Dependable Networks: Trends and Challenges

Stefan Schmid (TU Berlin & Sabbatical at Hebrew University)

INET @ TU Berlin

We aim at the investigation of future communication **networks** and future applications offered through these networks:

- **Algorithms** and mechanisms to design and operate communication networks
- Network **architectures** and **protocols** for future communication technologies
- **Performance** evaluation of networked and distributed systems
- Network **security**
- **Wireless** and cellular networks

Our vision is that networked systems should become ***self-**** (i.e., self-optimizing, self-repairing, self-configuring).

Accordingly, we are currently particularly interested in ***automated*** and ***data-driven*** approaches to design, optimize, and verify networked systems.

INET @ TU Berlin

We aim at the investigation of future communication **networks** and future applications offered through these networks:

- **Algorithms** and mechanisms to design and operate communication networks
- Network **architectures** and **protocols** for future communication technologies
- **Performance** evaluation of networked and distributed systems
- Network **security**
- **Wireless** and cellular networks

Our vision is that networked systems should become ***self-**** (i.e., self-optimizing, self-repairing, self-configuring).

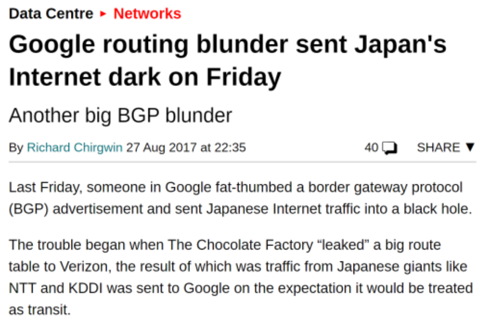
Accordingly, we are currently particularly interested in ***automated*** and ***data-driven*** approaches to design, optimize, and verify networked systems.

But why?? Networks are working well today!
Internet is huge success, handled all trends!

A Major Issue: Complexity

Many outages due to **misconfigurations** and **human errors**.

Entire countries disconnected…



Data Centre ▸ Networks

Google routing blunder sent Japan's Internet dark on Friday

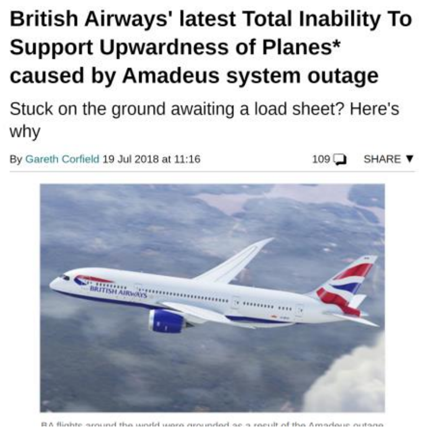
Another big BGP blunder

By Richard Chirgwin 27 Aug 2017 at 22:35 40 SHARE ▼

Last Friday, someone in Google fat-thumbed a border gateway protocol (BGP) advertisement and sent Japanese Internet traffic into a black hole.

The trouble began when The Chocolate Factory "leaked" a big route table to Verizon, the result of which was traffic from Japanese giants like NTT and KDDI was sent to Google on the expectation it would be treated as transit.

… 1000s passengers stranded…



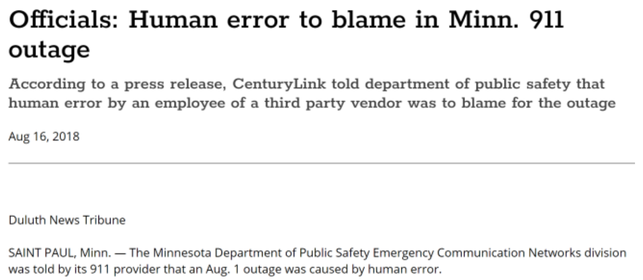
British Airways' latest Total Inability To Support Upwardness of Planes* caused by Amadeus system outage

Stuck on the ground awaiting a load sheet? Here's why

By Gareth Corfield 19 Jul 2018 at 11:16 109 SHARE ▼

BA flights around the world were grounded as a result of the Amadeus outage

… even 911 services affected!



Officials: Human error to blame in Minn. 911 outage

According to a press release, CenturyLink told department of public safety that human error by an employee of a third party vendor was to blame for the outage

Aug 16, 2018

Duluth News Tribune

SAINT PAUL, Minn. — The Minnesota Department of Public Safety Emergency Communication Networks division was told by its 911 provider that an Aug. 1 outage was caused by human error.

Slide credit: Laurent Vanbever

Even Tech-Savvy Companies Struggle to Provide Reliable Networks

We discovered a misconfiguration on this pair of switches that caused what's called a "bridge loop" in the network.

A network change was […] executed incorrectly […] more "stuck" volumes and added more requests to the re-mirroring storm

Service outage was due to a series of internal network events that corrupted router data tables

Experienced a network connectivity issue […] interrupted the airline's flight departures, airport processing and reservations systems

Slide credit: Nate Foster

Another Issue: *Lack of Tools*
Anecdote “Wall Street Bank”

- Outage of a data center of a Wall Street investment bank
- Lost revenue measured in USD 10^6 / min
- Quickly, an emergency team was assembled with experts in compute, storage and networking:
  - **The compute team:** soon came armed with **reams of logs**, showing how and when the applications failed, and had already written experiments to reproduce and **isolate the error**, along with candidate prototype programs to workaround the failure.
  - **The storage team:** similarly equipped, showing which file **system logs** were affected, and already progressing with **workaround programs**.
  - “All the **networking team** had were **two tools invented over 20y ago** to merely test end-to-end connectivity. Neither tool could reveal **problems with switches**, the **congestion** experienced by individual packets, or provide any means to create experiments to identify, quarantine and resolve the problem. Whether or not the problem was in the network, the **networking team would be blamed** since they were unable to demonstrate otherwise.”

Source: «The world’s fastest and most programmable networks»
White Paper Barefoot Networks

Roadmap

- Opportunity: emerging networking technologies
  - Automation and „self-driving networks“
  - Programmable networks for improved visibility

- Challenge: emerging network technologies
  - New threat models

It’s an ***exciting period***! New tools, simple abstractions, disburdening human operators, etc.

Roadmap

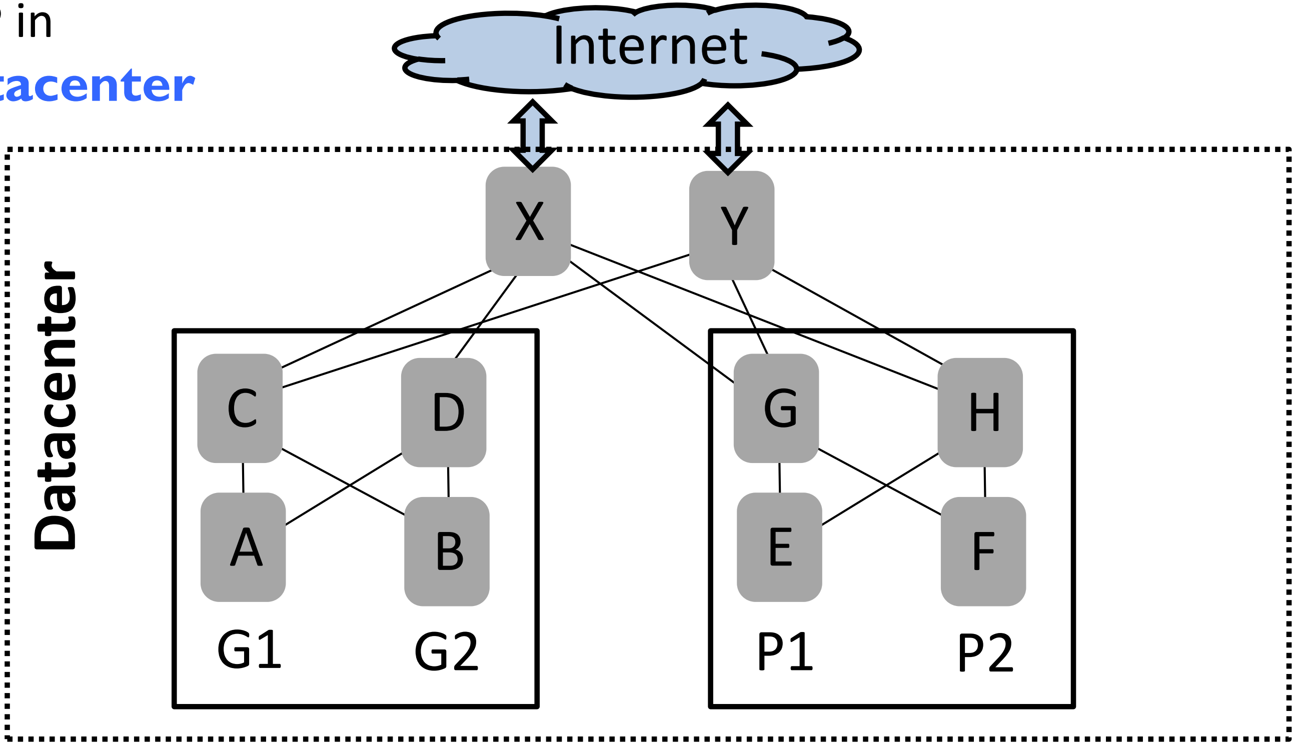
- Opportunity: emerging networking technologies
  - **Automation and „self-driving networks“**
  - Programmable networks for improved visibility

- Challenge: emerging network technologies
  - New threat models

It’s an ***exciting period***! New tools, simple abstractions, disburdening human operators, etc.

Why is it so complex? Example.

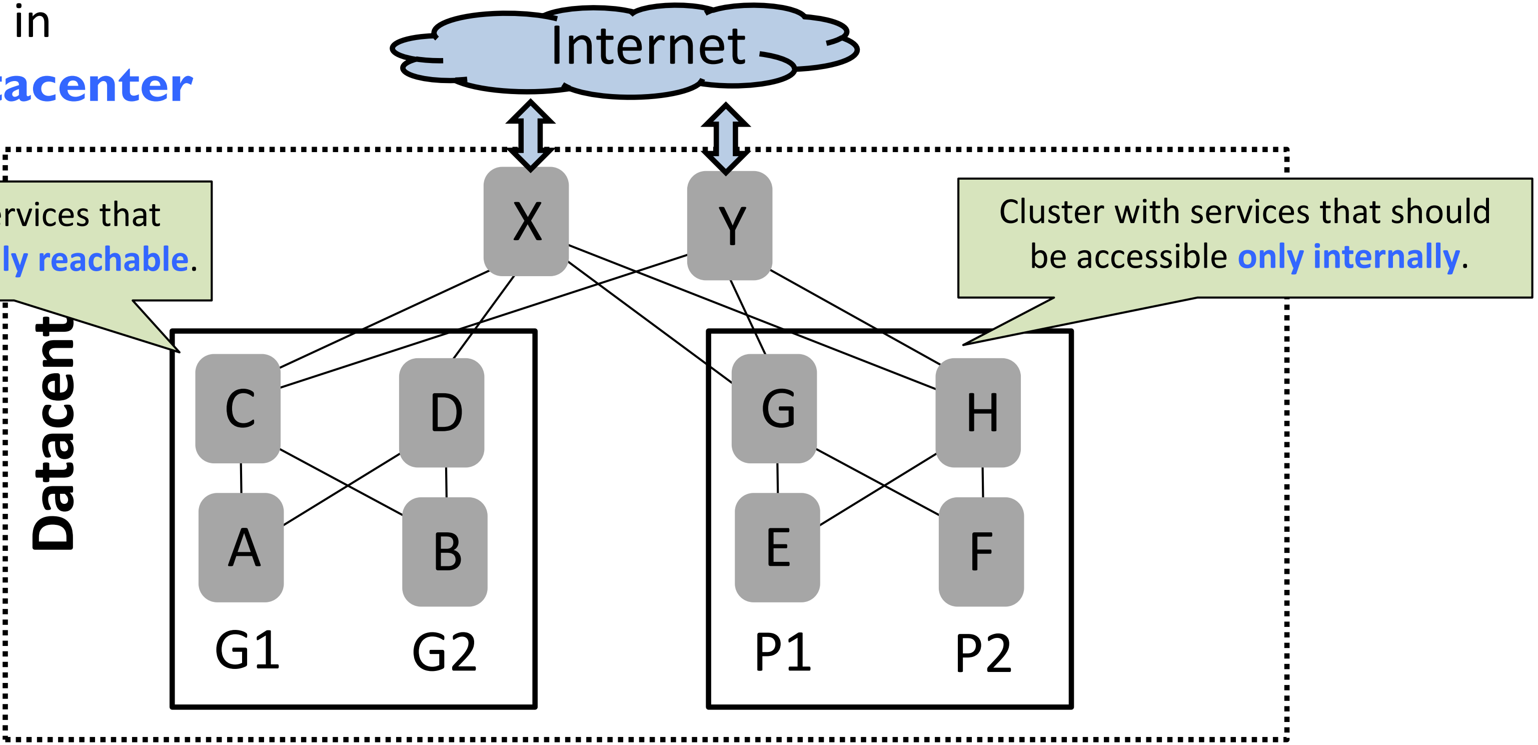
Example: BGP in Microsoft **datacenter**



Credits: Beckett et al. (SIGCOMM 2016): Bridging Network-wide Objectives and Device-level Configurations.

Why is it so complex? Example.

Example: BGP in Microsoft **datacenter**



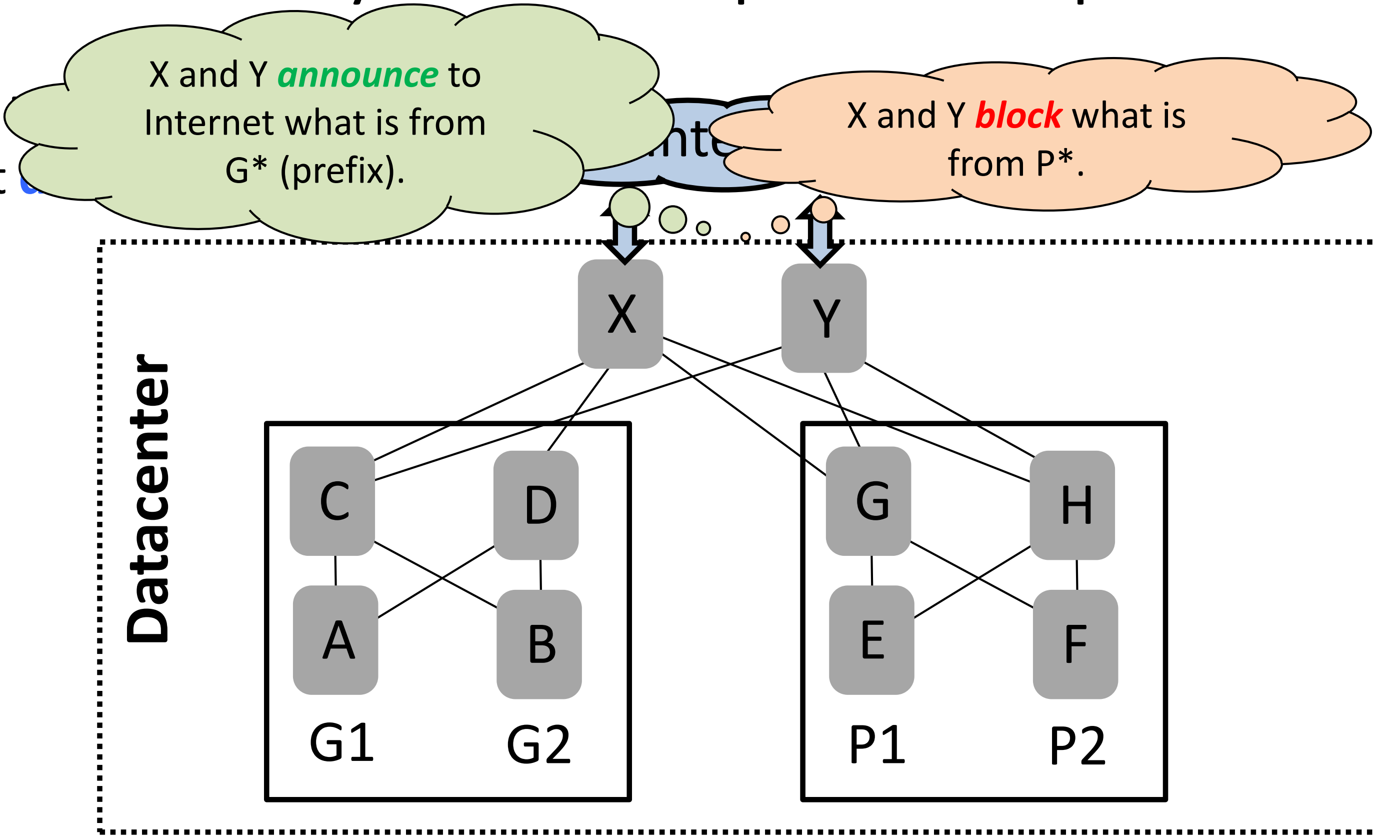
Cluster with services that should be **globally reachable**.

Cluster with services that should be accessible **only internally**.

Credits: Beckett et al. (SIGCOMM 2016): Bridging Network-wide Objectives and Device-level Configurations.

Why is it so complex? Example.

