SPONGE : Software-Defined Traffic Engineering to Absorb Influx of Network Traffic

Benoit Henry, **Shihabur R. Chowdhury,** Abdelkader Lahmadi, Romain Azais, Jérôme François, and Raouf Boutaba

IMT Lille Douai
École Mines-Télécom
IMT-Université de Lille

UNIVERSITY OF WATERLOO

UNIVERSITÉ DE LORRAINE

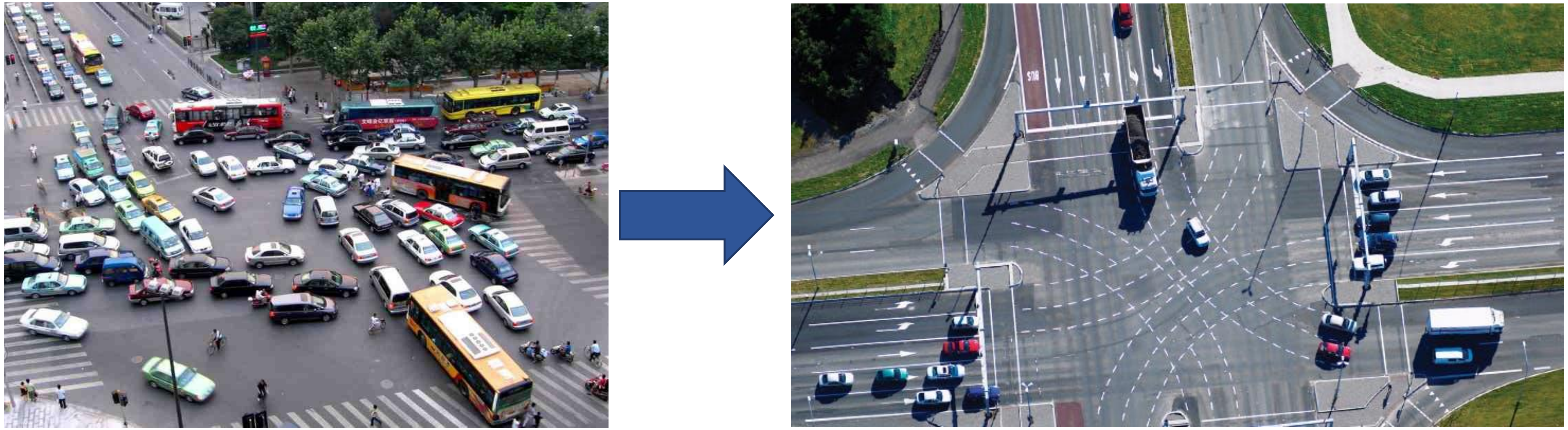
Inria

Traffic Engineering

The art of assigning (network) traffic to paths while optimizing certain objective(s) (*e.g.*, minimize congestion, maximize residual capacity)

Traffic Engineering

The art of assigning (network) traffic to paths while optimizing certain objective(s) (*e.g.*, minimize congestion, maximize residual capacity)



State-of-the-art: Birds eye view

Static & Load-unaware (pre-SDN era)	OSPF, MPLSE-TE, ECMP
Topology-specific (post-SDN era)	DC – Hedera[1], MicroTE[2] WAN - B4[3], SWAN[4]
Event specific	Congestion & Failure events; Attack events (*e.g.*, link-flooding attack)

[1]M. Al-Fares, *et al.* "Hedera: dynamic flow scheduling for data center networks." In Proc. of NSDI 2010.
[2]T. Benson, *et al.* "MicroTE: Fine grained traffic engineering for data centers." In Proc. of ACM CoNeXT 2011.
[3]S. Jain, *et al.* "B4: Experience with a globally-deployed software defined WAN." In Proc. of ACM SIGCOMM 2013.
[4]C. Hong, *et al.* "Achieving high utilization with software-driven WAN." In Proc. of ACM SIGCOMM 2013.

Our Contribution: **SPONGE**

A traffic engineering mechanism not specific to any network topology, traffic pattern, objective function, and network events

SPONGE : Overview

Topology agnostic — The network is modeled as a graph with system of queues on link end-points

Traffic matrix & network event agnostic — Network dynamics is characterized by a stochastic process on queues

Objective function agnostic — A pluggable objective function supports different operational policies

SPONGE: Network Model

The network is modeled as a graph with a system of queues and a routing table at each node

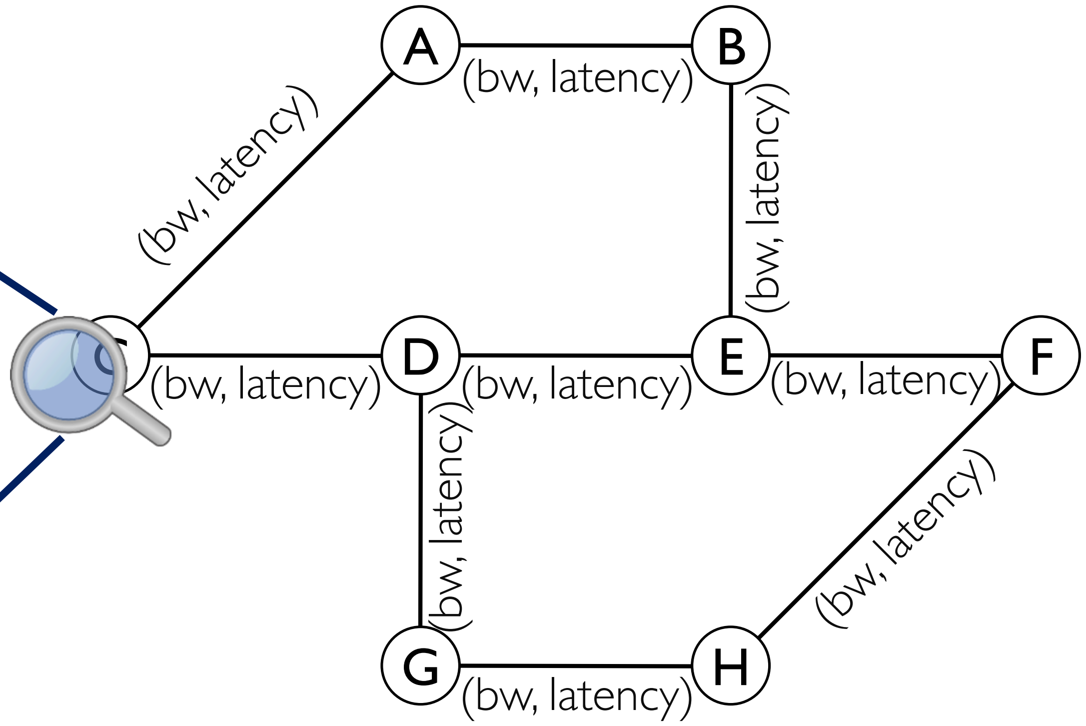
Routing Table

Dest.	Next
A	A
F	D
…	…
…	…
H	A

Queue of packets at time t on node C

X_t^C	A	E	.	.	.	H

Queue contains packet destinations



SPONGE: Network Model

The network is modeled as a graph with a system of queues and a routing table at each node

