	0	4	1
Total alerts	9	17	2

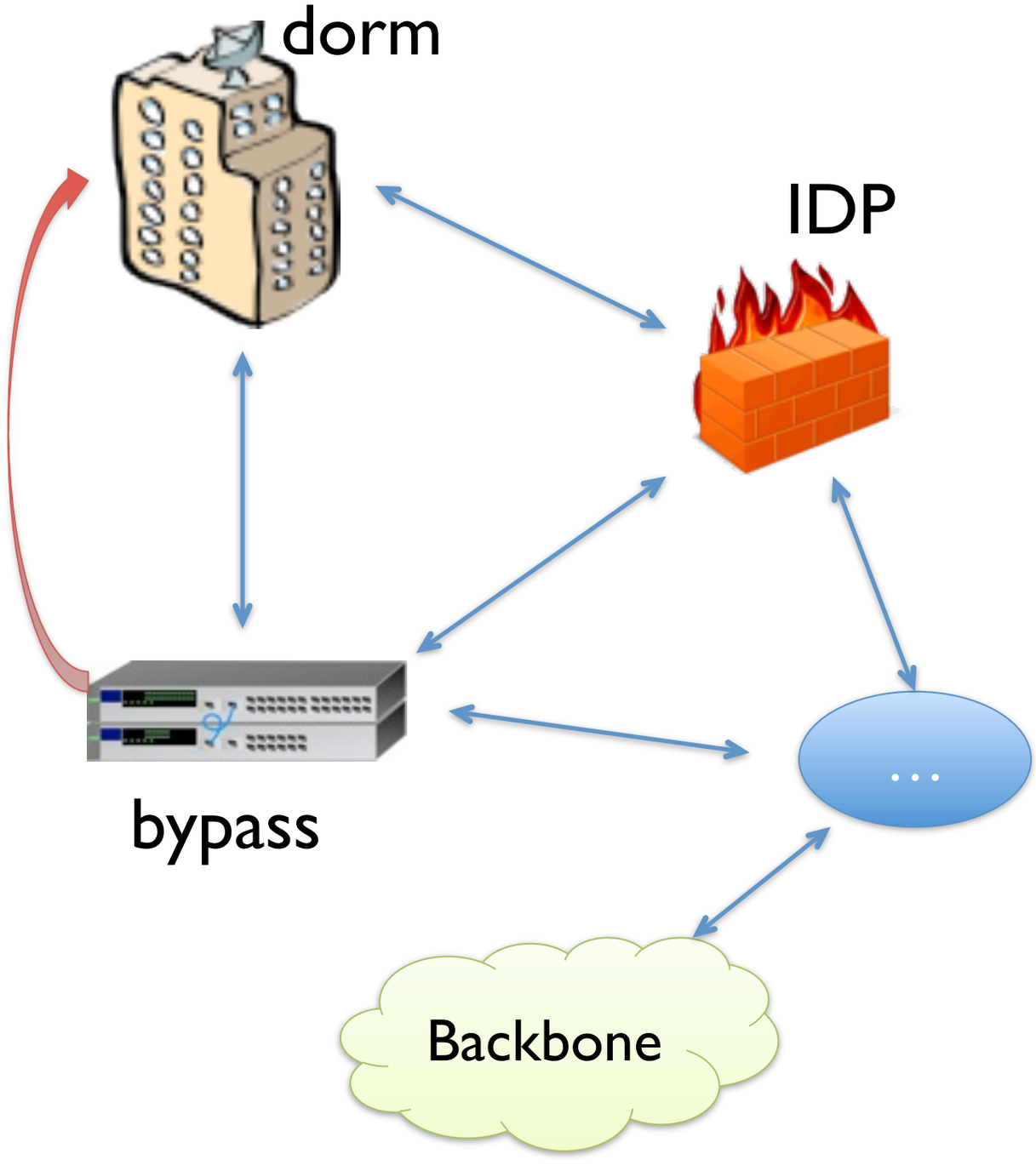
Forwarding loops

IDP was overloaded, operator introduced bypass

- IDP only inspected traffic for campus

bypass routed campus traffic to IDP through static routes

Introduced 9 loops



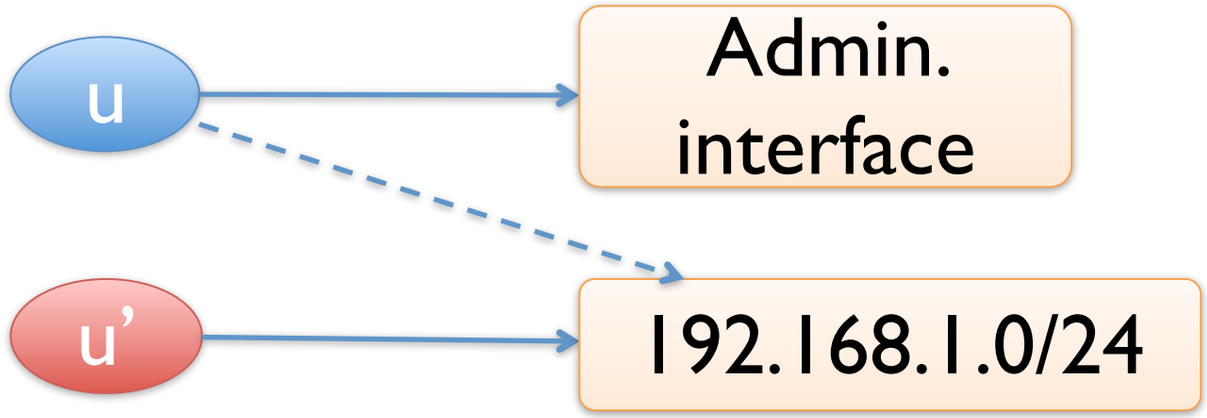
Bugs found by other invariants

Packet loss

- Blocking compromised machines at IP level
- Stale configuration

  From Sep, 2008

