Simplifying Datacenter Network Debugging with PathDump

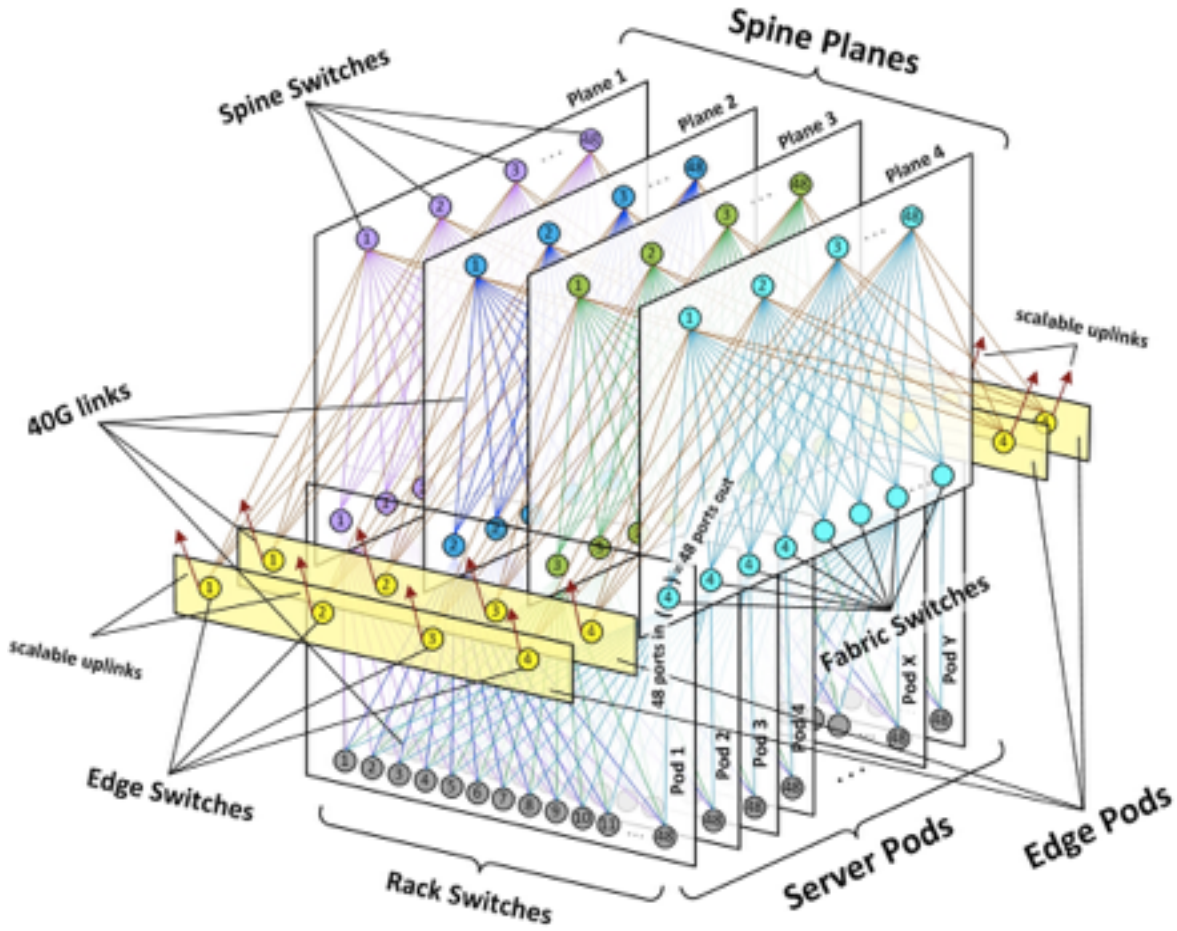


Praveen Tammana† Rachit Agarwal‡ Myungjin Lee†

†University of Edinburgh, ‡Cornell University

Datacenter networks are complex

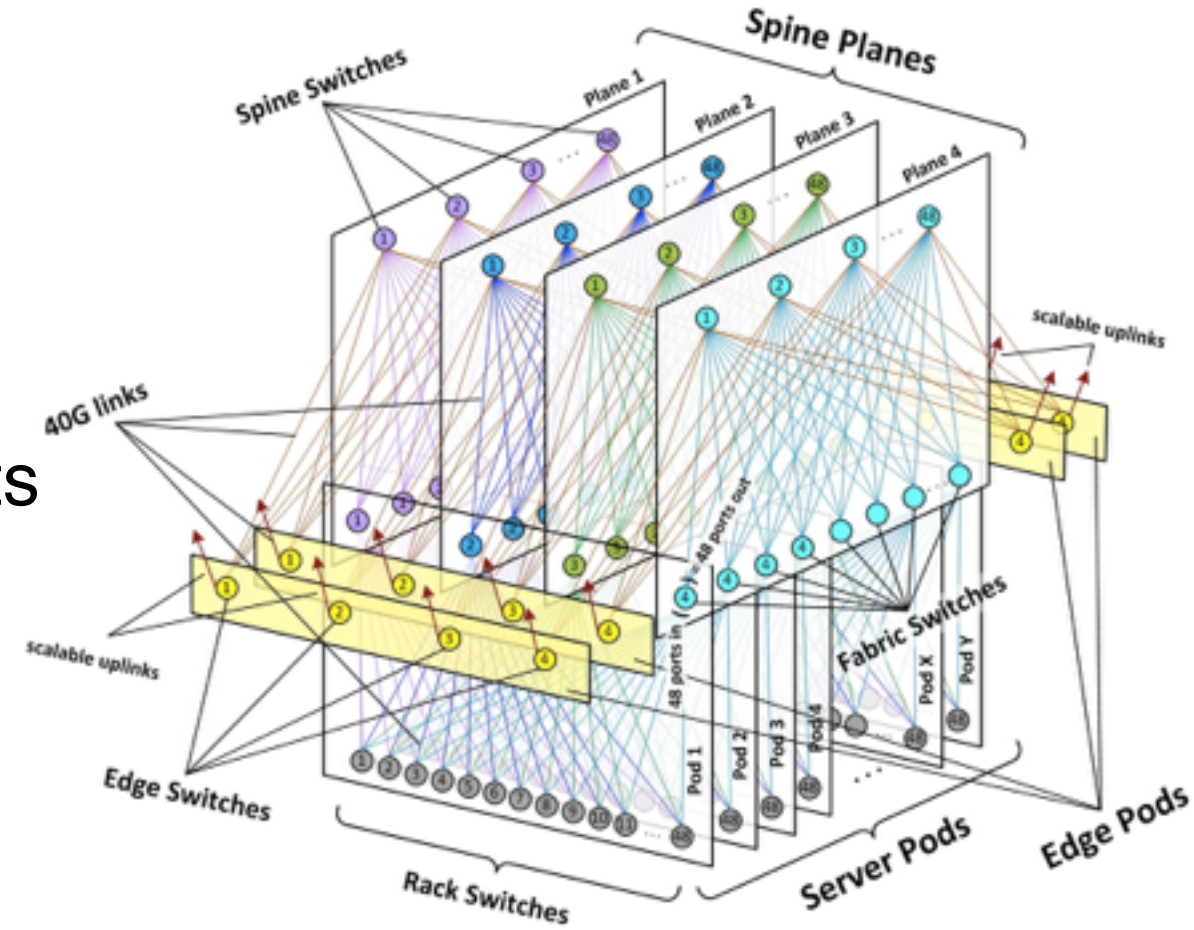
- Increasingly larger scale
  - Over 10k switches, 100k servers
  - Each server with 10 to 40 Gbps
  - Aggregate traffic > 100 Tbps



--source: TechRepublic.com

Datacenter networks are complex

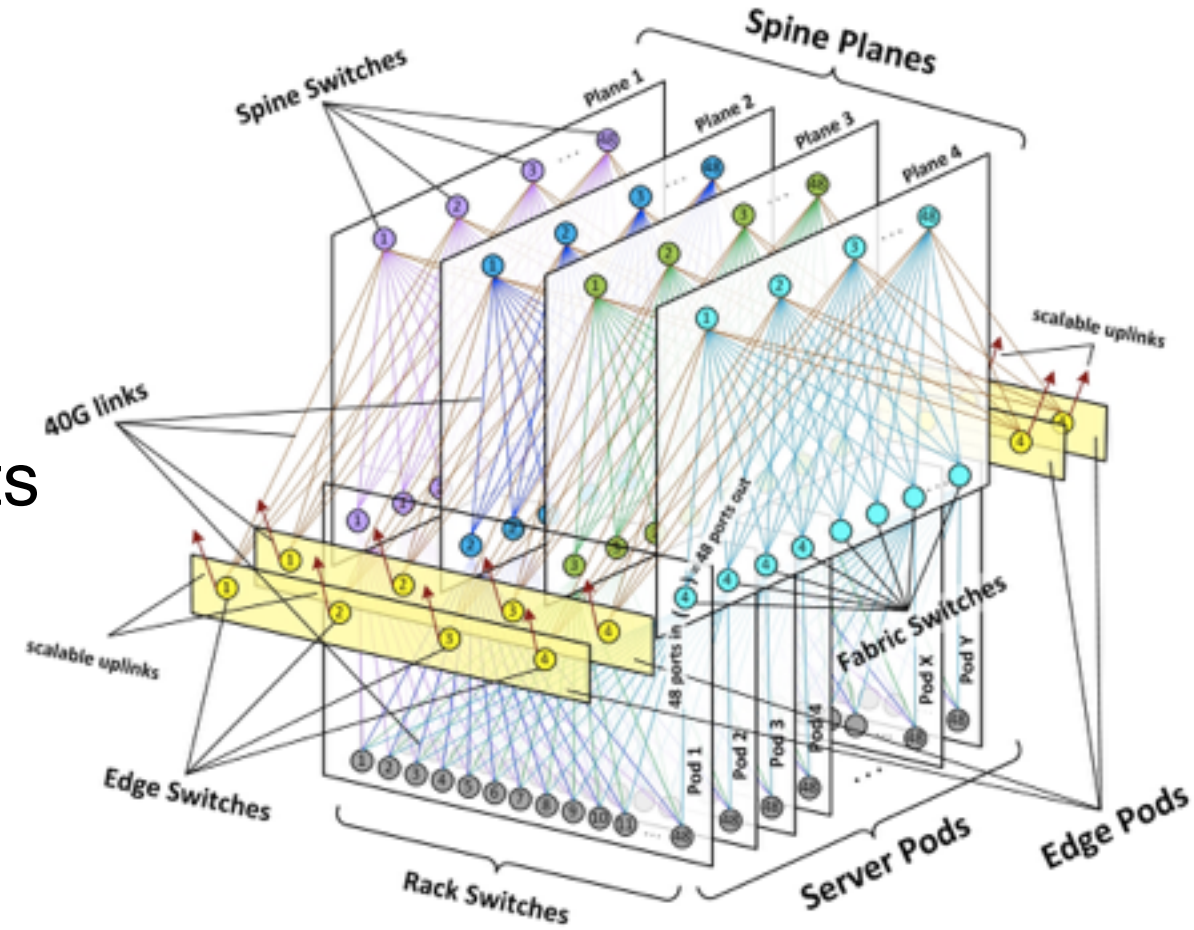
- Increasingly larger scale
  - Over 10k switches, 100k servers
  - Each server with 10 to 40 Gbps
  - Aggregate traffic > 100 Tbps
- Stringent performance requirements
  - E.g., Amazon and Google studies



--source: TechRepublic.com

Datacenter networks are complex

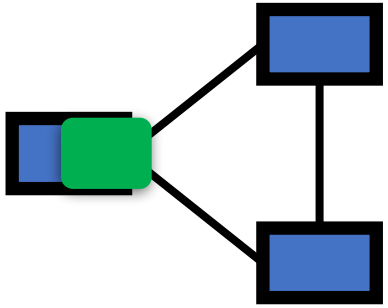
- Increasingly larger scale
  - Over 10k switches, 100k servers
  - Each server with 10 to 40 Gbps
  - Aggregate traffic > 100 Tbps
- Stringent performance requirements
  - E.g., Amazon and Google studies
- Complex policies
  - Security, isolation, etc.
- Network programmability
  - Too many possible configurations



--source: TechRepublic.com

Network problems are inevitable

Loops

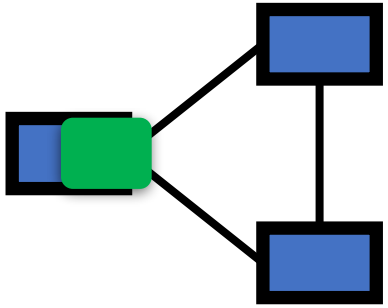


Failures, bugs

- Result: **Mismatch** between network behavior and operator intent

Network problems are inevitable

Loops

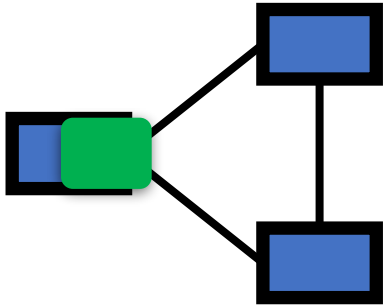


Failures, bugs

- Result: **Mismatch** between network behavior and operator intent

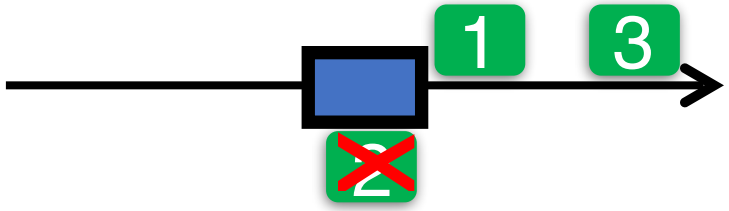
Network problems are inevitable

Loops



Failures, bugs

Silent random packet drops

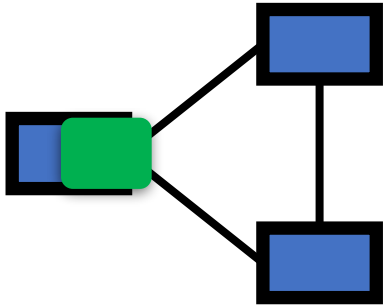


Faulty interface

- Result: **Mismatch** between network behavior and operator intent

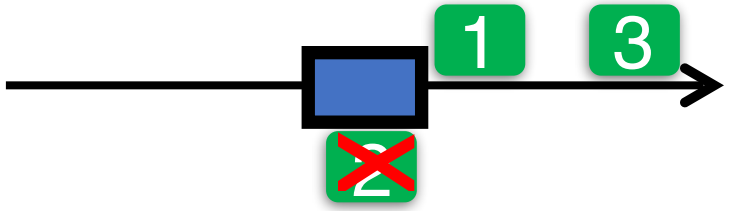
Network problems are inevitable

Loops



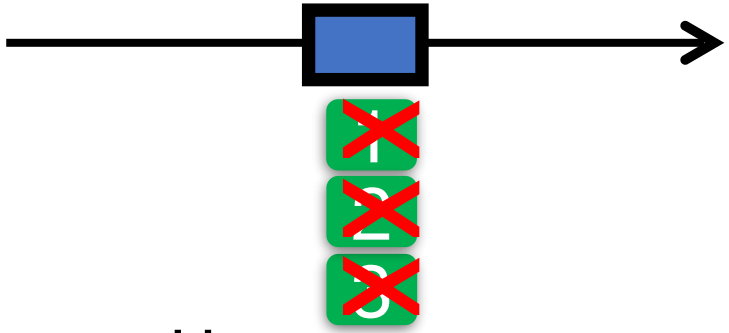
Failures, bugs

Silent random packet drops



Faulty interface

Black hole

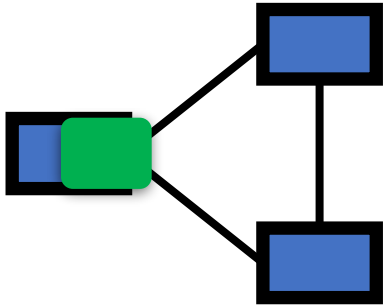


Human errors

- Result: **Mismatch** between network behavior and operator intent

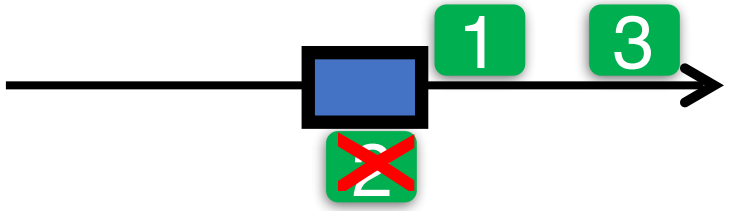
Network problems are inevitable

Loops



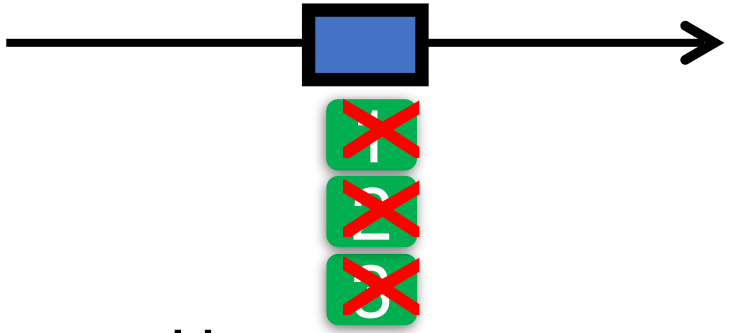
Failures, bugs

Silent random packet drops



Faulty interface

Black hole

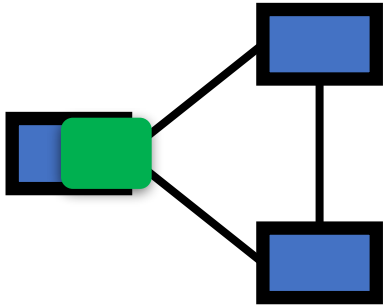


Human errors

- Result: **Mismatch** between network behavior and operator intent

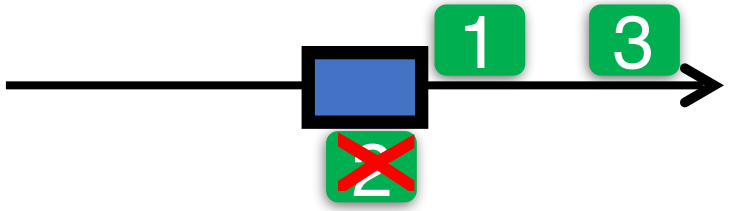
Network problems are inevitable

Loops



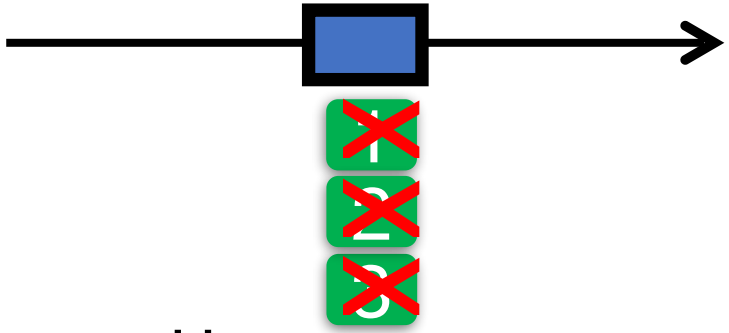
Failures, bugs

Silent random packet drops



Faulty interface

Black hole



Human errors

- Result: **Mismatch** between network behavior and operator intent
- Lots of research efforts in building network debuggers

Network debuggers are even more complex

Existing designs: **in-network techniques**

Network debuggers are even more complex

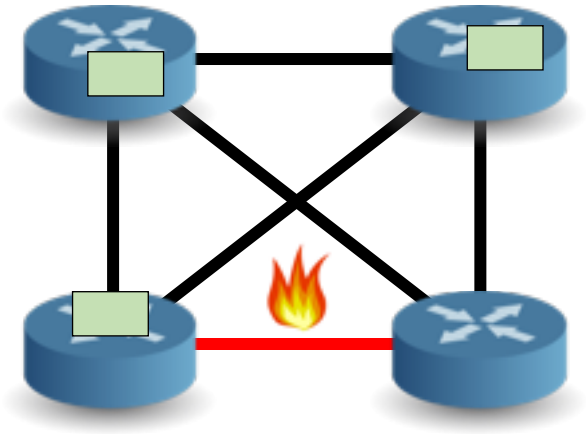
Existing designs: **in-network techniques**

Idea: Use programmability of network switches to capture debugging information

Network debuggers are even more complex

Static analysis of data plane snapshots

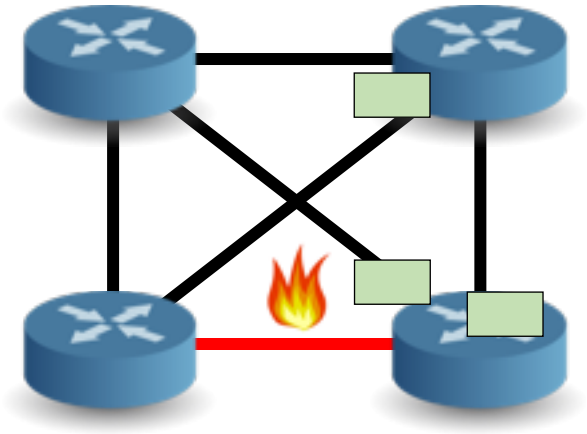
E.g.: HSA [NSDI'12], Anteater [SIGCOMM'11]