Result of optimization

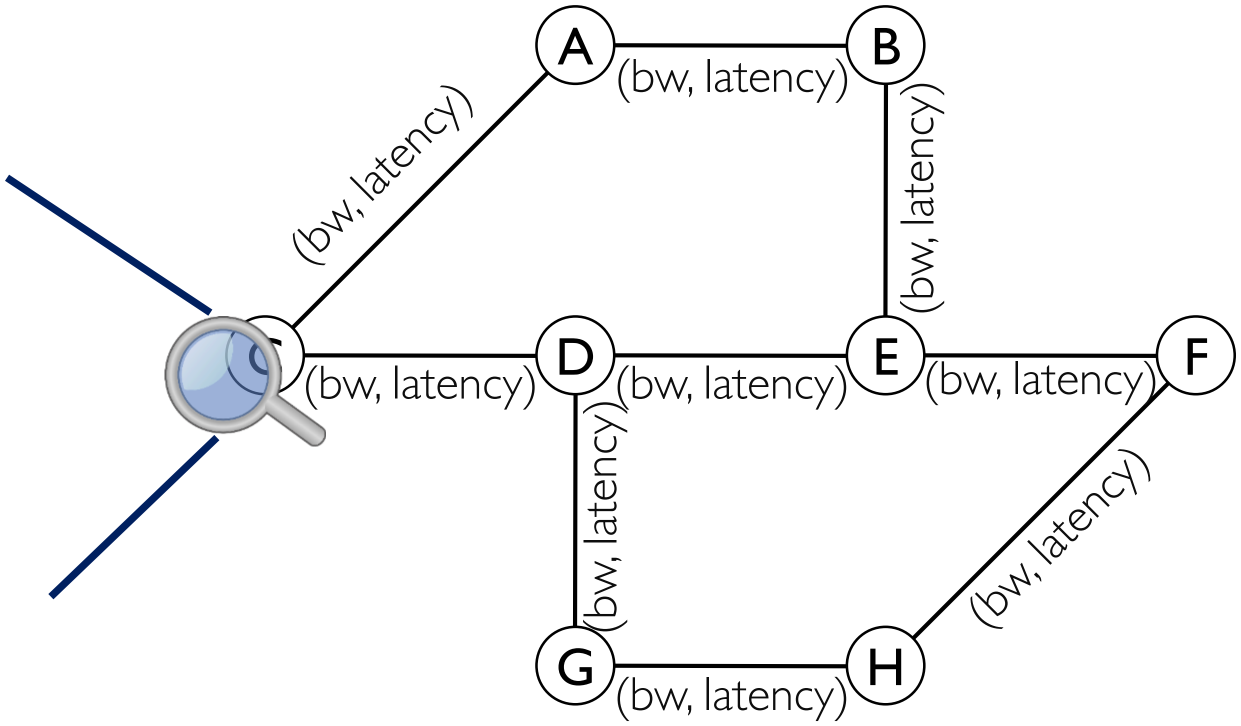
Routing Table

Dest.	Next
A	A
F	D
…	…
…	…
H	A

Queue of packets at time t on node C

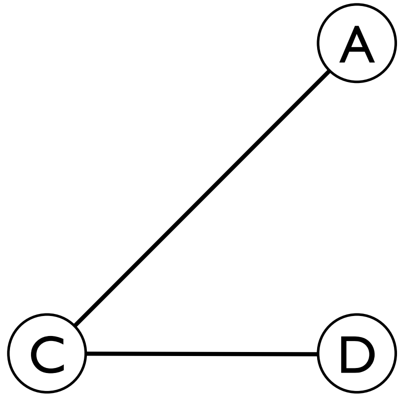
X_t^C | A | E | . | . | . | H |

Queue contains packet destinations



SPONGE: Network Dynamics

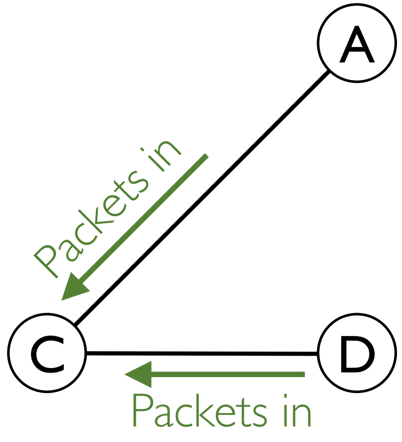
Network dynamics is modeled as variation of the queues at network nodes



SPONGE: Network Dynamics

Network dynamics is modeled as variation of the queues at network nodes

$\delta^{+}X_t^C$ Positive variation of queue, *i.e.*, packets in (function of negative variation of C's neighbors)

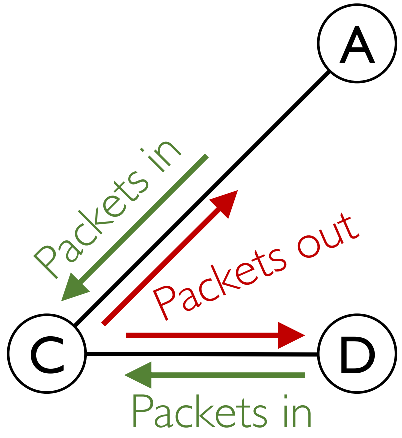


SPONGE: Network Dynamics

Network dynamics is modeled as variation of the queues at network nodes

$\delta^{+} X_t^C$ Positive variation of queue, *i.e.*, packets in (function of negative variation of C's neighbors)

$\delta^{-} X_t^C$ Negative variation of queue, *i.e.*, packets out (function of C's processing time)



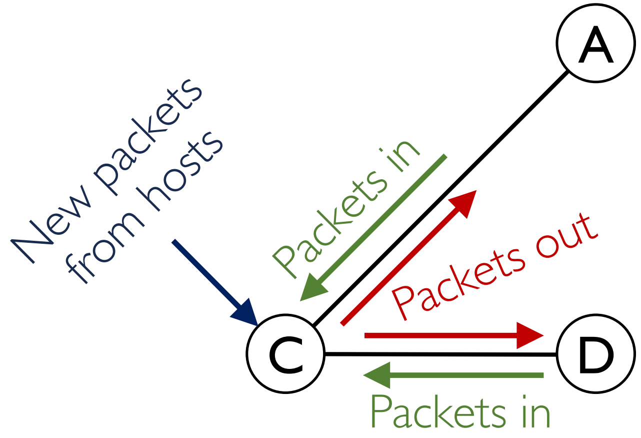
SPONGE: Network Dynamics

Network dynamics is modeled as variation of the queues at network nodes

$\delta^+ X_t^C$ Positive variation of queue, *i.e.*, packets in (function of negative variation of C's neighbors)

$\delta^- X_t^C$ Negative variation of queue, *i.e.*, packets out (function of C's processing time)

(I_k^C, D_k^C) Arrival time & destination of k-th new packet at C



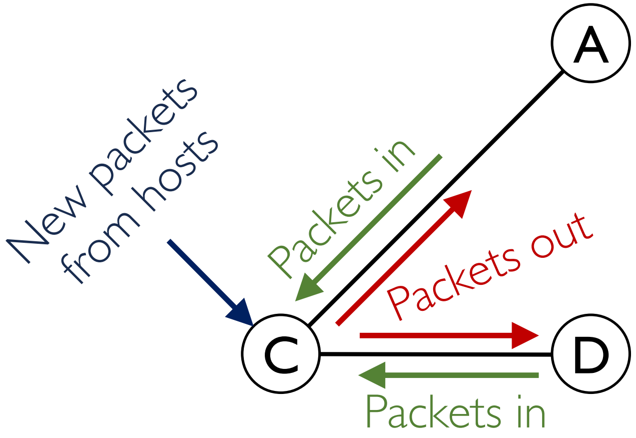
SPONGE: Network Dynamics

Network dynamics is modeled as variation of the queues at network nodes

$\delta^+ X_t^C$ Positive variation of queue, *i.e.*, packets in (function of negative variation of C's neighbors)

$\delta^- X_t^C$ Negative variation of queue, *i.e.*, packets out (function of C's processing time)

(I_k^C, D_k^C) Arrival time & destination of k-th new packet at C



Dynamics of X_t^C after time t $= f(\delta^+ X_t^C, \delta^- X_t^C, (I_k^C, D_k^C))$

SPONGE: Objective

Given a network graph and functions representing variations of queues, compute routing tables that bring the network to a ***healthy state***

SPONGE: Objective

Given a network graph and functions representing variations of queues, compute routing tables that bring the network to a ***healthy state***

How do we quantify ***healthy network state***?