Consistency



- One router exposed web admin interface in FIB
- Different policy on private IP address range

Refs: Offline Data Plane Verification

Static reachability in IP networks [Bush et al'03, Xie et al'05]

FlowChecker [Al-Shaer, Al-Haj, SafeConfig '10]

ConfigChecker [Al-Shaer, Al-Saleh, SafeConfig '11]

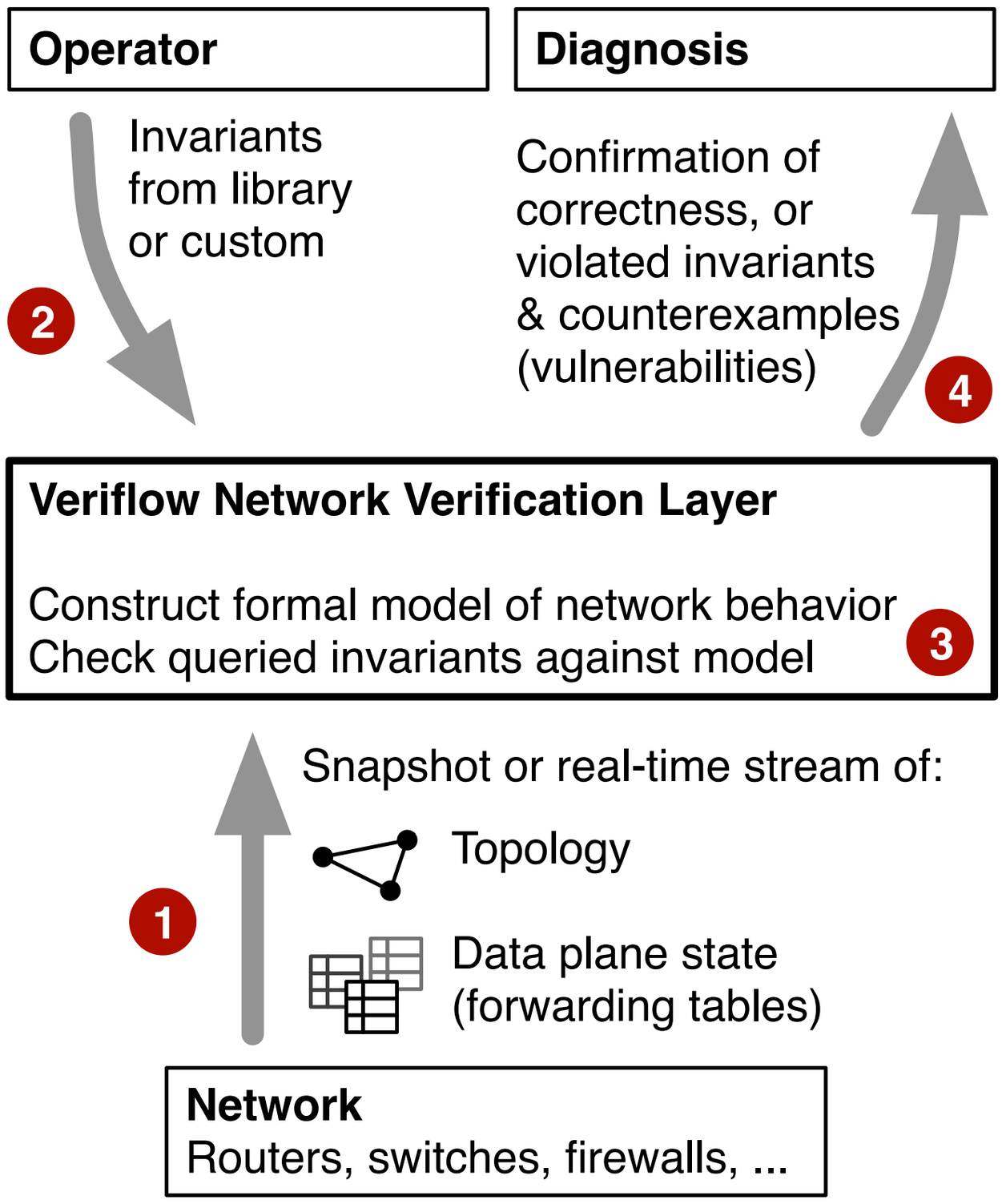
Anteater [SIGCOMM'11] http://code.google.com/p/anteater