Network debuggers are even more complex

Static analysis of data plane snapshots

E.g.: HSA [NSDI'12], Anteater [SIGCOMM'11]



Network debuggers are even more complex

Static analysis of data plane snapshots

E.g.: HSA [NSDI'12], Anteater [SIGCOMM'11]

Network debuggers are even more complex

Static analysis of data plane snapshots

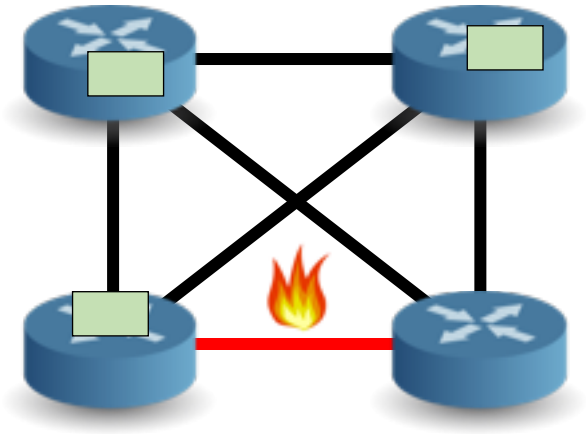
E.g.: HSA [NSDI'12], Anteater [SIGCOMM'11]

- Capturing consistent network state is a hard problem

Network debuggers are even more complex

Per-switch per-packet logging

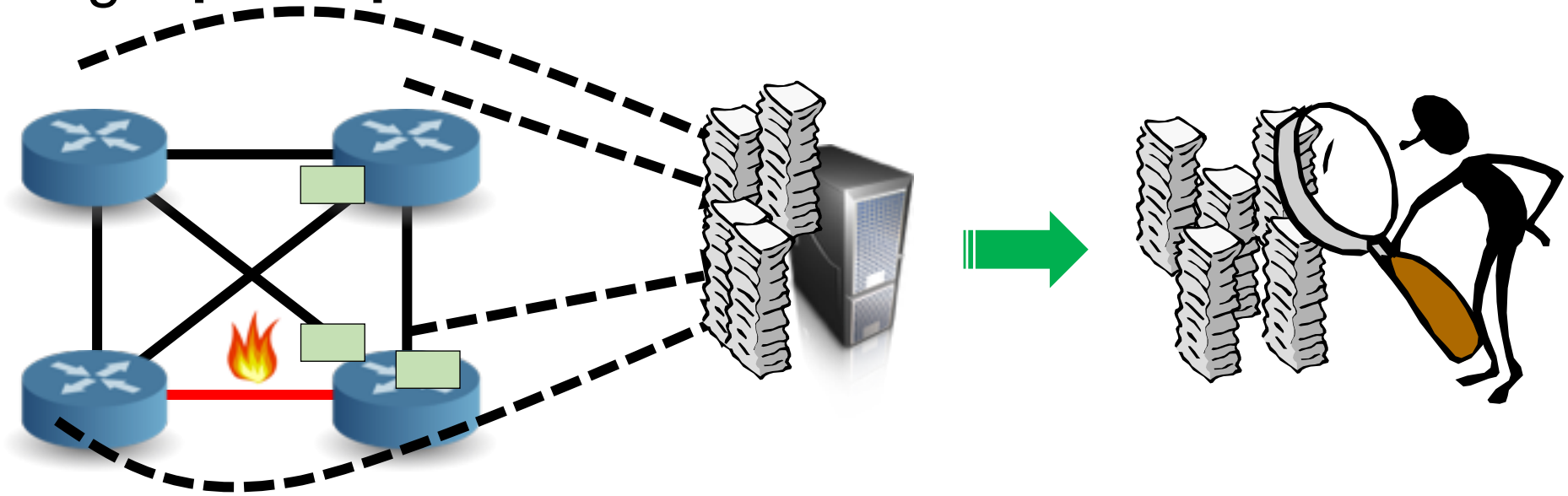
E.g.: NetSight [NSDI'14]



Network debuggers are even more complex

Per-switch per-packet logging

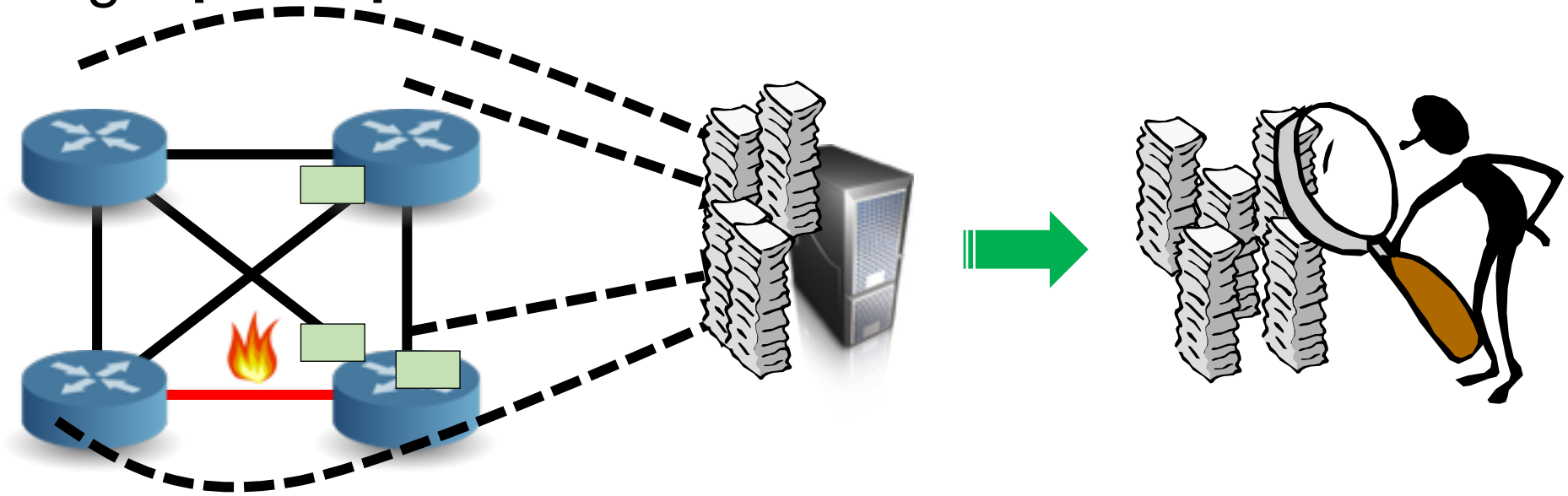
E.g.: NetSight [NSDI'14]



Network debuggers are even more complex

Per-switch per-packet logging

E.g.: NetSight [NSDI'14]

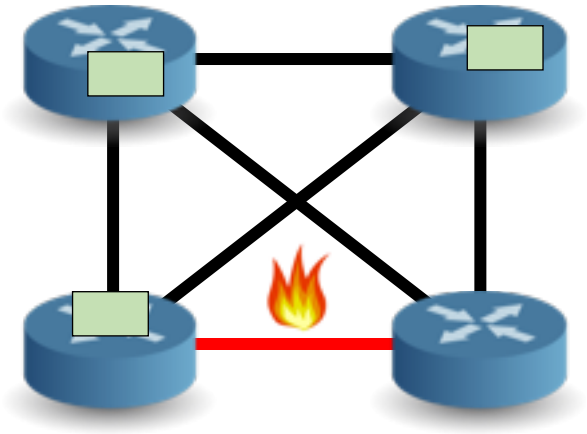


- High bandwidth and processing overhead

Network debuggers are even more complex

Selective packet sampling and mirroring

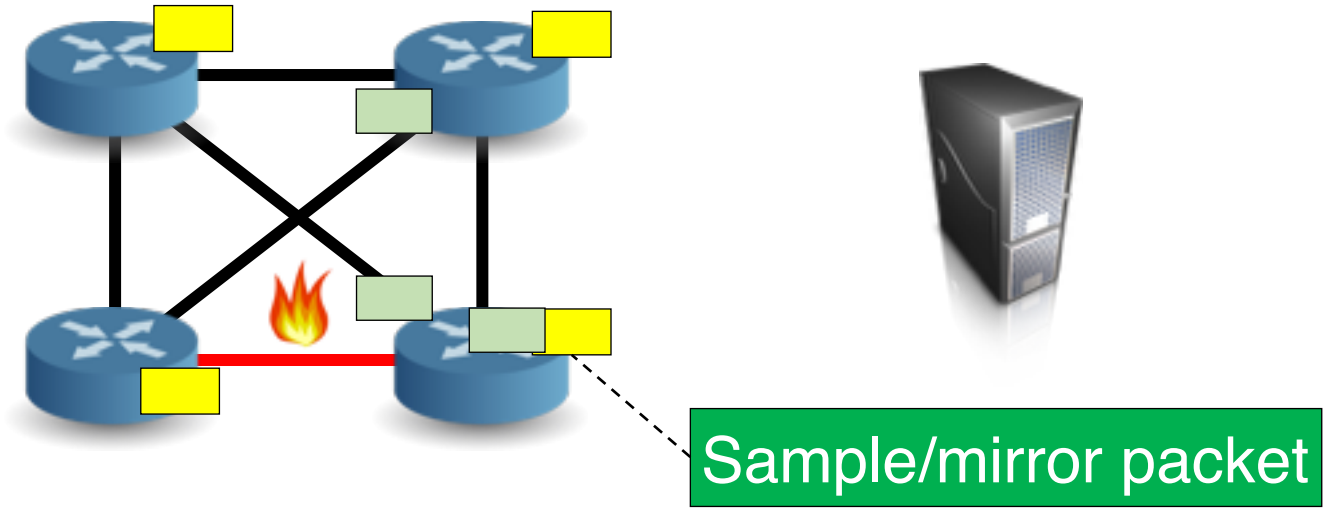
E.g.: Everflow [SIGCOMM'15], Planck [SIGCOMM'14], sFlow



Network debuggers are even more complex

Selective packet sampling and mirroring

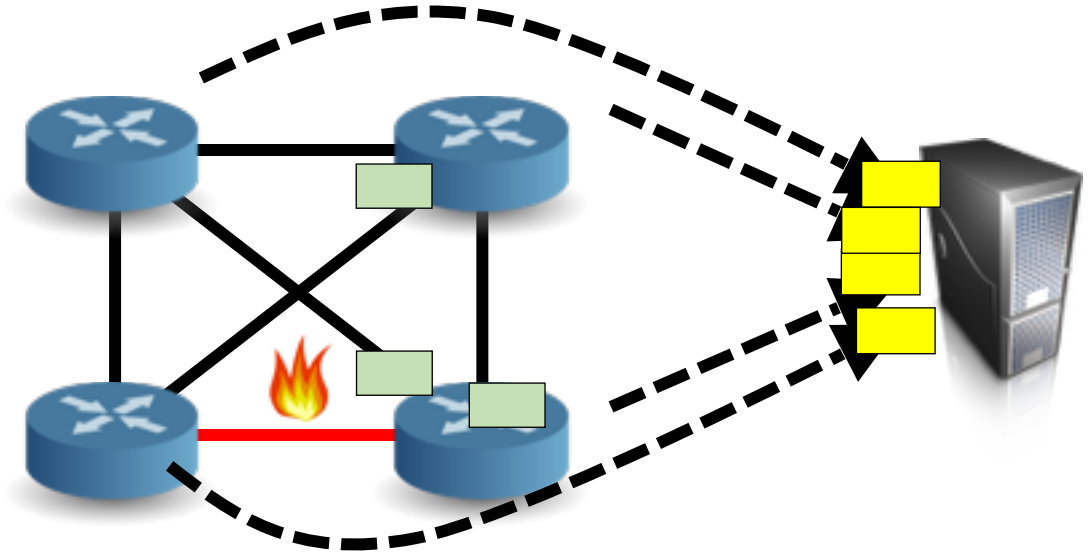
E.g.: Everflow [SIGCOMM'15], Planck [SIGCOMM'14], sFlow



Network debuggers are even more complex

Selective packet sampling and mirroring

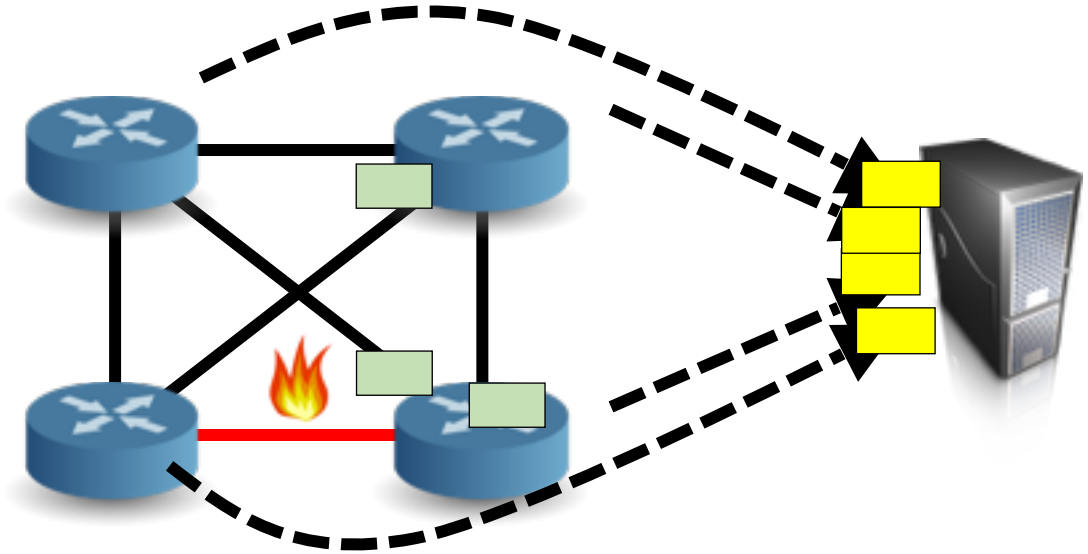
E.g.: Everflow [SIGCOMM'15], Planck [SIGCOMM'14], sFlow



Network debuggers are even more complex

Selective packet sampling and mirroring

E.g.: Everflow [SIGCOMM'15], Planck [SIGCOMM'14], sFlow

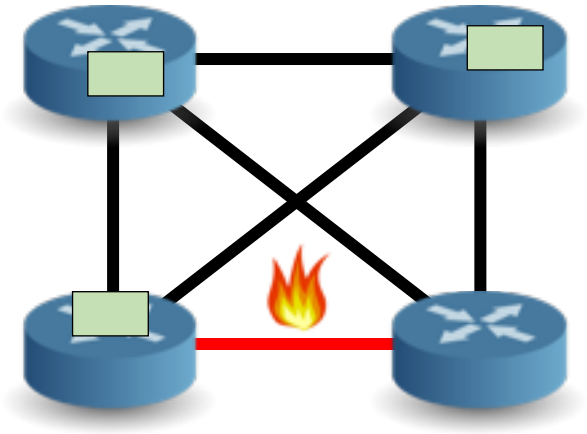


- Identifying packets to sample for debugging problems is complex

Network debuggers are even more complex

SQL-like queries on switches

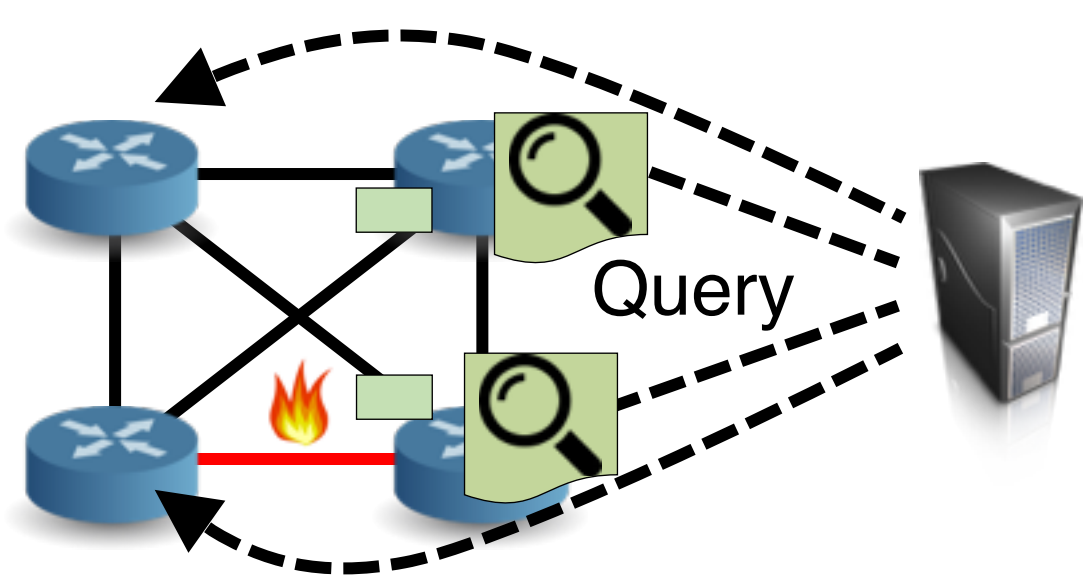
E.g.: Pathquery [NSDI'16]



Network debuggers are even more complex

SQL-like queries on switches

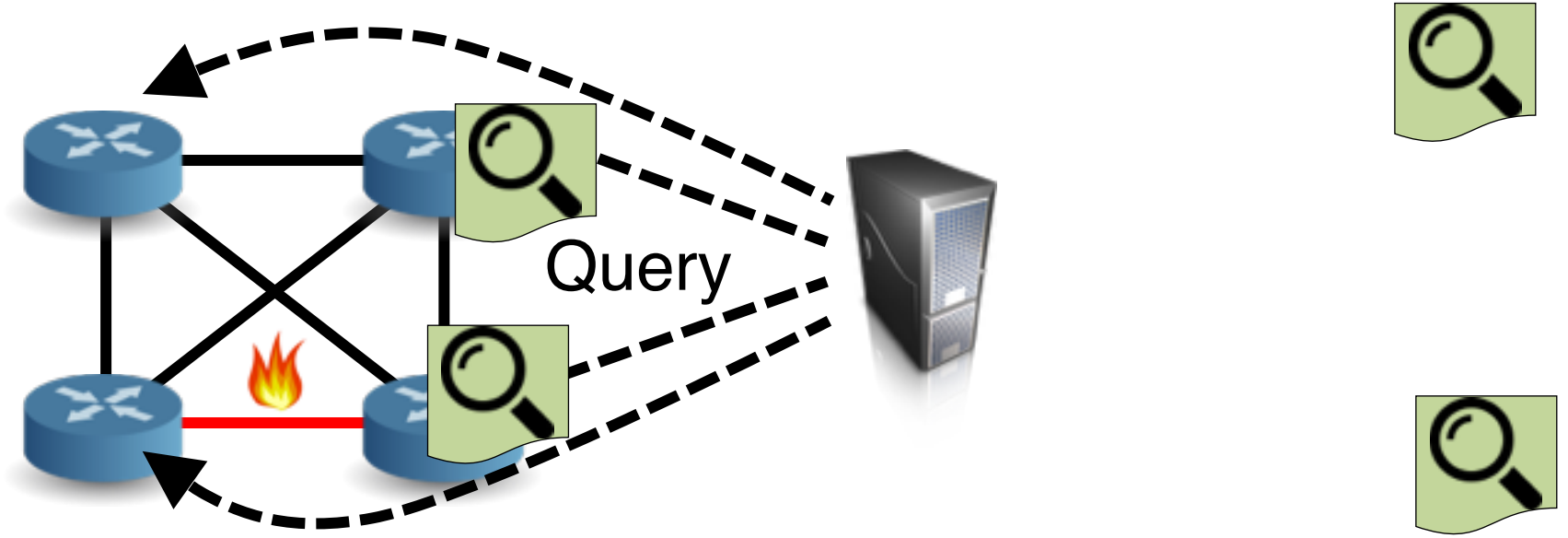
E.g.: Pathquery [NSDI'16]



