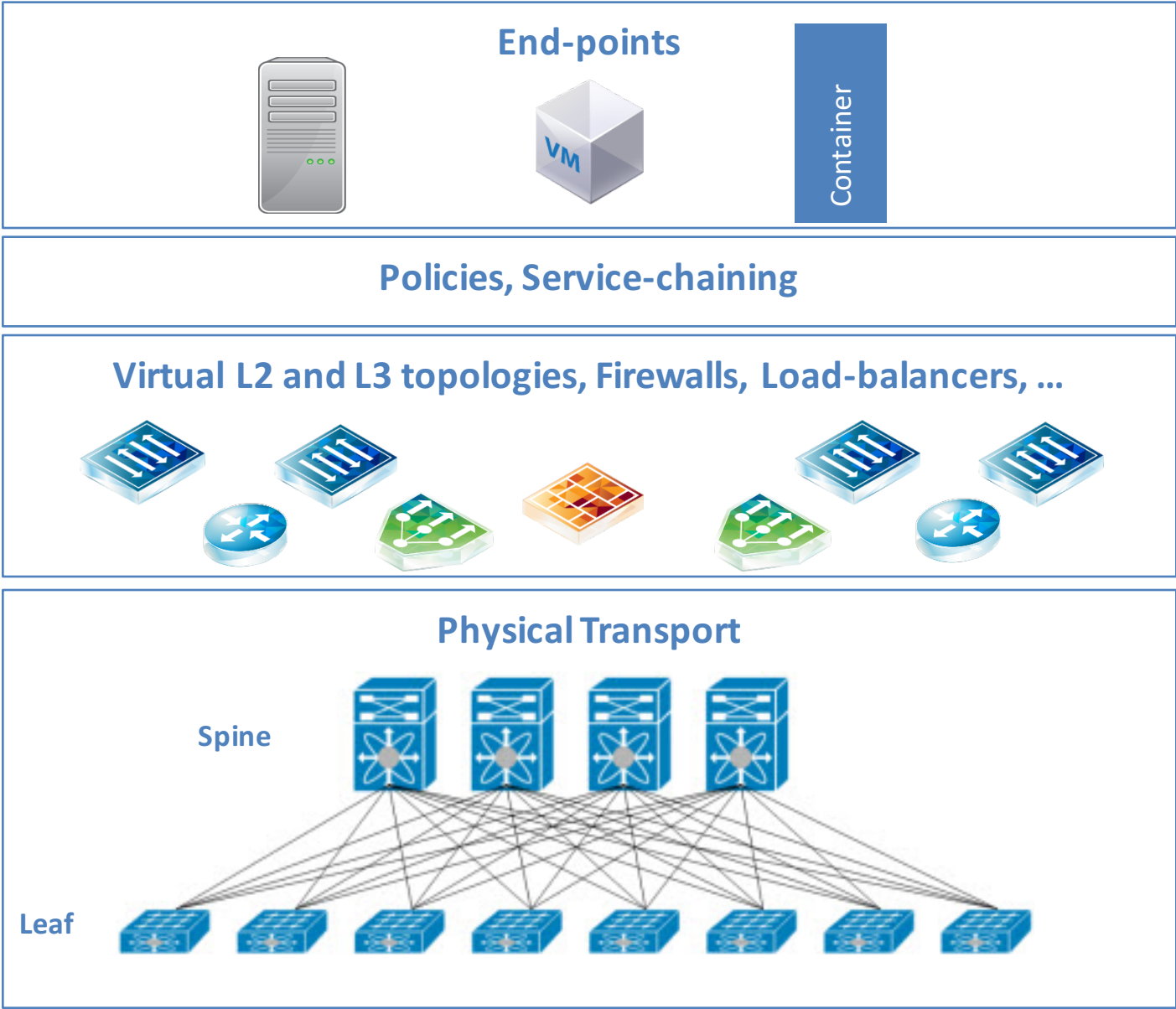


In-band Network Telemetry (INT)

Mukesh Hira, VMware

Naga Katta, Princeton University

Datacenter Network Topologies



Current monitoring methods are inadequate

- Not fast enough
 - Involve CPU and control planes
 - Network state changes rapidly
- Do not provide end-to-end state
 - Difficult to correlate per-element state with the actual path of a flow

INT : In-band Network Telemetry

- Mechanism for collecting network state in the dataplane
 - As close to **realtime** as possible
 - At current and future **line rates**
 - With a framework that can **adapt** over time
- Examples of network state
 - Switch ID, Ingress Port ID, Egress Port ID
 - Egress Link Utilization
 - Hop Latency
 - Egress Queue Occupancy
 - Egress Queue Congestion Status
 -