Healthy Network State: Example-I

Example-I: Direct routing potential (H_{route})

$\sum$(distances of the packets in every queue from their destination)

Example-II: Low load potential (H_{load})

$$f\left(\frac{1}{e^{(residual\_queue\_capacity\_at\_all\_nodes)}}\right)$$

Healthy Network State: Example-I

Example-I: Direct routing potential (H_{route})

$\sum$(distances of the packets in every queue from their destination)

Example-II: Low load potential (H_{load})

$$f\left(\frac{1}{e^{(residual\_queue\_capacity\_at\_all\_nodes)}}\right)$$

Healthy Network State: Example-I

Example-I: Direct routing potential (H_{route})

$\sum$(distances of the packets in every queue from their destination)

Our choice

Example-II: Low load potential (H_{load})

Weighted sum of two potentials = $\alpha H_{route} + (1 - \alpha) H_{load}$

$$f\left(\frac{1}{e^{(residual\_queue\_capacity\_at\_all\_nodes)}}\right)$$

SPONGE: Control Optimization

Any numerical optimization method can be used to compute a routing table that minimizes the potential function given the current network status

SPONGE: Control Optimization

Any numerical optimization method can be used to compute a routing table that minimizes the potential function given the current network status

Our choice: *Simulated Annealing*

SPONGE: Simulated annealing

Neighborhood Generation	Metropolis-Hasting algorithm
Fitness Function	Gibbs measure. It inherently prioritizes the low potential states, *i.e.*, healthier network states.

Evaluation

Use Cases — Link-flooding attack; Data-center congestion mitigation

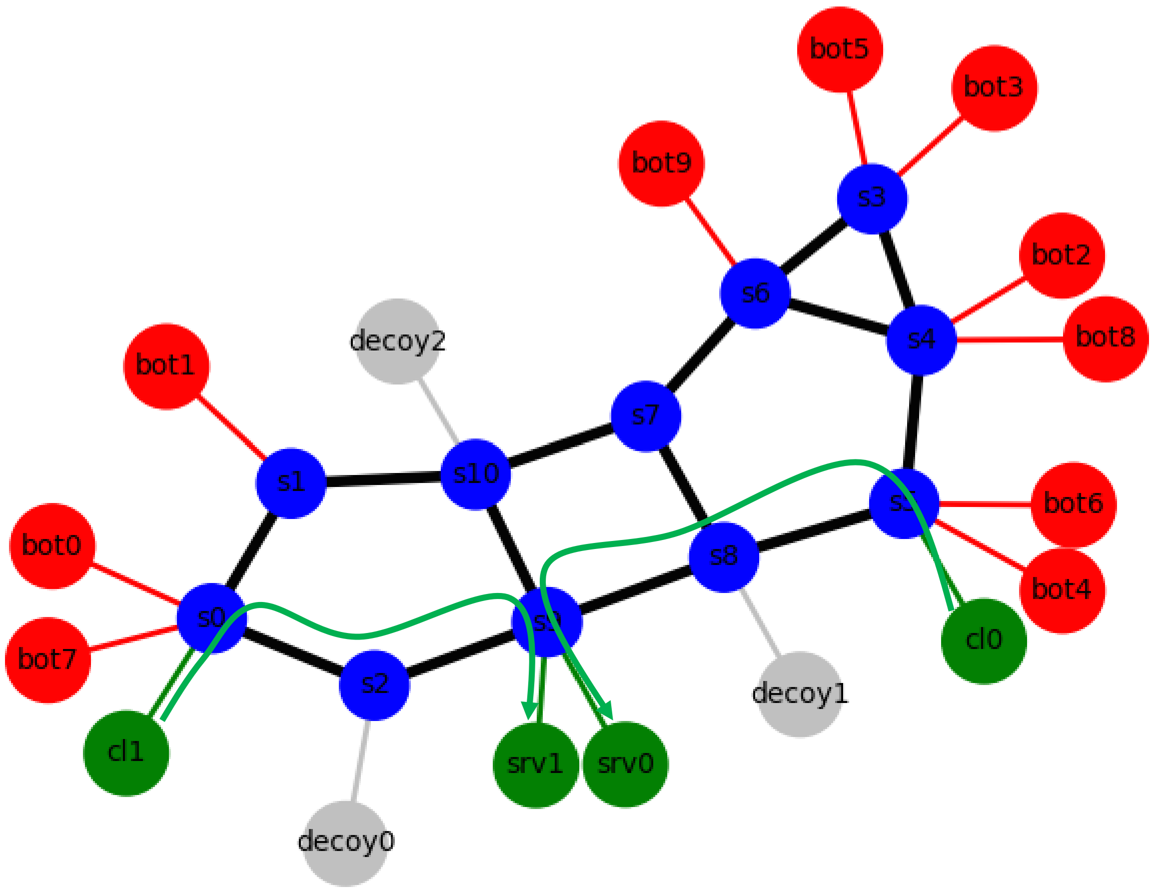
Network Topology — ISP networks (Abilene, Bell Canada), Data center (leaf-spine)

Traffic Pattern — Crossfire attack[1], Many-to-any aggregation traffic pattern in data centers

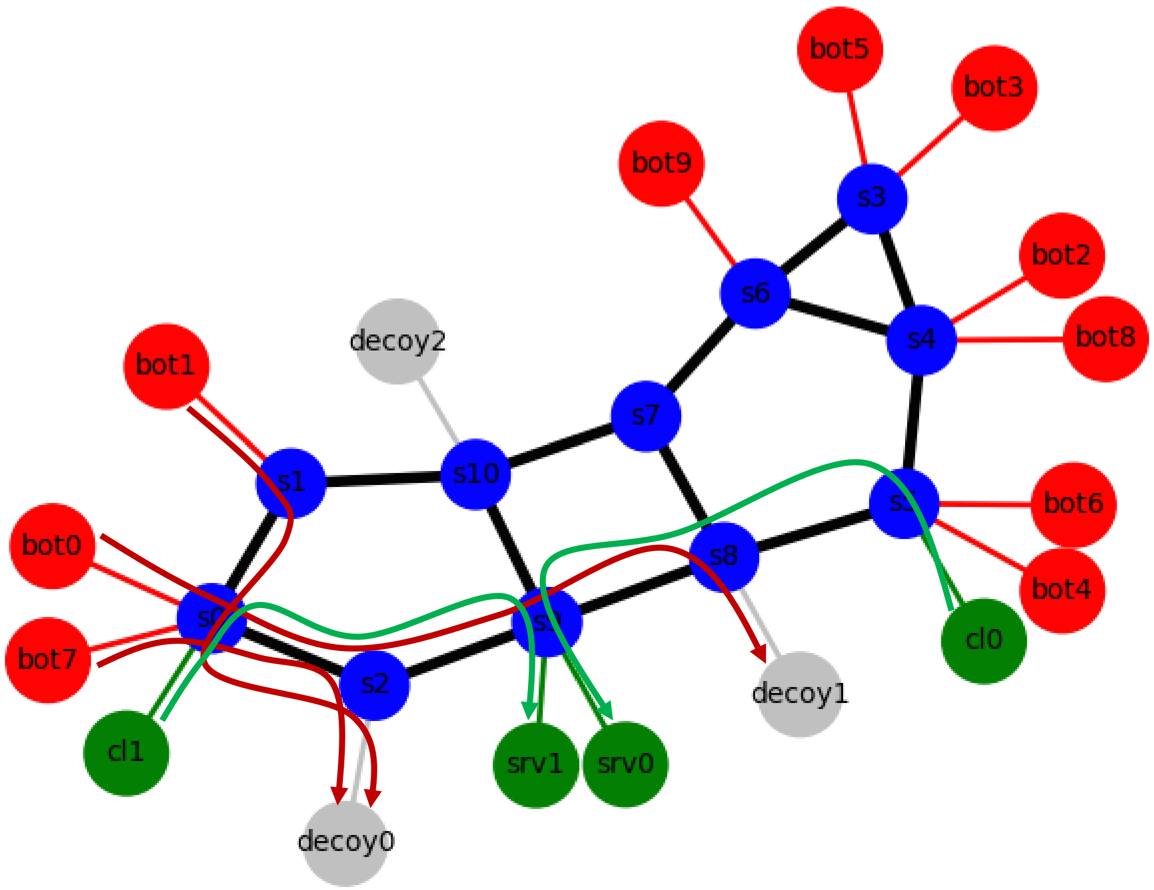
Methodology — Matlab simulation; Mininet emulation

[1] M. S. Kang, *et al.*, "The Crossfire attack," in Proceedings of IEEE S&P, 2013, pp. 127–141.