Example:
Microsoft

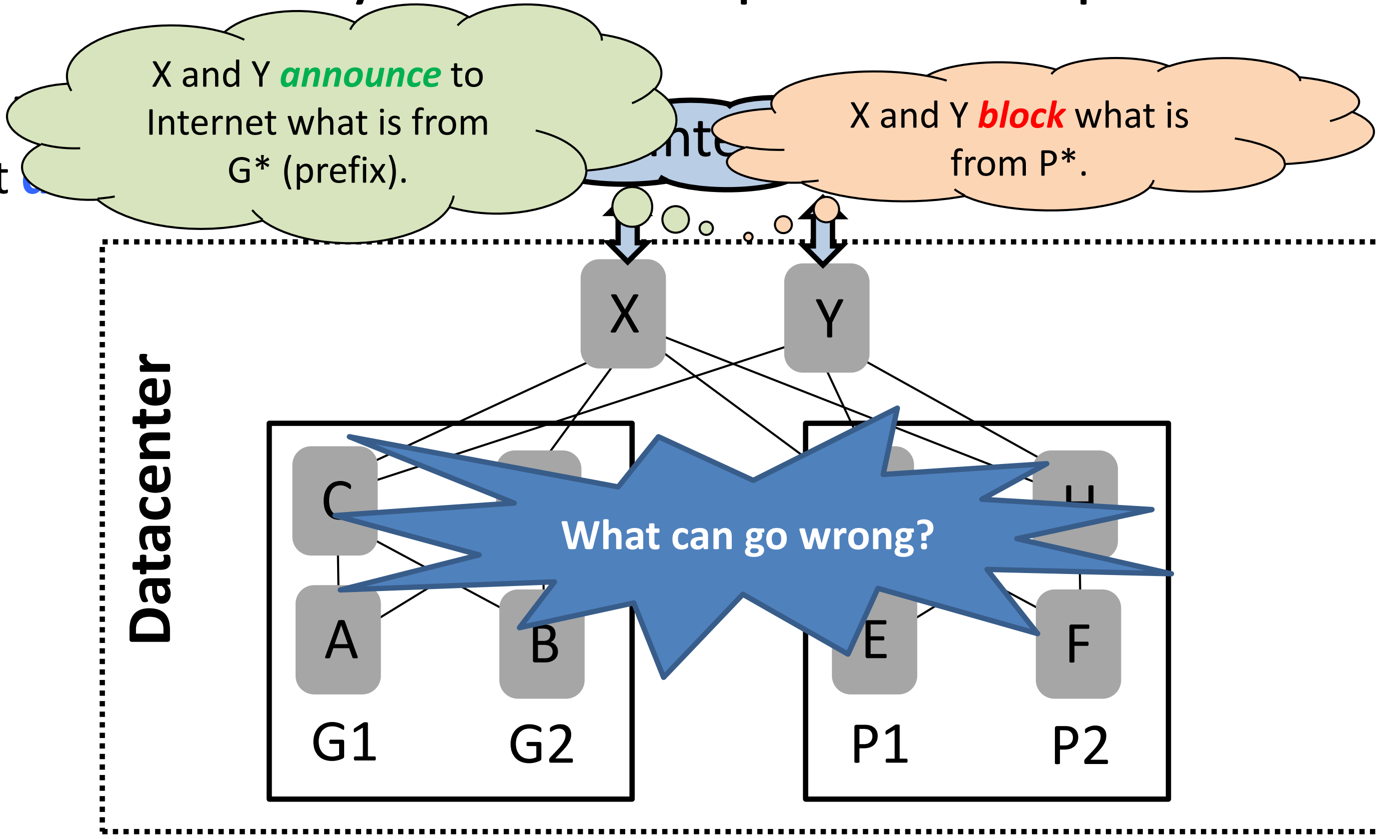

X and Y ***announce*** to Internet what is from G* (prefix).
X and Y ***block*** what is from P*.
Inte
Datacenter
X
Y
C
D
A
B
G
H
E
F
G1
G2
P1
P2

Credits: Beckett et al. (SIGCOMM 2016): Bridging Network-wide Objectives and Device-level Configurations.

Why is it so complex? Example.

Example:

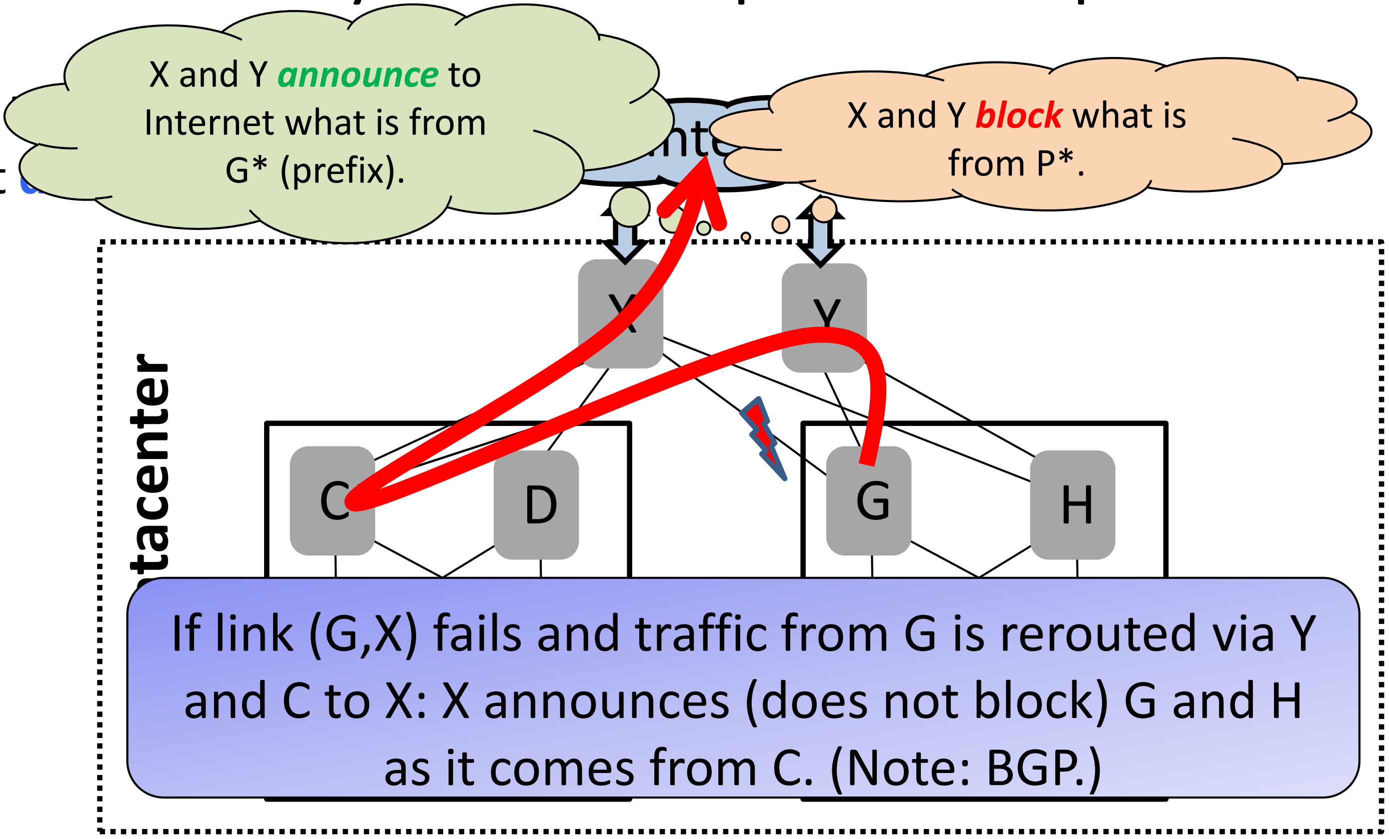
Microsoft


X and Y ***announce*** to Internet what is from G* (prefix).
X and Y ***block*** what is from P*.
Datacenter
X
Y
C
A
B
E
F
G1
G2
P1
P2
What can go wrong?

Credits: Beckett et al. (SIGCOMM 2016): Bridging Network-wide Objectives and Device-level Configurations.

Why is it so complex? Example.

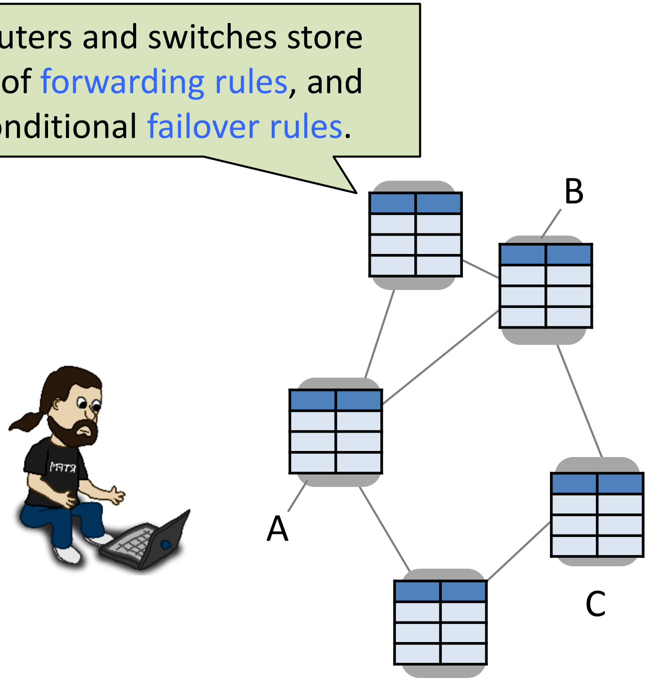
Example:
Microsoft


X and Y *announce* to Internet what is from G* (prefix).
X and Y *block* what is from P*.
Inte
tacenter
X
Y
C
D
G
H

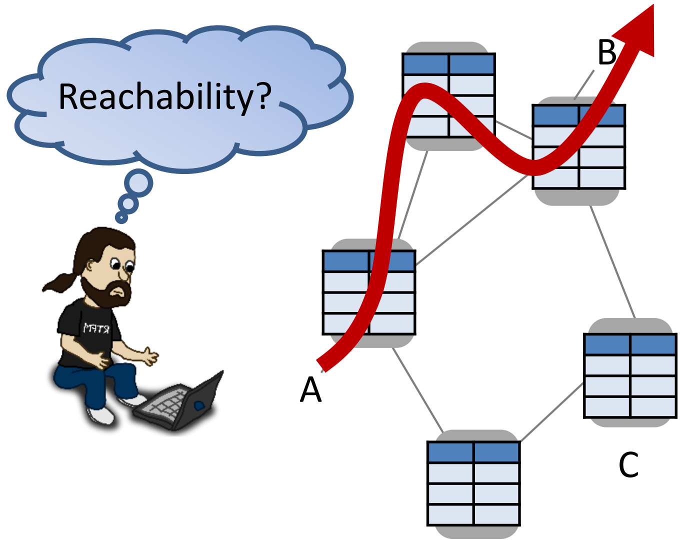
If link (G,X) fails and traffic from G is rerouted via Y and C to X: X announces (does not block) G and H as it comes from C. (Note: BGP.)

Credits: Beckett et al. (SIGCOMM 2016): Bridging Network-wide Objectives and Device-level Configurations.

Responsibilities of a Sysadmin


Routers and switches store list of forwarding rules, and conditional failover rules.
B
A
C

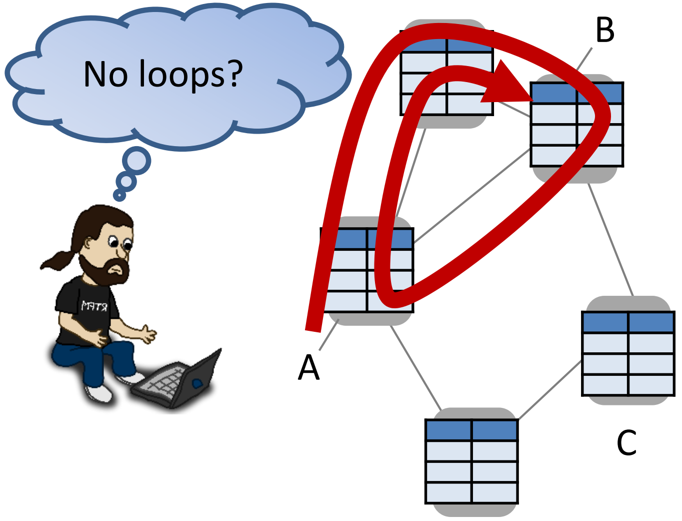
Responsibilities of a Sysadmin



Sysadmin responsible for:

- **Reachability:** Can traffic from ingress port A reach egress port B?

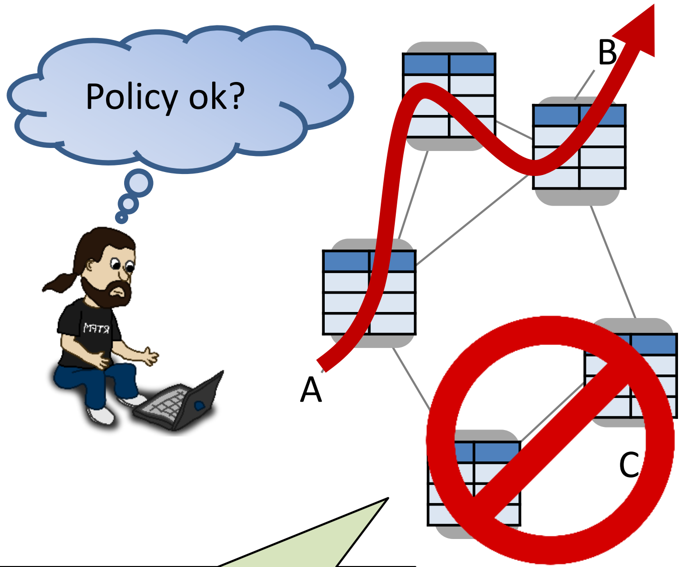
Responsibilities of a Sysadmin



Sysadmin responsible for:

- **Reachability:** Can traffic from ingress port A reach egress port B?
- **Loop-freedom:** Are the routes implied by the forwarding rules loop-free?

Responsibilities of a Sysadmin

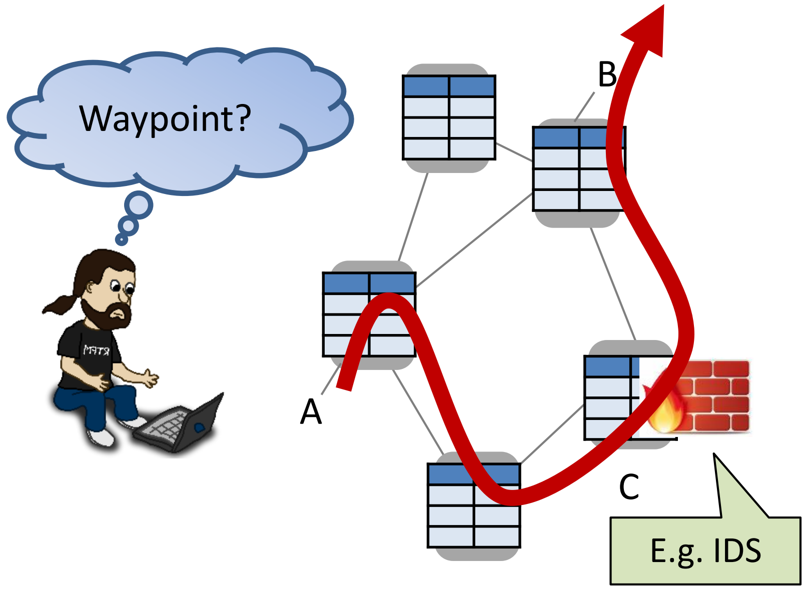


E.g. ***NORDUnet***: no traffic via Iceland (expensive!).

Sysadmin responsible for:

- **Reachability:** Can traffic from ingress port A reach egress port B?
- **Loop-freedom:** Are the routes implied by the forwarding rules loop-free?
- **Policy:** Is it ensured that traffic from A to B never goes via C?

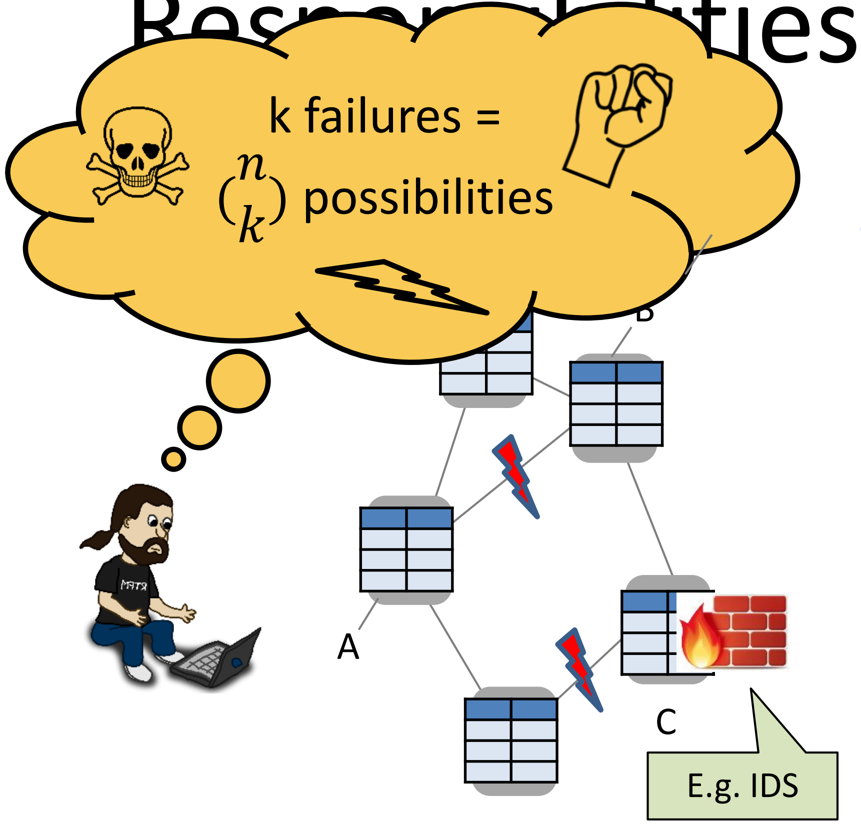
Responsibilities of a Sysadmin



Sysadmin responsible for:

- **Reachability:** Can traffic from ingress port A reach egress port B?
- **Loop-freedom:** Are the routes implied by the forwarding rules loop-free?
- **Policy:** Is it ensured that traffic from A to B never goes via C?
- **Waypoint enforcement:** Is it ensured that traffic from A to B is always routed via a node C (e.g., intrusion detection system or a firewall)?