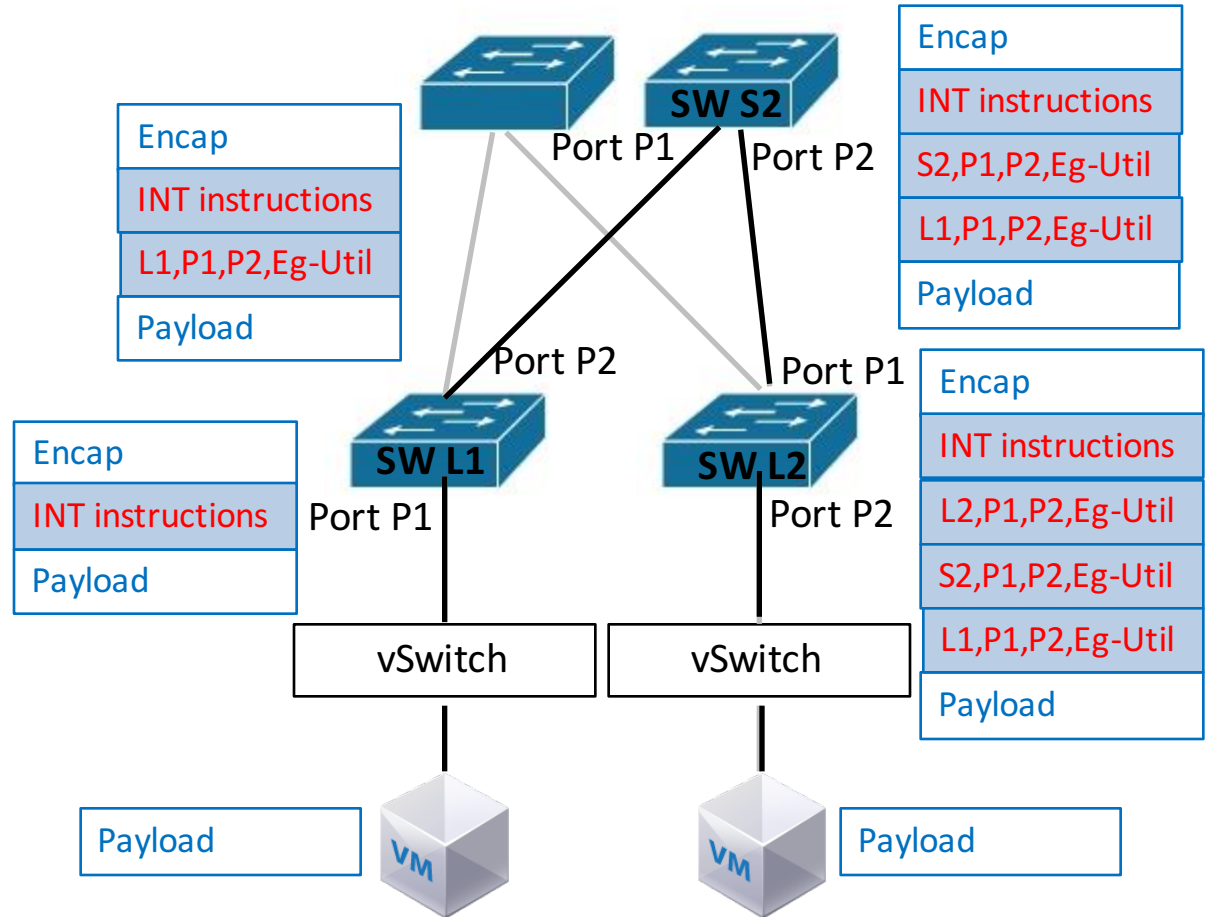
INT Example

Switch ID

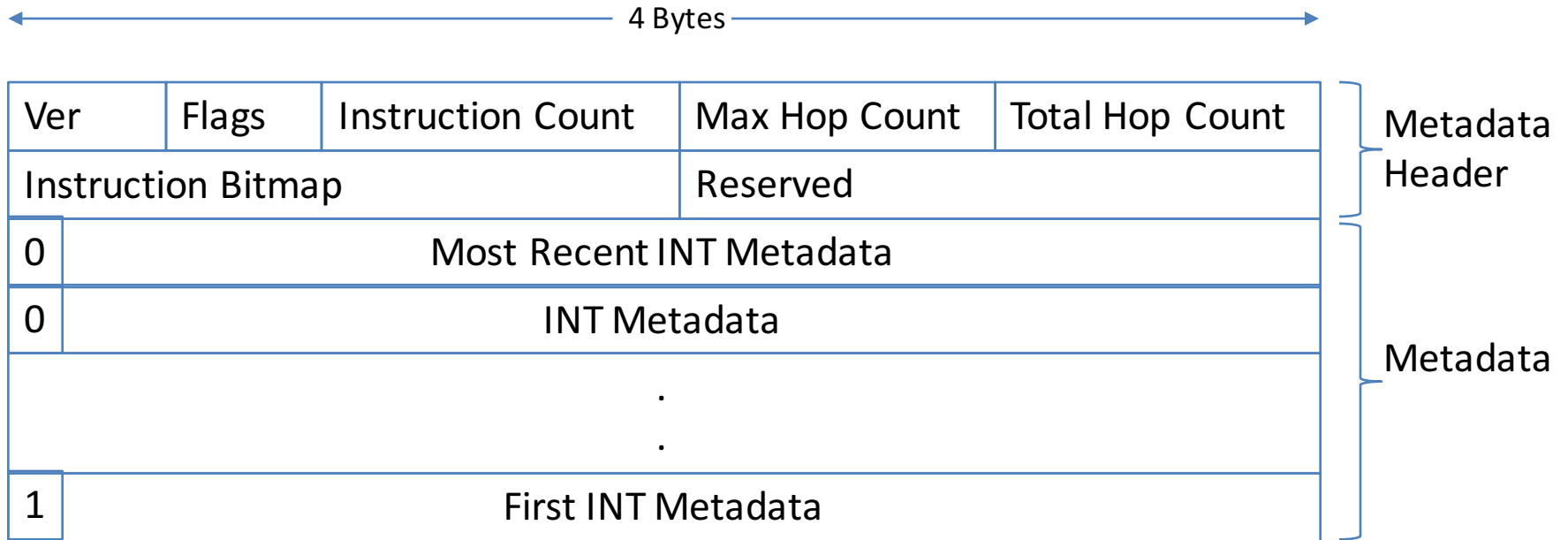
Ingress Port ID

Egress Port ID

Egress Link Utilization

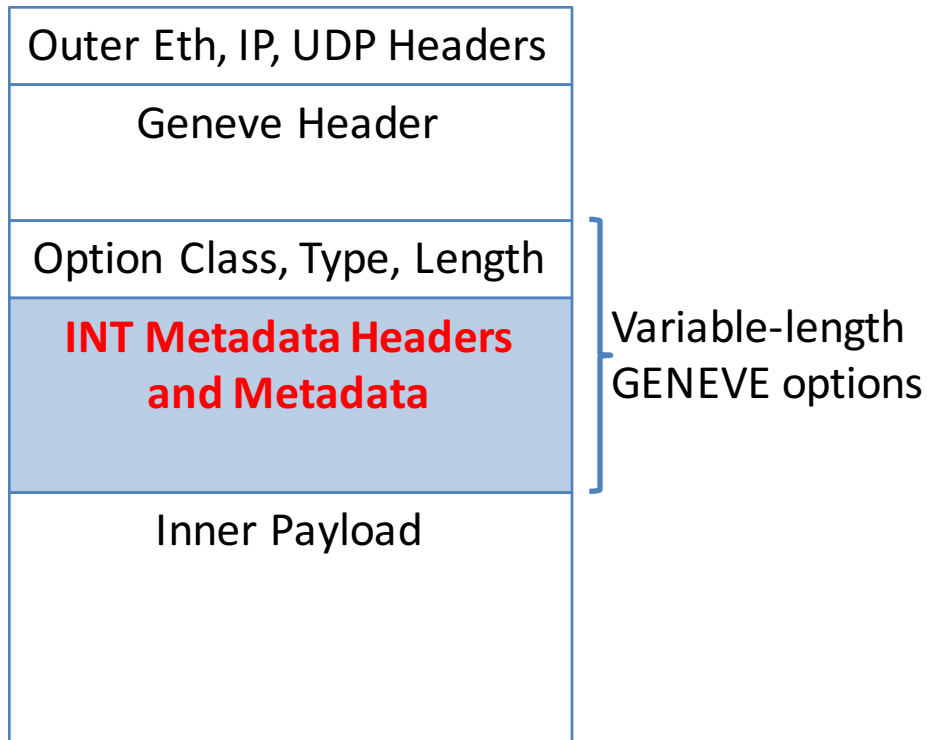


INT Header Format

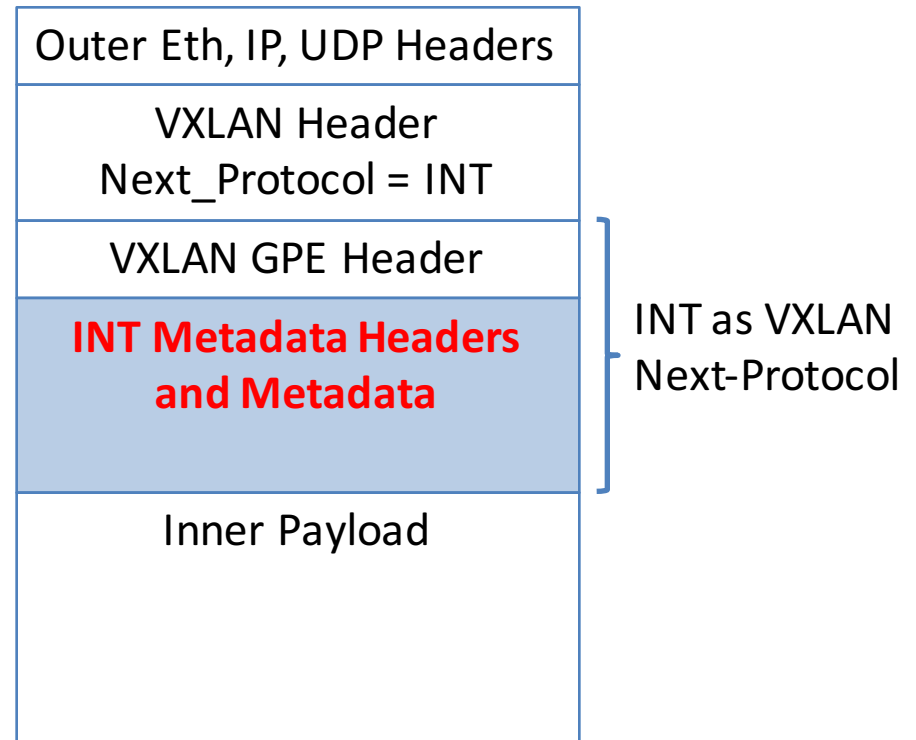


INT Header: Potential Locations

GENEVE



VXLAN-GPE



INT metadata may also be carried as

- Network Service Header Metadata
- TCP options/payload
- UDP payload

INT using P4

- P4 enables flexible packet parsing and modification for INT
- P4 allows INT to adapt to
 - Any Encapsulation format
 - Any State required to be collected
 - Any feature, protocol – current and future

INT : P4 Code Snippet

Header Definitions

```
header_type vxlan_gpe_t
{
  fields {
    flags : 8;
    reserved : 16;
    next_proto : 8;
    vni : 24;
    reserved2 : 8;
  }
}
```

```
header_type vxlan_gpe_int_header_t
{
  fields {
    int_type : 8;
    rsvd : 8;
    len : 8;
    next_proto : 8;
  }
}
```

```
header_type int_header_t {
  fields {
    ver : 2;
    flags : 9;
    ins_cnt : 5;
    max_hop_cnt : 8;
    total_hop_cnt : 8;
    instruction_mask : 16;
  }
}
```