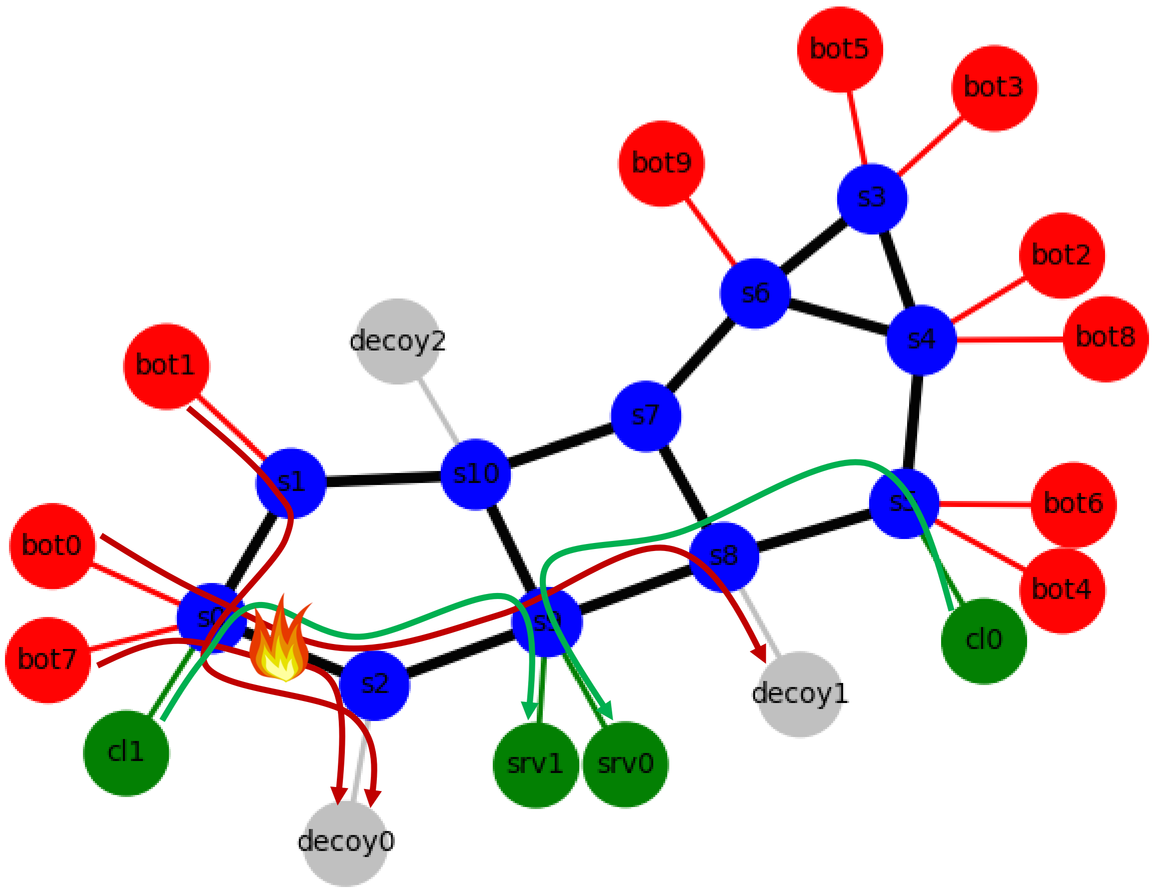
Use-case: Link flooding attack



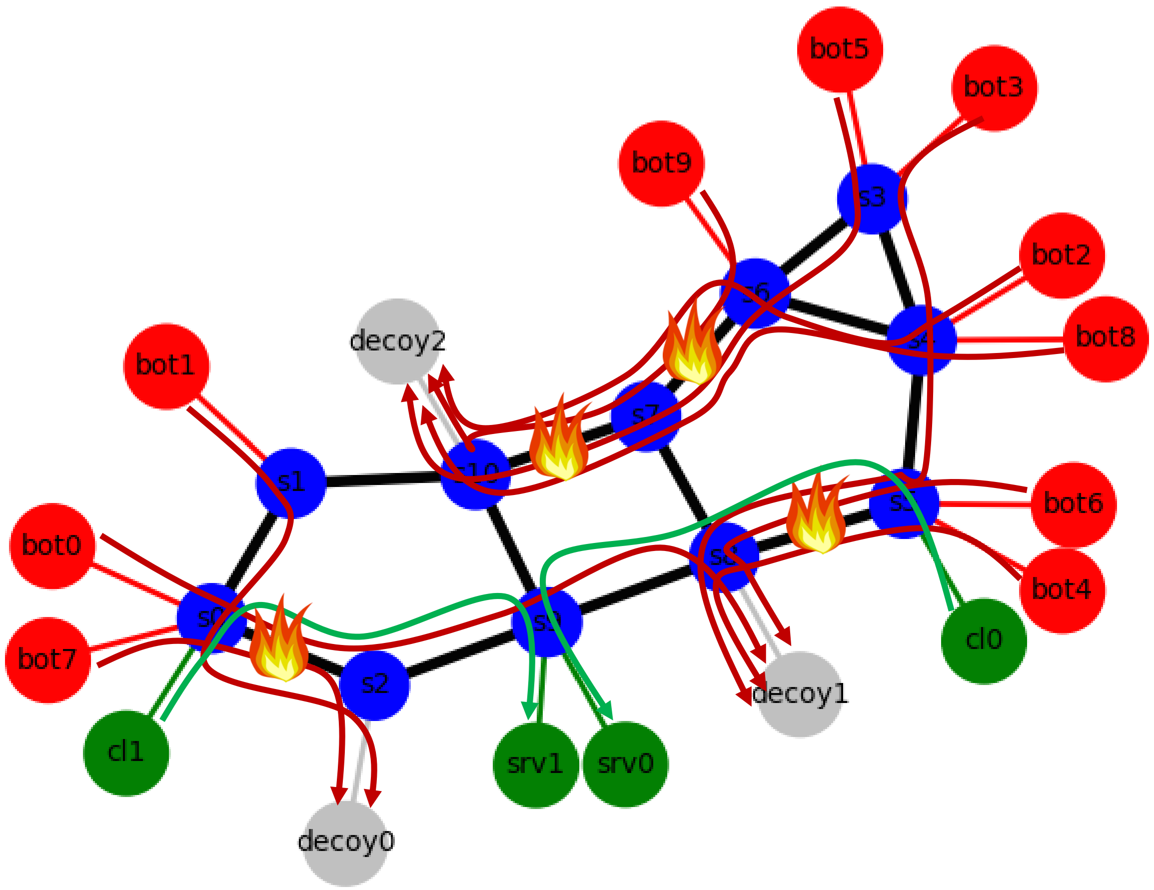
Use-case: Link flooding attack



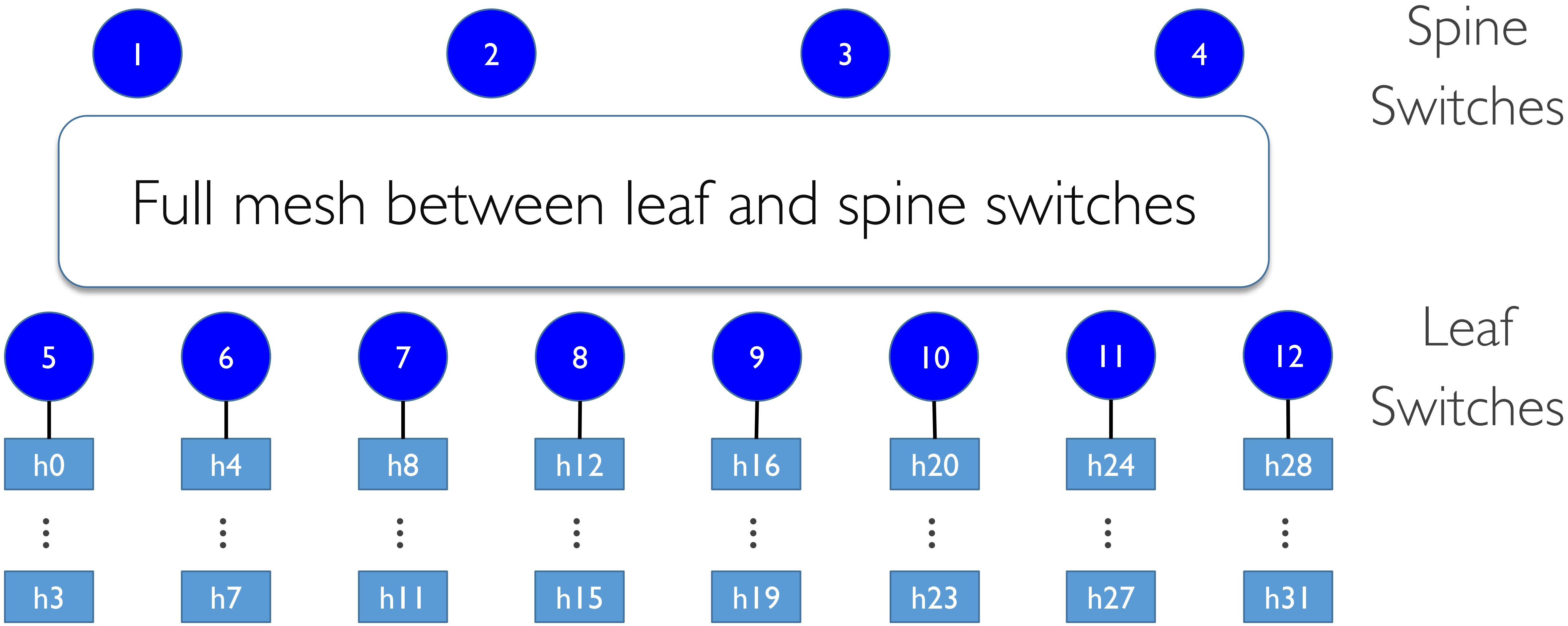
Use-case: Link flooding attack