Responsibilities of a Sysadmin

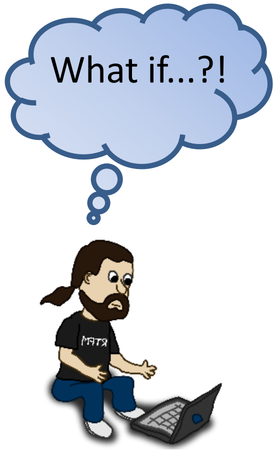


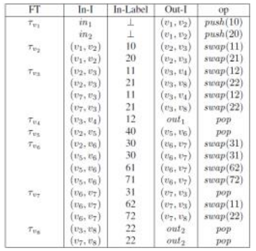
Sysadmin responsible for:

- **Reachability:** Can traffic from ingress port A reach egress port B?
- **Loop-freedom:** Are the routes implied by the forwarding rules loop-free?
- **Policy:** Is it ensured that traffic from A to B never goes via C?
- **Waypoint enforcement:** Is it ensured that traffic from A to B is always routed via a node C (e.g., intrusion detection system or a firewall)?

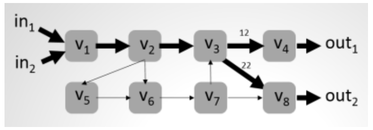
… and everything even under multiple failures?!

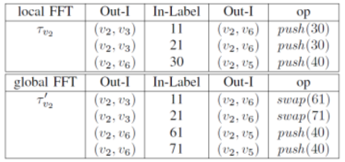
Vision: Automation and Formal Methods





FT	In-I	In-Label	Out-I	op
τ_{v_1}	in_1	$\perp$	(v_1, v_2)	$push(10)$
	in_2	$\perp$	(v_1, v_2)	$push(20)$
τ_{v_2}	(v_1, v_2)	10	(v_2, v_3)	$swap(11)$
	(v_1, v_2)	20	(v_2, v_3)	$swap(21)$
τ_{v_3}	(v_2, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_2, v_3)	21	(v_3, v_8)	$swap(22)$
	(v_7, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_7, v_3)	21	(v_3, v_8)	$swap(22)$
τ_{v_4}	(v_3, v_4)	12	out_1	pop
τ_{v_5}	(v_2, v_5)	40	(v_5, v_6)	pop
τ_{v_6}	(v_2, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	61	(v_6, v_7)	$swap(62)$
	(v_5, v_6)	71	(v_6, v_7)	$swap(72)$
τ_{v_7}	(v_6, v_7)	31	(v_7, v_3)	pop
	(v_6, v_7)	62	(v_7, v_3)	$swap(11)$
	(v_6, v_7)	72	(v_7, v_8)	$swap(22)$
τ_{v_8}	(v_3, v_8)	22	out_2	pop
	(v_7, v_8)	22	out_2	pop

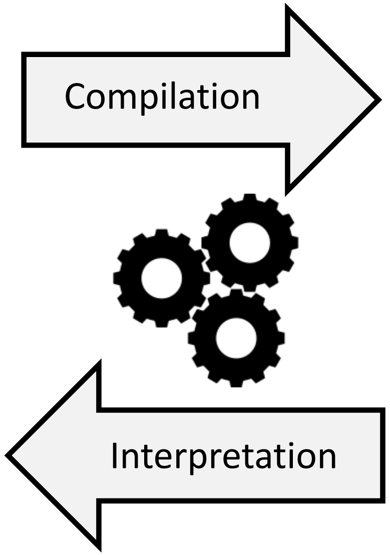




local FFT	Out-I	In-Label	Out-I	op
τ_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$push(30)$
	(v_2, v_3)	21	(v_2, v_6)	$push(30)$
	(v_2, v_6)	30	(v_2, v_5)	$push(40)$

global FFT	Out-I	In-Label	Out-I	op
τ'_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$swap(61)$
	(v_2, v_3)	21	(v_2, v_6)	$swap(71)$
	(v_2, v_6)	61	(v_2, v_5)	$push(40)$
	(v_2, v_6)	71	(v_2, v_5)	$push(40)$

Router **configurations**, Segment Routing etc.

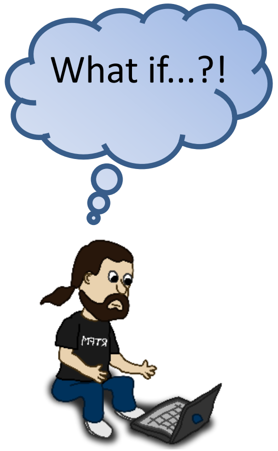


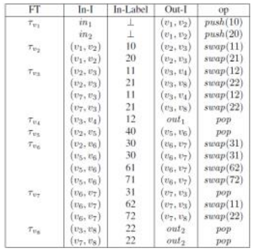
$$pX \Rightarrow qXX$$
$$pX \Rightarrow qYX$$
$$qY \Rightarrow rYY$$
$$rY \Rightarrow r$$
$$rX \Rightarrow pX$$

Pushdown Automaton and **Prefix Rewriting Systems** Theory

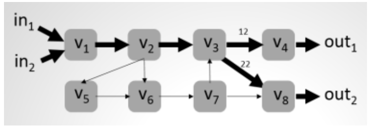
Vision: Automati ods

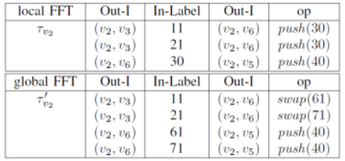
Use cases: Sysadmin ***issues queries*** to test certain properties, or do it on a ***regular basis*** automatically!





FT	In-I	In-Label	Out-I	op
τ_{v_1}	in_1	$\bot$	(v_1, v_2)	$push(10)$
	in_2	$\bot$	(v_1, v_2)	$push(20)$
τ_{v_2}	(v_1, v_2)	10	(v_2, v_3)	$swap(11)$
	(v_1, v_2)	20	(v_2, v_3)	$swap(21)$
τ_{v_3}	(v_2, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_2, v_3)	21	(v_3, v_8)	$swap(22)$
	(v_7, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_7, v_3)	21	(v_3, v_8)	$swap(22)$
τ_{v_4}	(v_3, v_4)	12	out_1	pop
τ_{v_5}	(v_2, v_5)	40	(v_5, v_6)	pop
τ_{v_6}	(v_2, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	61	(v_6, v_7)	$swap(62)$
	(v_5, v_6)	71	(v_6, v_7)	$swap(72)$
τ_{v_7}	(v_6, v_7)	31	(v_7, v_3)	pop
	(v_6, v_7)	62	(v_7, v_3)	$swap(11)$
	(v_6, v_7)	72	(v_7, v_8)	$swap(22)$
τ_{v_8}	(v_3, v_8)	22	out_2	pop
	(v_7, v_8)	22	out_2	pop





local FFT	Out-I	In-Label	Out-I	op
τ_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$push(30)$
	(v_2, v_3)	21	(v_2, v_6)	$push(30)$
	(v_2, v_6)	30	(v_2, v_5)	$push(40)$

global FFT	Out-I	In-Label	Out-I	op
τ'_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$swap(61)$
	(v_2, v_3)	21	(v_2, v_6)	$swap(71)$
	(v_2, v_6)	61	(v_2, v_5)	$push(40)$
	(v_2, v_6)	71	(v_2, v_5)	$push(40)$

Compilation

Interpretation

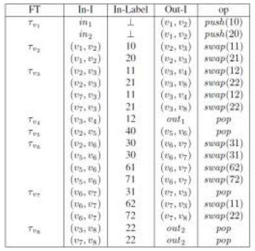
$$pX \Rightarrow qXX$$
$$pX \Rightarrow qYX$$
$$qY \Rightarrow rYY$$
$$rY \Rightarrow r$$
$$rX \Rightarrow pX$$

Router **configurations**, Segment Routing etc.

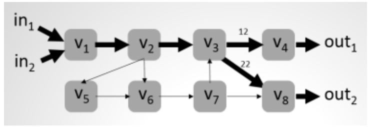
Pushdown Automaton and **Prefix Rewriting Systems** Theory

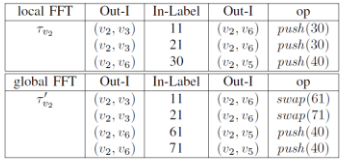
Vision: Automati ods

Use cases: Sysadmin ***issues queries*** to test certain properties, or do it on a ***regular basis*** automatically!



FT	In-I	In-Label	Out-I	op
τ_{v_1}	in_1	$\bot$	(v_1, v_2)	$push(10)$
	in_2	$\bot$	(v_1, v_2)	$push(20)$
τ_{v_2}	(v_1, v_2)	10	(v_2, v_3)	$swap(11)$
	(v_1, v_2)	20	(v_2, v_3)	$swap(21)$
τ_{v_3}	(v_2, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_2, v_3)	21	(v_3, v_8)	$swap(22)$
	(v_7, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_7, v_3)	21	(v_3, v_8)	$swap(22)$
τ_{v_4}	(v_3, v_4)	12	out_1	pop
τ_{v_5}	(v_2, v_5)	40	(v_5, v_6)	pop
τ_{v_6}	(v_2, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	30	(v_6, v_7)	$swap(31)$
	(v_5, v_6)	61	(v_6, v_7)	$swap(62)$
	(v_5, v_6)	71	(v_6, v_7)	$swap(72)$
τ_{v_7}	(v_6, v_7)	31	(v_7, v_3)	pop
	(v_6, v_7)	62	(v_7, v_3)	$swap(11)$
	(v_6, v_7)	72	(v_7, v_8)	$swap(22)$
τ_{v_8}	(v_3, v_8)	22	out_2	pop
	(v_7, v_8)	22	out_2	pop

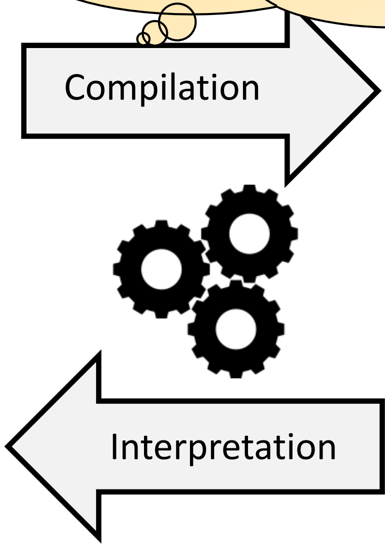




local FFT	Out-I	In-Label	Out-I	op
τ_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$push(30)$
	(v_2, v_3)	21	(v_2, v_6)	$push(30)$
	(v_2, v_6)	30	(v_2, v_5)	$push(40)$

global FFT	Out-I	In-Label	Out-I	op
τ'_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$swap(61)$
	(v_2, v_3)	21	(v_2, v_6)	$swap(71)$
	(v_2, v_6)	61	(v_2, v_5)	$push(40)$
	(v_2, v_6)	71	(v_2, v_5)	$push(40)$

Router **configurations**, Segment Routing etc.



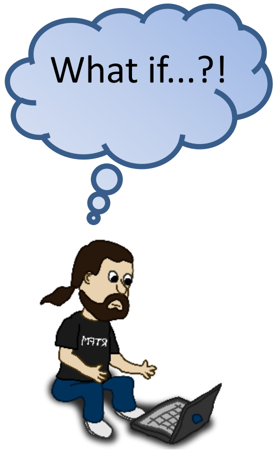
$$pX \Rightarrow qXX$$
$$pX \Rightarrow qYX$$
$$qY \Rightarrow rYY$$
$$rY \Rightarrow r$$
$$rX \Rightarrow pX$$

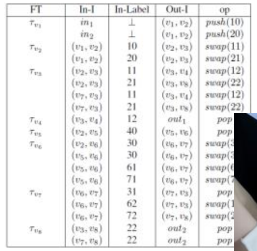
Pushdown Automaton and **Prefix Rewriting Systems** Theory

P-Rex: Fast Verification of MPLS Networks with Multiple Link Failures. Jensen et al., ACM CoNEXT, 2018.

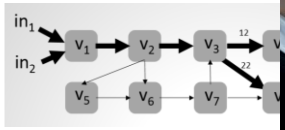
Vision: Automati ods

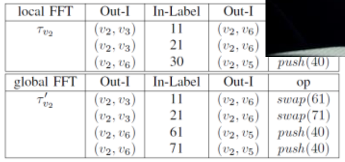
Use cases: Sysadmin ***issues queries*** to test certain properties, or do it on a ***regular basis*** automatically!





FT	In-I	In-Label	Out-I	op
τ_{v_1}	in_1	$\perp$	(v_1, v_2)	$push(10)$
	in_2	$\perp$	(v_1, v_2)	$push(20)$
τ_{v_2}	(v_1, v_2)	10	(v_2, v_3)	$swap(11)$
	(v_1, v_2)	20	(v_2, v_3)	$swap(21)$
τ_{v_3}	(v_2, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_2, v_3)	21	(v_3, v_8)	$swap(22)$
	(v_7, v_3)	11	(v_3, v_4)	$swap(12)$
	(v_7, v_3)	21	(v_3, v_8)	$swap(22)$
τ_{v_4}	(v_3, v_4)	12	out_1	pop
τ_{v_5}	(v_2, v_5)	40	(v_5, v_6)	pop
τ_{v_6}	(v_2, v_6)	30	(v_6, v_7)	$swap(3$
	(v_5, v_6)	30	(v_6, v_7)	$swap(3$
	(v_5, v_6)	61	(v_6, v_7)	$swap($
	(v_5, v_6)	71	(v_6, v_7)	$swap($
τ_{v_7}	(v_6, v_7)	31	(v_7, v_3)	pop
	(v_6, v_7)	62	(v_7, v_3)	$swap(1$
	(v_6, v_7)	72	(v_7, v_8)	$swap($
τ_{v_8}	(v_3, v_8)	22	out_2	pop
	(v_7, v_8)	22	out_2	pop





local FFT	Out-I	In-Label	Out-I	
τ_{v_2}	(v_2, v_3)	11	(v_2, v_6)	
	(v_2, v_3)	21	(v_2, v_6)	
	(v_2, v_6)	30	(v_2, v_5)	$push(40)$

global FFT	Out-I	In-Label	Out-I	op
τ'_{v_2}	(v_2, v_3)	11	(v_2, v_6)	$swap(61)$
	(v_2, v_3)	21	(v_2, v_6)	$swap(71)$
	(v_2, v_6)	61	(v_2, v_5)	$push(40)$
	(v_2, v_6)	71	(v_2, v_5)	$push(40)$

$X \Rightarrow qXX$

$X \Rightarrow qYX$

$Y \Rightarrow rYY$

$rY \Rightarrow r$

$X \Rightarrow pX$

Router **configurations**, Segment Routing etc.

Pushdown Automaton and **Prefix Rewriting Systems** Theory

P-Rex: Fast Verification of MPLS Networks with Multiple Link Failures. Jensen et al., ACM CoNEXT, 2018.

Case Study: MPLS Networks

- Widely deployed networks by Internet Service Providers (**ISPs**)

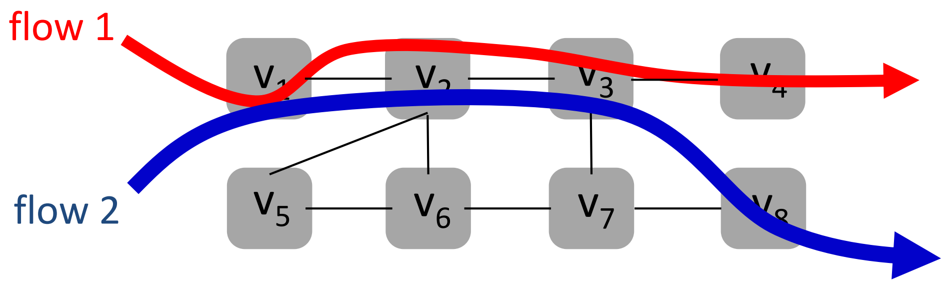
- Often used for **traffic engineering**
  - Avoid congestion by going non-shortest paths

- Allows for ***header re-writing*** upon failures
  - Header based on **stack of labels**

Enables efficient verification!