Parser Definitions

```
parser parse_gpe_int_header {
  extract(vxlan_gpe_int_header);
  set_metadata(int_metadata.gpe_int_hdr_len,
               latest.len);
  return parse_int_header;
}
```

```
parser parse_int_header {
  extract(int_header);
  ....
}
```

INT : P4 Code Snippet

Exact-match Table Definition

```
table int_inst {
  reads {
    int_header.instruction_mask : exact;
  }
  actions {
    int_set_header_i0;
    int_set_header_i1;
    int_set_header_i2;
    int_set_header_i3;
    .....
  }
}
```

Action Definitions

```
action int_set_header_i0() {
}
action int_set_header_i1() {
  int_set_header_3();
}
action int_set_header_i2() {
  int_set_header_2();
}
action int_set_header_i3() {
  int_set_header_3();
  int_set_header_2();
}
.....
```

INT Application

Real-time monitoring and troubleshooting

Overlay Network Monitoring today

vmware NSX Dashboard Network Components Controller Cluster Tools & Troubleshooting Welcome, admin Cluster: Demo Admin

Port Connection

Select a logical port source type

Logical Switch

Enter a Logical Switch UUID

9e06793a-5a52-4843-84ba-82aeb4c36389

[Demo-Logical-Switch](#)

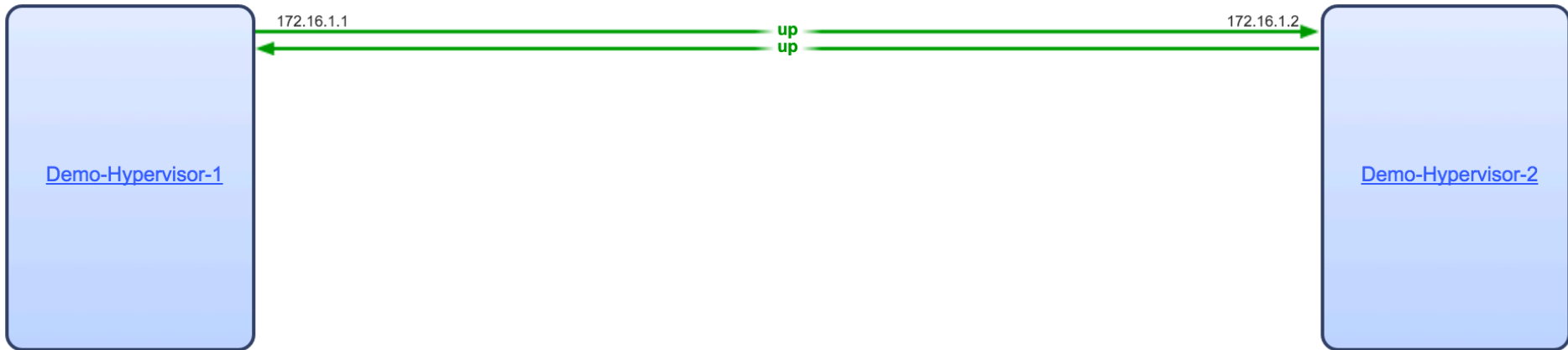
Select a Logical Switch Port

Demo-Logical-Port-1 (1) 00:0c:29:84:29:47

Go

Select a Logical Switch Port

Demo-Logical-Port-2 (2) 00:0c:29:f5:54:7b



Real-time Network Monitoring

vmware NSX Dashboard Network Components Controller Cluster Tools & Troubleshooting Welcome, admin Cluster: Demo Admin

Port Connection

Select a logical port source type

Logical Switch

Enter a Logical Switch UUID

9e06793a-5a52-4843-84ba-82aeb4c36389

[Demo-Logical-Switch](#)

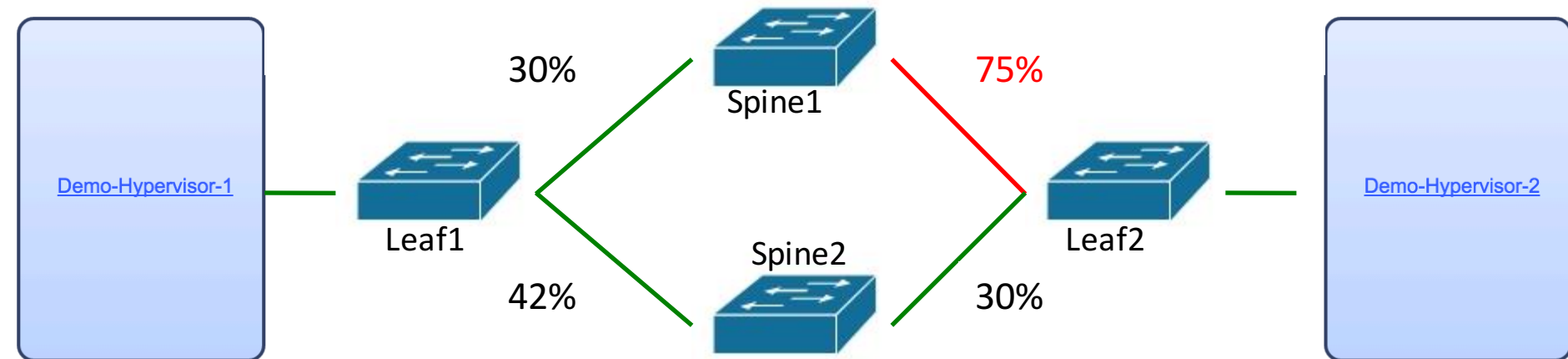
Select a Logical Switch Port

Demo-Logical-Port-1 (1) 00:0c:29:84:29:47

Go

Select a Logical Switch Port

Demo-Logical-Port-2 (2) 00:0c:29:f5:54:7b



Next: Pick a flow on the source logical port and view the path it takes and exact network state it experiences