Network debuggers are even more complex

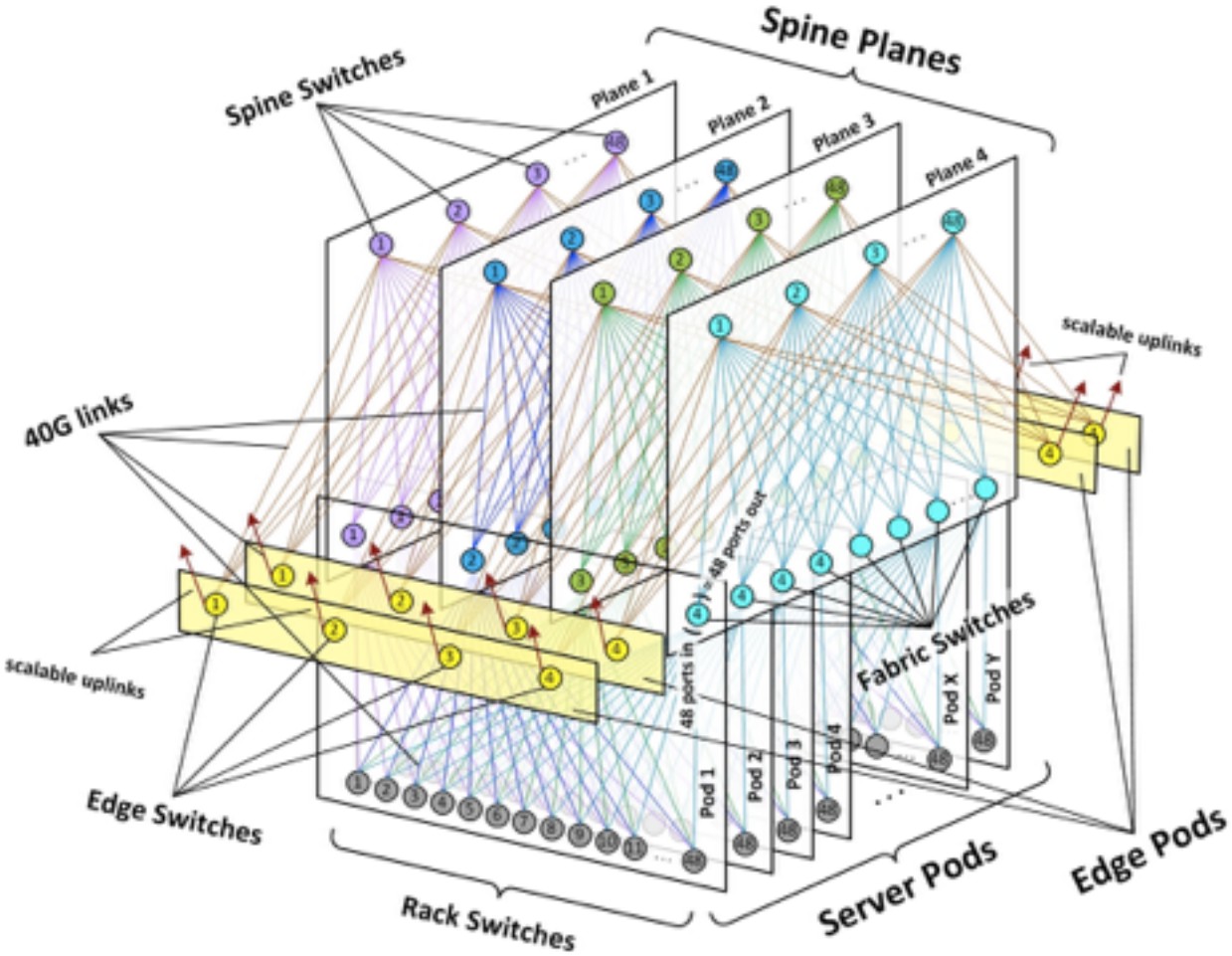
SQL-like queries on switches

E.g.: Pathquery [NSDI'16]



- Requires dynamic installation of switch rules

Summary: Complex networks and debuggers



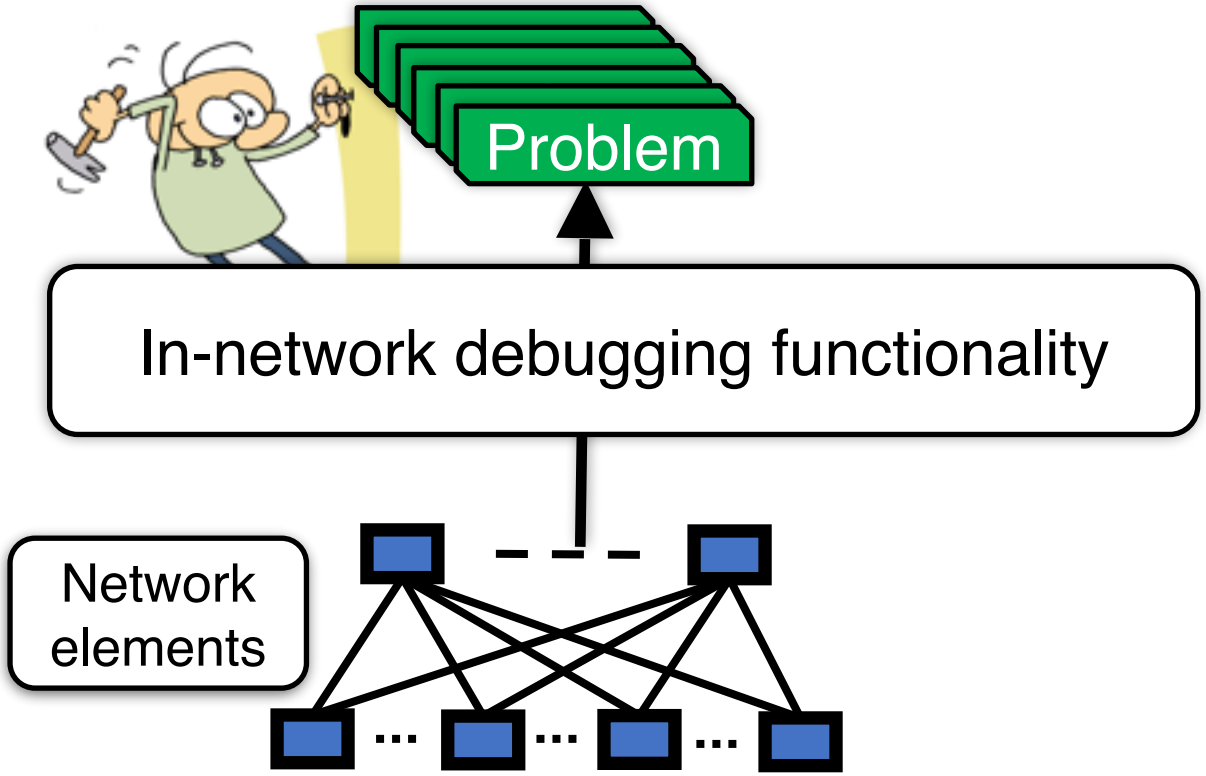
--source: TechRepublic.com

Complex networks

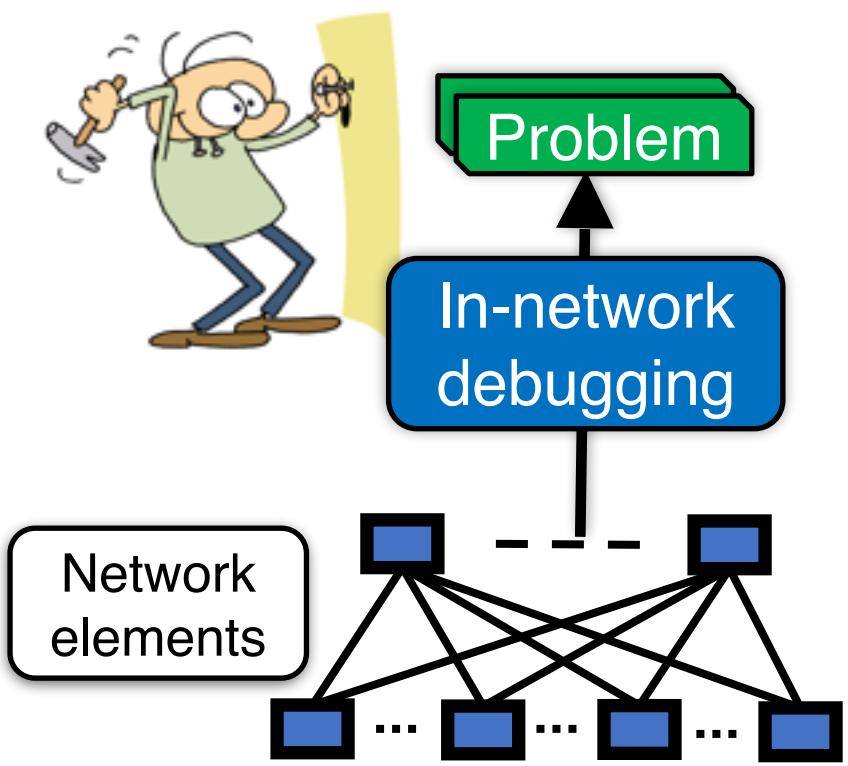
- Data plane snapshots
- Per-switch per-packet logs
- Packet mirroring
- Packet sampling
- Dynamic rule installation

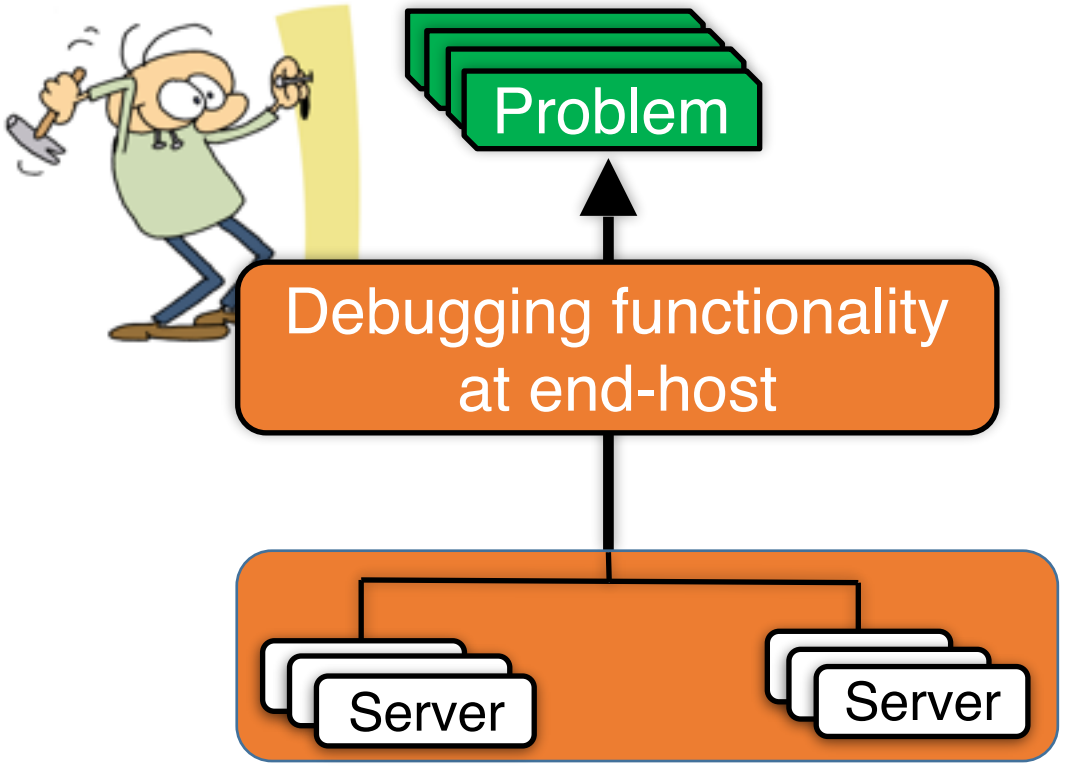
Network debuggers even more complex

PathDump: (Simple) In-network + End-hosts



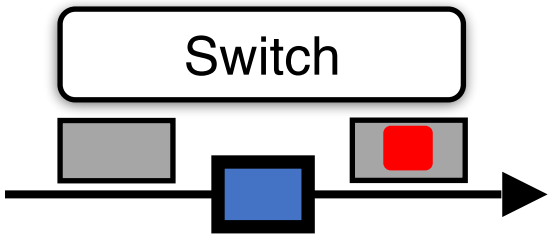
PathDump: (Simple) In-network + End-hosts





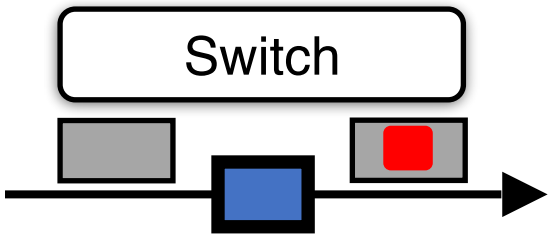
- Use end-hosts for most debugging problems
- In-network functionality for a small number of debugging problems

PathDump in a nutshell



- Before forwarding a packet, checks a condition
- If met, embeds its ID into packet header

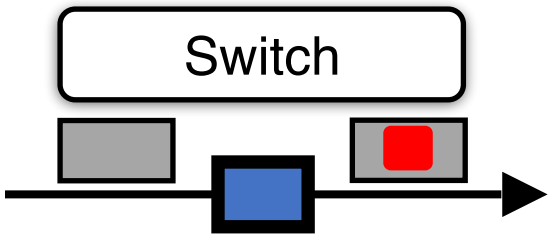
PathDump in a nutshell



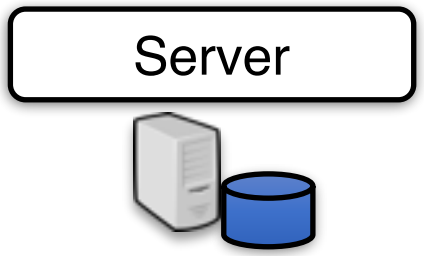
- Before forwarding a packet, checks a condition
- If met, embeds its ID into packet header

- No data plane snapshots
- No per-switch per-packet logs
- No packet sampling
- No packet mirroring
- No dynamic rule installation

PathDump in a nutshell

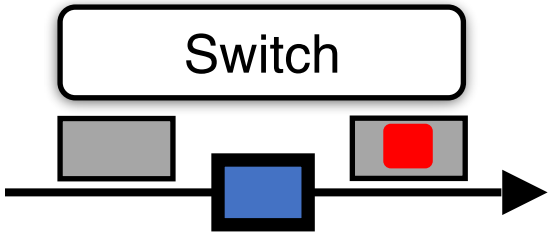


- Before forwarding a packet, checks a condition
- If met, embeds its ID into packet header

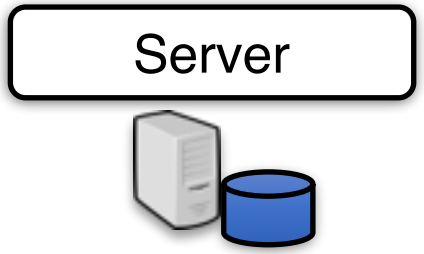


- Captures each and every packet header
- Stores and updates flow-level statistics
- Exposes API for debugging purposes

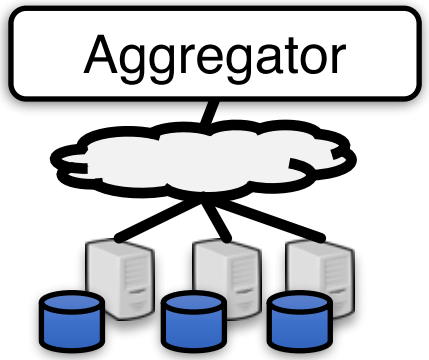
PathDump in a nutshell



- Before forwarding a packet, checks a condition
- If met, embeds its ID into packet header



- Captures each and every packet header
- Stores and updates flow-level statistics
- Exposes API for debugging purposes



- Enables slicing-and-dicing of statistics across flows (potentially stored at various end-hosts)

PathDump: Three challenges

PathDump: Three challenges

Coverage

How to support large class of debugging functionalities?

Debug more than 85% of reported network problems

PathDump: Three challenges

Coverage

How to support large class of debugging functionalities?

Debug more than 85% of reported network problems

Packets not reaching destination

How to handle packet drops and loops caused by network problems?

Exploit load balancing (e.g. ECMP) and identify spurious packet drops

PathDump: Three challenges

Coverage

How to support large class of debugging functionalities?

Debug more than 85% of reported network problems

Packets not reaching destination

How to handle packet drops and loops caused by network problems?

Exploit load balancing (e.g. ECMP) and identify spurious packet drops

Data plane/end-host resources

Switch resources and packet header space are limited

CherryPick: Per-packet path tracing technique

10s of flow rules at switch

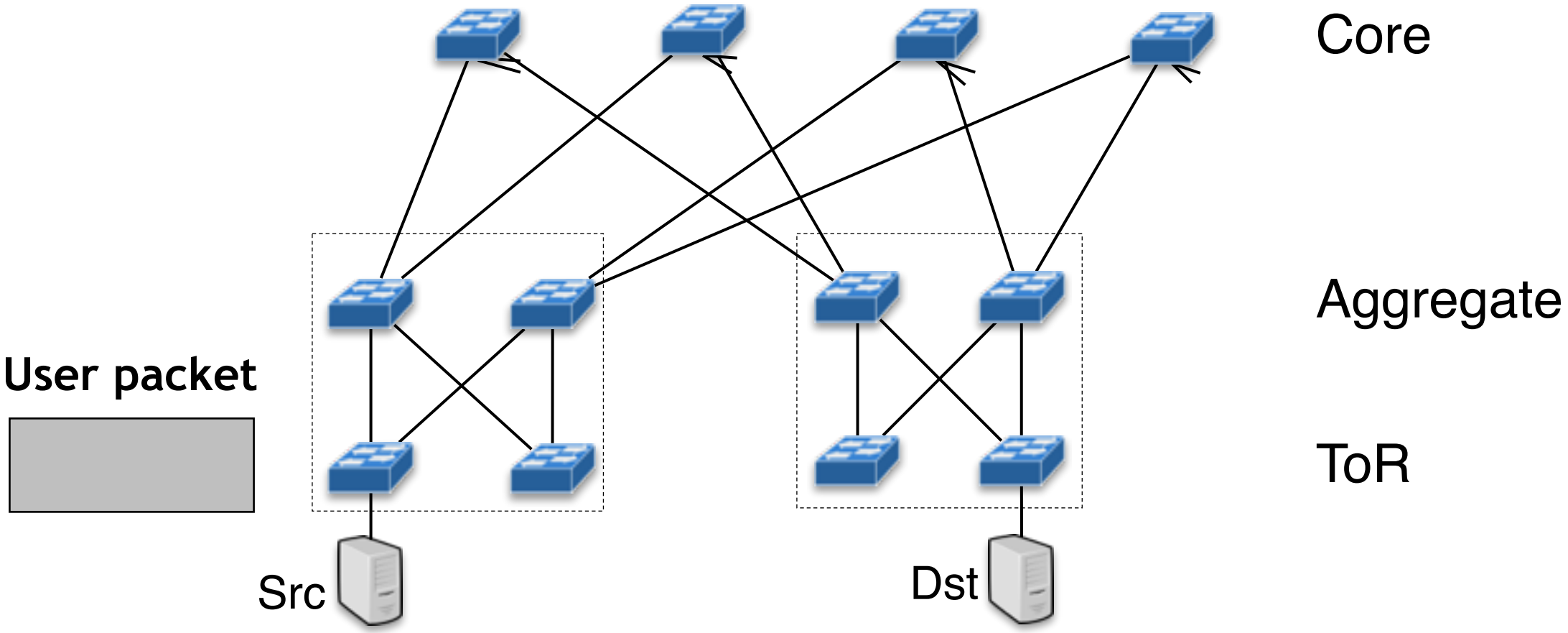
PathDump should not hog user app's resources at end-host