How (MPLS) Networks Work

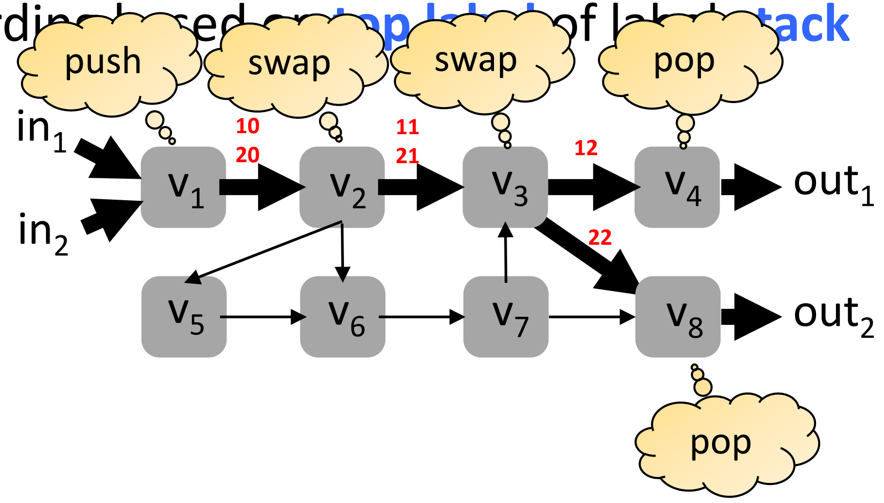
- Forwarding based on **top label** of label **stack**



Default routing of two flows

How (MPLS) Networks Work

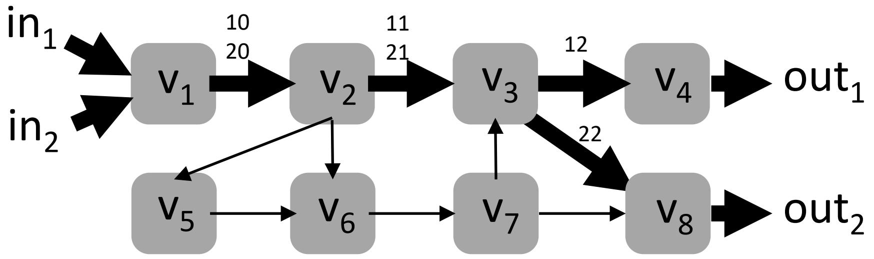
- Forwarding based on **top label** of label **stack**



Default routing of two flows

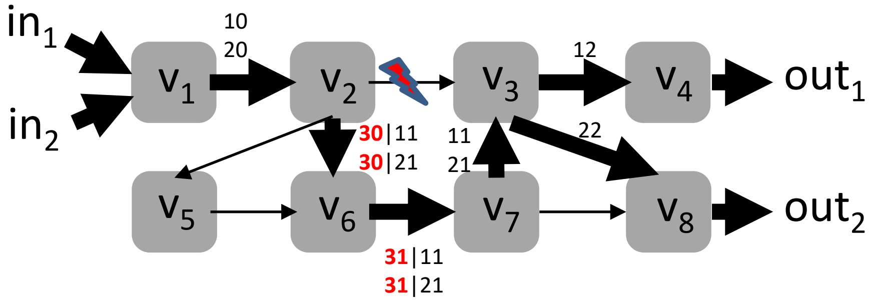
Fast Reroute Around *1 Failure*

- Forwarding based on **top label** of label **stack** (in packet header)



Default routing of two flows

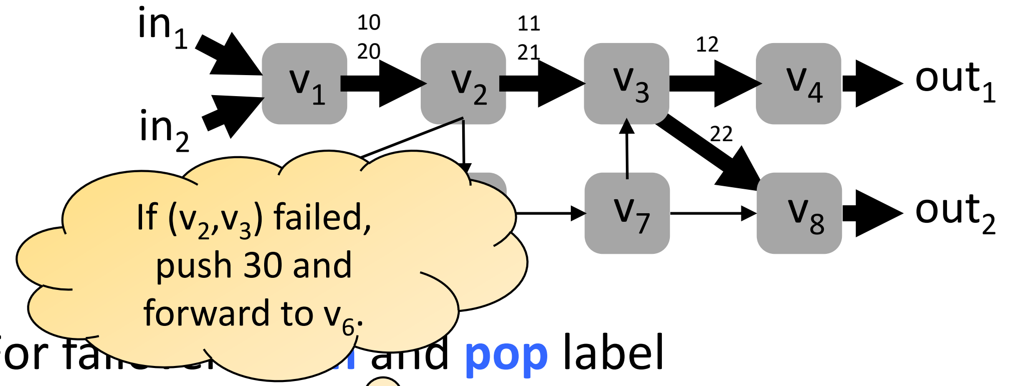
- For failover: **push** and **pop** label



One failure: push 30: route around (v_2,v_3)

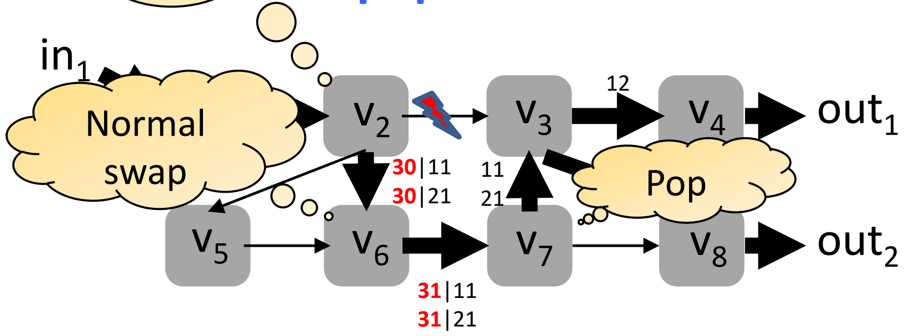
Fast Reroute Around *1 Failure*

- Forwarding based on **top label** of label **stack** (in packet header)



Default routing of two flows

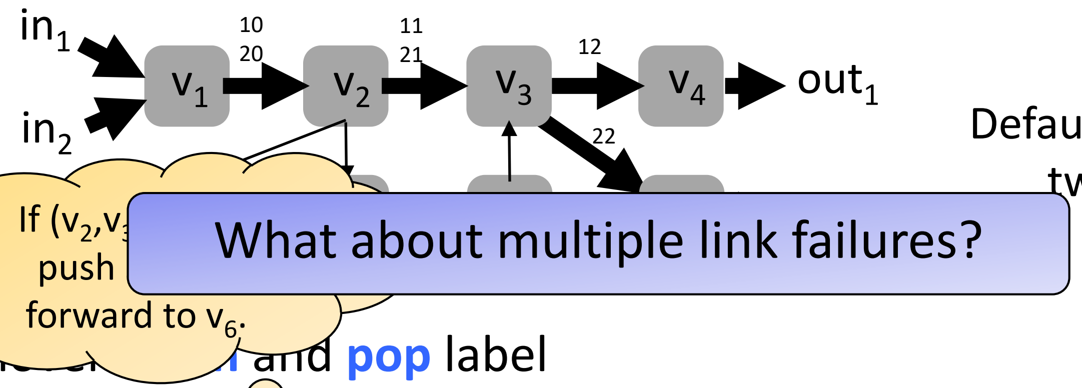
- For fa... **...** and **pop** label



One failure: push 30: route around (v_2,v_3)

Fast Reroute Around *1 Failure*

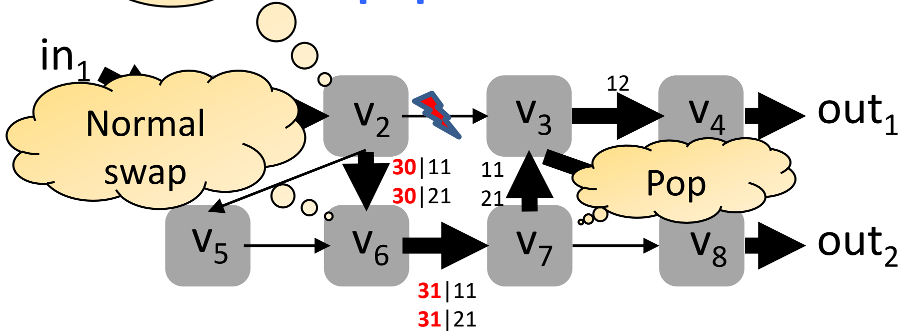
- Forwarding based on **top label** of label **stack** (in packet header)



What about multiple link failures?

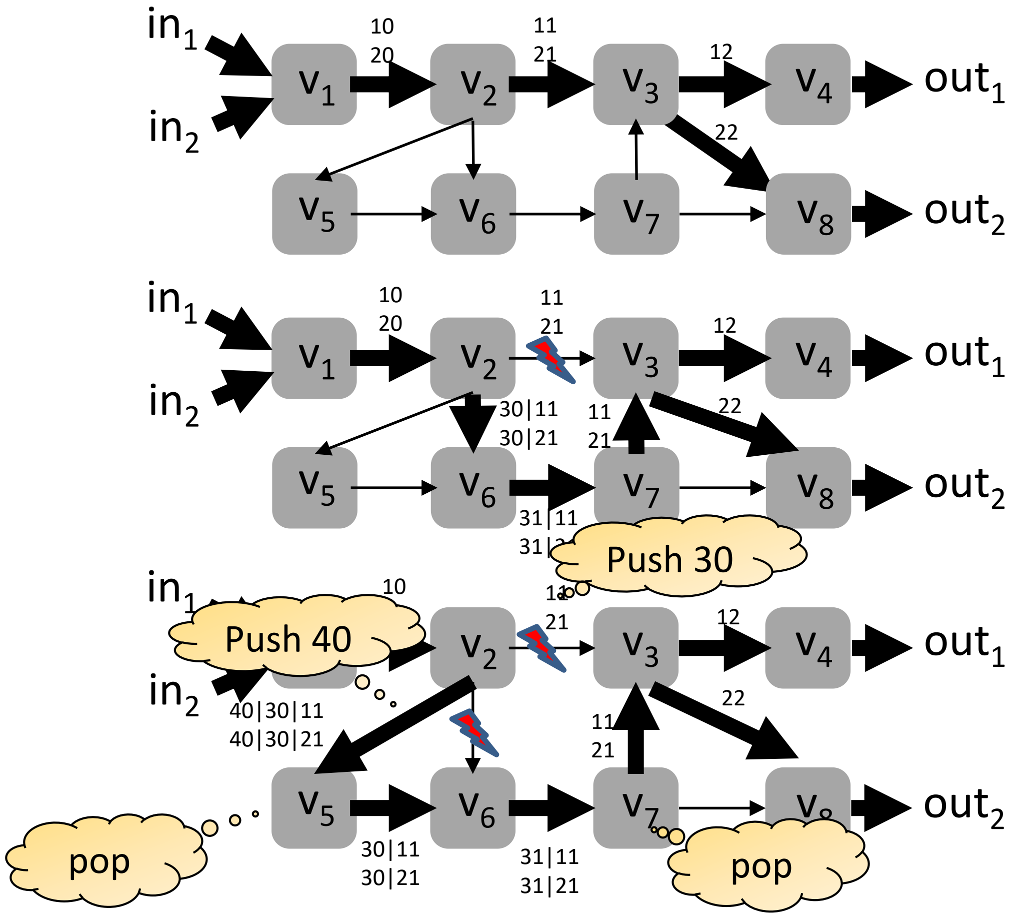
Default routing of two flows

- For fa... and **pop** label



One failure: push 30: route around (v$_2$,v$_3$)

2 Failures: Push *Recursively*



Original Routing

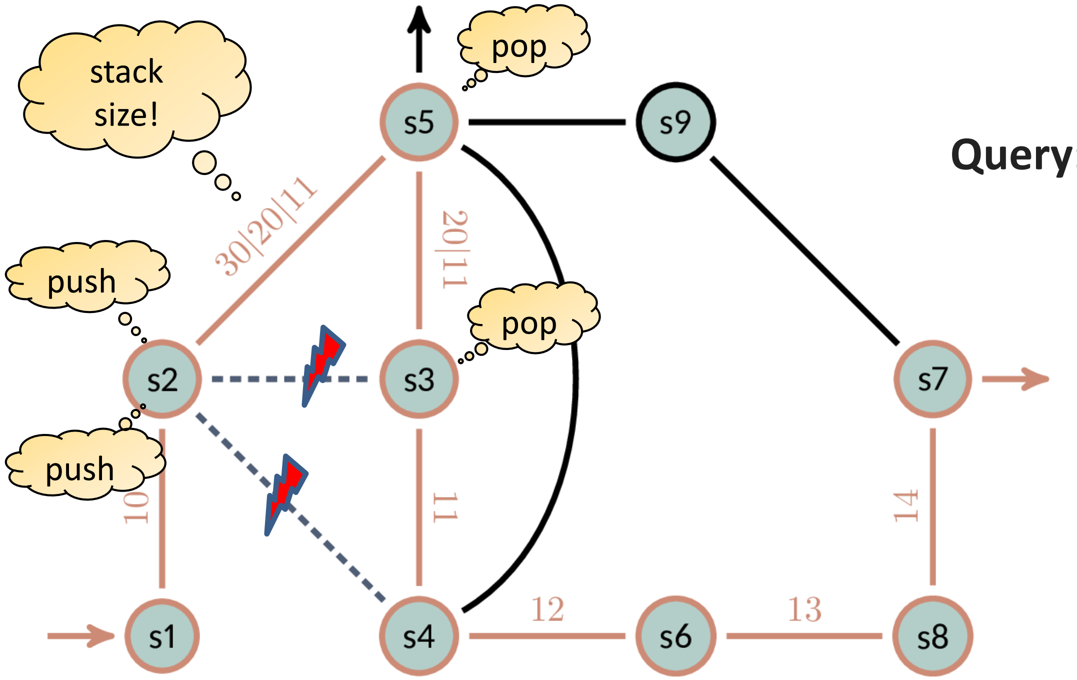
One failure: push 30: route around (v_2,v_3)

Two failures: first push 30: route around (v_2,v_3)

Push recursively 40: route around (v_2,v_6)

Example: P-Rex for MPLS Networks

Can traffic starting with [] go through s5, under up to k=2 failures?



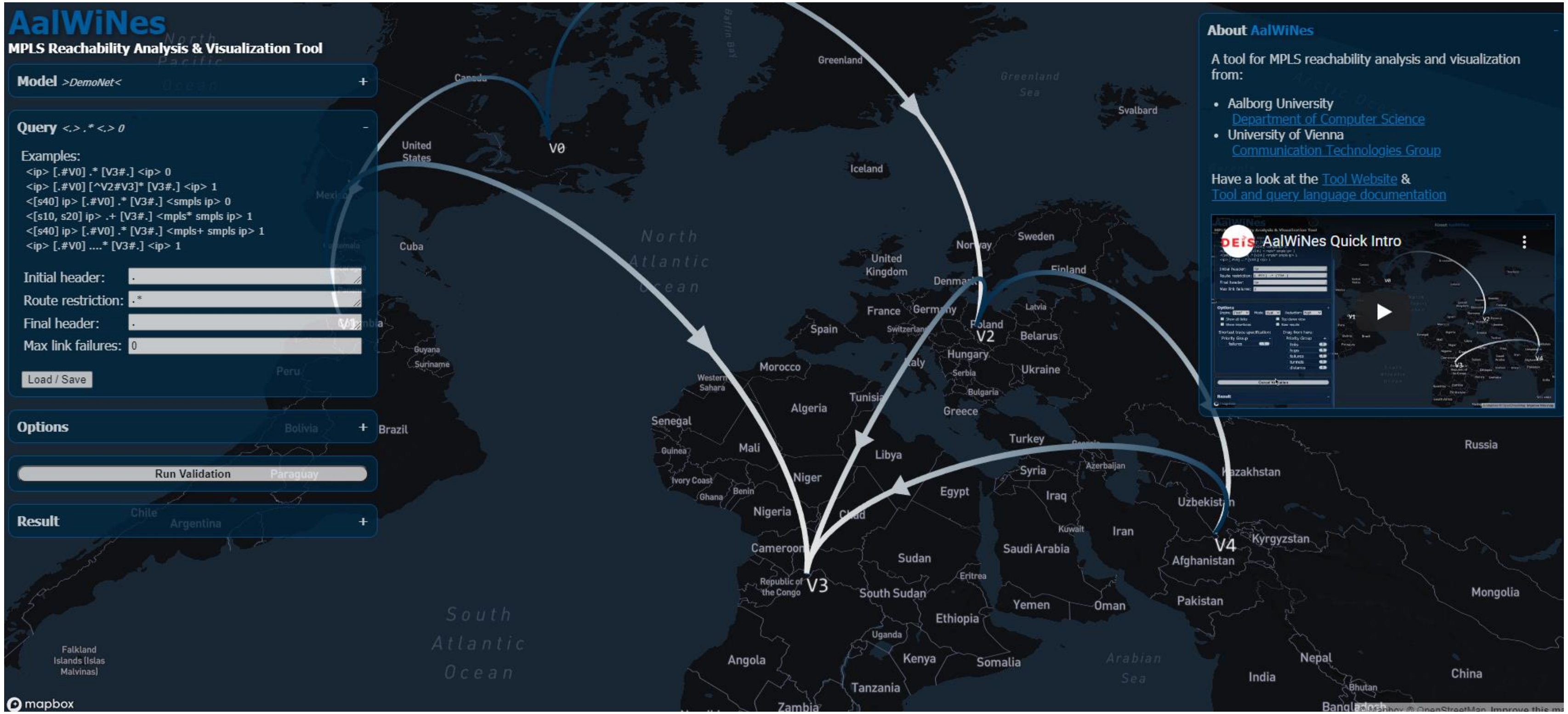
2 failures

Query: k=2 [] s1 >> s5 >> s7 []

YES

(Polynomial time!)

Demo of P-Rex / AalWiNes Tool



AalWiNes
MPLS Reachability Analysis & Visualization Tool

Model *>DemoNet<*

Query *<.> .* <.> 0*

Examples:

```
<ip> [.#V0] .* [V3#.] <ip> 0
<ip> [.#V0] [^V2#V3]* [V3#.] <ip> 1
<[s40] ip> [.#V0] .* [V3#.] <smpls ip> 0
<[s10, s20] ip> .+ [V3#.] <mpls* smpls ip> 1
<[s40] ip> [.#V0] .* [V3#.] <mpls+ smpls ip> 1
<ip> [.#V0] ....* [V3#.] <ip> 1
```

Initial header:	.
Route restriction:	.*
Final header:	.
Max link failures:	0

About AalWiNes

A tool for MPLS reachability analysis and visualization from:

- Aalborg University
  Department of Computer Science
- University of Vienna
  Communication Technologies Group

Have a look at the Tool Website & Tool and query language documentation

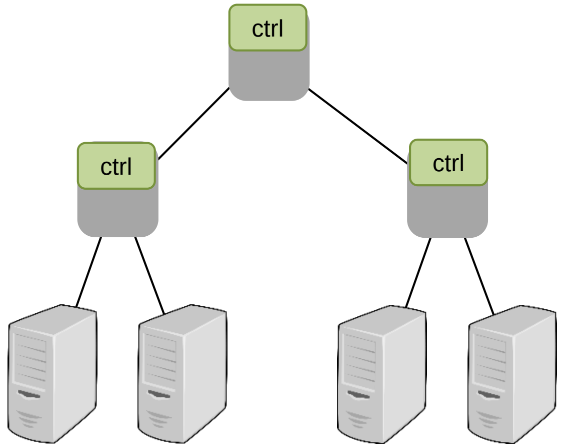
Tool: https://demo.aalwines.cs.aau.dk/, Youtube: https://www.youtube.com/watch?v=mvXAn9i7_Q0

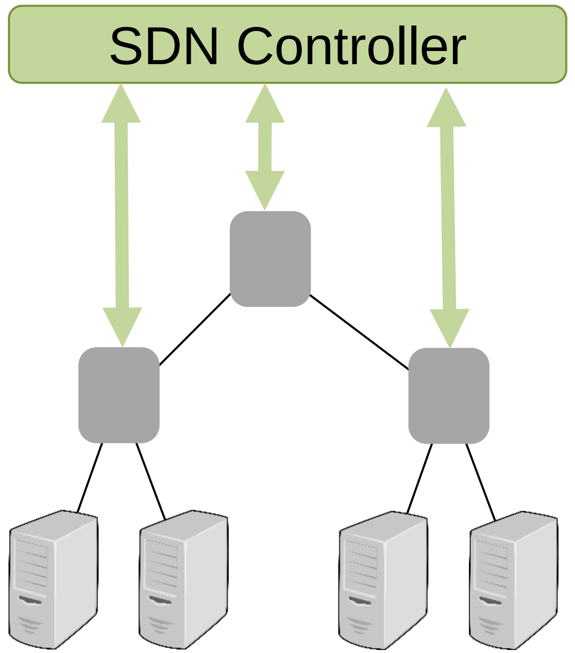
Roadmap

- Opportunity: emerging networking technologies
  - Automation and „self-driving networks“
  - **Programmable networks for improved visibility**
- Challenge: emerging network technologies
  - New threat models

It’s an ***exciting period***! New tools, simple abstractions, disburdening human operators, etc.