Real-time troubleshooting demo

The screenshot displays a VirtualBox VM environment with the following components:

- Terminal (Left):** Shows the execution of `mininet` commands to configure a network topology. The output includes: `disable ipv6`, `arp -l eth0`, `route add def`, `Adding table`, `sw1`, `Cleaning stat`, `sw2`, `Cleaning stat`, `sw3`, `Running backg`, `Starting visu`, `Running Apach`, `Starting Web`, `Done`, and `*** Starting mininet`.
- Network Visualization (Center):** A network diagram showing hosts `h1`, `h2`, `h3` and switches `sw1`, `sw2`, `sw3`. Traffic flows are indicated by green arrows with labels: `h1` to `sw1` (859.0K bps), `sw1` to `h1` (95.0K bps), `sw1` to `sw3` (800.0K bps), `sw3` to `h3` (859.0K bps), `h3` to `sw3` (95.0K bps), `sw3` to `sw2` (95.0K bps), and `sw2` to `h2` (95.0K bps). A control panel at the top allows setting node attributes like `packet_drop_rate` (0 to 300) and edge attributes like `rate_tx` (0 to 1000000). A text box overlay reads `h3 wget h1`.
- Terminal (Right):** Shows the execution of `h3 wget h1` and `h3 xterm` commands, along with the execution of `./fetch_and_p` and `./int_plot.py` scripts.
- Figure 1 (Bottom Right):** A line graph titled "Figure 1" showing "HTTP GET latency (seconds)" on the y-axis (0.0 to 0.5) and "Time (seconds)" on the x-axis (5 to 40). The graph displays a baseline latency of approximately 0.05 seconds, with several sharp spikes reaching up to 0.5 seconds at approximately 18, 22, and 33 seconds.