Two VLAN tags in the packet

25% of one core / 100MB of mem. at end-host

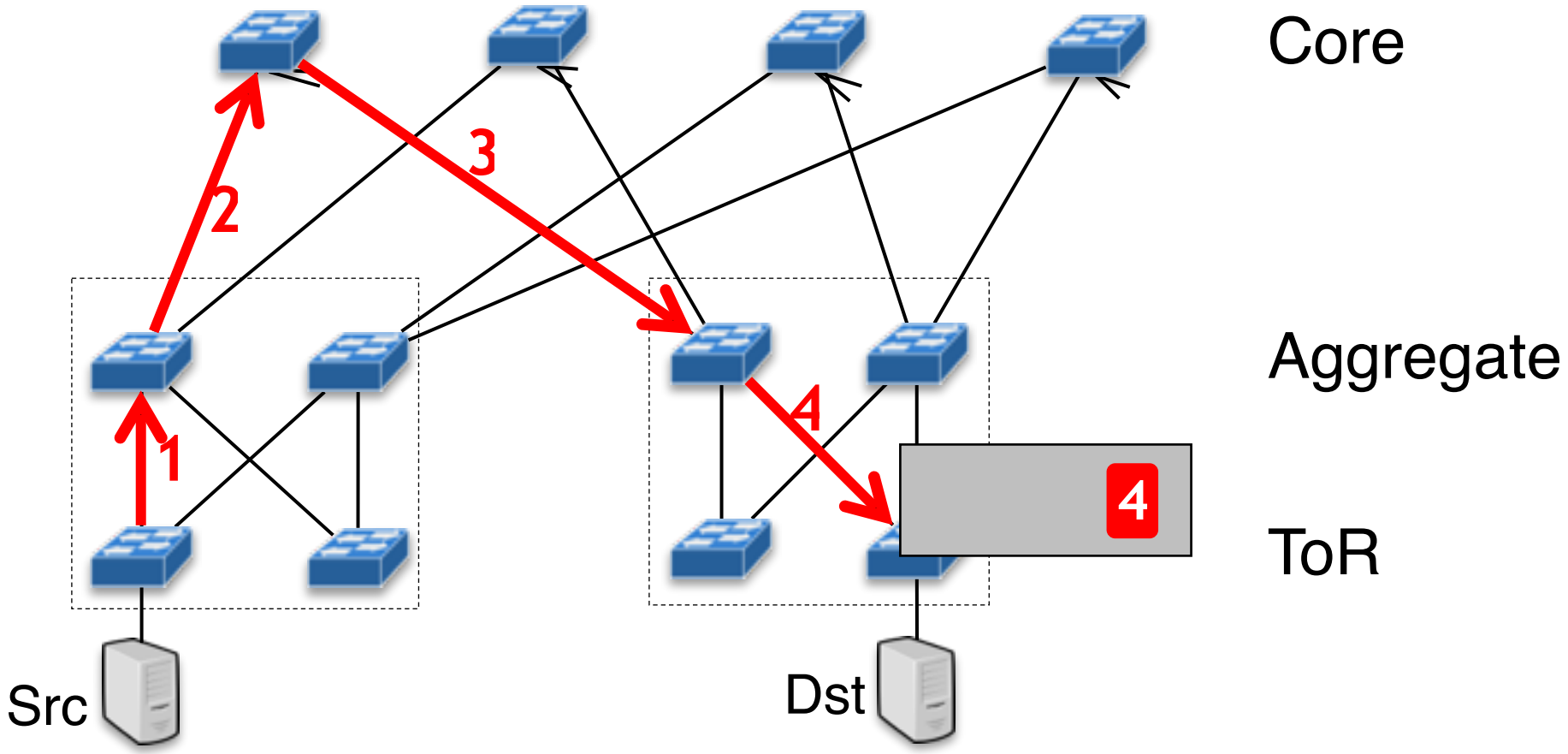
PathDump architecture

1. Switch embeds unique ID (e.g., link ID)



PathDump architecture

1. Switch embeds unique ID (e.g., link ID)

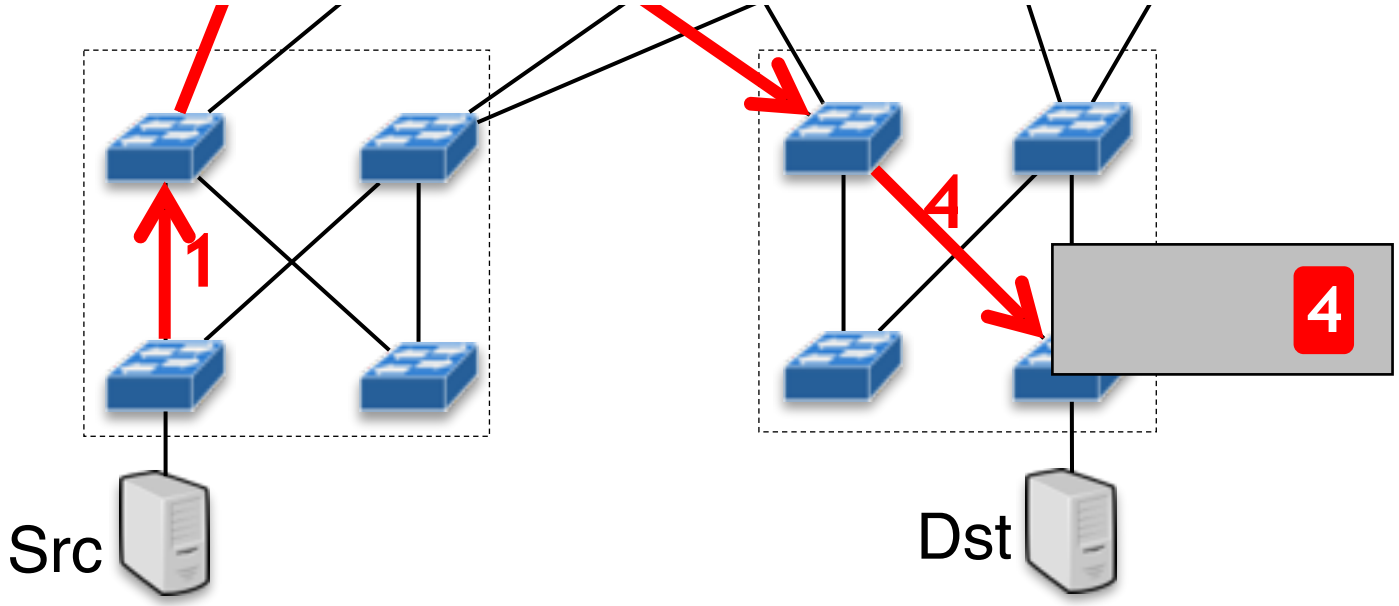


PathDump architecture

1. Switch embeds unique ID (e.g., link ID)

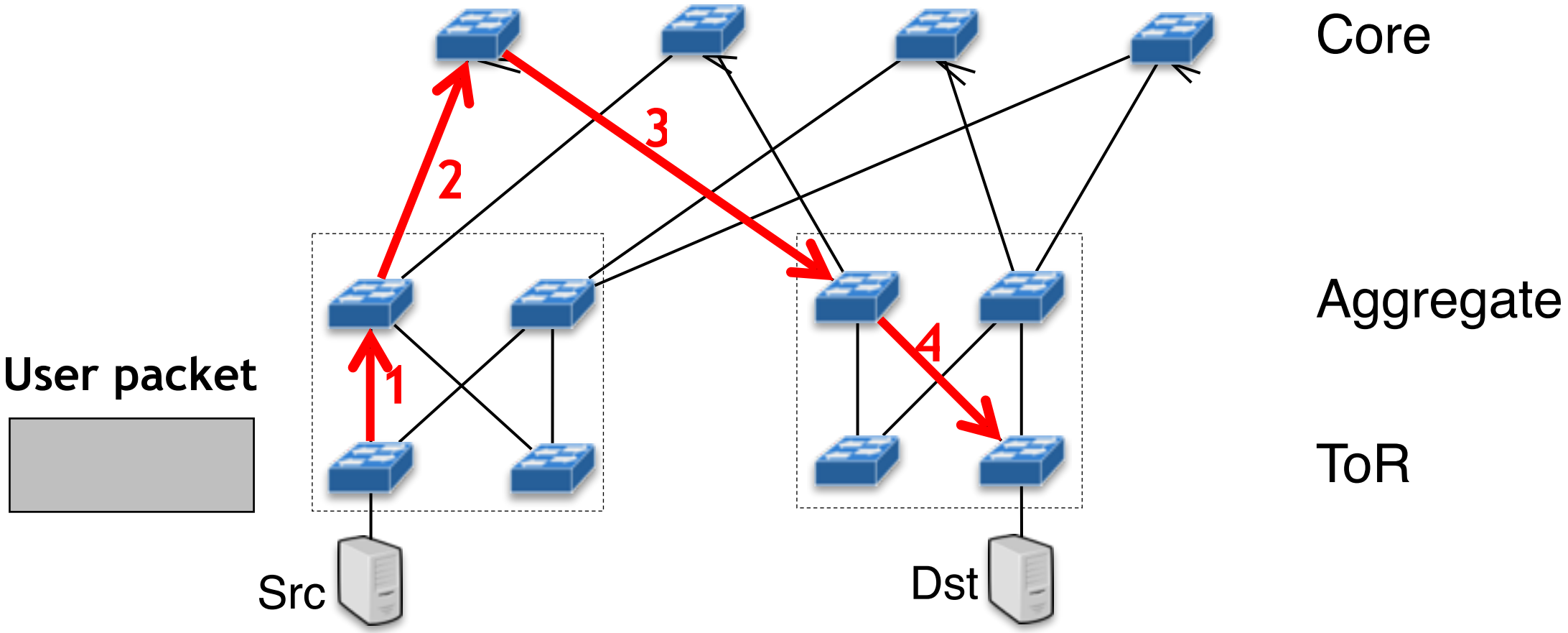
- Packet header space limitation

- Cherrypick [SOSR'15] for current deployments



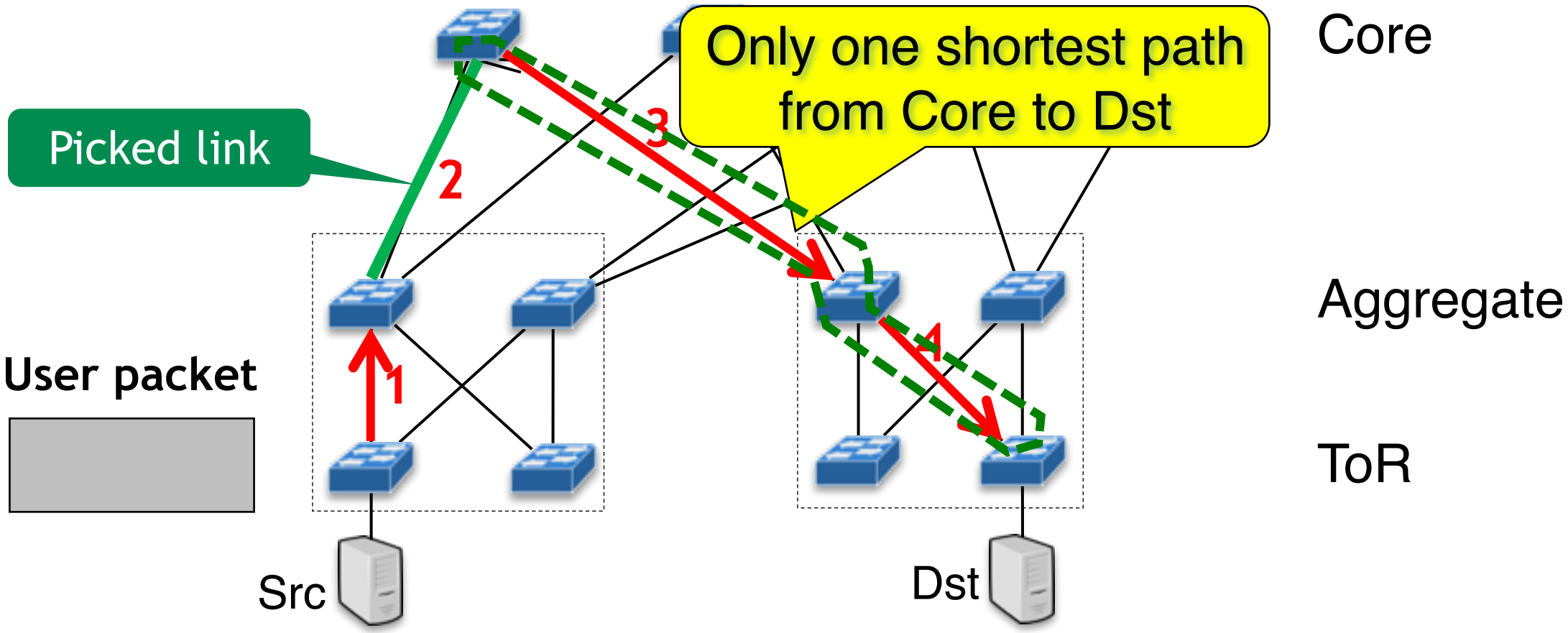
PathDump architecture

1. Switch embeds unique ID (e.g., link ID)



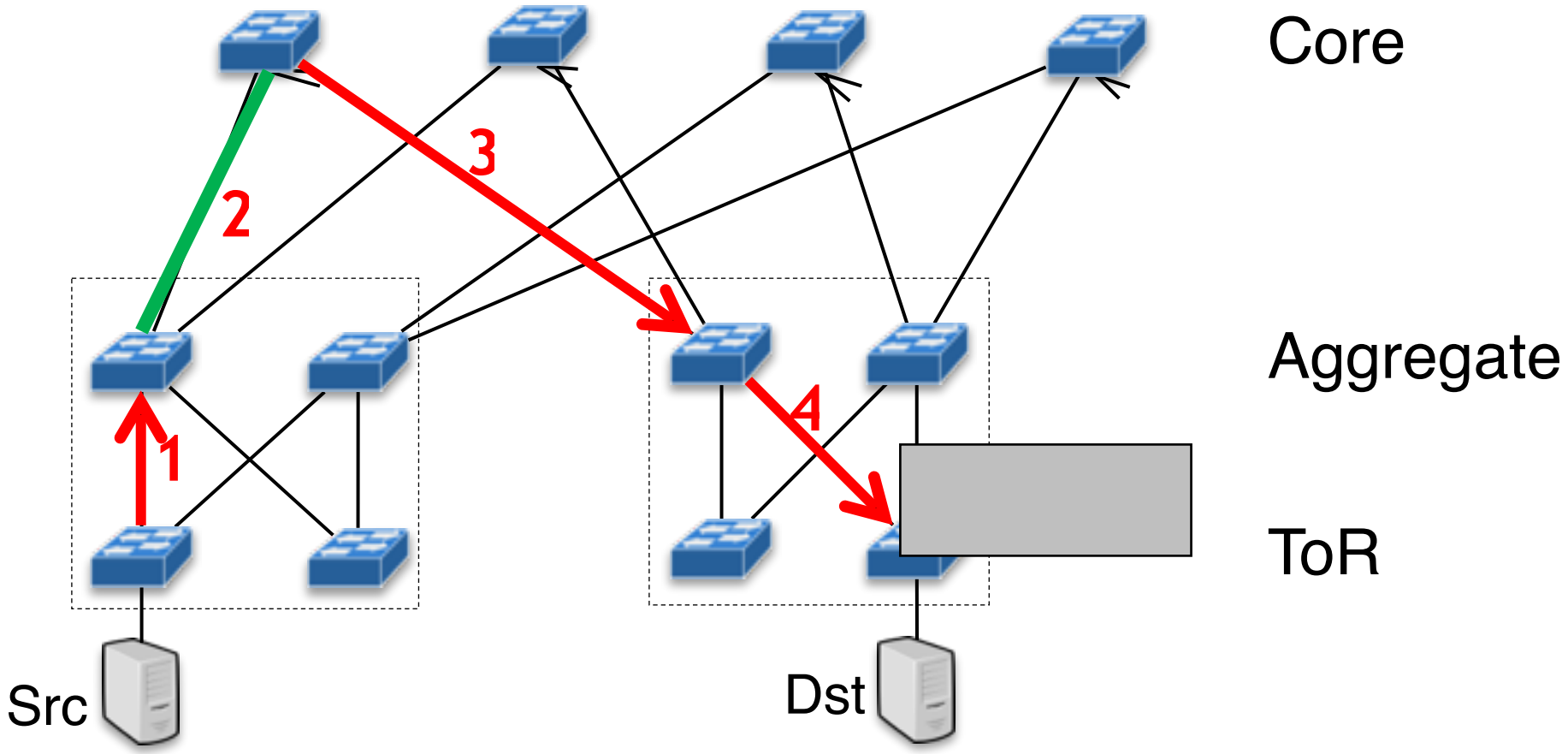
PathDump architecture

1. Switch embeds unique ID (e.g., link ID)



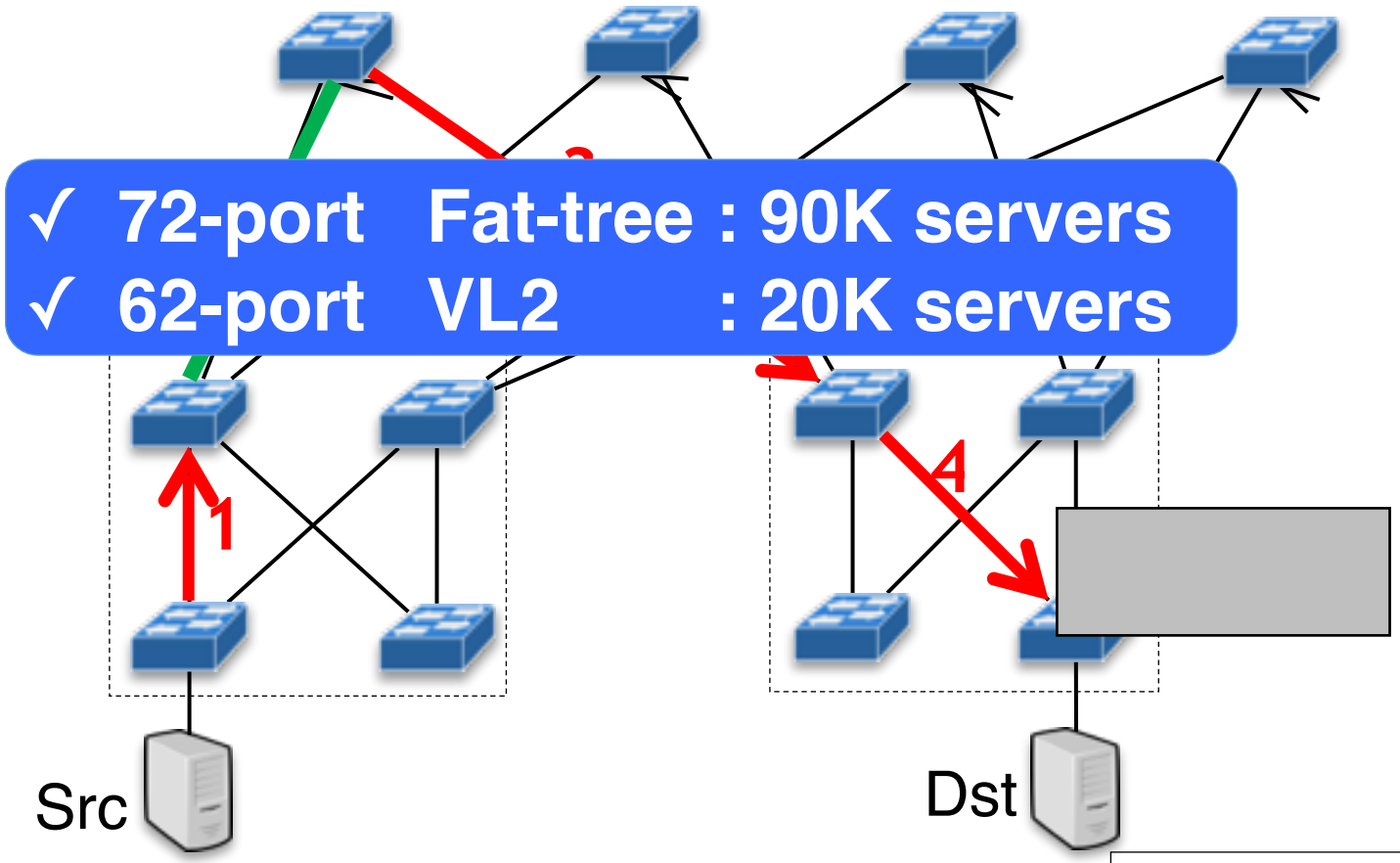
PathDump architecture

1. Switch embeds unique ID (e.g., link ID)



PathDump architecture

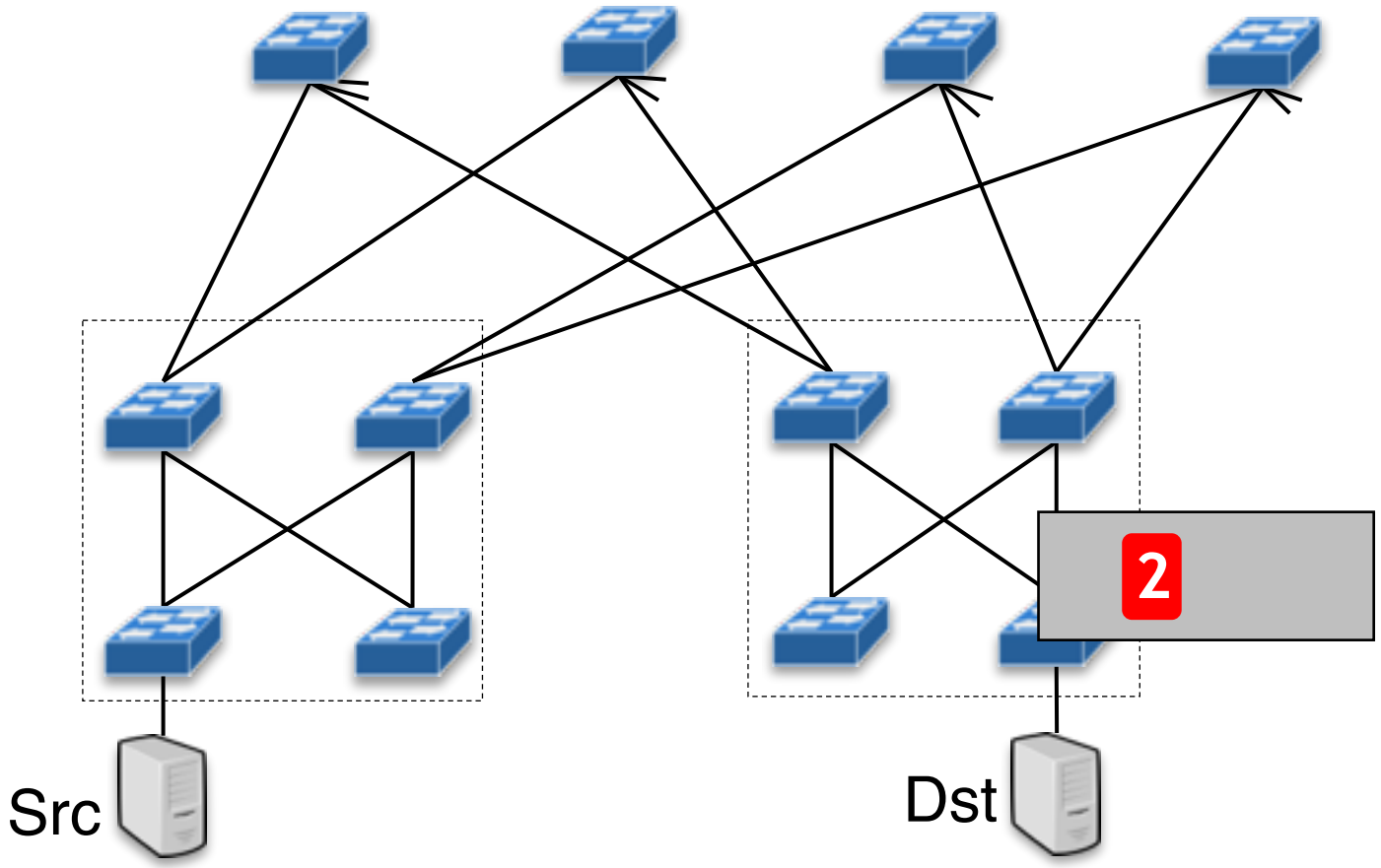
1. Switch embeds unique ID (e.g., link ID)