Software-Defined Networks





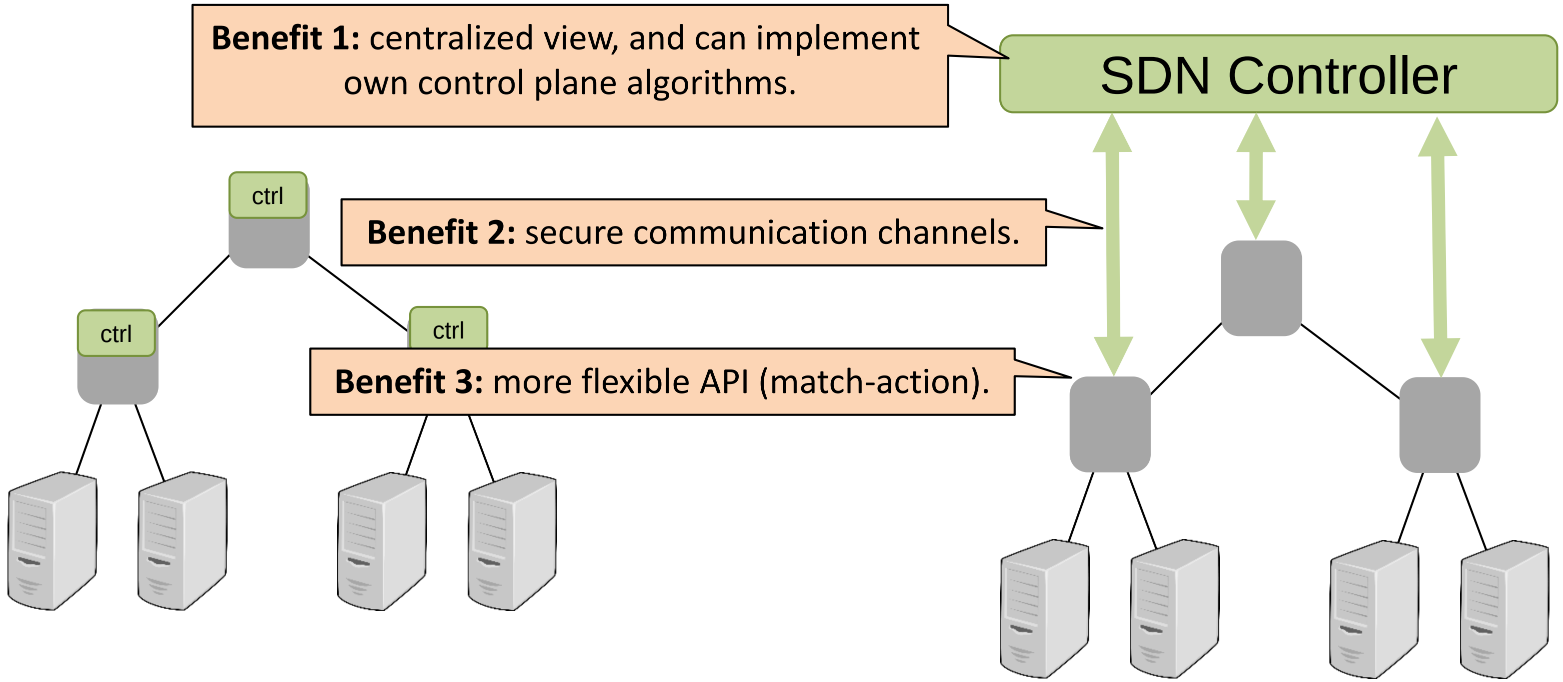
Traditionally:

- Distributed control plane
- Blackbox, not programmable

Software-defined Networs (SDN):

- Logically centralized control
- Programmable, match-action

Software-Defined Networks



Benefit 1: centralized view, and can implement own control plane algorithms.

Benefit 2: secure communication channels.

Benefit 3: more flexible API (match-action).

Traditionally:

- Distributed control plane
- Blackbox, not programmable

Software-defined Networs (SDN):

- Logically centralized control
- Programmable, match-action

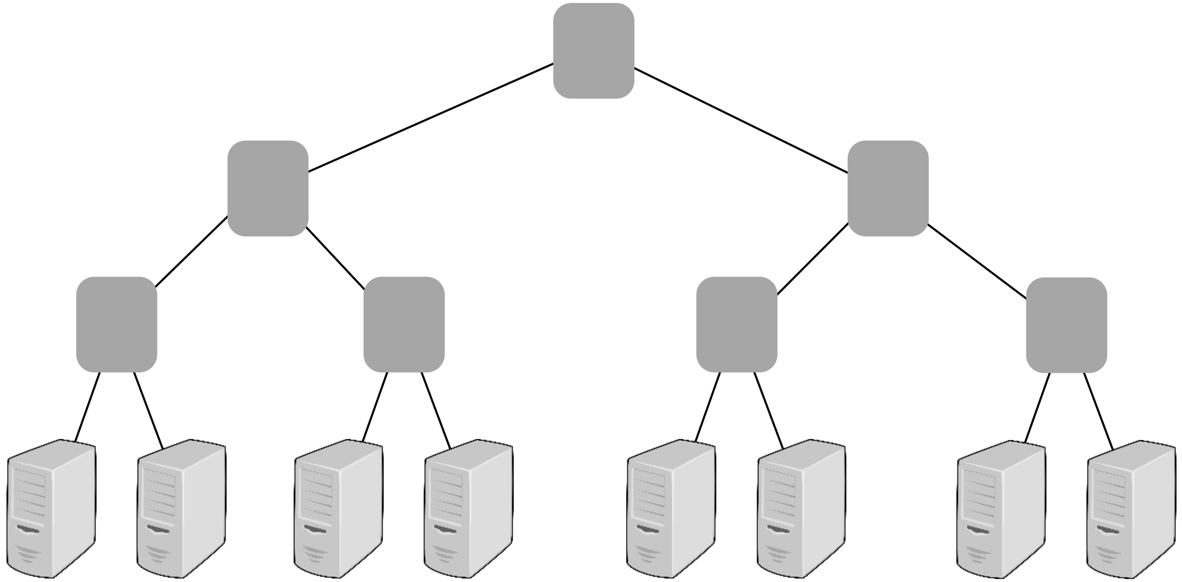
Example Application for SDN: Detecting Misbehavior

Dealing with Untrusted Hardware: Secure Trajectory Sampling

Monitor packets, traditionally:

trajectory sampling

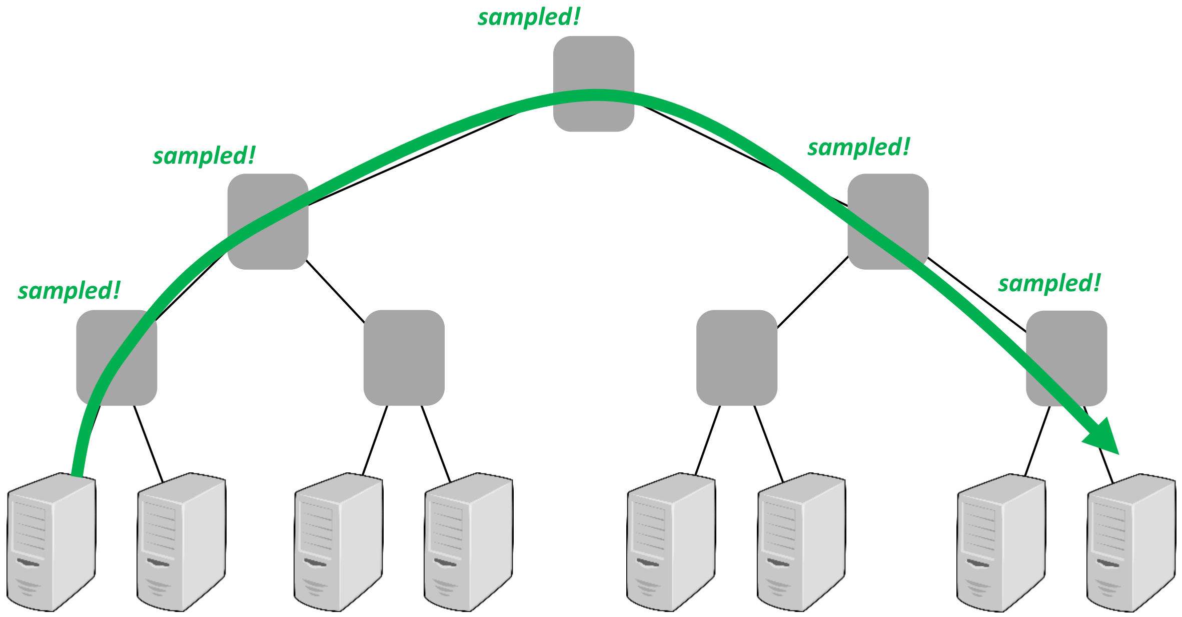
- *Globally* sample packets with ***hash(imm. header)∈[x,y]***
- See full routes *of some packets*



Dealing with Untrusted Hardware: Secure Trajectory Sampling

Monitor packets, traditionally:
trajectory sampling

- *Globally* sample packets with ***hash(imm. header)∈[x,y]***
- See full routes *of some packets*

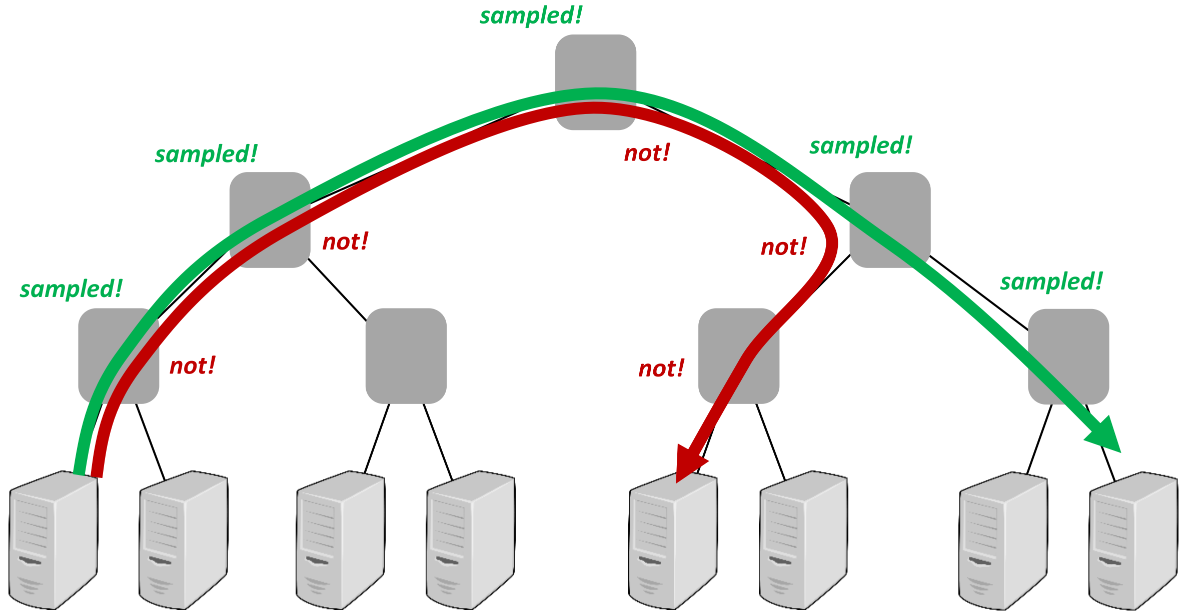


Dealing with Untrusted Hardware: Secure Trajectory Sampling

Monitor packets, traditionally:

trajectory sampling

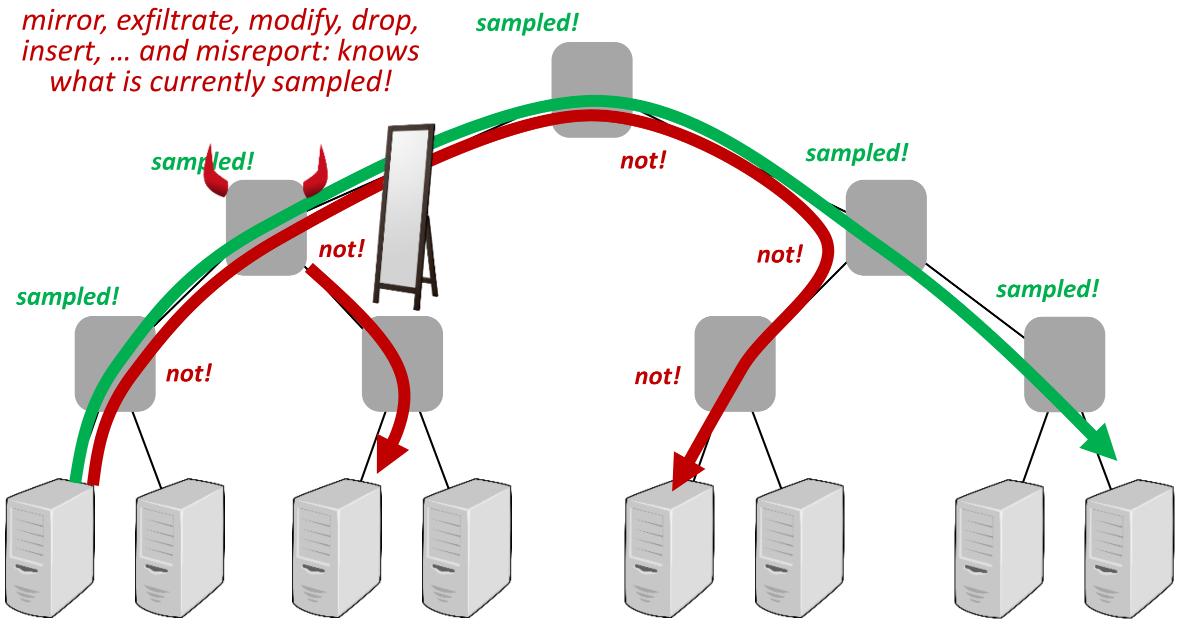
- *Globally* sample packets with ***hash(imm. header)∈[x,y]***
- See full routes *of some packets*
- But *not others!* (resp. later)



Dealing with Untrusted Hardware: Secure Trajectory Sampling

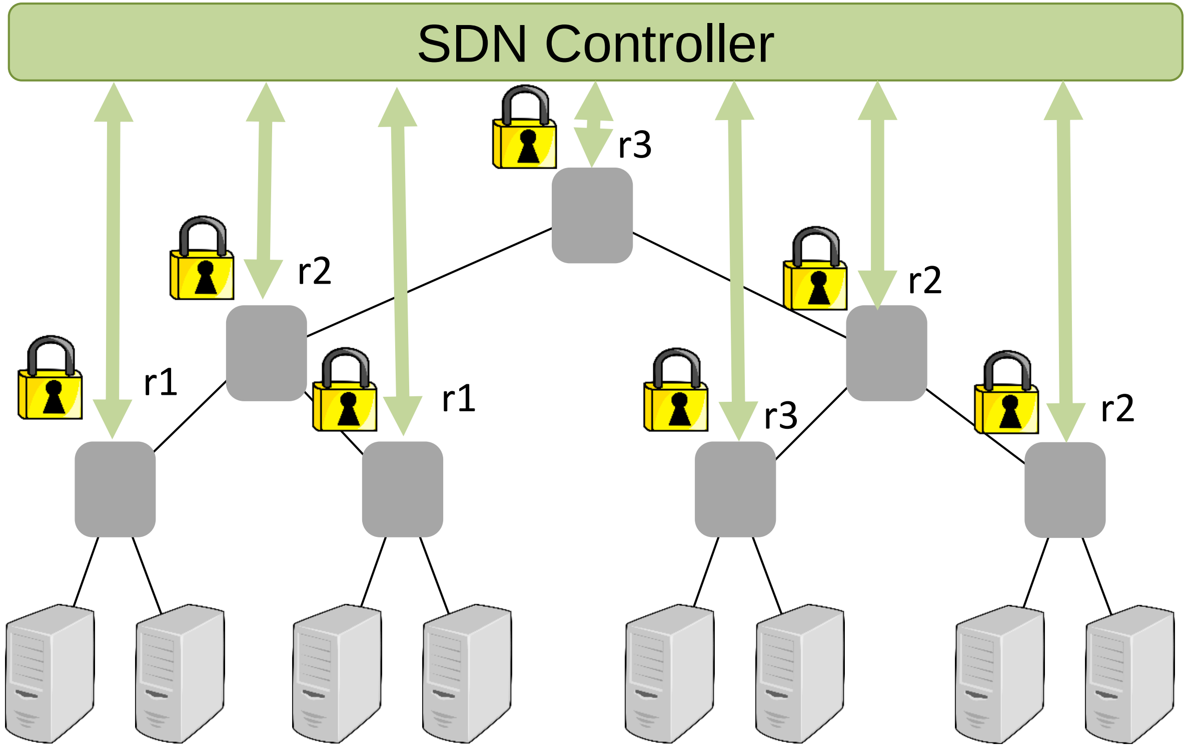
Monitor packets, traditionally:
trajectory sampling

- *Globally* sample packets with ***hash(imm. header)∈[x,y]***
- See full routes *of some packets*
- But *not others!* (resp. later)


mirror, exfiltrate, modify, drop, insert, … and misreport: knows what is currently sampled!
sampled!
sampled!
not!
sampled!
not!
not!
sampled!
not!
not!
sampled!