INT Application

Hop-by-Hop Utilization-Aware Load-balancing Architecture

HULA: INT + Flowlet routing

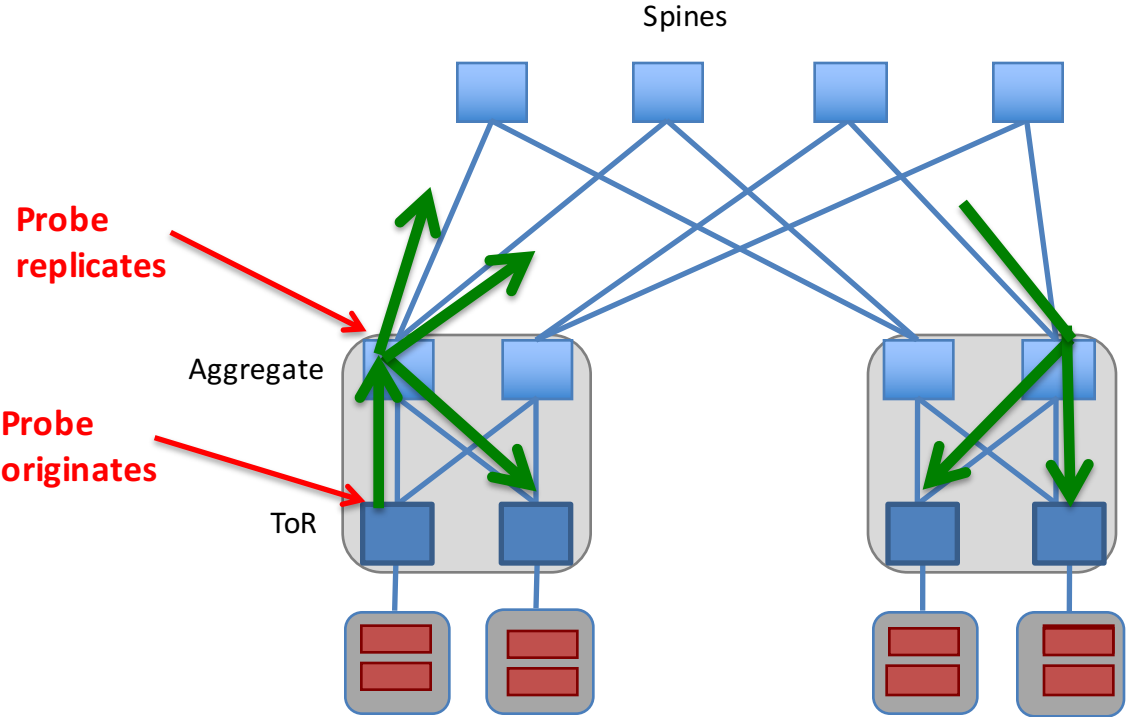
1. Periodic INT probes

- disseminate path utilization to switches

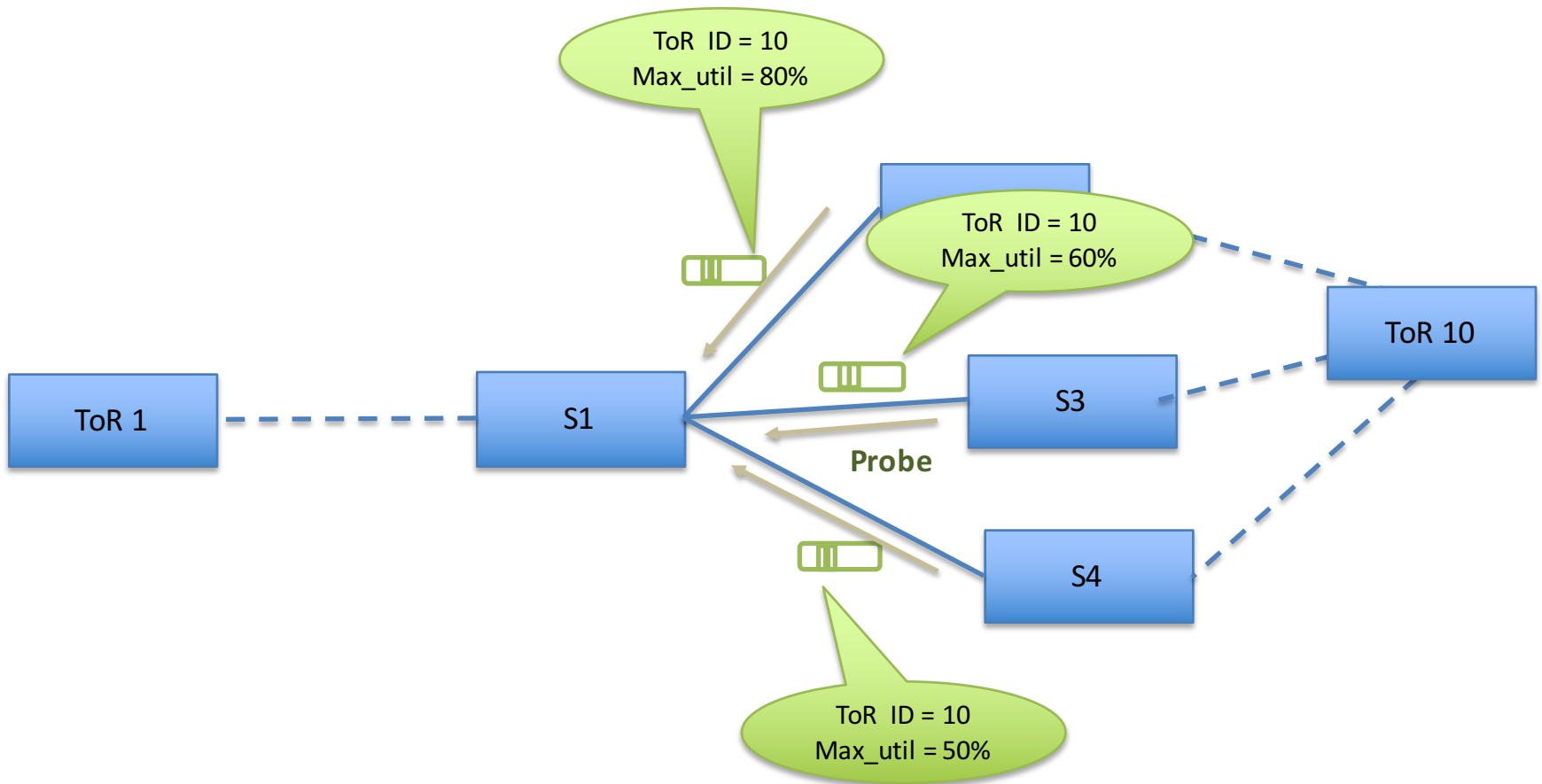
2. Flowlet detection and path selection

- happens at **all** switches
- hop-by-hop adaptive routing