Use-case: Link flooding attack

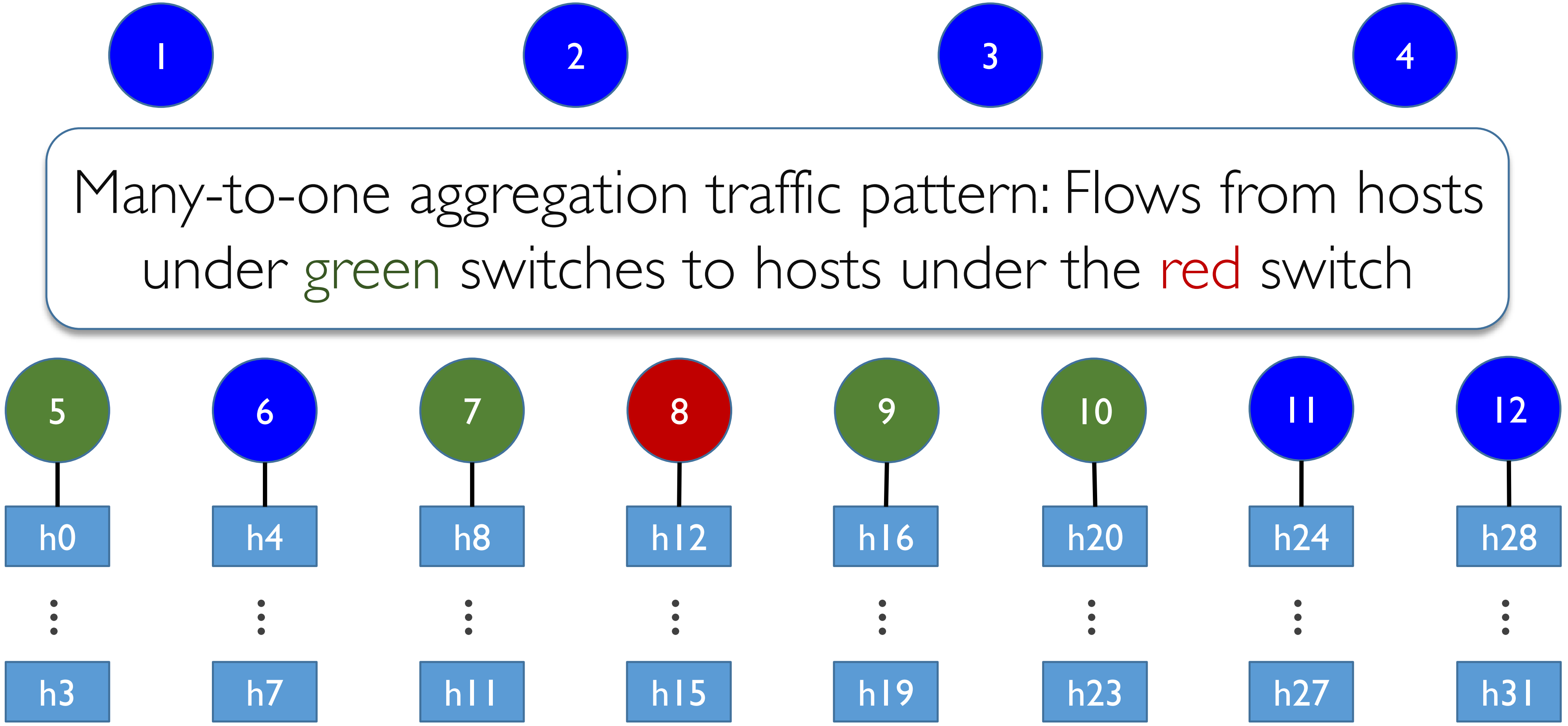


Use-case: Data-center



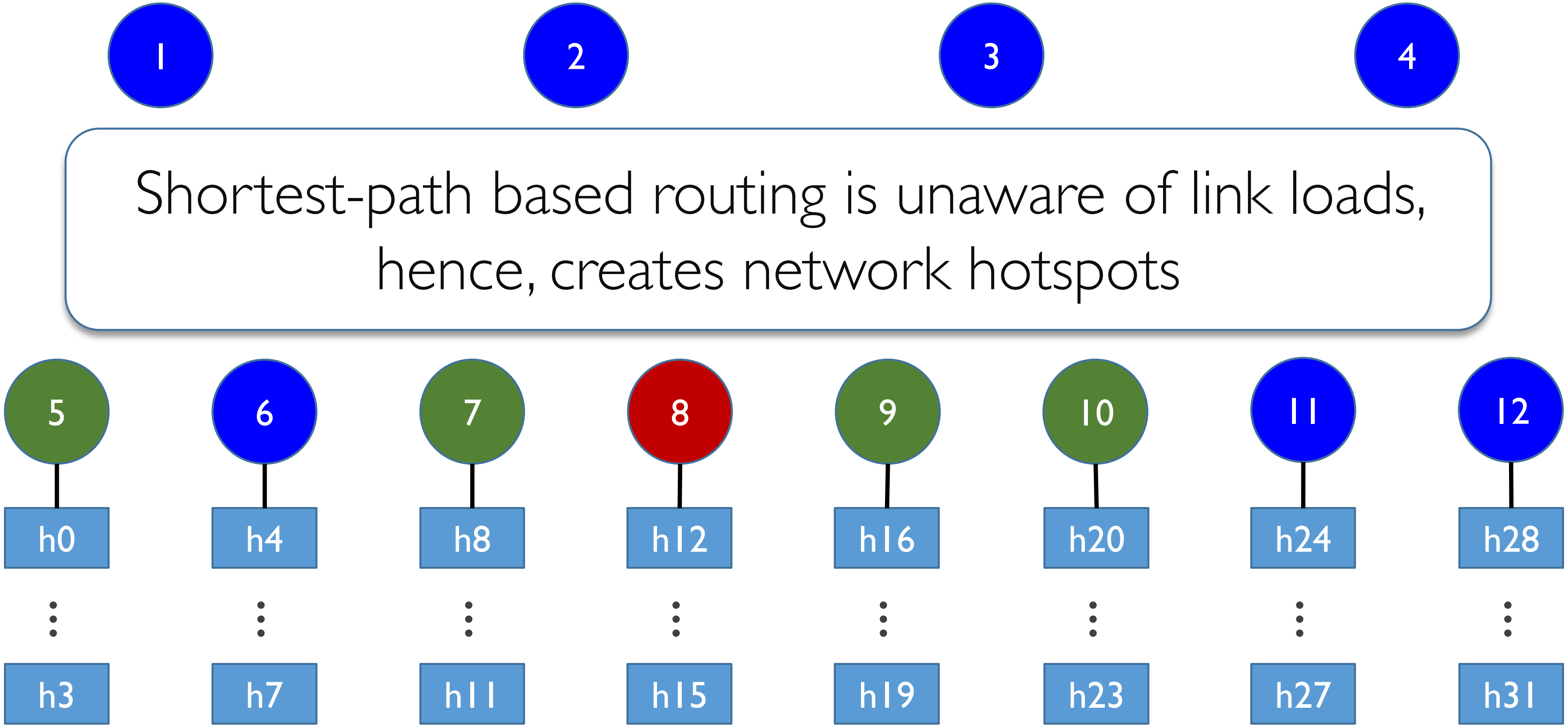
Use-case: Data-center

Many-to-one aggregation traffic pattern: Flows