More details in our paper

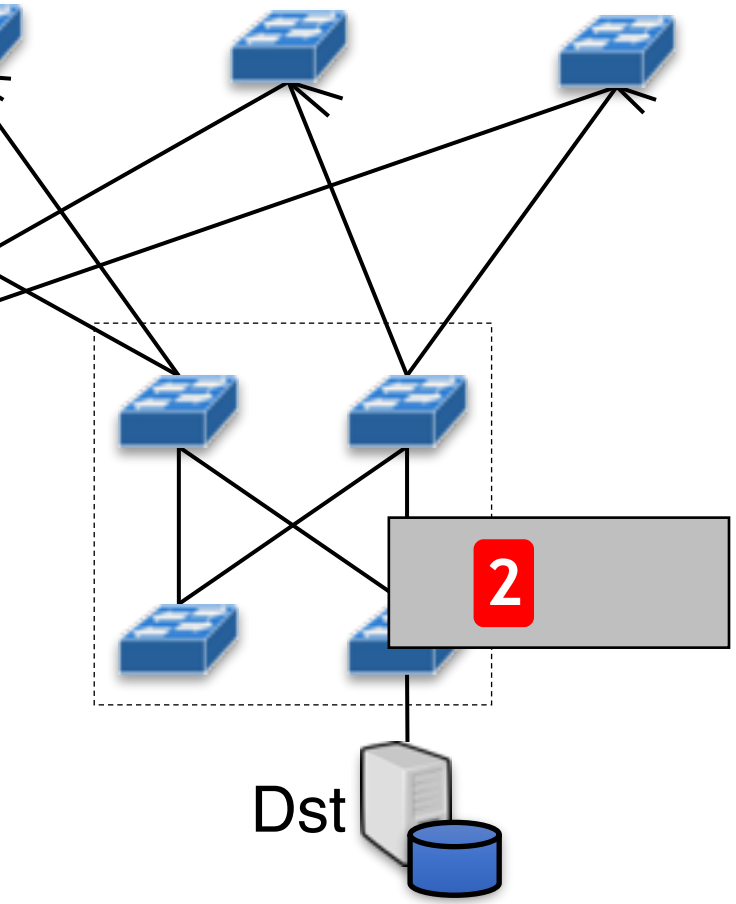
PathDump architecture

2. End-host captures packet path and updates flow-level statistics



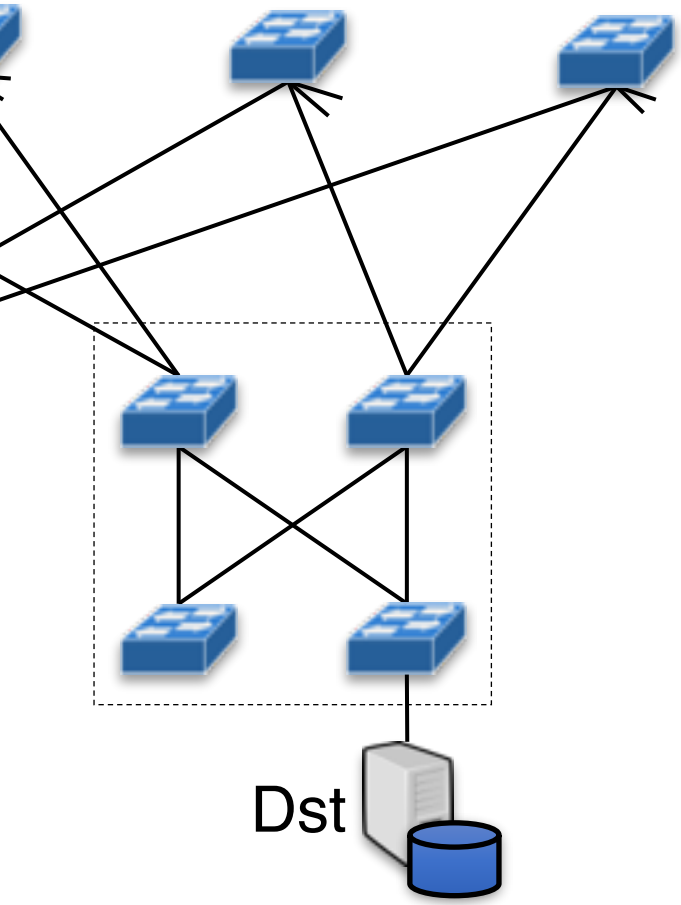
PathDump architecture

2. End-host captures packet path and updates flow-level statistics



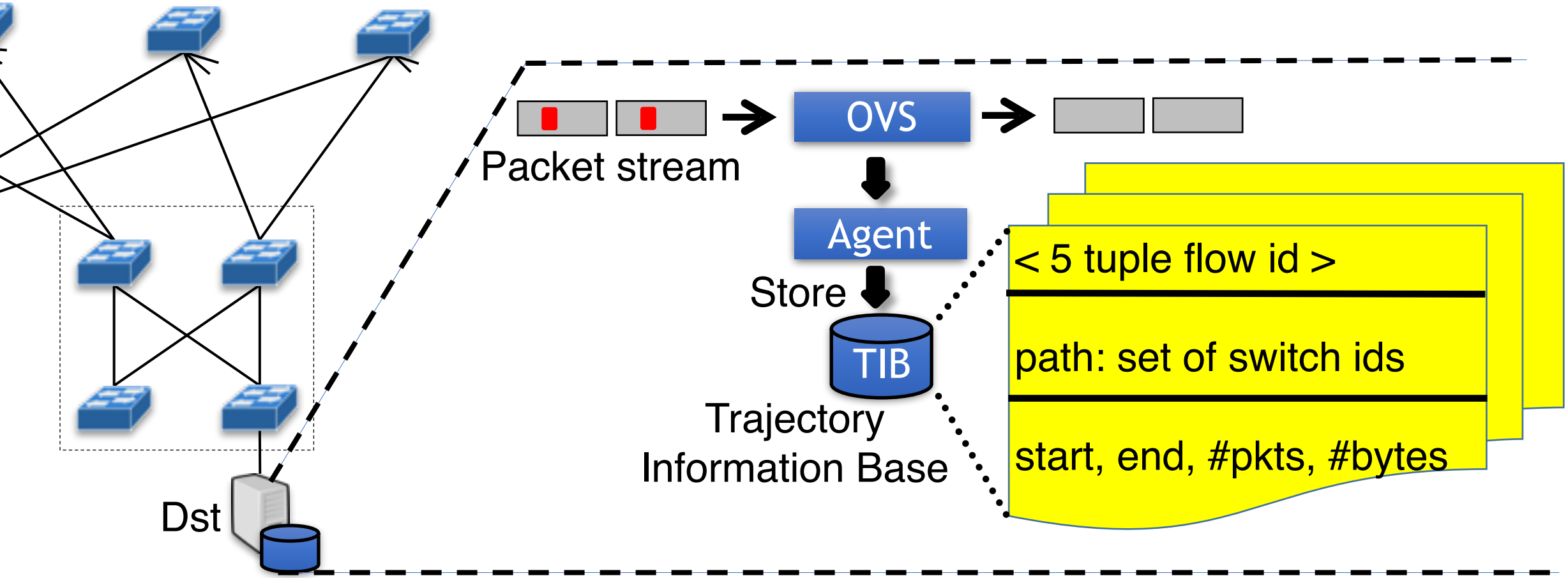
PathDump architecture

2. End-host captures packet path and updates flow-level statistics



PathDump architecture

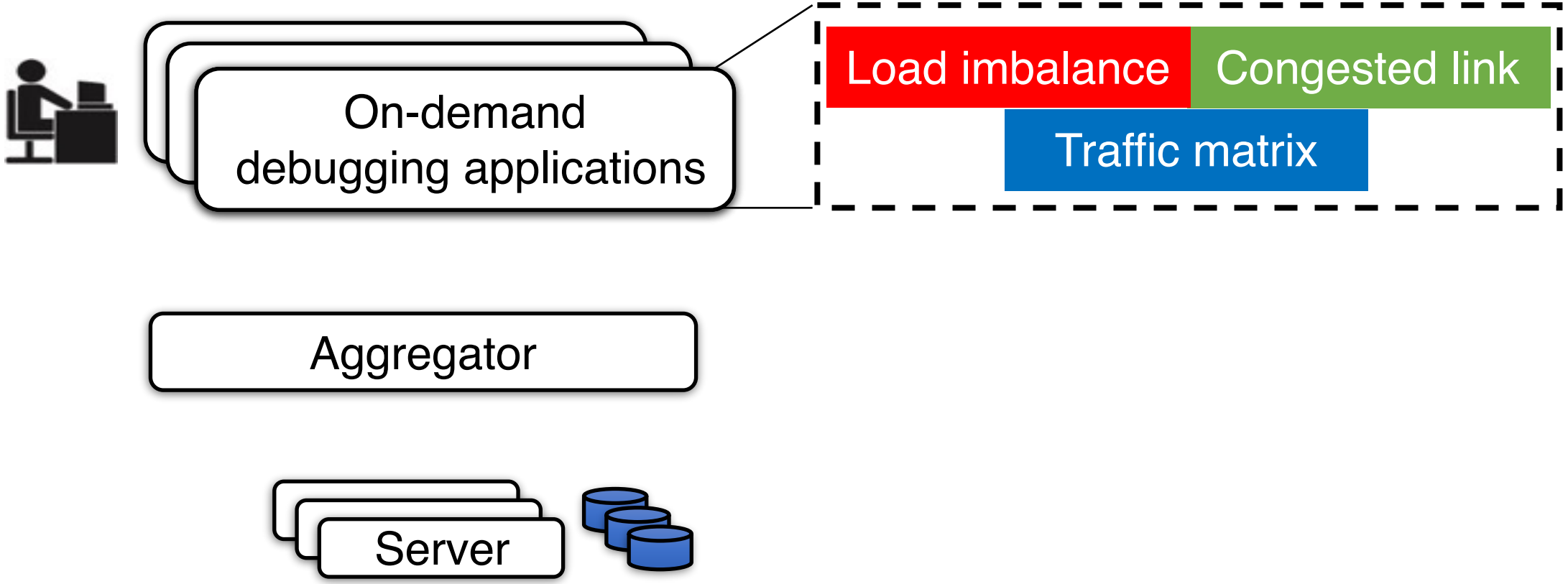
2. End-host captures packet path and updates flow-level statistics



PathDump architecture

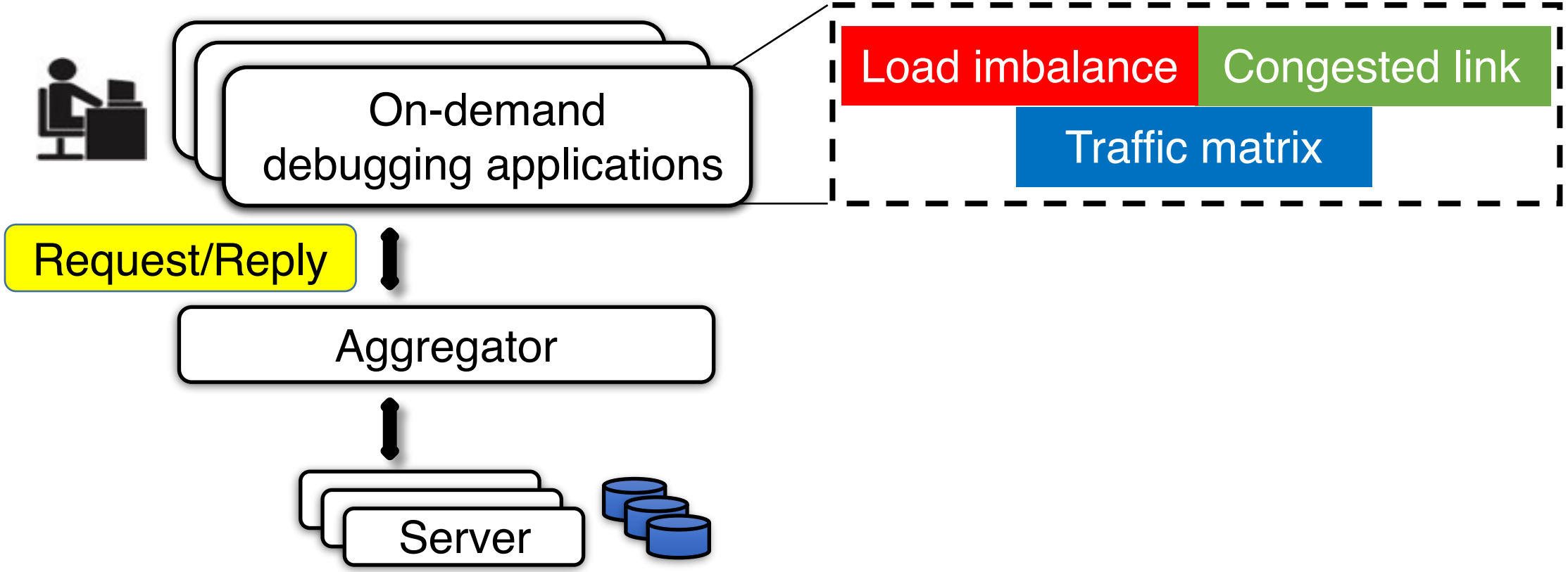
3. Aggregator runs debugging applications

On-demand vs. Event-driven



PathDump architecture

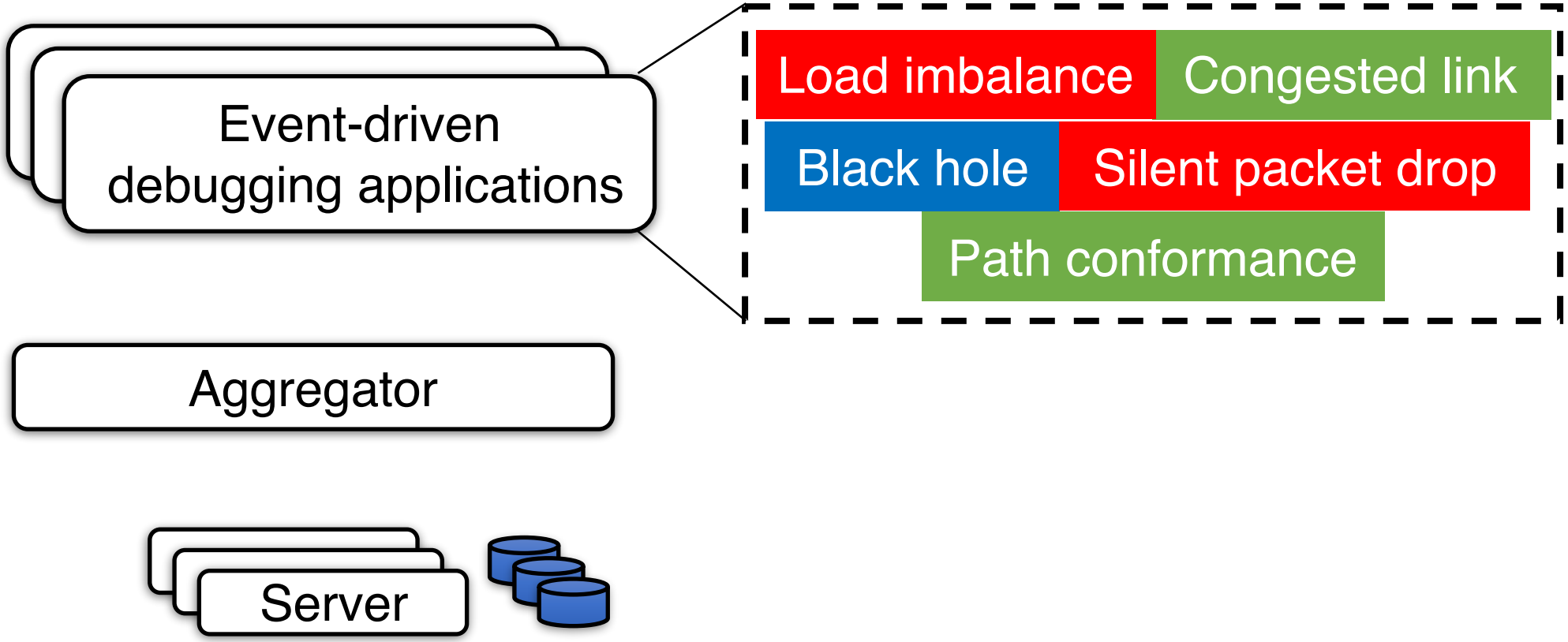
3. Aggregator runs debugging applications
On-demand vs. Event-driven



PathDump architecture

3. Aggregator runs debugging applications

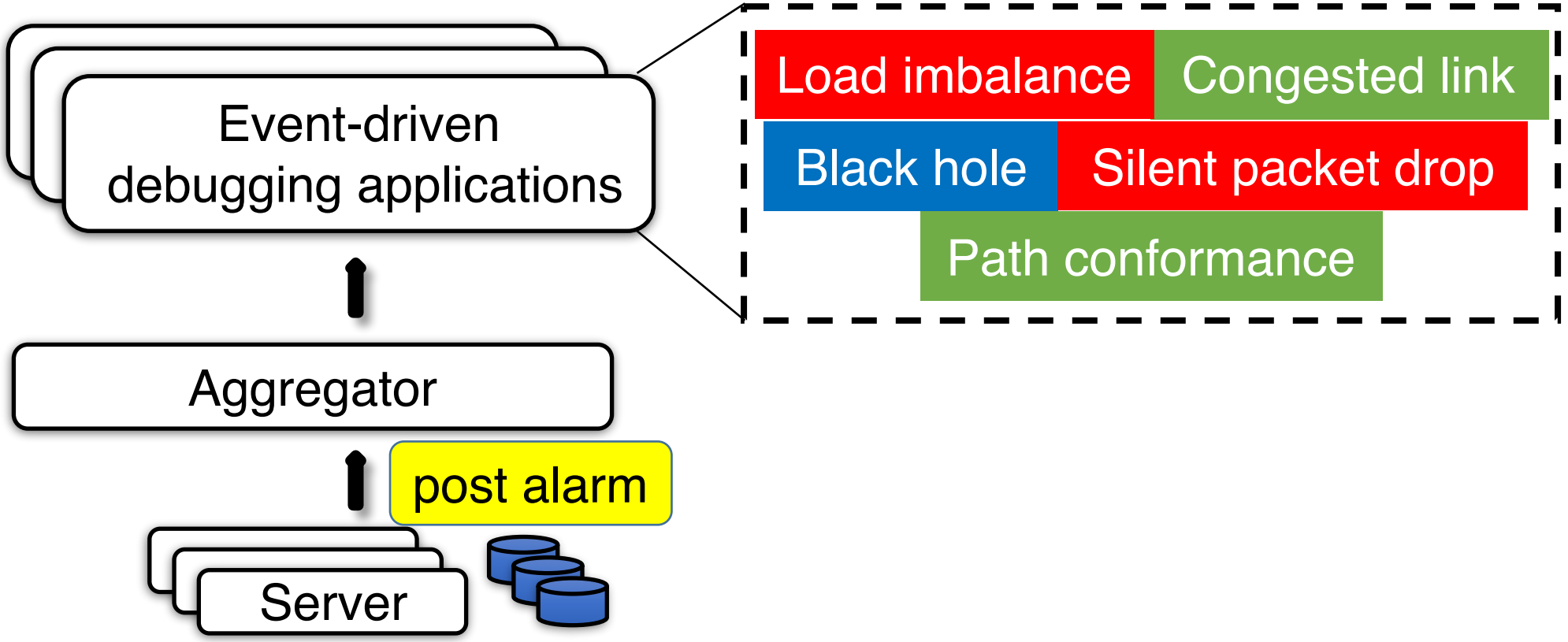
On-demand vs. Event-driven



PathDump architecture

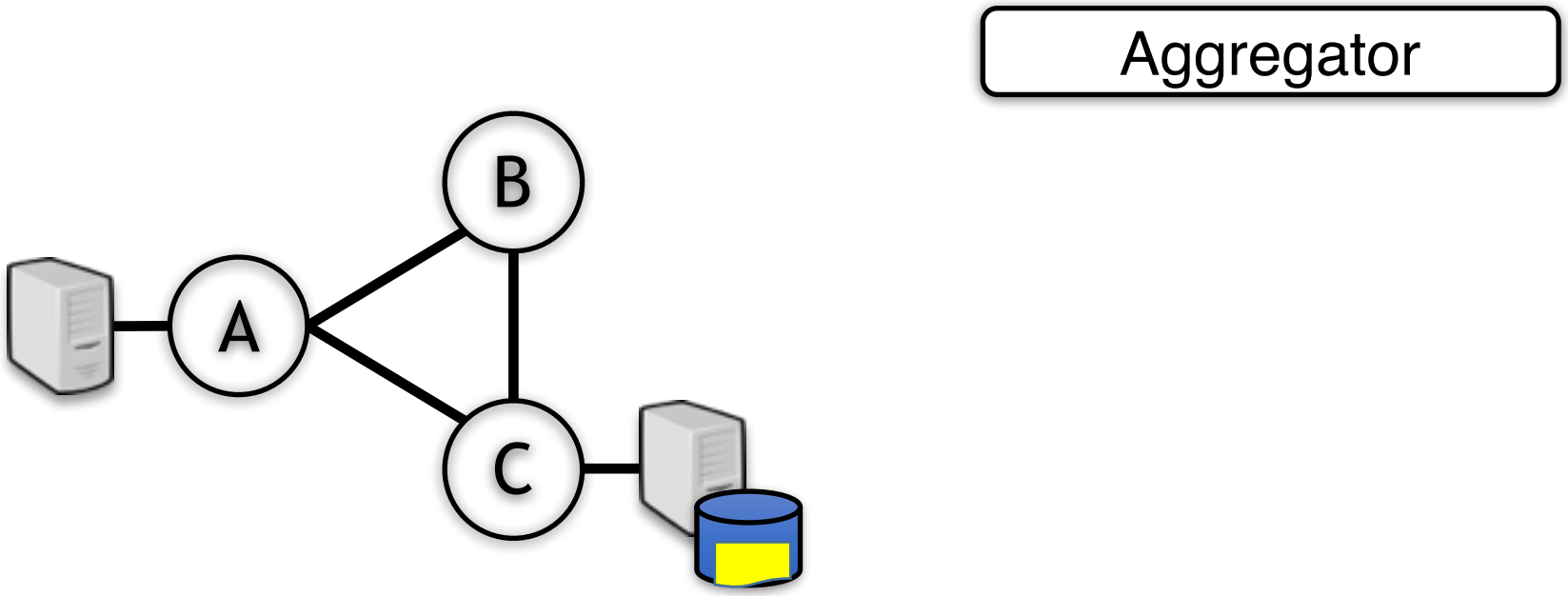
3. Aggregator runs debugging applications