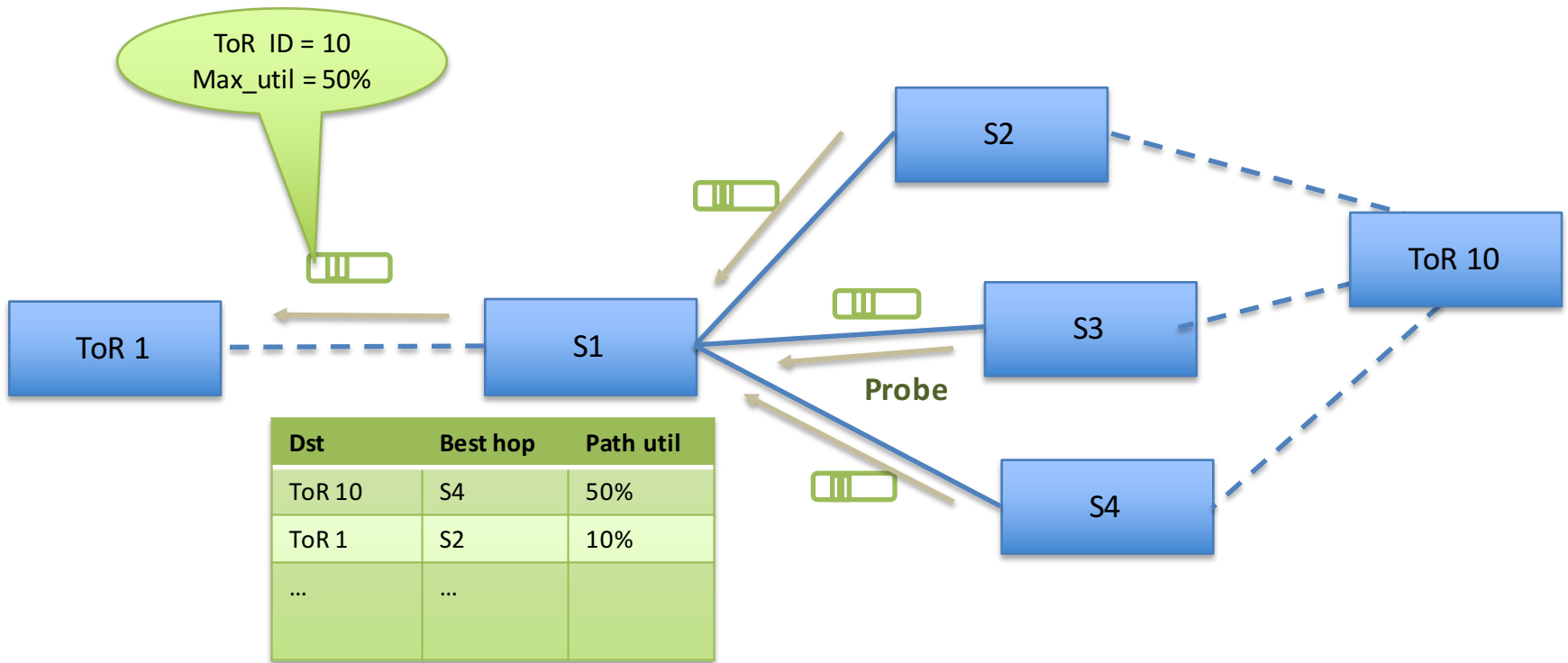
INT probes traverse multiple paths



Probes carry path utilization



Probes update switch state

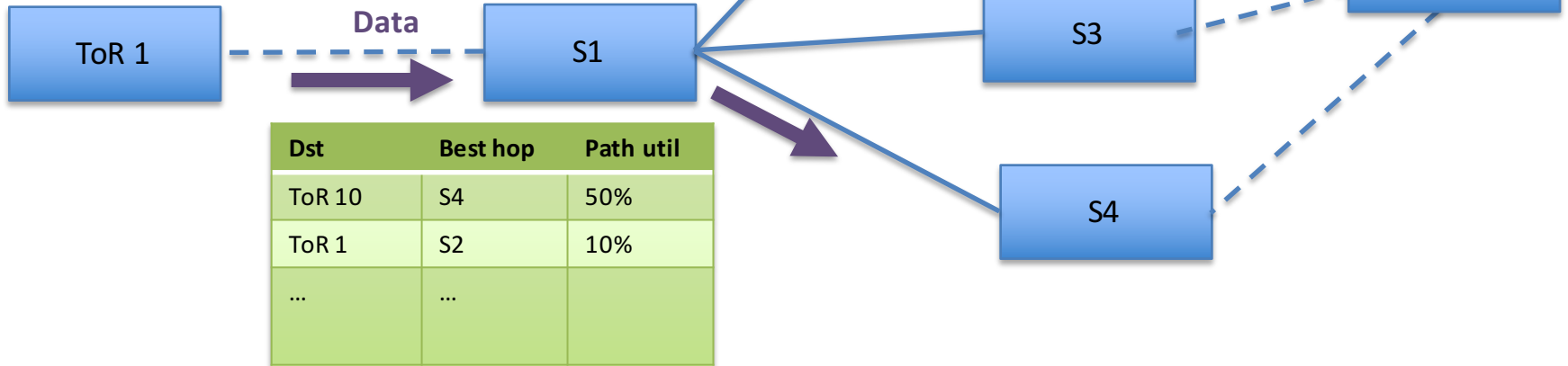


Path Util table

Switches load balance flowlets

Flowlet table

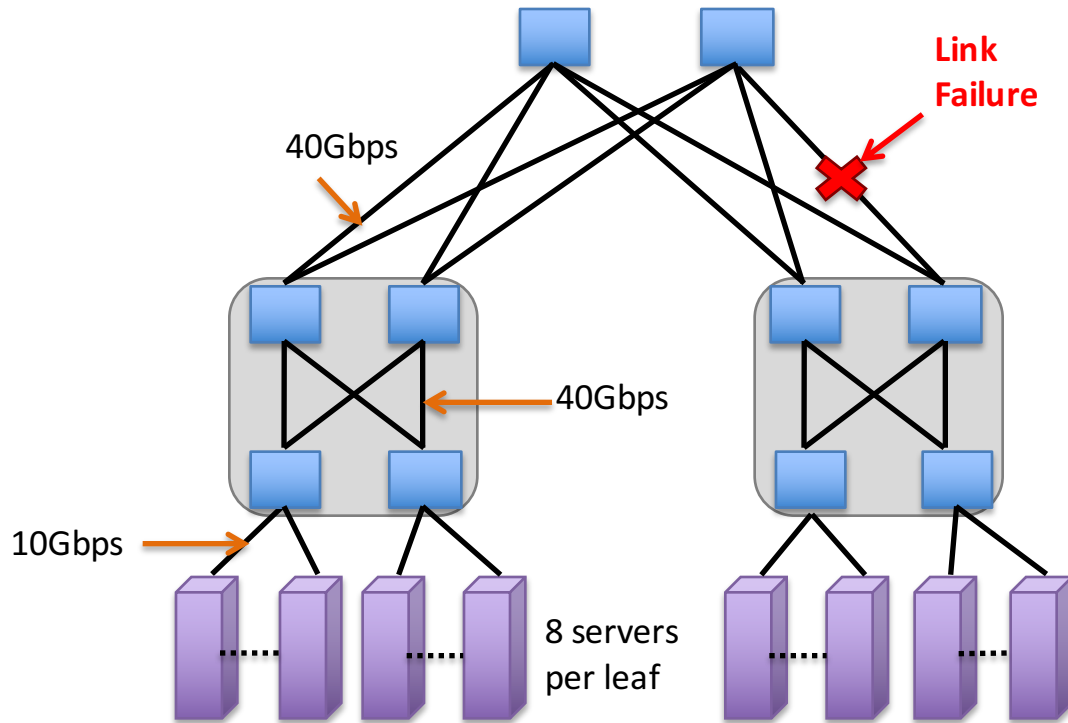
Dst	Flowlet #	Next hop
Tor_10	1	S4
...
...