Solution: Use SDN for *Secure* Trajectory Sampling

- Idea:
  - Use ***secure*** channels between controller and switches to distribute hash ranges
  - Give ***different hash ranges*** hash ranges to different switches, but add some ***redundancy***: risk of being caught!

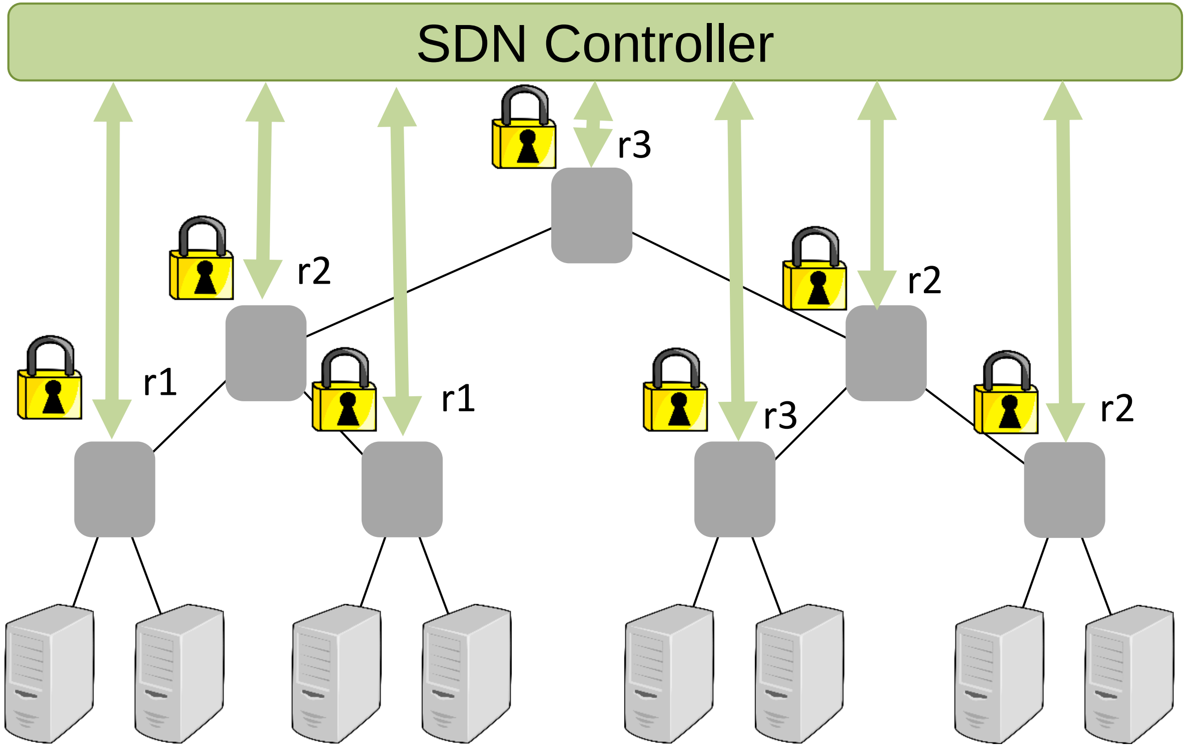


Network Policy Checker for Adversarial Environments.
Kashyap Thimmaraju, Liron Schiff, and S. SRDS 2019.

Solution: Use SDN for *Secure* Trajectory Sampling

- Idea:
  - Use ***secure*** channels between controller and switches to distribute hash ranges
  - Give ***different hash ranges*** hash ranges to different switches, but add some ***redundancy***: risk of being caught!

- In general: obtaining live data from the network ***becomes easier!***



Network Policy Checker for Adversarial Environments. Kashyap Thimmaraju, Liron Schiff, and S. SRDS 2019.

Roadmap

- Opportunity: emerging networking technologies
  - Automation and „self-driving networks“
  - Programmable networks for improved visibility

- Challenge: emerging network technologies
  - New threat models

It’s an ***exciting period***! New tools, simple abstractions, disburdening human operators, etc.

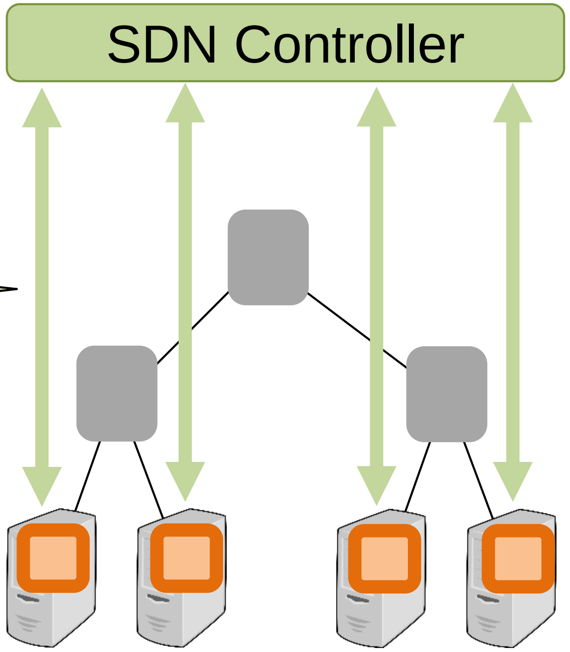
Roadmap

- Opportunity: emerging networking technologies
  - Automation and „self-driving networks“
  - Programmable networks for improved visibility

- **Challenge: emerging network technologies**
  - **New threat models**

It’s an ***exciting period***! New tools, simple abstractions, disburdening human operators, etc.

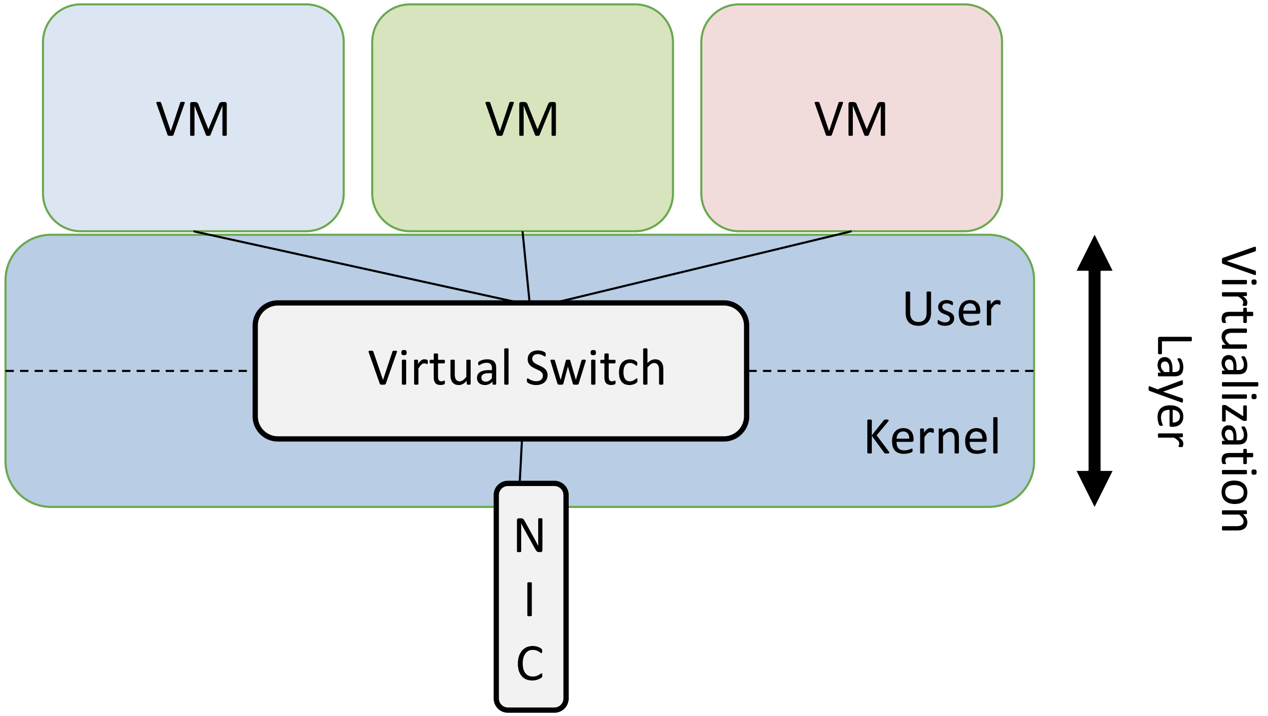
SDN in Datacenters: Virtual Switches



Simple in core: hardware switches.

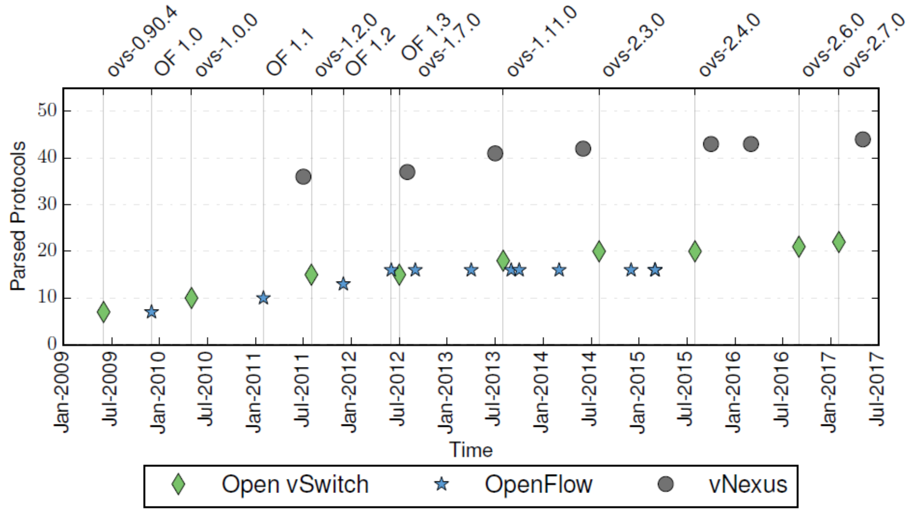
Clever at edge: virtualized and programmable.

The Virtual Switch



Virtual switches reside in the **server's virtualization layer** (e.g., Xen's Dom0). Goal: provide connectivity and isolation.

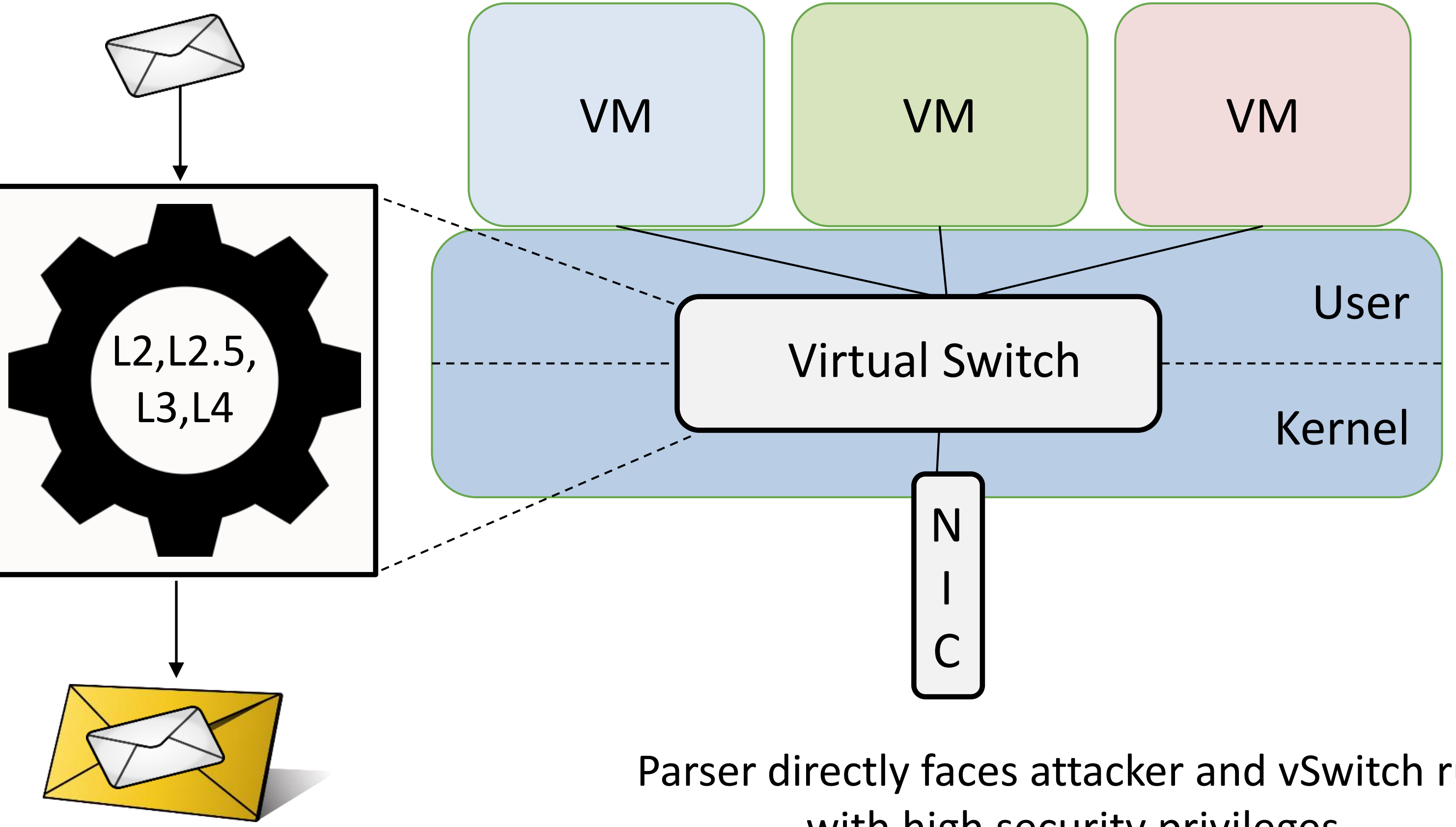
A Challenge: Complexity



Number of parsed high-level protocols constantly increases…

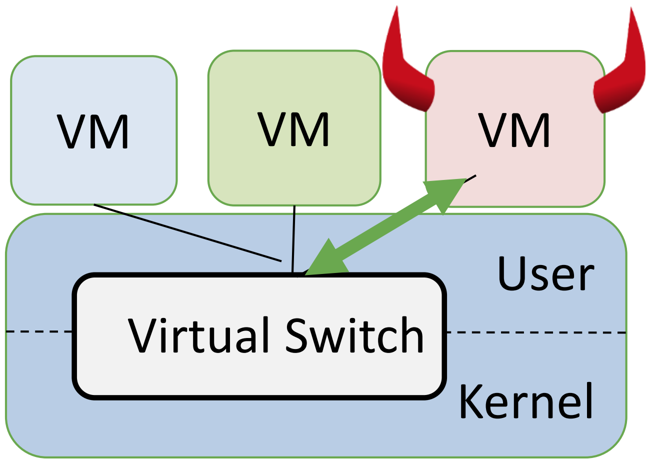
Complexity: Parsing

Ethernet
LLC
VLAN
MPLS
IPv4
ICMPv4
TCP
UDP
ARP
SCTP
IPv6
ICMPv6
IPv6 ND
GRE
LISP
VXLAN
PBB
IPv6 EXT HDR
TUNNEL-ID
IPv6 ND
IPv6 EXT HDR
IPv6HOPOPTS
IPv6ROUTING
IPv6Fragment
IPv6DESTOPT
IPv6ESP
IPv6 AH
RARP
IGMP

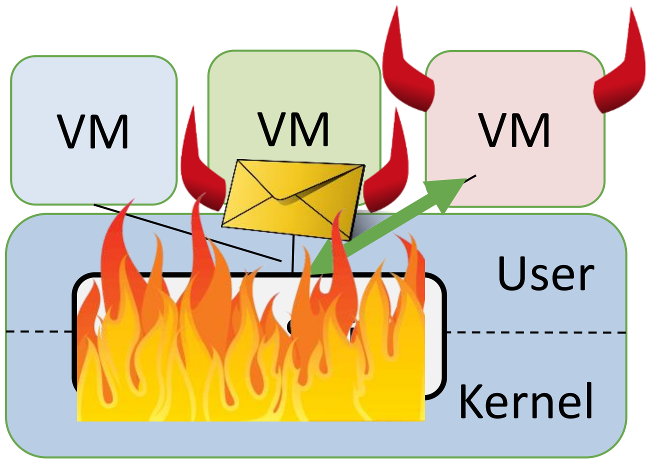


Parser directly faces attacker and vSwitch runs with high security privileges.