Header Space Analysis [Kazemian, Varghese, McKeown, NSDI '12]

Abstractions for Network Update [Reitblatt, Foster, Rexford, Schlesinger, Walker, SIGCOMM'12]

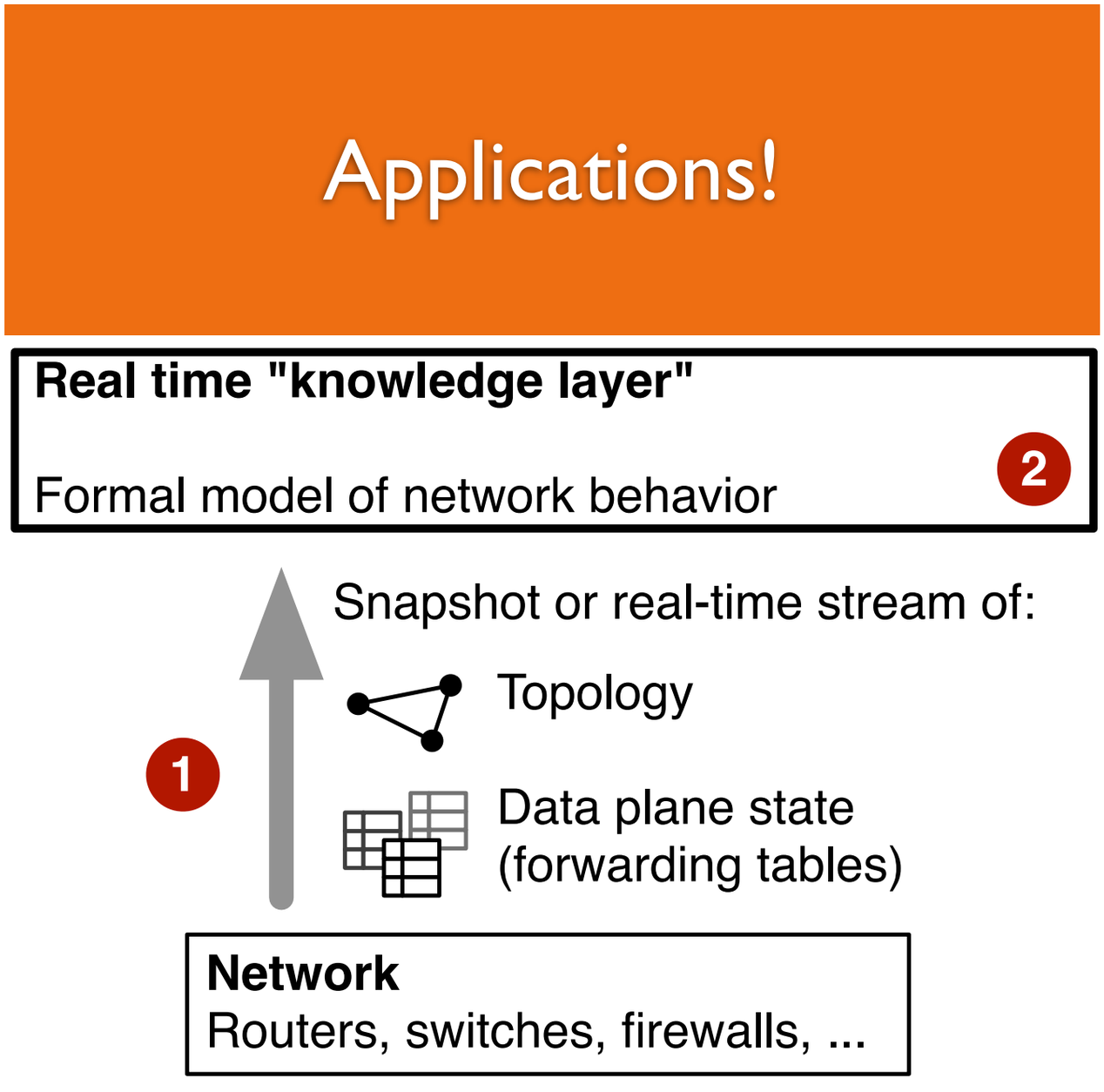
Verification of Computer Switching Networks: An Overview [Shuyun Zhang, Sharad Malik, Rick McGeer]

Looking ahead: An Opportunity



Looking ahead: An Opportunity

Applications!



Data Plane Verification Discussion

1. Expressing policies can be hard. How can we make network verification easy for operators?

2. What apps can we build on top of a real-time understanding of network's behavior?

3. Can DPV be extended to stateful networks?

4. How should DPV connect with policy generation?

5. Can formal methods dramatically improve network reliability?

Email to: pbg@illinois.edu