from hosts under green switches to hosts under the red switch

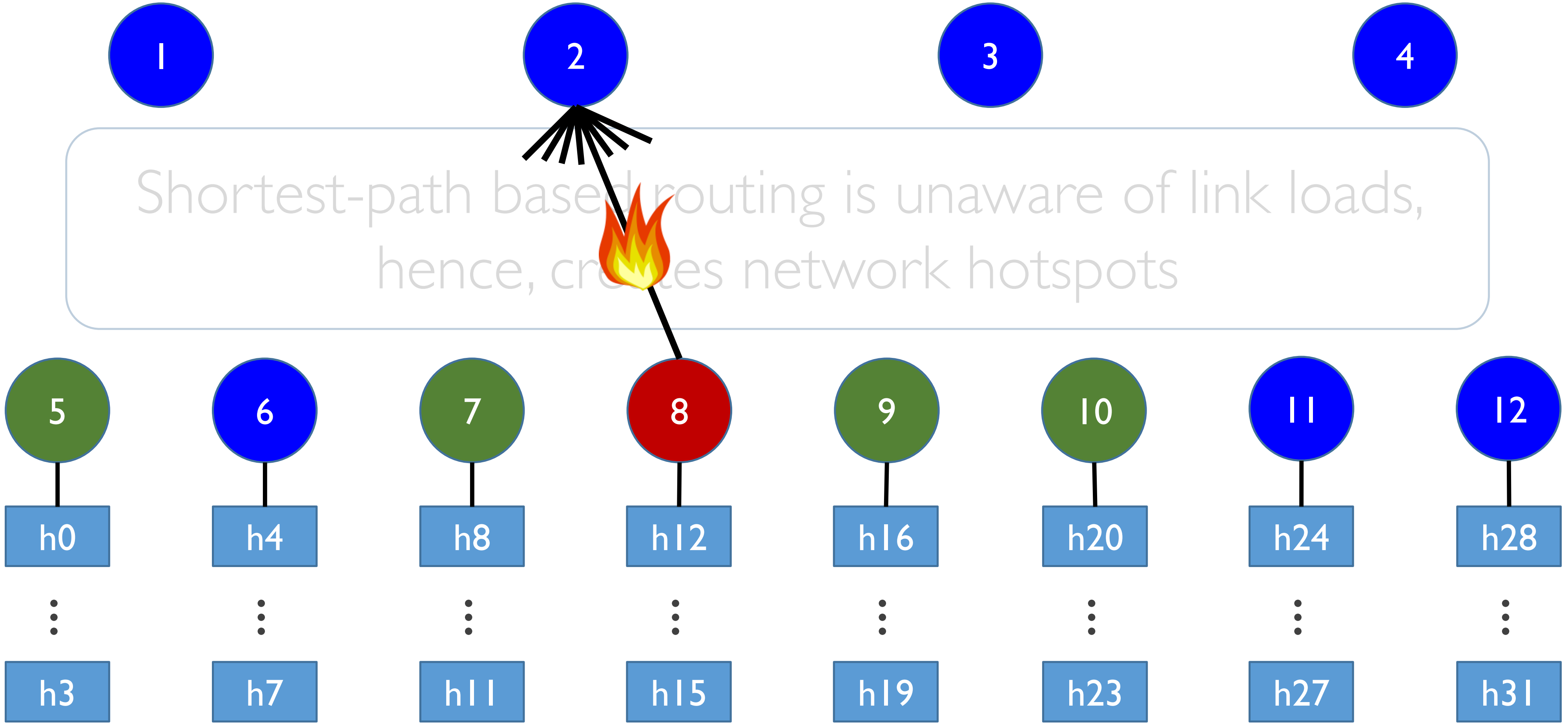


Use-case: Data-center

Shortest-path based routing is unaware of link loads, hence, creates network hotspots