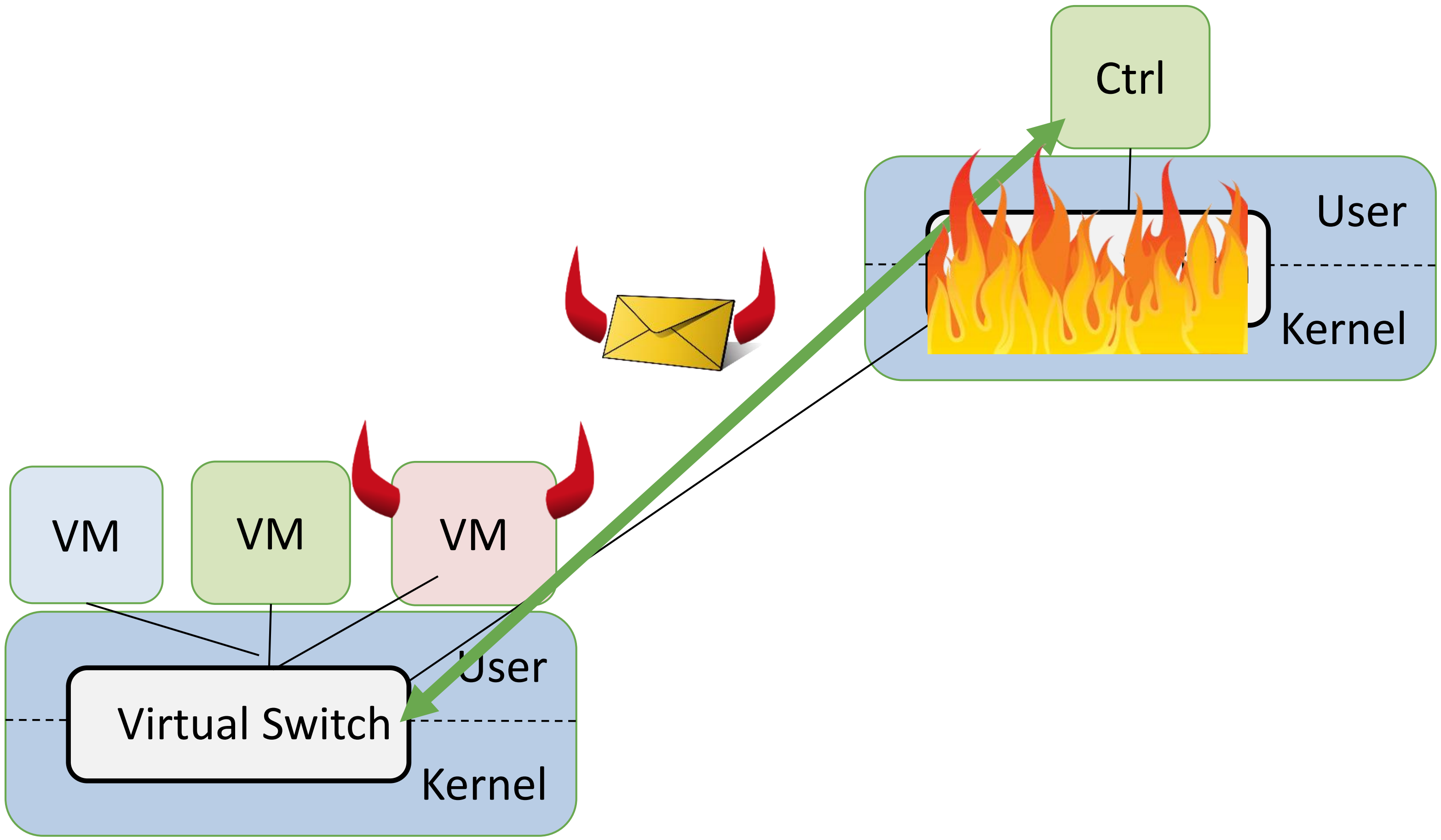
Enables Very Low-Cost Attacks



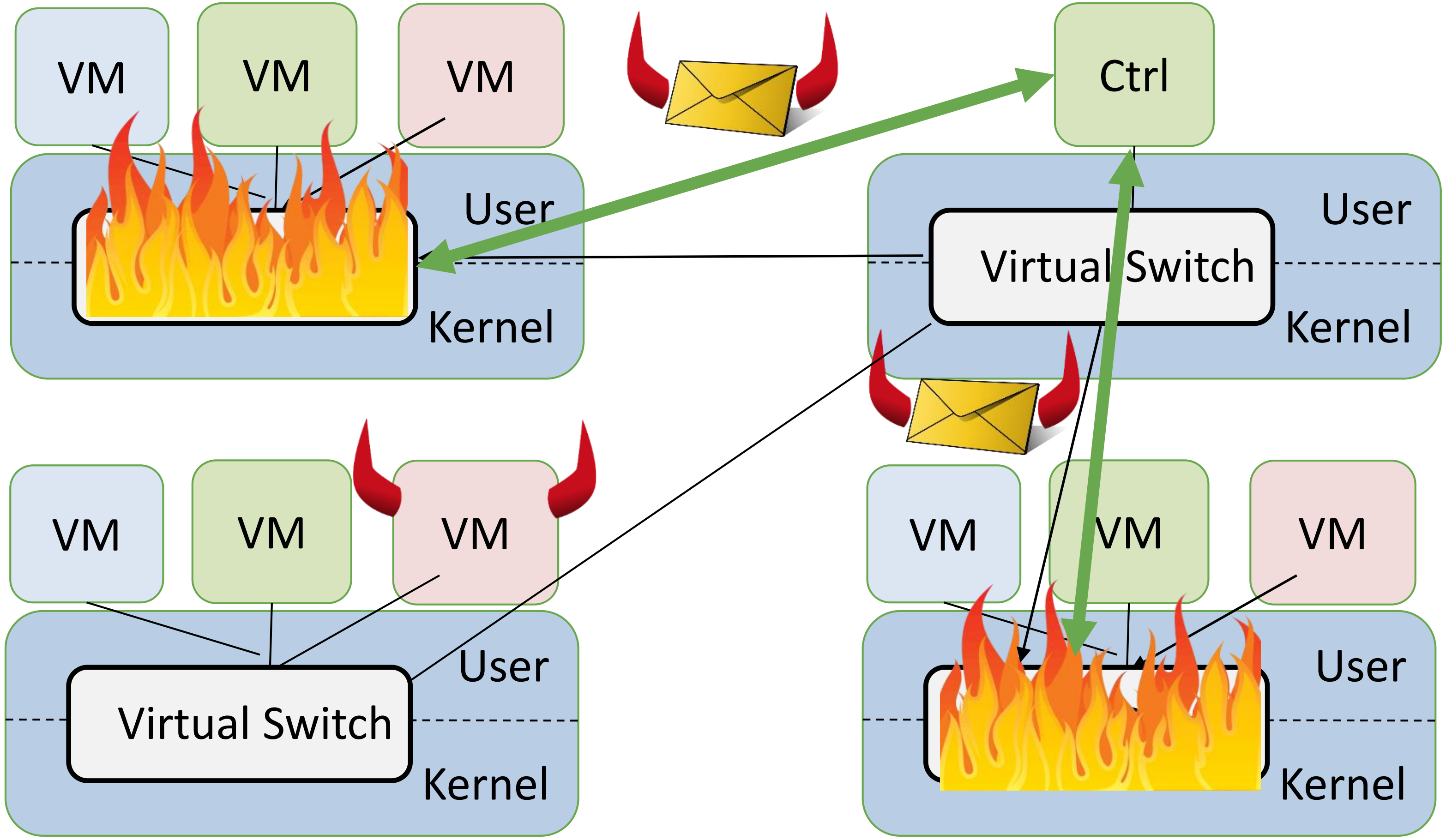
Enables Very Low-Cost Attacks



Enables Very Low-Cost Attacks

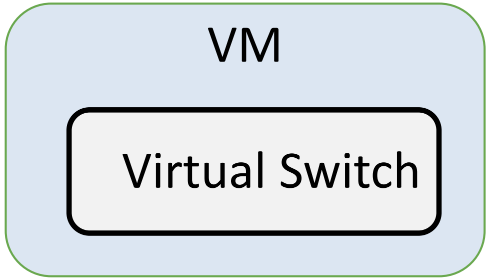


Enables Very Low-Cost Attacks



Challenge: How to provide better isolation *efficiently*?

- Idea for better *isolation*: put vSwitch in a VM
- But what about *performance*?
- Or container?

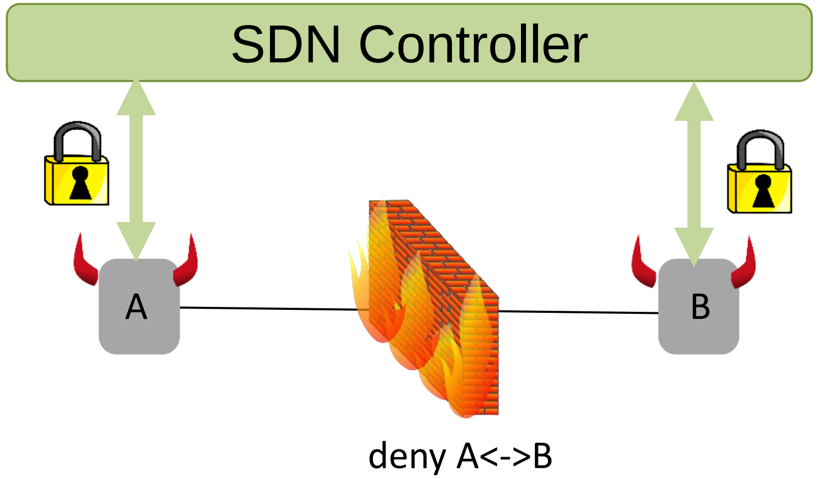


MTS: Bringing Multi-Tenancy to Virtual Switches Kashyap Thimmaraju, Saad Hermak, Gabor Retvari, and S. USENIX ATC, 2019.

Also A Challenge:
Covert and Side Channels

New Types of Attacks: Via SDN Controller

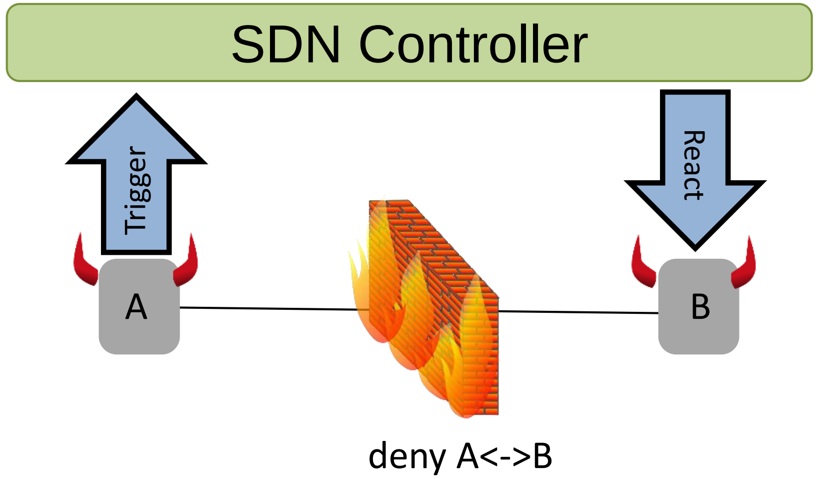
- **Controller** may be attacked or exploited



Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

New Types of Attacks: Via SDN Controller

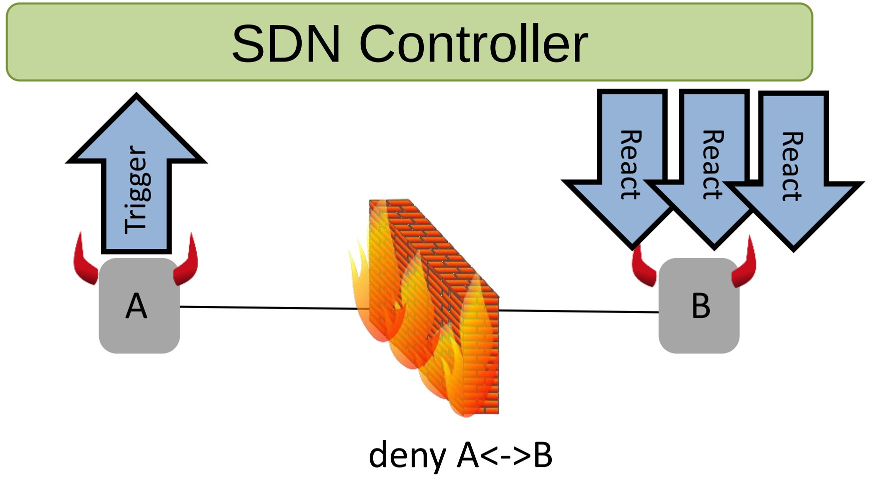
- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs



Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

New Types of Attacks: Via SDN Controller

- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs
  - Or even *multicast*: **pave-path technique** more efficient than hop-by-hop

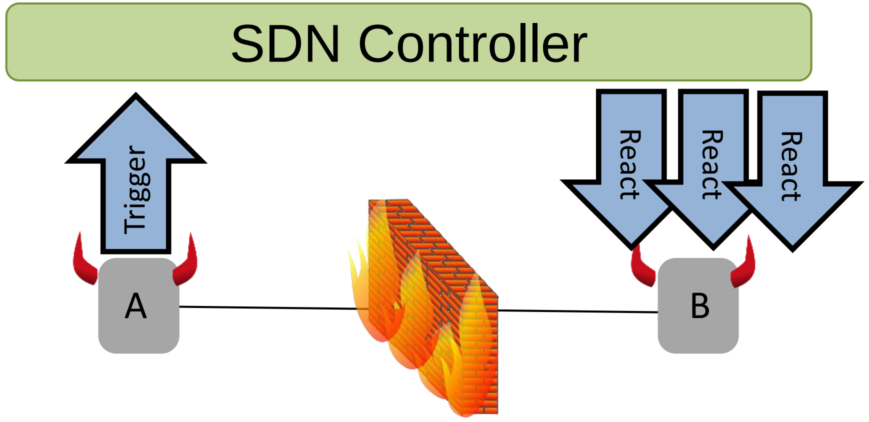


Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

New Types of Attacks: Via SDN Controller

- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs
  - Or even *multicast*: **pave-path technique** more efficient than hop-by-hop

May introduce ***new communication paths*** which can be used in unintendend ways!

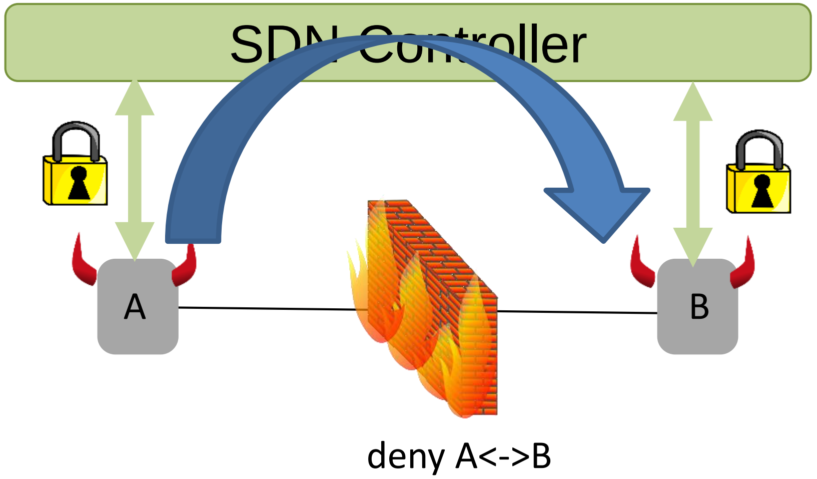


deny A<->B

Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

New Types of Attacks: Via SDN Controller

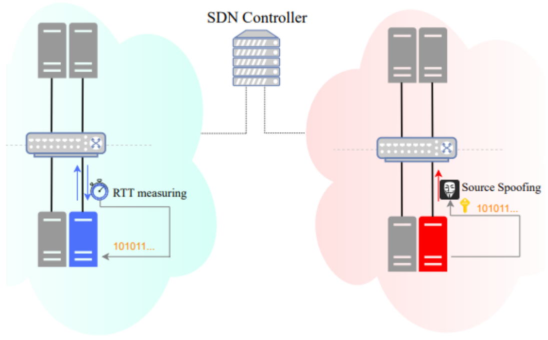
- In particular: new **covert communication** channels
  - E.g., exploit MAC learning (use codeword „0xBADDAD“) or modulate information with timing
- May *bypass security-critical elements*: e.g., firewall in the dataplane
- *Hard to catch*: along „normal communication paths“ and encrypted

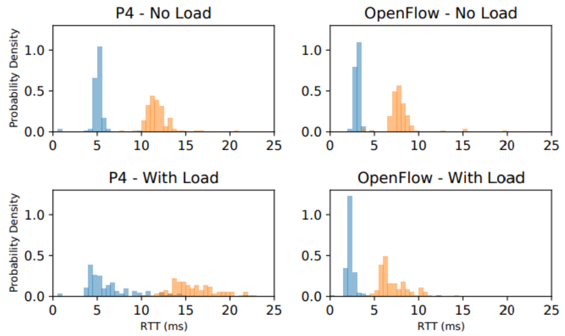


Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

Example: Macchiato Side Channels

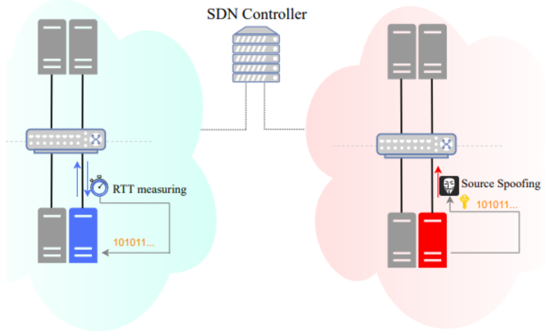
- Side channel *without malicious switches*: just between hosts
- Also leverages MAC learning for *mobility applications*: the fact that old rules associated with a MAC address are cleared when they marked in a new location
- The switch connected to the red host contacts the controller to report a new MAC address discovery. The controller interprets this that the blue host migrated to a new location and performs flow reconfigurations. Introduces *measurable differences* to the blue host that can be used repeatedly by the red host to modulate confidential data.
- Throughput can be improved with **deep neural networks**

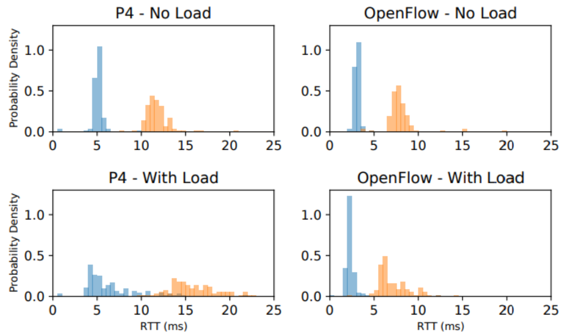




Example: Macchiato Side Channels

- Side channel *without malicious switches*: just between hosts
- Also leverages MAC learning for *mobility applications*: the fact that old rules associated with a MAC address are cleared when they marked in a new location
- The switch connected to the red host contacts the controller to report a new MAC address discovery. The controller interprets this that the blue host migrated to a new location and performs flow reconfigurations. Introduces *measurable differences* to the blue host that can be used repeatedly by the red host to modulate confidential data.
- Throughput can be improved with **deep neural networks**



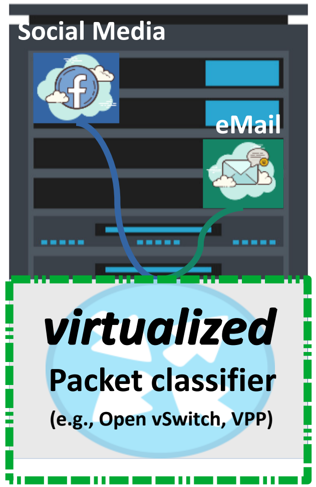


Macchiato: Importing Cache Side Channels to SDNs (Best Paper Award). Amir Sabzi, Liron Schiff, Kashyap Thimmaraju, Andreas Blenk, and Stefan Schmid. ANCS 2021.