On-demand vs. Event-driven



PathDump interface

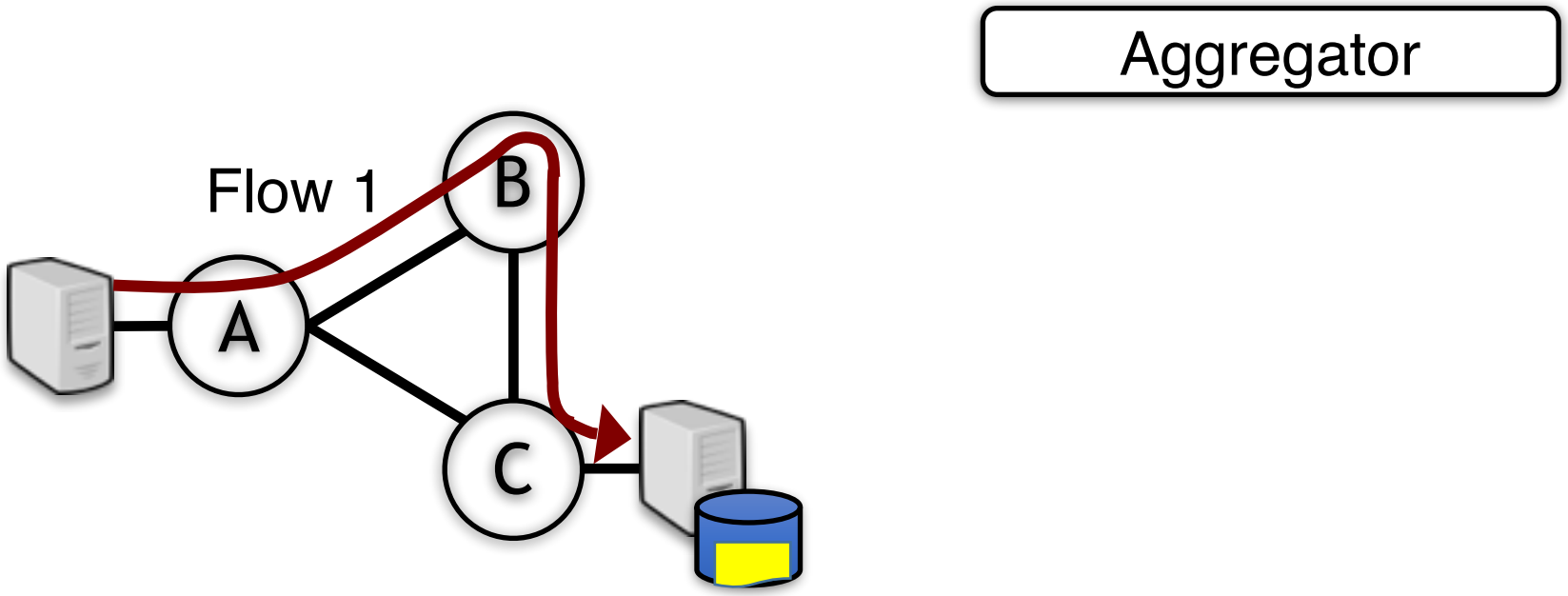
A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

PathDump interface

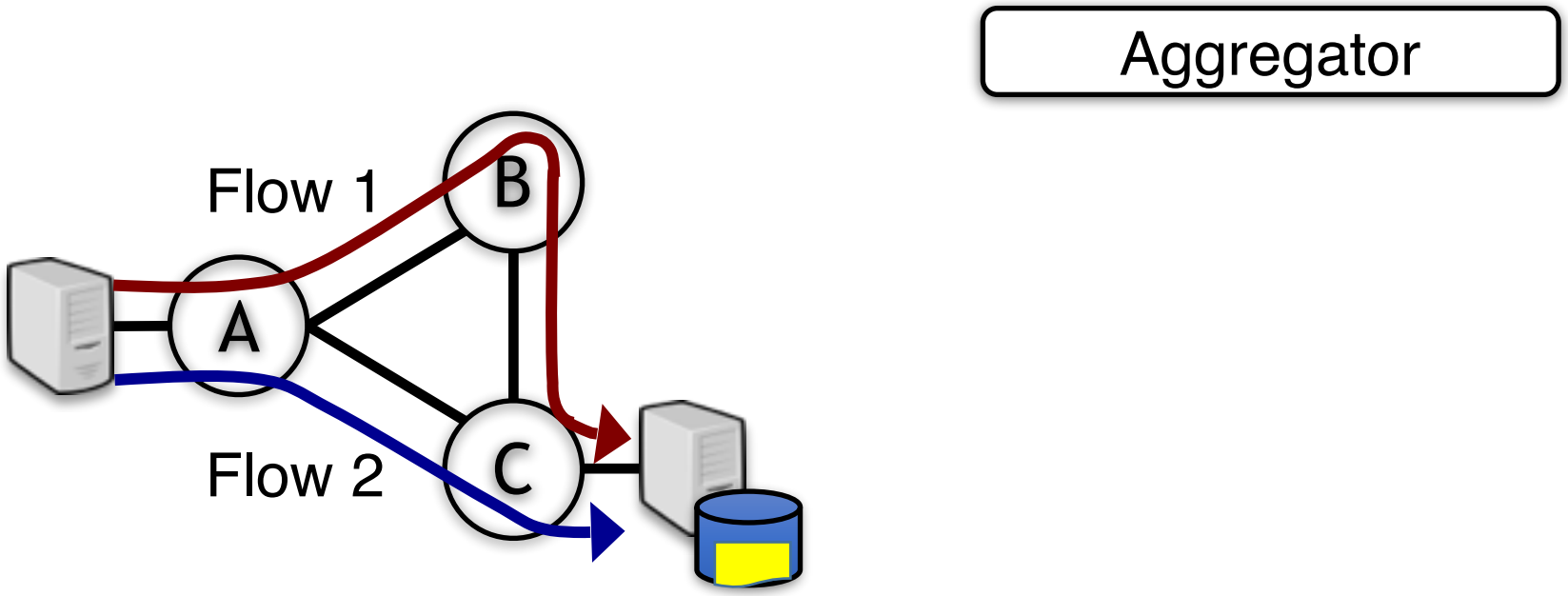
A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

PathDump interface

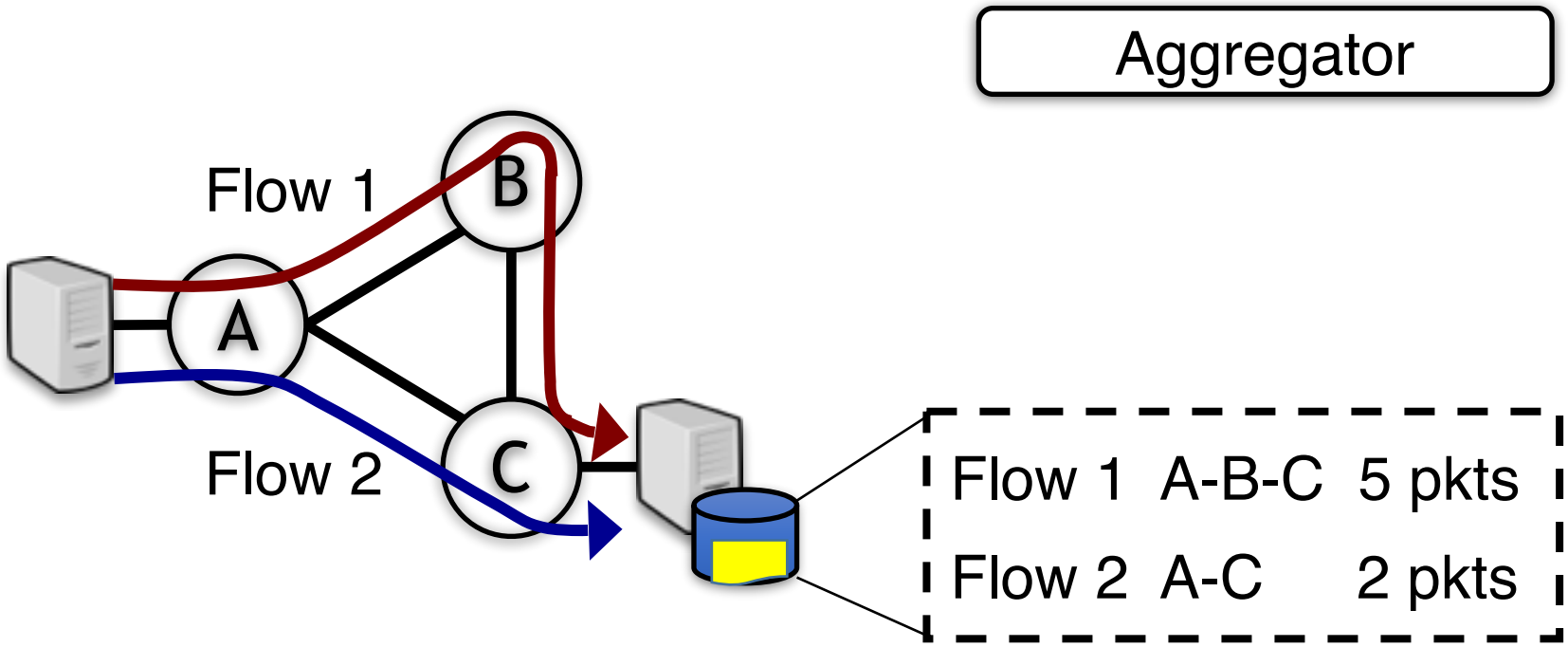
A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

PathDump interface

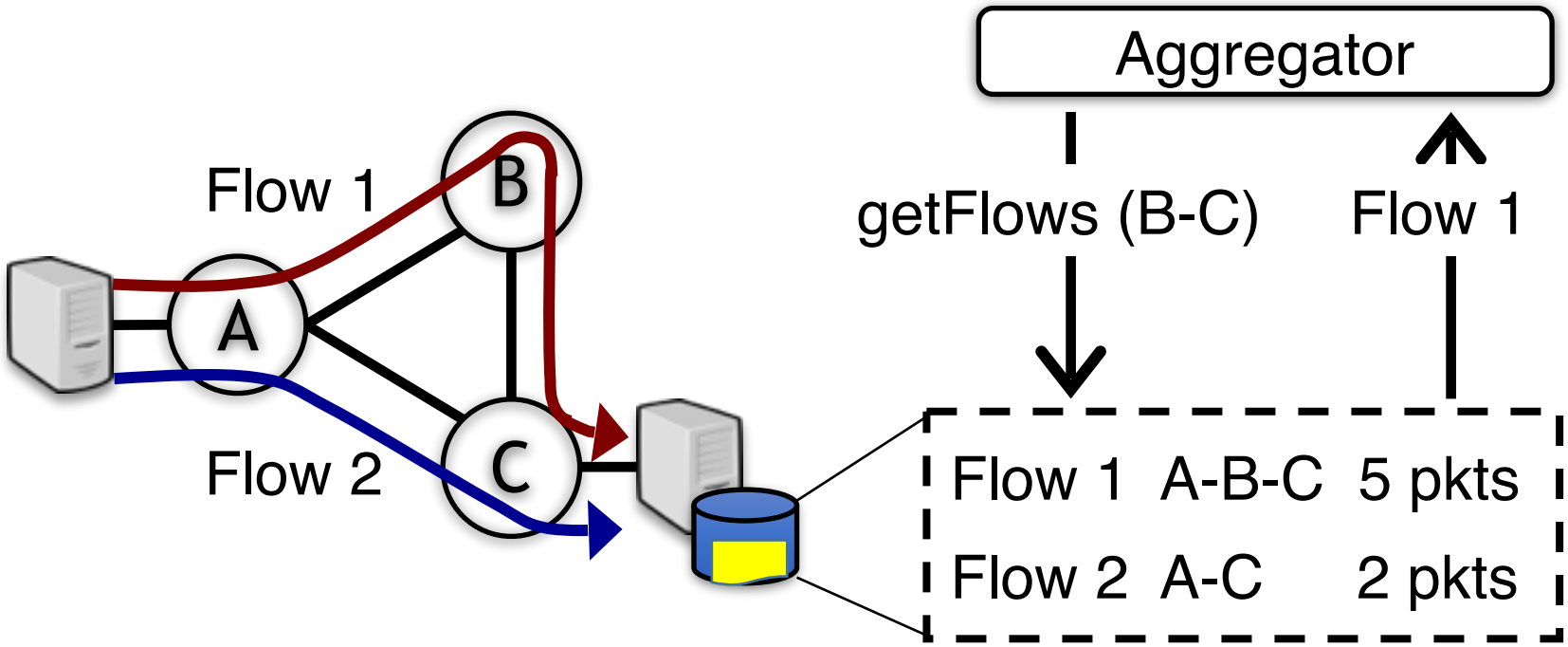
A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

PathDump interface

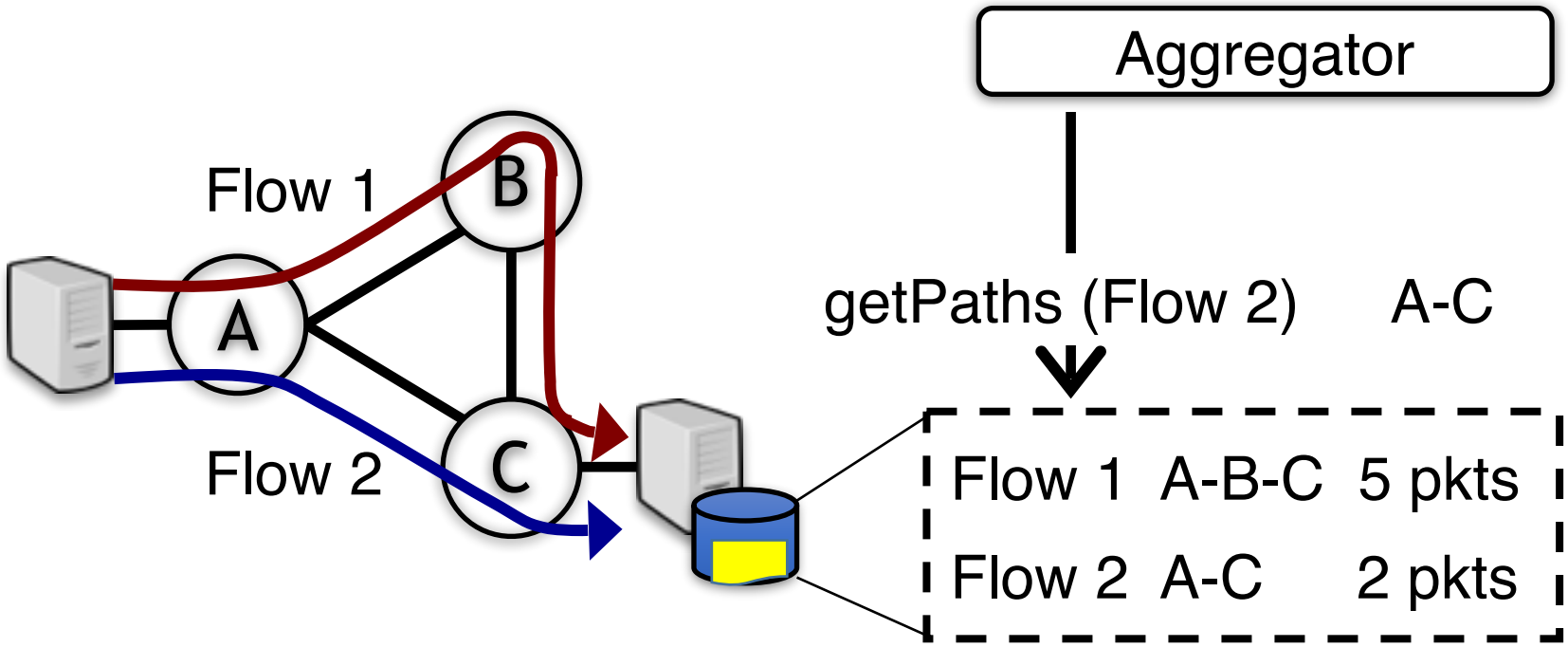
A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