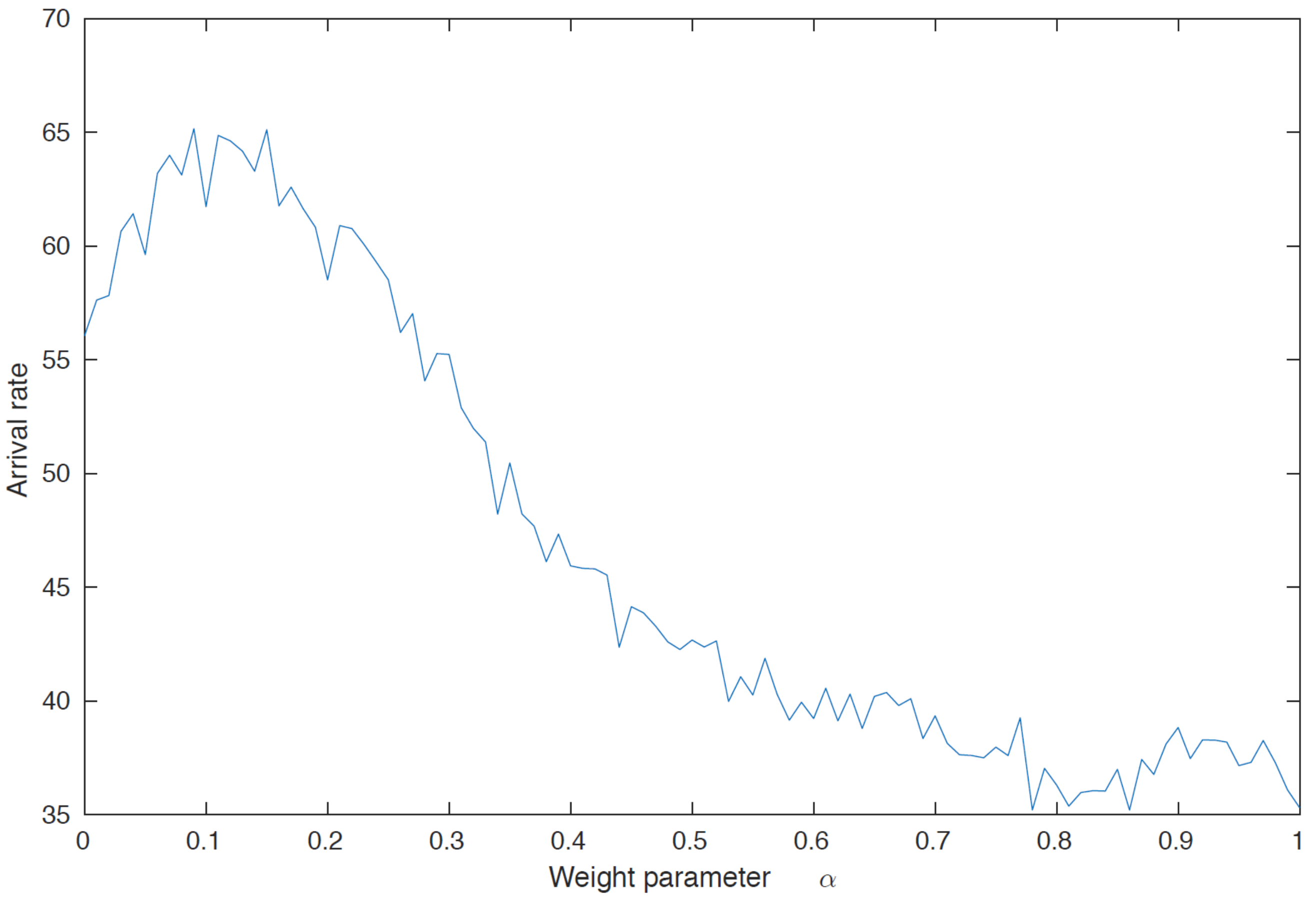
Use-case: Data-center



Matlab Simulation

Packet Arrival	Markovian with random packet sizes
Network Topology	Bell Canada
Processing time	Constant per packet
Use case	Link-flooding attack mitigation

Successful packet delivery



Low load potential ⟷ Direct routing potential

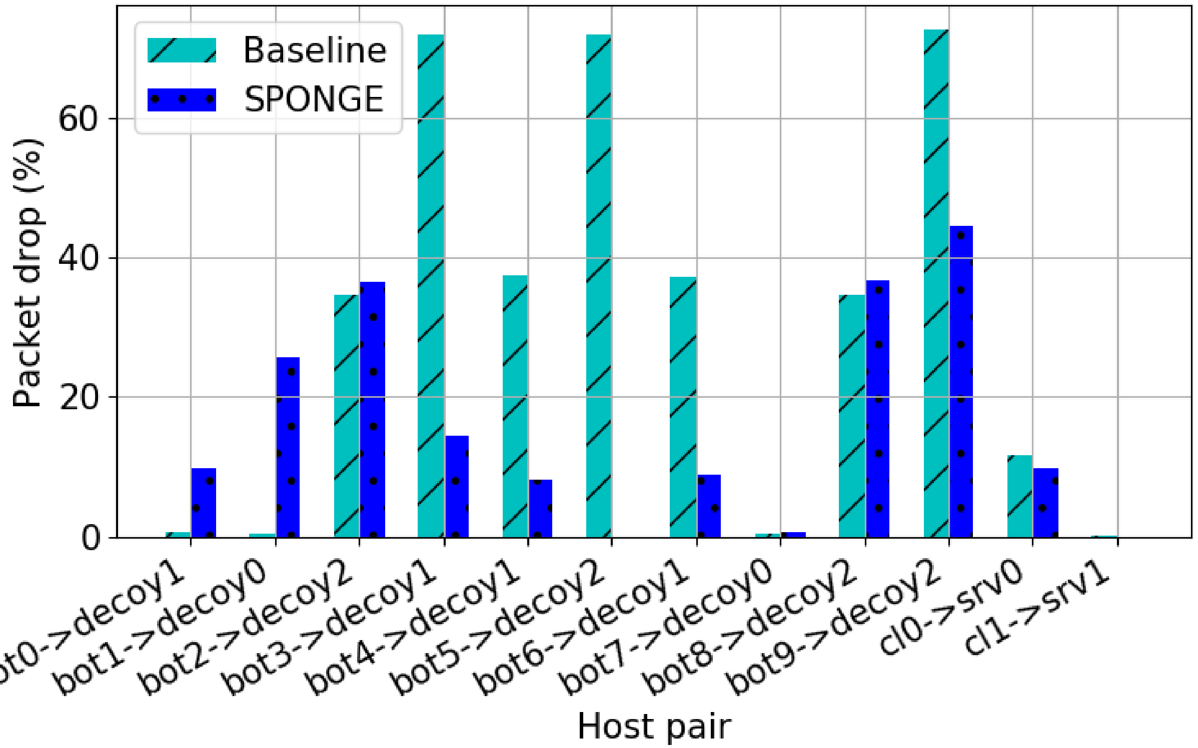
Mininet Emulation

Use Cases: Link-flooding attack, Data-center congestion mitigation

Network Topology: ISP networks (Abilene, Bell Canada), Data center (leaf-spine)