Another Challenge: Algorithmic Complexity Attacks

Algorithmic Complexity Attacks

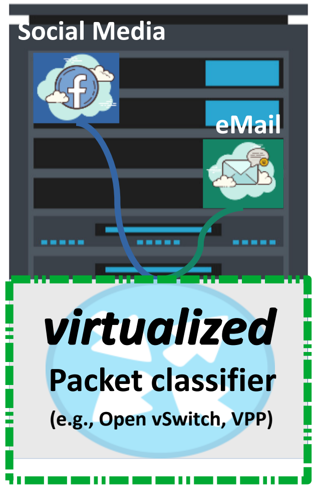
- Network dataplane runs many **complex algorithms**: may perform poorly under specific or *adversarial inputs*
- E.g., packet classifier: runs **Tuple Space Search** algorithm (e.g., in OVS)
- Can be exploited: adversary can *degrade performance* to ~10% of the baseline (10 Gbps) with only <1 Mbps (!) attack traffic
- Idea:
  - Tenants can use the Cloud Management System (CMS) to set up their **ACLs** to access-control, redirect, log, etc.
  - Attacker's goal: send some *packet towards the virtual switch* that when subjected to the ACLs will *exhaust resources*



Tuple Space Explosion: A Denial-of-Service Attack Against a Software Packet Classifier. Levente Csikor et al. ACM CoNEXT, 2019.

Algorithmic Complexity Attacks

- Network dataplane runs many **complex algorithms**: may perform poorly under specific or *adversarial inputs*
- E.g., packet classifier: runs **Tuple Space Search** algorithm (e.g., in OVS)
- Can be exploited: adversary can *degrade performance* to ~10% of the baseline (10 Gbps) with only <1 Mbps (!) attack traffic
- Idea:
  - Tenants can use the Cloud Management System (CMS) to set up their **ACLs** to access-control, redirect, log, etc.
  - Attacker's goal: send some *packet towards the virtual switch* that when subjected to the ACLs will *exhaust resources*



Use AI to find such attacks?!

Tuple Space Explosion: A Denial-of-Service Attack Against a Software Packet Classifier. Levente Csikor et al. ACM CoNEXT, 2019.

Conclusion

- Can we trust our networks today? Challenges, due to complexity, **security assumptions** and lack of tools

- Opportunities of emerging network technologies
  - Automation and programmability: new tools and improved **network monitoring**

- Challenges of emerging network technologies
  - New threat models: e.g., ***propagate*** worm in datacenter
  - Algorithmic complexity attacks: e.g., make virtual switch ***crawl***

תודה רבה

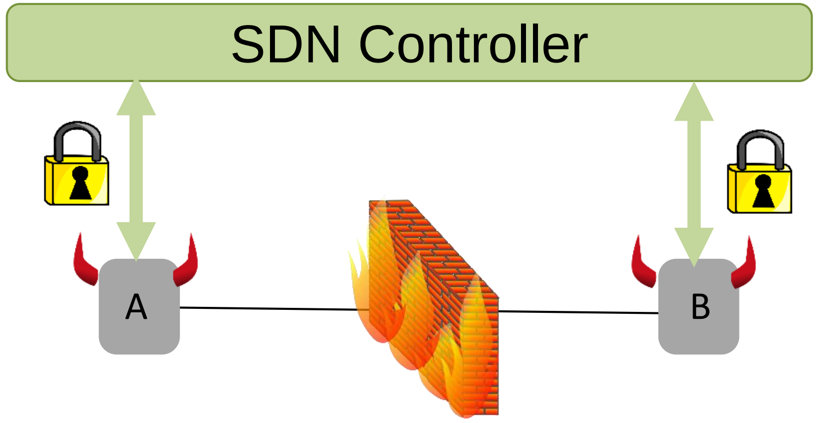
Toda raba!

Further Reading

P-Rex: Fast Verification of MPLS Networks with Multiple Link Failures
Jesper Stenbjerg Jensen, Troels Beck Krogh, Jonas Sand Madsen, Stefan Schmid, Jiri Srba, and Marc Tom Thorgersen.
14th International Conference on emerging Networking EXperiments and Technologies (**CoNEXT**), Heraklion, Greece, December 2018.
Macchiato: Importing Cache Side Channels to SDNs (Best Paper Award)
Amir Sabzi, Liron Schiff, Kashyap Thimmaraju, Andreas Blenk, and Stefan Schmid.
16th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (**ANCS**), Virtual Conference, December 2021.
NetBOA: Self-Driving Network Benchmarking
Johannes Zerwas, Patrick Kalmbach, Laurenz Henkel, Gabor Retvari, Wolfgang Kellerer, Andreas Blenk, and Stefan Schmid.
ACM SIGCOMM Workshop on Network Meets AI & ML (**NetAI**), Beijing, China, August 2019.
MTS: Bringing Multi-Tenancy to Virtual Switches
Kashyap Thimmaraju, Saad Hermak, Gabor Retvari, and Stefan Schmid.
USENIX Annual Technical Conference (**ATC**), Renton, Washington, USA, July 2019.
Taking Control of SDN-based Cloud Systems via the Data Plane (Best Paper Award)
Kashyap Thimmaraju, Bhargava Shastry, Tobias Fiebig, Felicitas Hetzelt, Jean-Pierre Seifert, Anja Feldmann, and Stefan Schmid.
ACM Symposium on SDN Research (**SOSR**), Los Angeles, California, USA, March 2018.
Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
2nd IEEE European Symposium on Security and Privacy (**EuroS&P**), Paris, France, April 2017.
Preacher: Network Policy Checker for Adversarial Environments
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
38th International Symposium on Reliable Distributed Systems (**SRDS**), Lyon, France, October 2019.

New Types of Attacks: Via SDN Controller

- **Controller** may be attacked or exploited

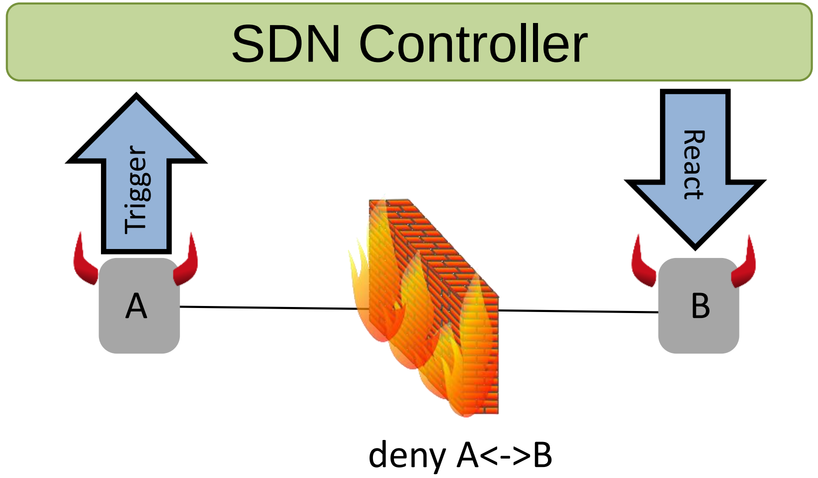


deny A<->B

Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and S.
EuroS&P, Paris, France, April 2017.

New Types of Attacks: Via SDN Controller

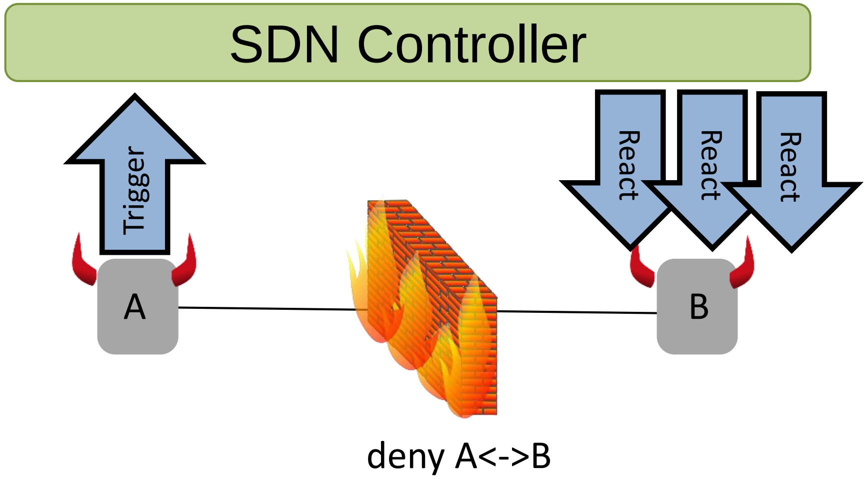
- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs



Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and S.
EuroS&P, Paris, France, April 2017.

New Types of Attacks: Via SDN Controller

- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs
  - Or even *multicast*: **pave-path technique** more efficient than hop-by-hop

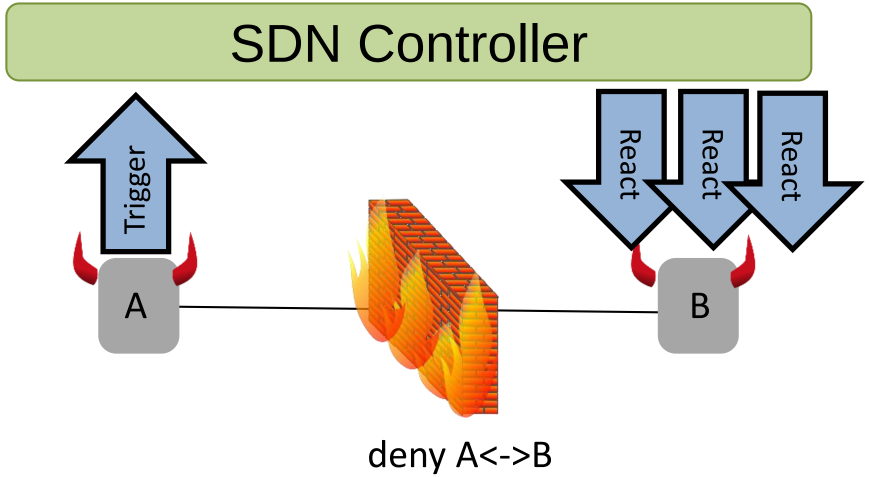


Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and S.
EuroS&P, Paris, France, April 2017.

New Types of Attacks: Via SDN Controller

- **Controller** may be attacked or exploited
  - By design, *reacts* to switch events, e.g., by packet-outs
  - Or even *multicast*: **pave-path technique** more efficient than hop-by-hop

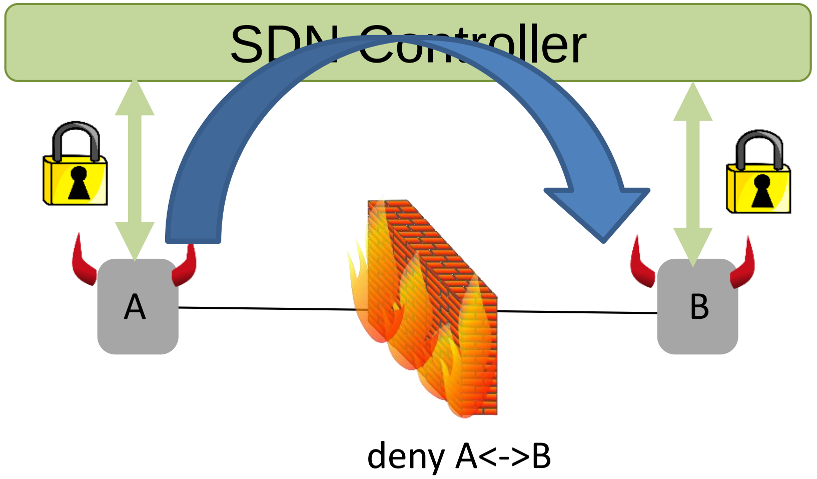
May introduce ***new communication paths*** which can be used in unintendend ways!



Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

New Types of Attacks: Via SDN Controller

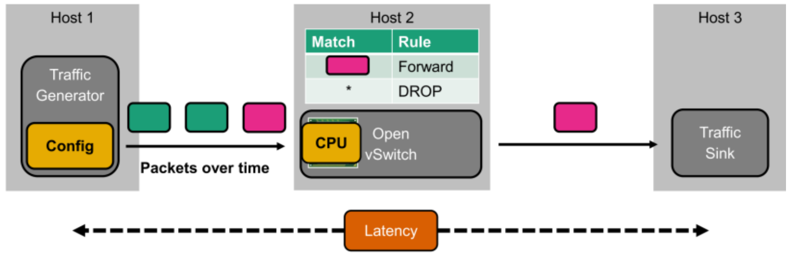
- In particular: new **covert communication** channels
  - E.g., exploit MAC learning (use codeword „0xBADDAD“) or modulate information with timing
- May *bypass security-critical elements*: e.g., firewall in the dataplane
- *Hard to catch*: along „normal communication paths“ and encrypted



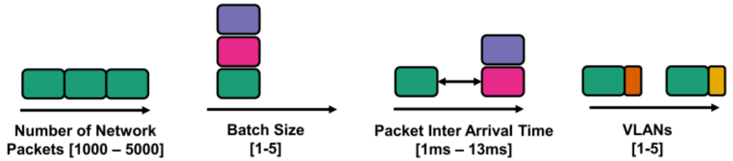
Outsmarting Network Security with SDN Teleportation
Kashyap Thimmaraju, Liron Schiff, and Stefan Schmid.
EuroS&P, Paris, France, April 2017 + ***CVEs***.

NetBOA: Automated Performance Benchmarking

- Idea: *automate*! Generate different input, measure impact (e.g., latency)
  - Similar to *fuzzing*



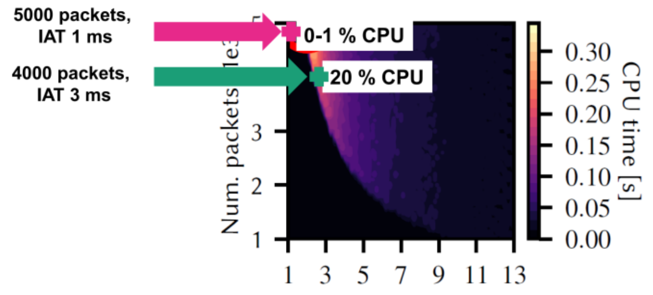
- Different dimensions:
  - Packet size, inter-arrival time, packet type, etc.


Number of Network Packets [1000 – 5000]
Batch Size [1-5]
Packet Inter Arrival Time [1ms – 13ms]
VLANs [1-5]

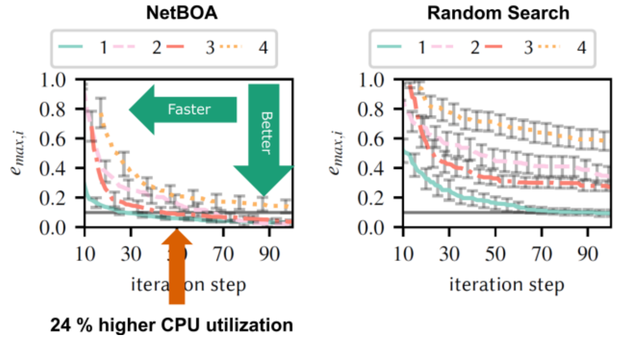
NetBOA: Self-Driving Network Benchmarking
Zerwas et al. ACM SIGCOMM Workshop on Network Meets AI & ML (NetAI), Beijing, China, August 2019.

Baysian Optimization Approach

- Complex systems (such as vSwitch) have complex behavior: e.g., sometimes sending less packets increases CPU load
  - Hard to find for humans



- Baysian optimization much faster than random baseline


NetBOA
Random Search
Faster
Better
iteration step
24 % higher CPU utilization