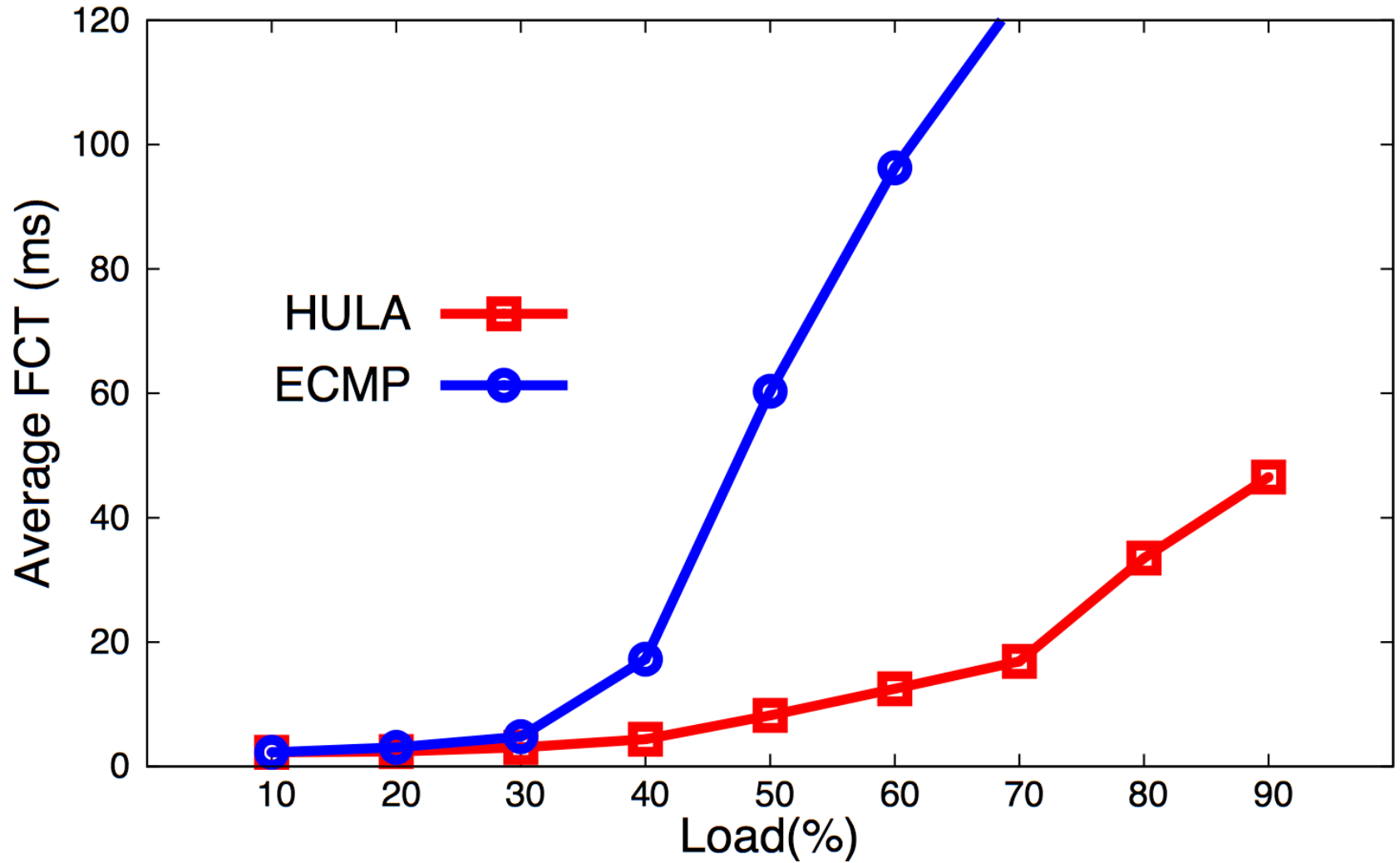
Dst	Best hop	Path util
ToR 10	S4	50%
ToR 1	S2	10%
...

Path Util table

Simulation: Topology Asymmetry



HULA Vs. ECMP



HULA - Advantages

- Topology oblivious
- **Adaptive** to network dynamics
- **Scalable** to large topologies
- No separate source routing required
- **Programmable** in P4!
 - Processing probes
 - Flowlet routing

Summary

- INT provides real-time network state directly in the dataplane
 - Scales to arbitrarily large networks
 - Scales to current and future link speeds
 - Can adapt to any network, any encap, any application
- Knowledge of real-time network state opens up new possibilities
 - Enhanced monitoring and troubleshooting
 - Network-state aware routing
 - ...

More information

<http://p4.org/p4/inband-network-telemetry/>

Blog post with links to

- INT demo video
- INT specification
- P4 source code repository

More information on Utilization aware routing will be posted on p4.org in the near future

INT Specification – Collaborative Effort

<http://p4.org/wp-content/uploads/fixed/INT/INT-current-spec.pdf>

In-band Network Telemetry (INT)

September 2015

Changhoon Kim, Parag Bhide, Ed Doe: *Barefoot Networks*

Hugh Holbrook: *Arista*

Anoop Ghanwani: *Dell*

Dan Daly: *Intel*

Mukesh Hira, Bruce Davie: *VMware*

[Introduction](#)

[Terms](#)

[What To Monitor](#)

[Switch-level Information](#)

[Ingress Information](#)

[Egress Information](#)

[Buffer Information](#)

[Processing INT Headers](#)

[INT Header Types](#)

[Handling INT Packets](#)

[Header Format and Location](#)

Thank You