Baseline: Shortest-path based routing

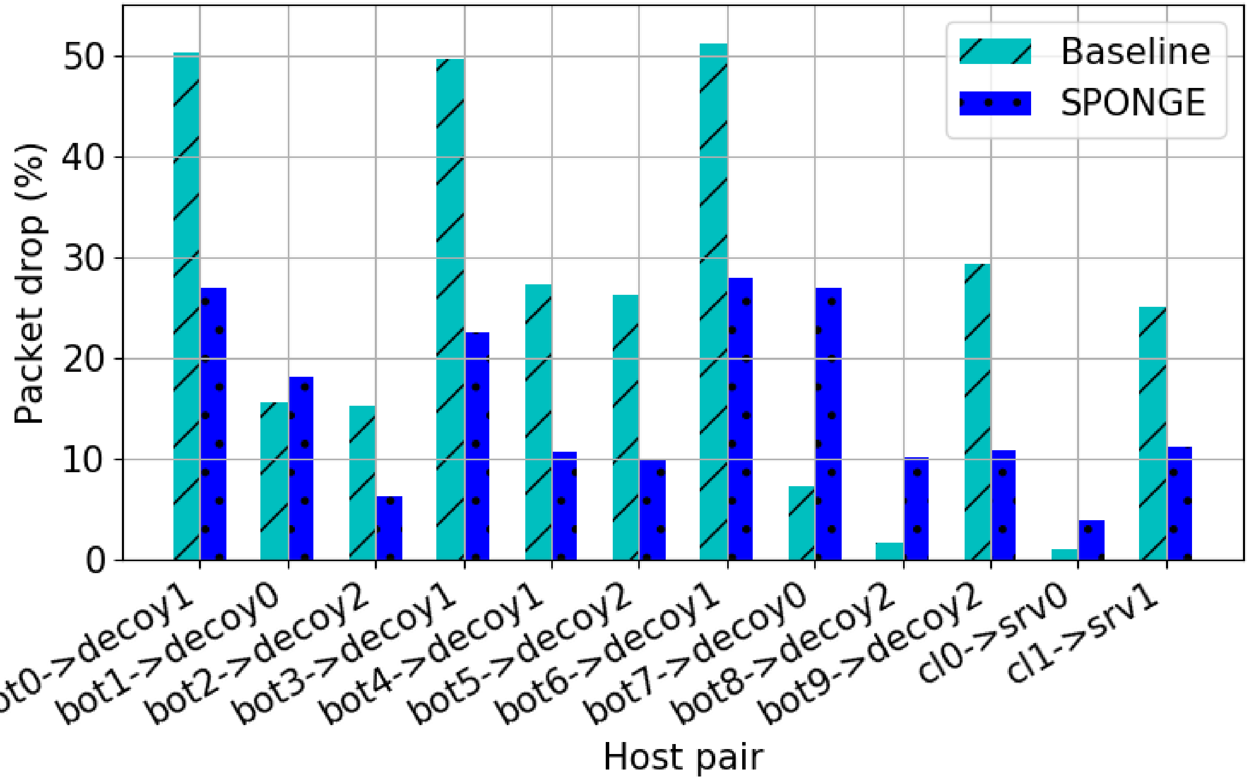
Use-case: Link flooding attack



~50% less packet drops on avg. with **SPONGE**

Abilene topology; Crossfire attack for 30 s

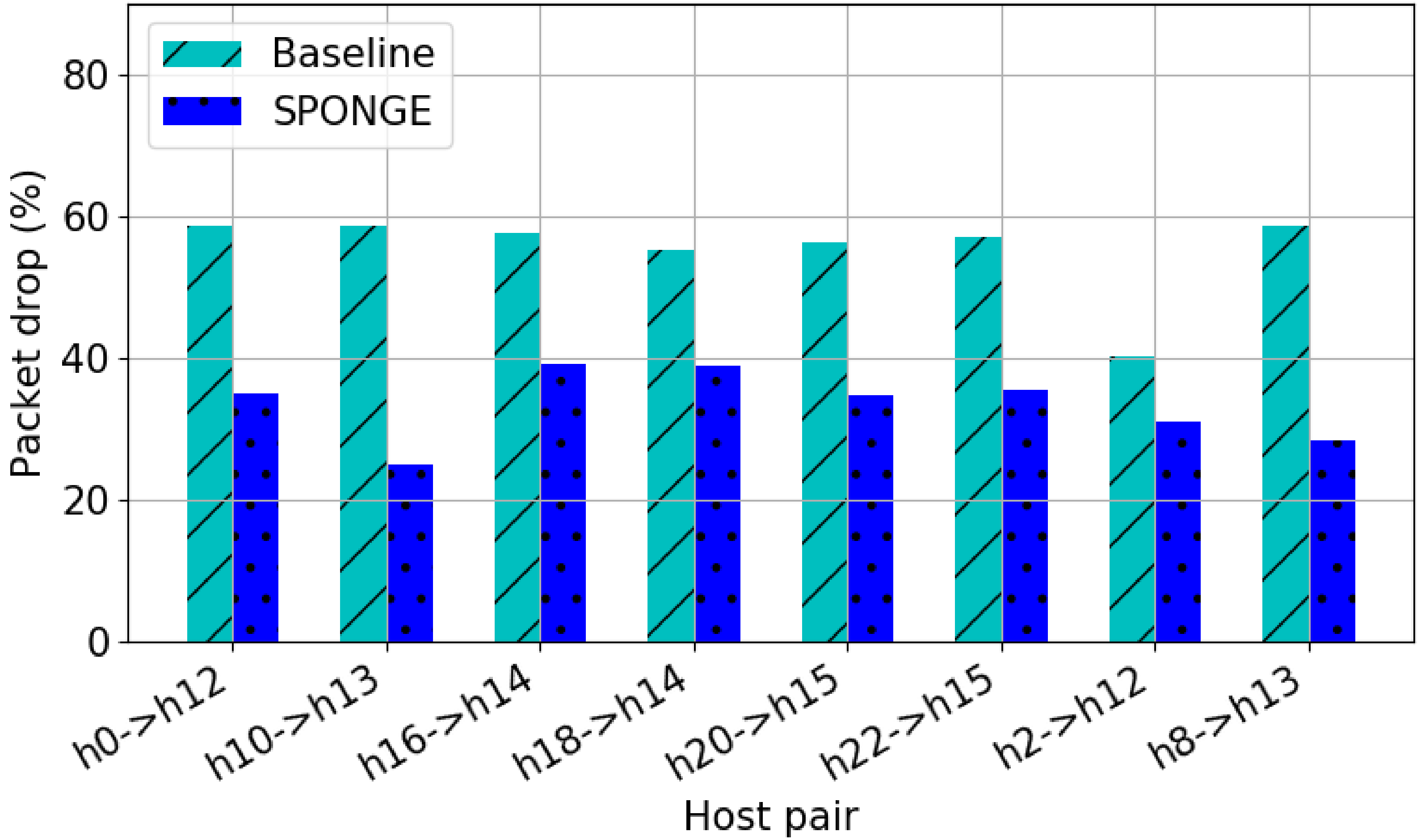
Use-case: Link flooding attack



~40% less packet drops on avg. with **SPONGE**

Bell Canada topology; Crossfire attack for 3 min

Use-case: Data-center



~39.6% less packet drops on avg. with **SPONGE**

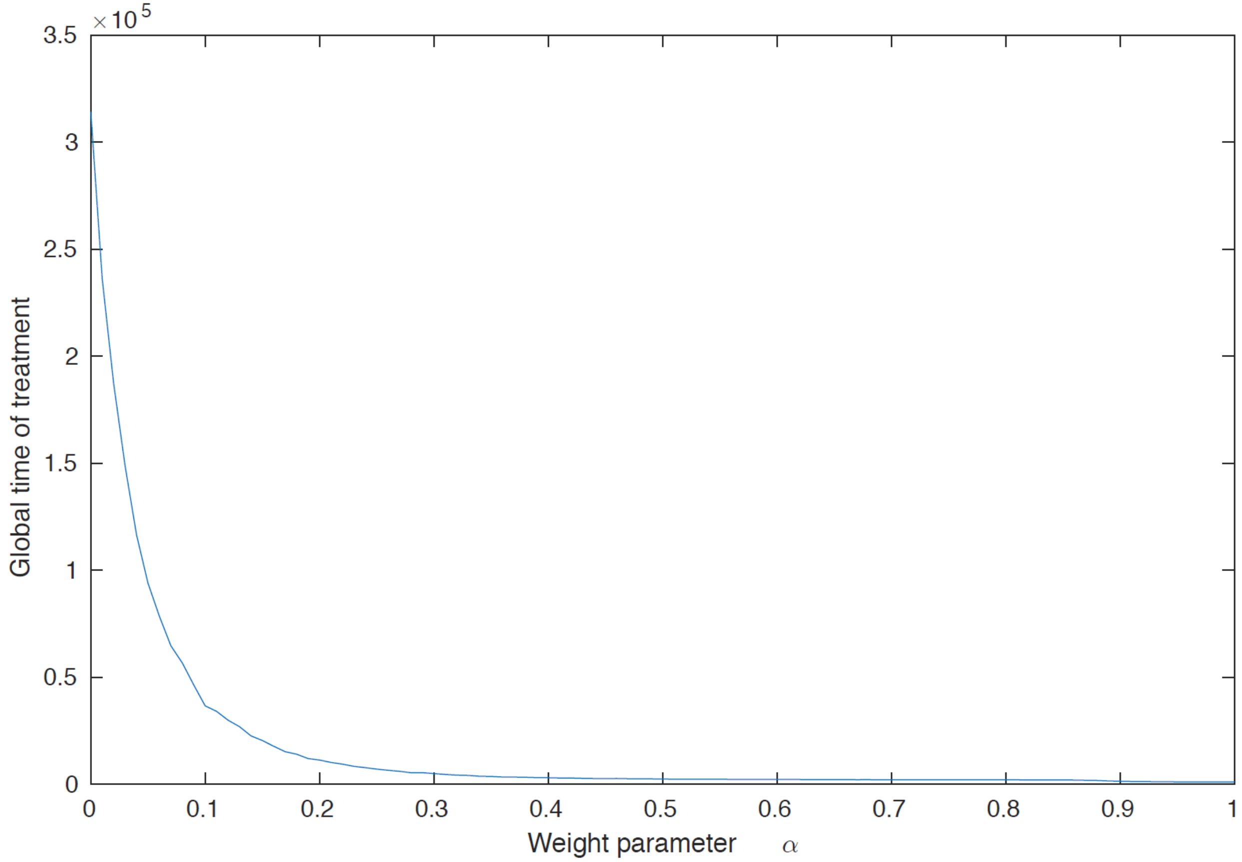
What's Next?

- Consideration for differential traffic classes
- Machine learning to automatically identify traffic classes (*e.g.*, malicious vs benign) and treat them accordingly

Questions?

Backup

Packet delivery time



Low load potential ⟷ Direct routing potential

Impact of α

α	Mean delay experienced by packets (s)
0.1	0.85
0.5	0.70
0.9	0.72

Low load potential → 0.1

Direct routing potential → 0.9