PathDump interface

A small set of simple APIs enables a variety of debugging applications



- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

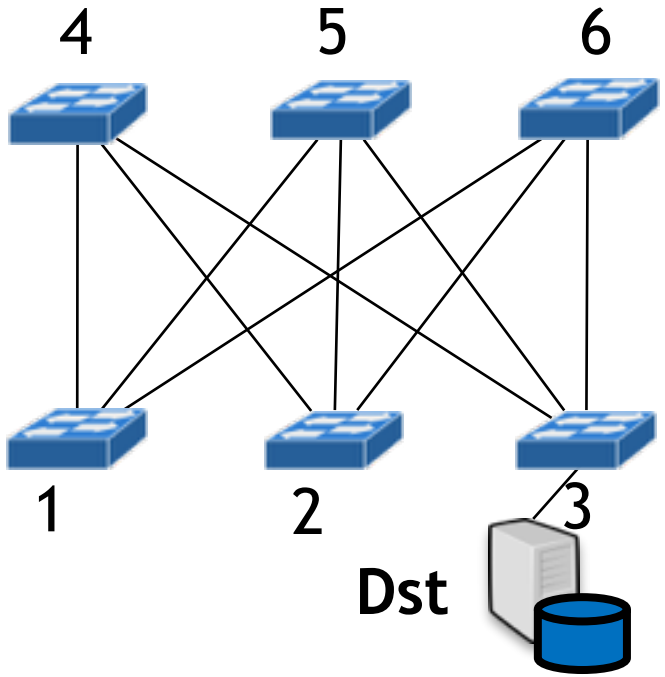
Example 1: Path conformance

Example 1: Path conformance

Check if actual forwarding path != network policy

- May occur due to switch faults or network state change

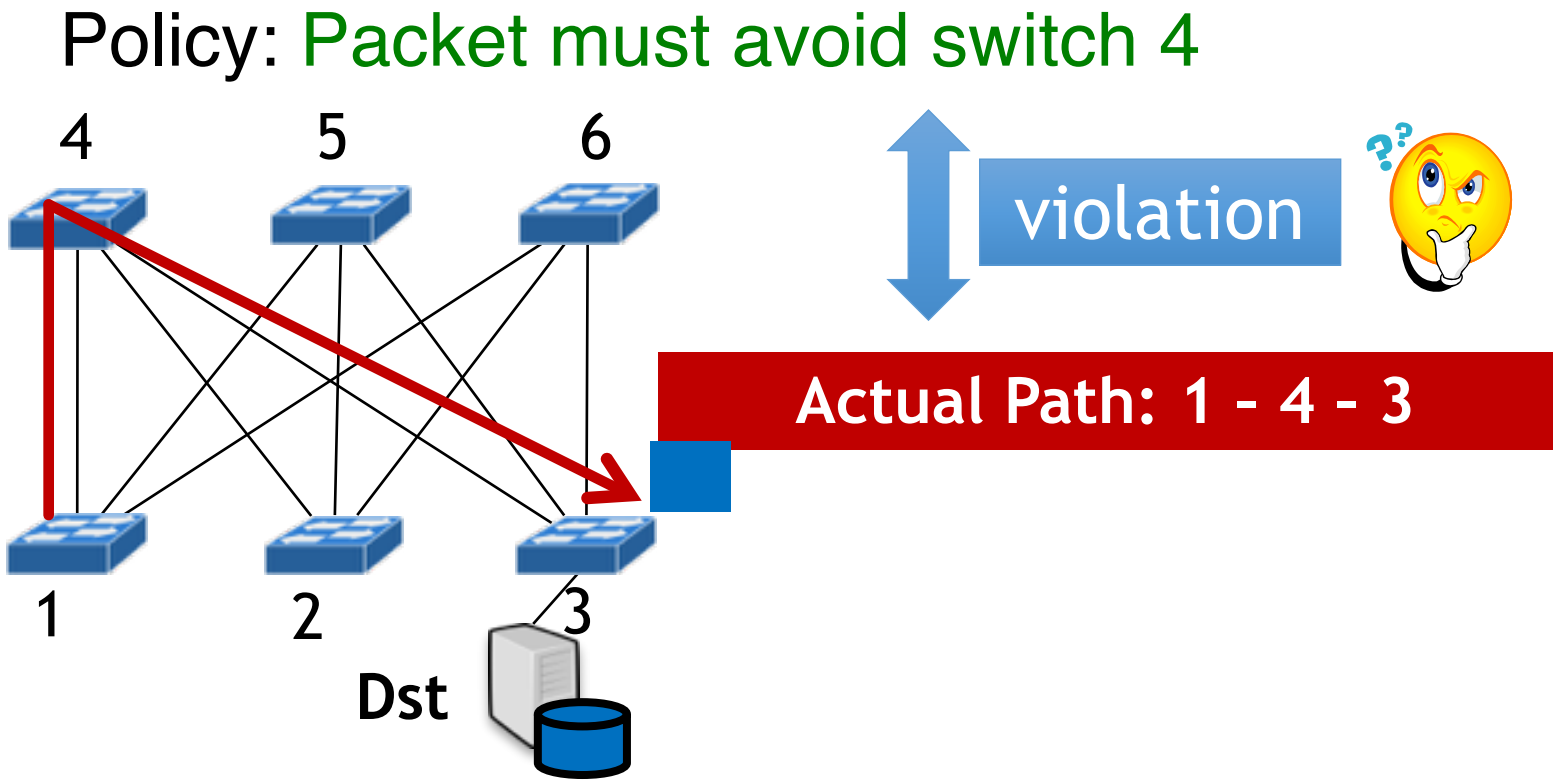
Policy: Packet must avoid switch 4



Example 1: Path conformance

Check if actual forwarding path != network policy

- May occur due to switch faults or network state change

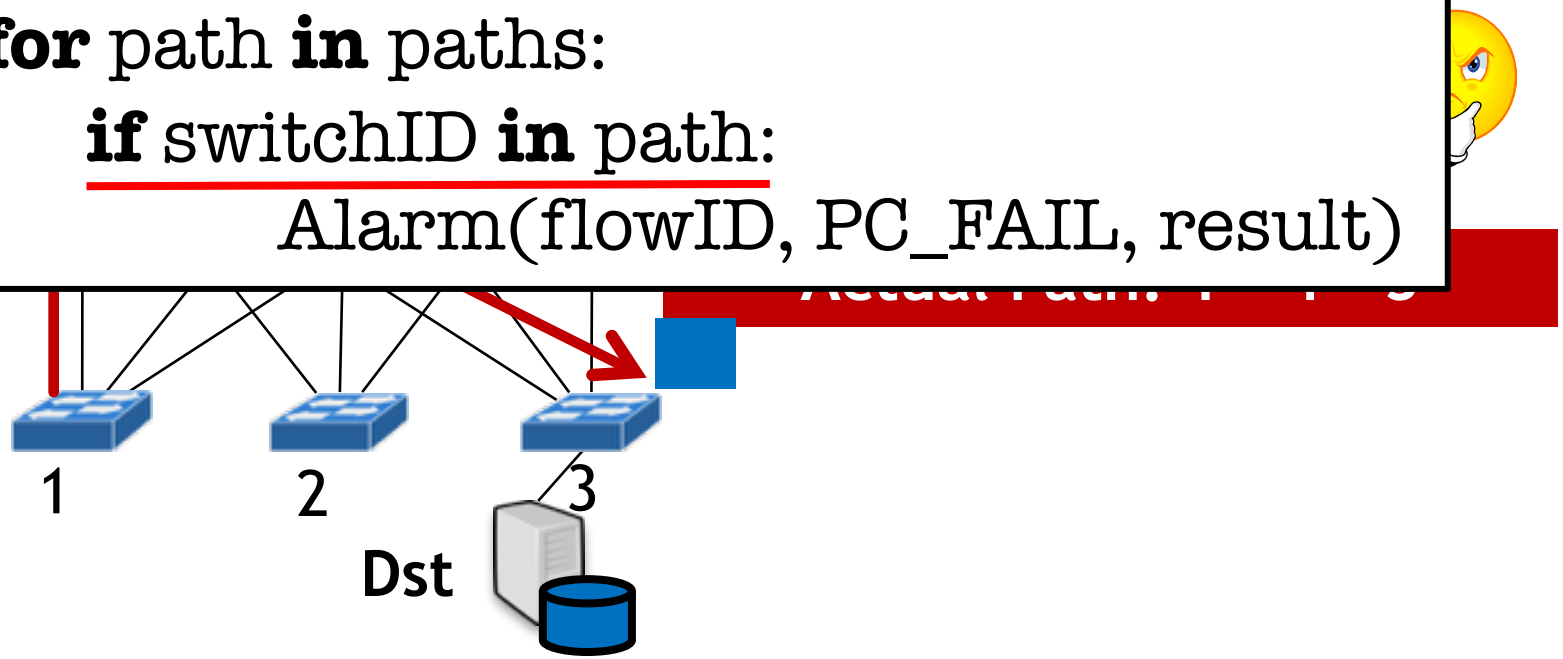


Example 1: Path conformance

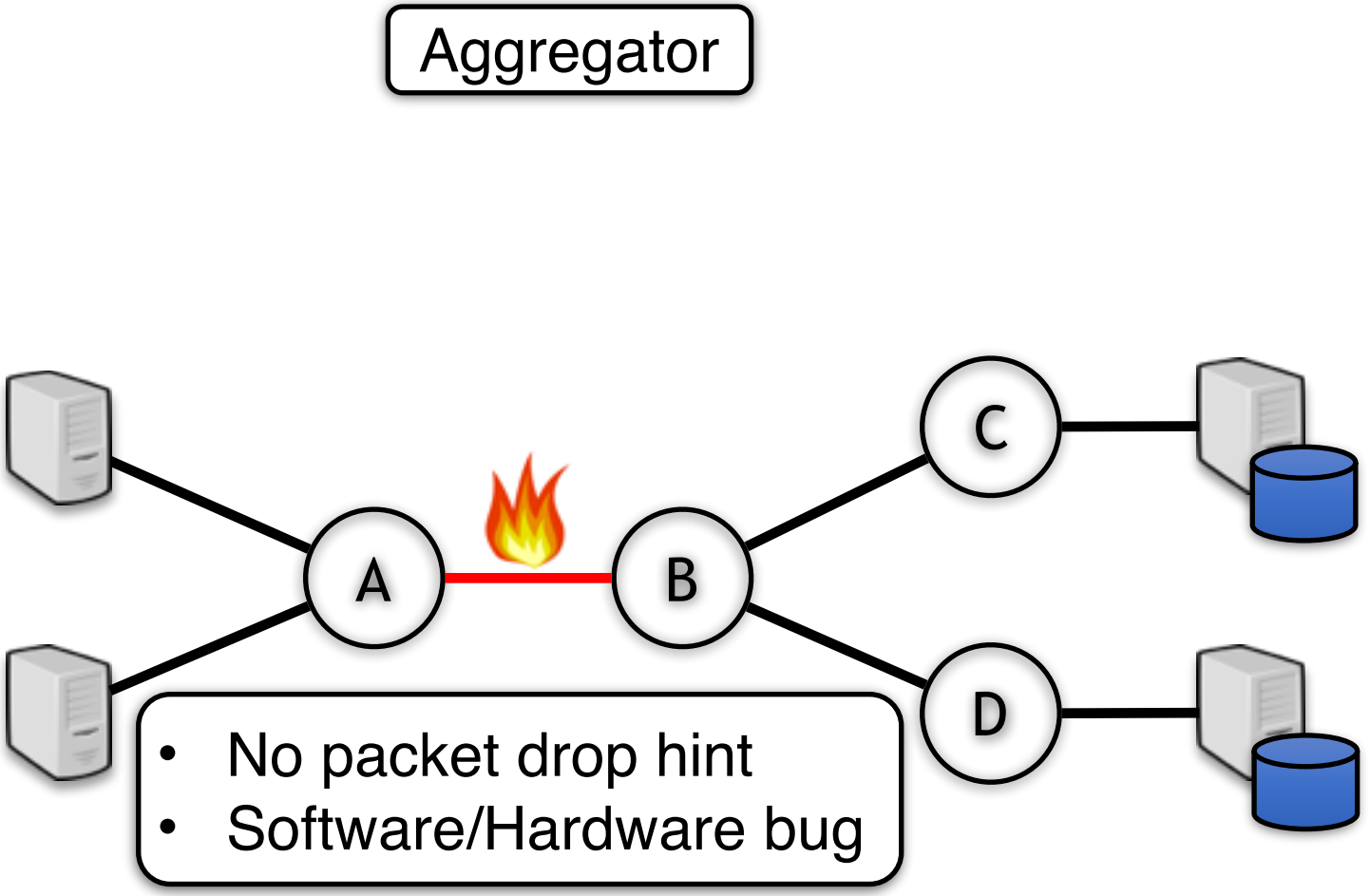
Check if actual forwarding path != network policy

- May occur due to switch faults or network state change

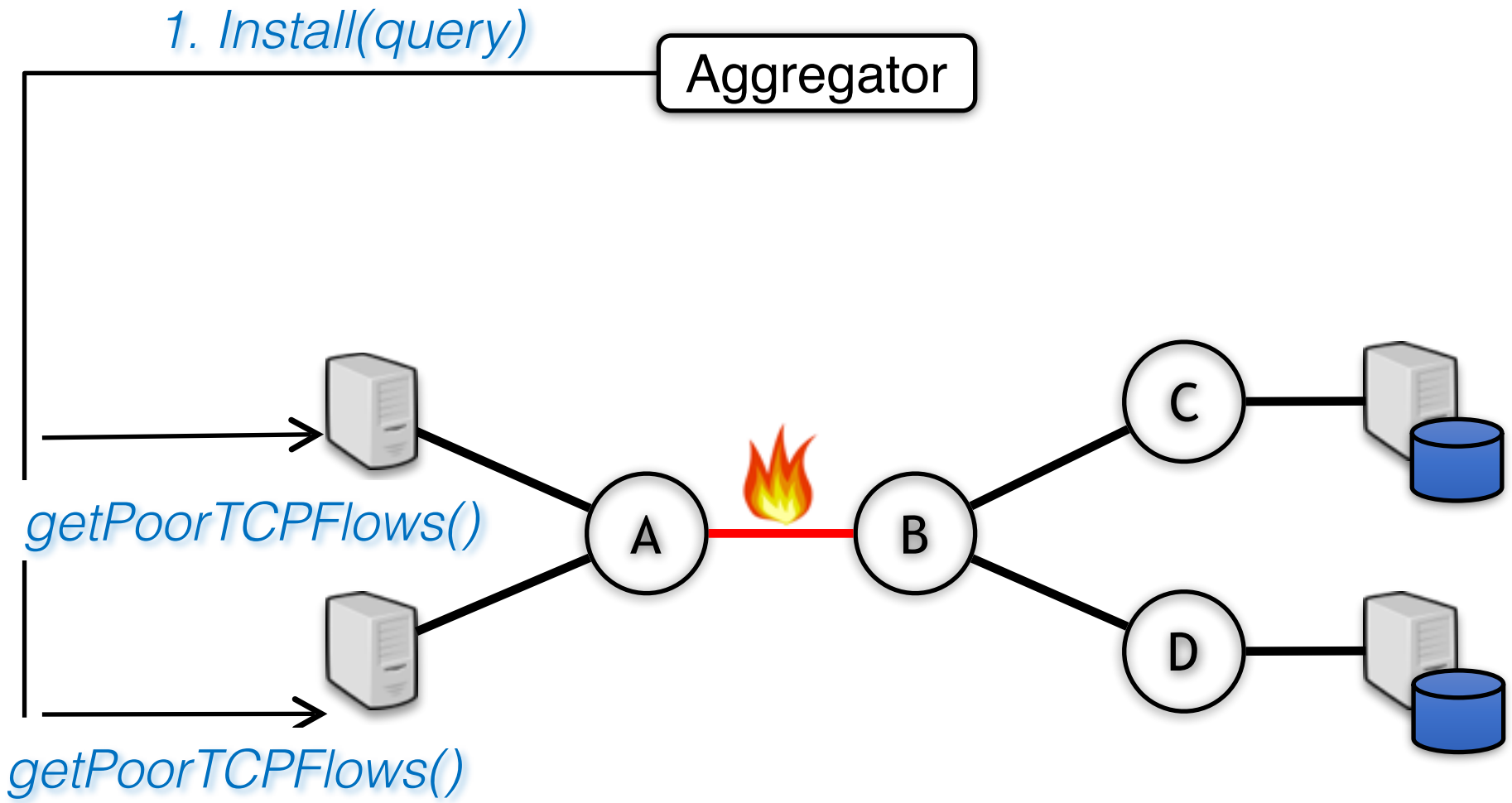
```
# Given flowID, paths, switchID

1: for path in paths:
2:     if switchID in path:
3:         Alarm(flowID, PC_FAIL, result)
```



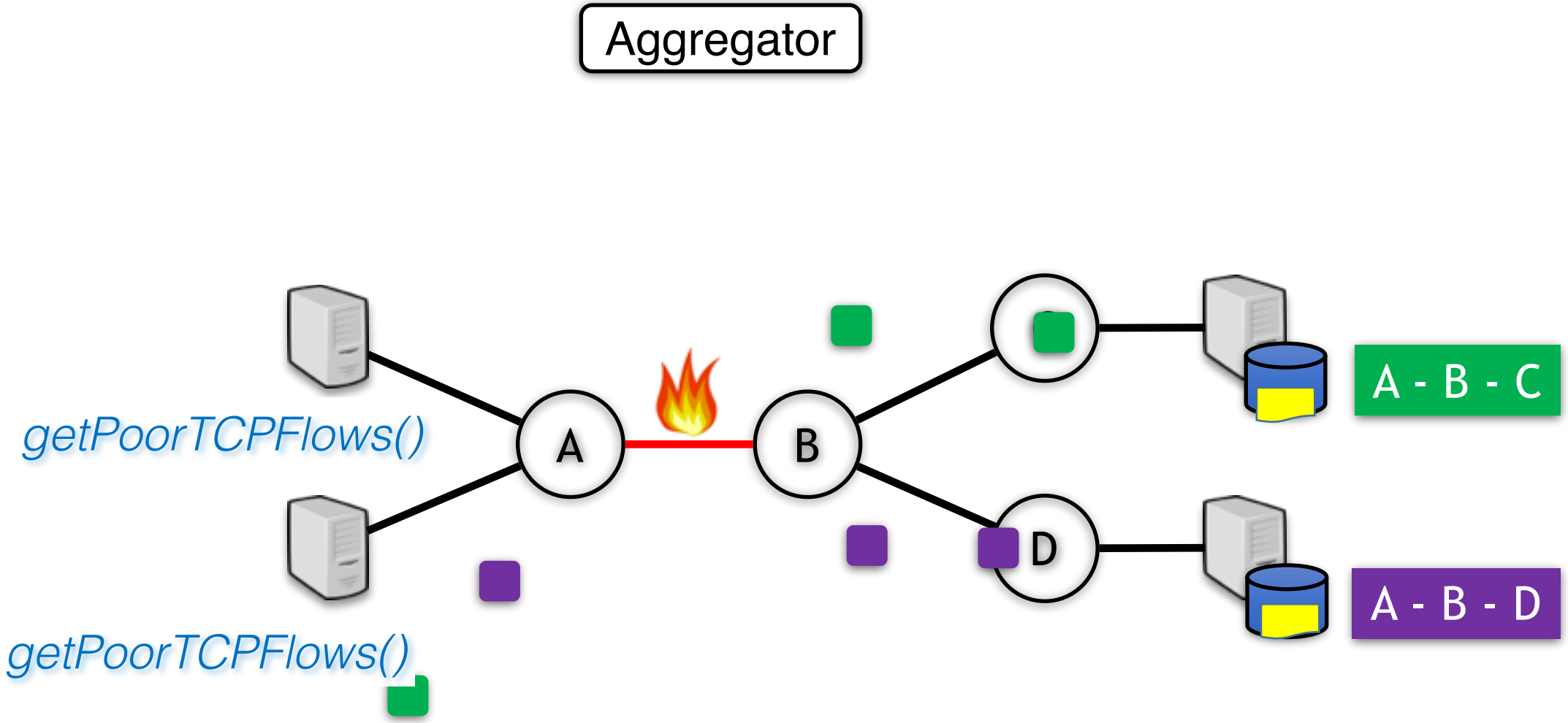
Example 2: Silent random packet drop diagnosis



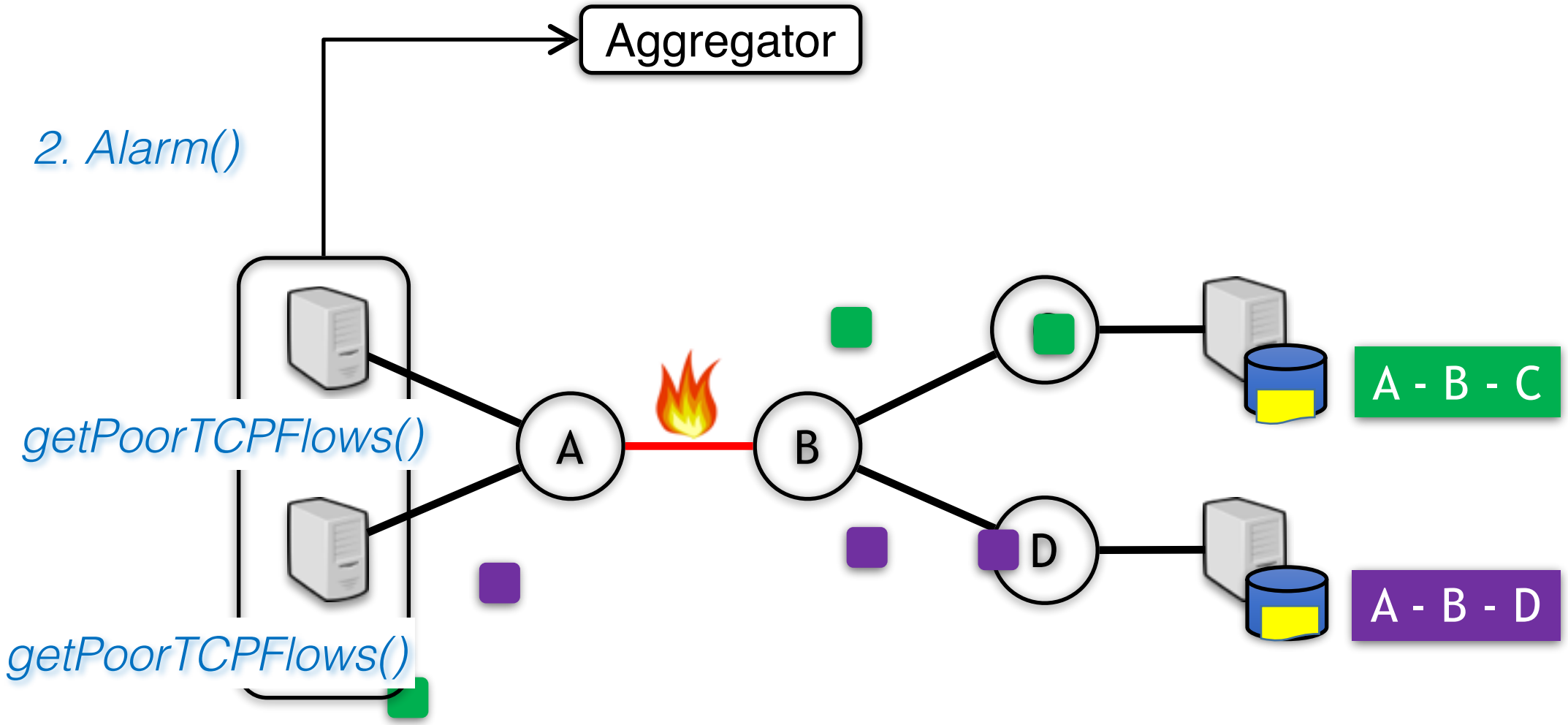
Example 2: Silent random packet drop diagnosis



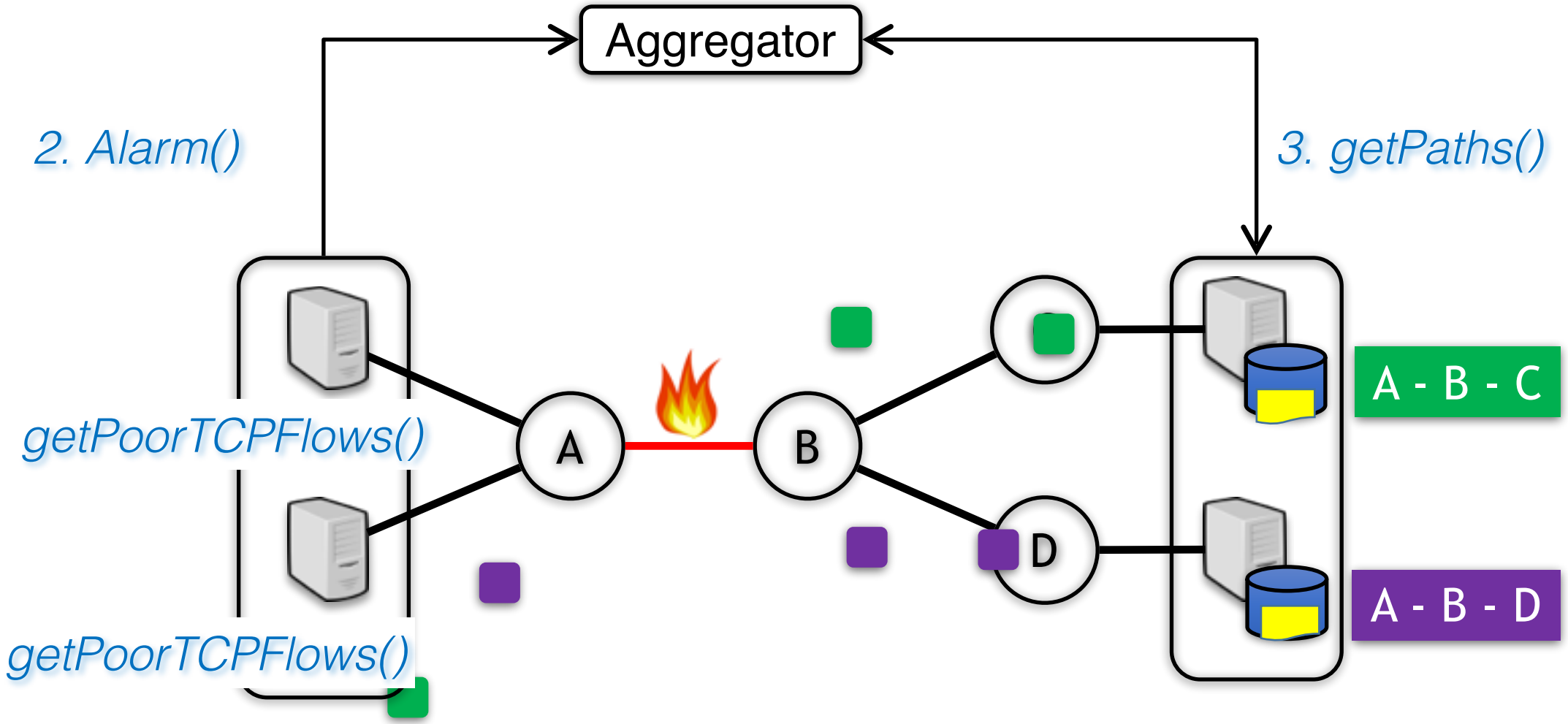
Example 2: Silent random packet drop diagnosis



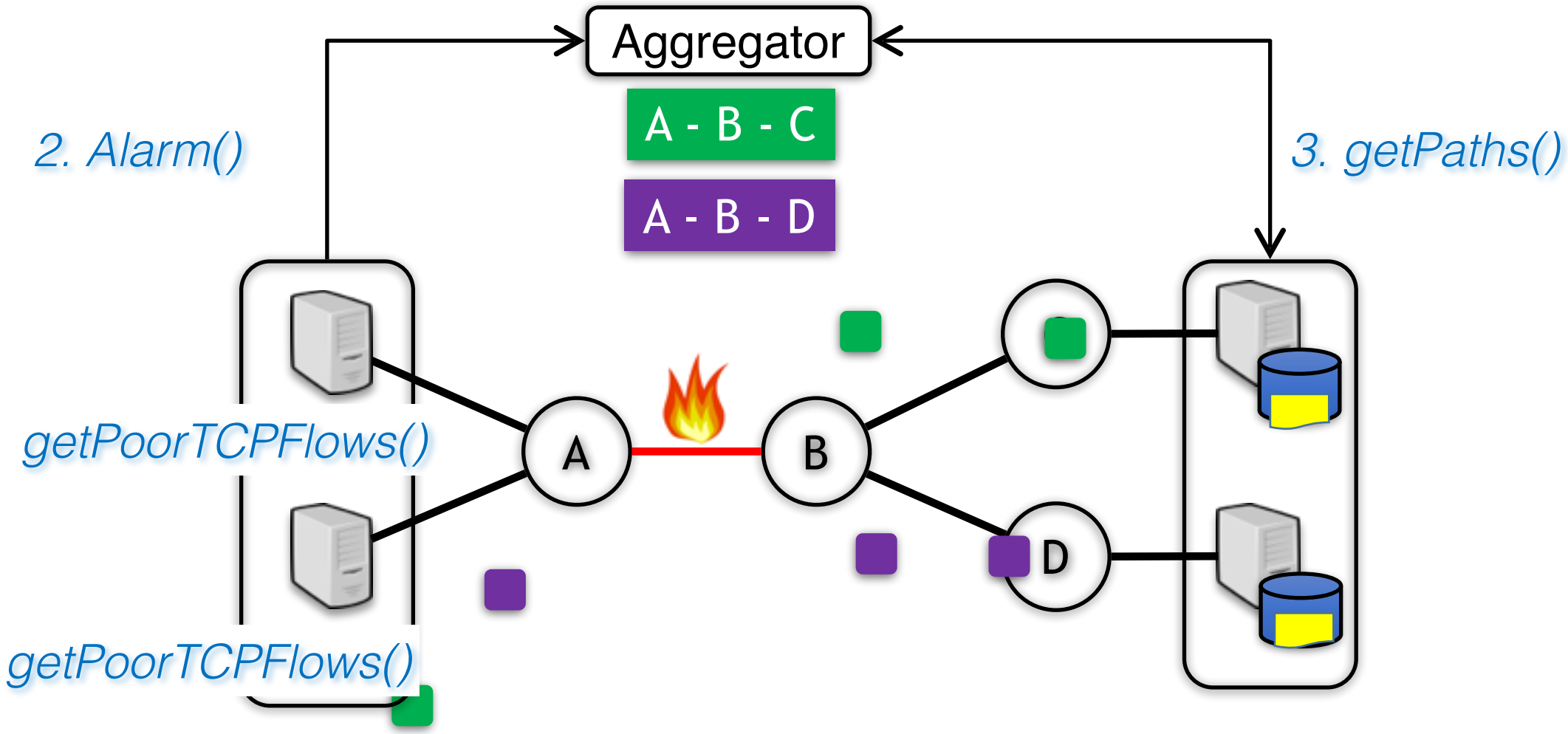
Example 2: Silent random packet drop diagnosis



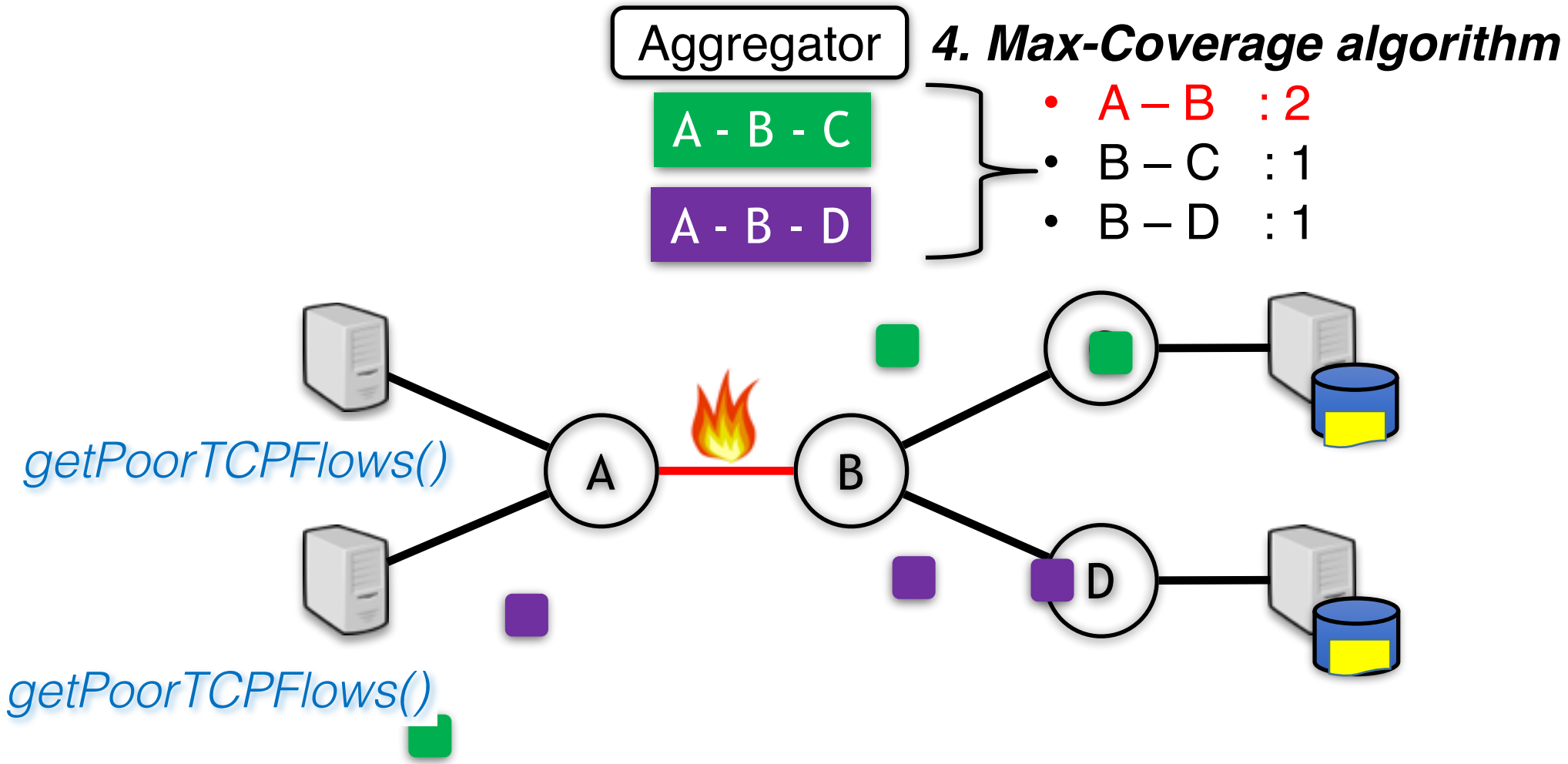
Example 2: Silent random packet drop diagnosis



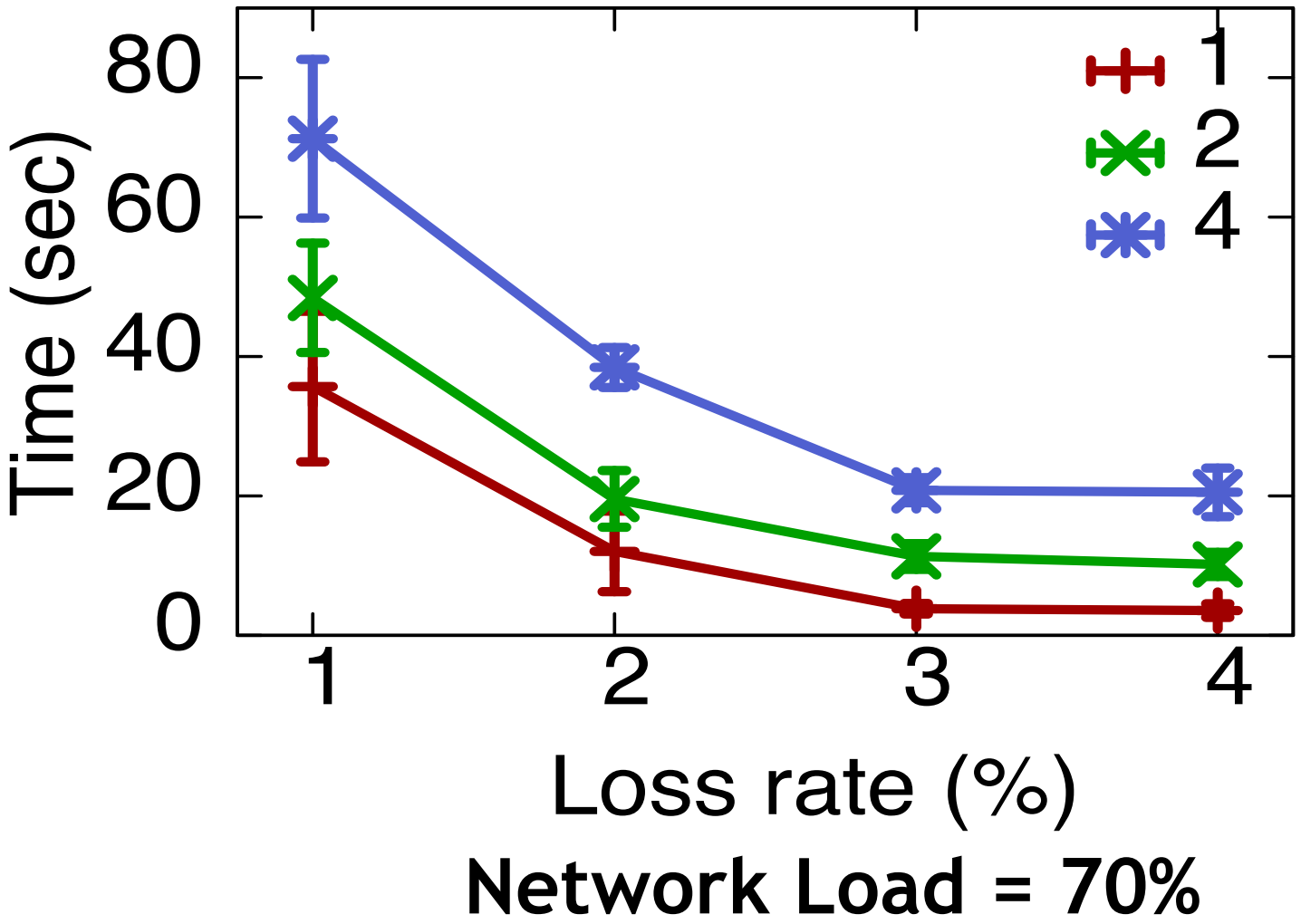
Example 2: Silent random packet drop diagnosis



Example 2: Silent random packet drop diagnosis



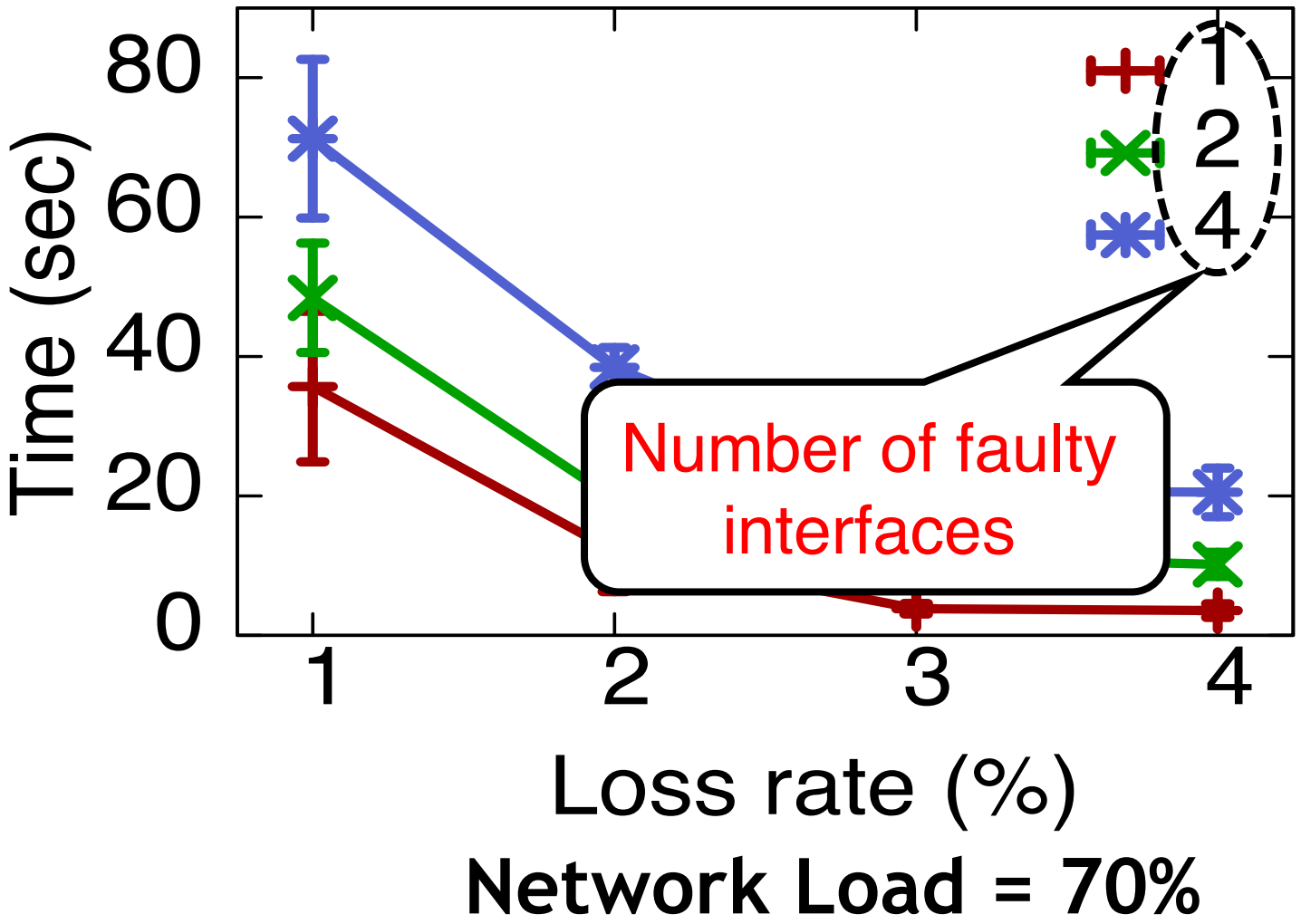
Example 2: Silent random packet drop diagnosis



Lab setup

- 4-ary fat-tree topology
- Web-traffic model

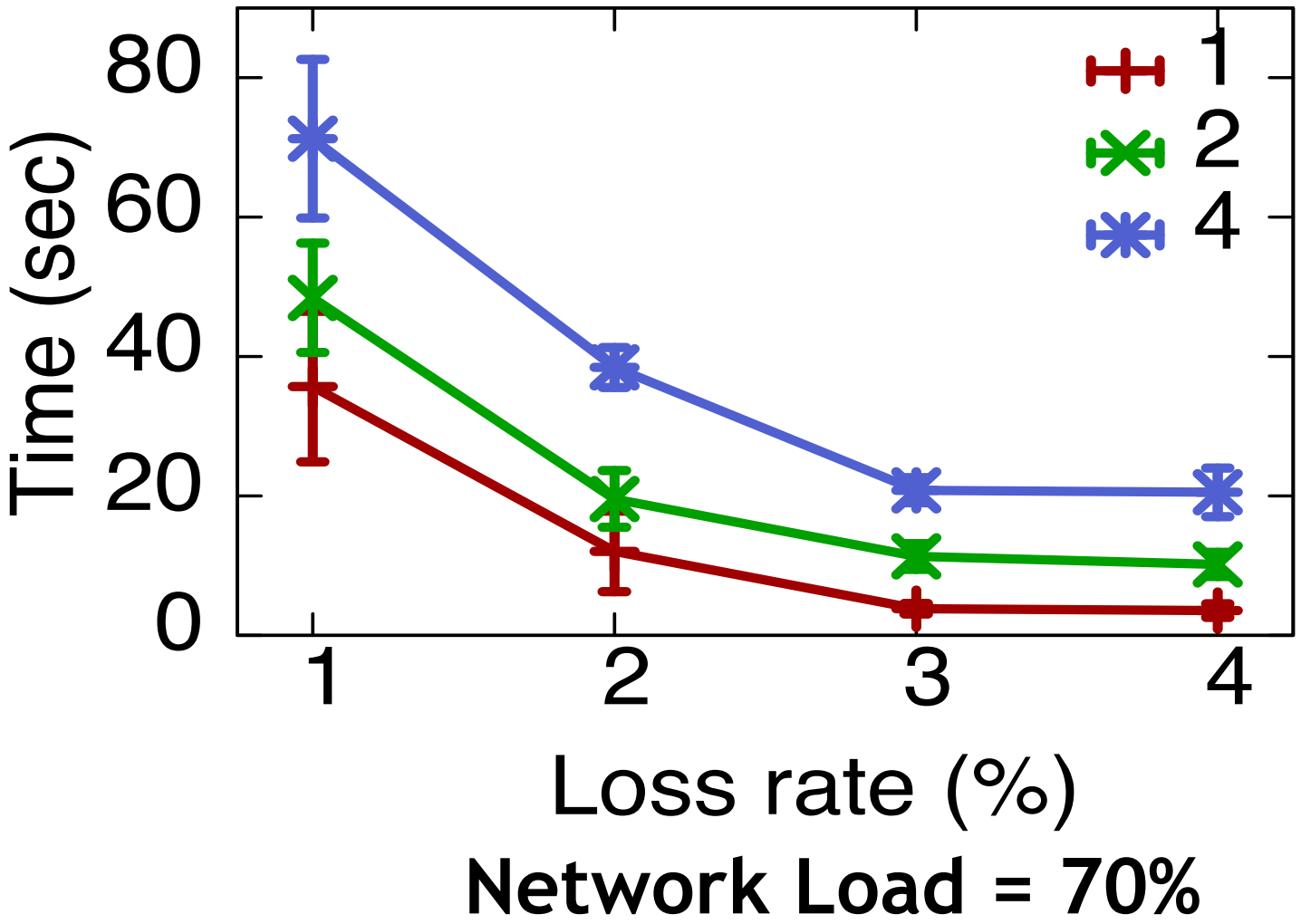
Example 2: Silent random packet drop diagnosis



Lab setup

- 4-ary fat-tree topology
- Web-traffic model

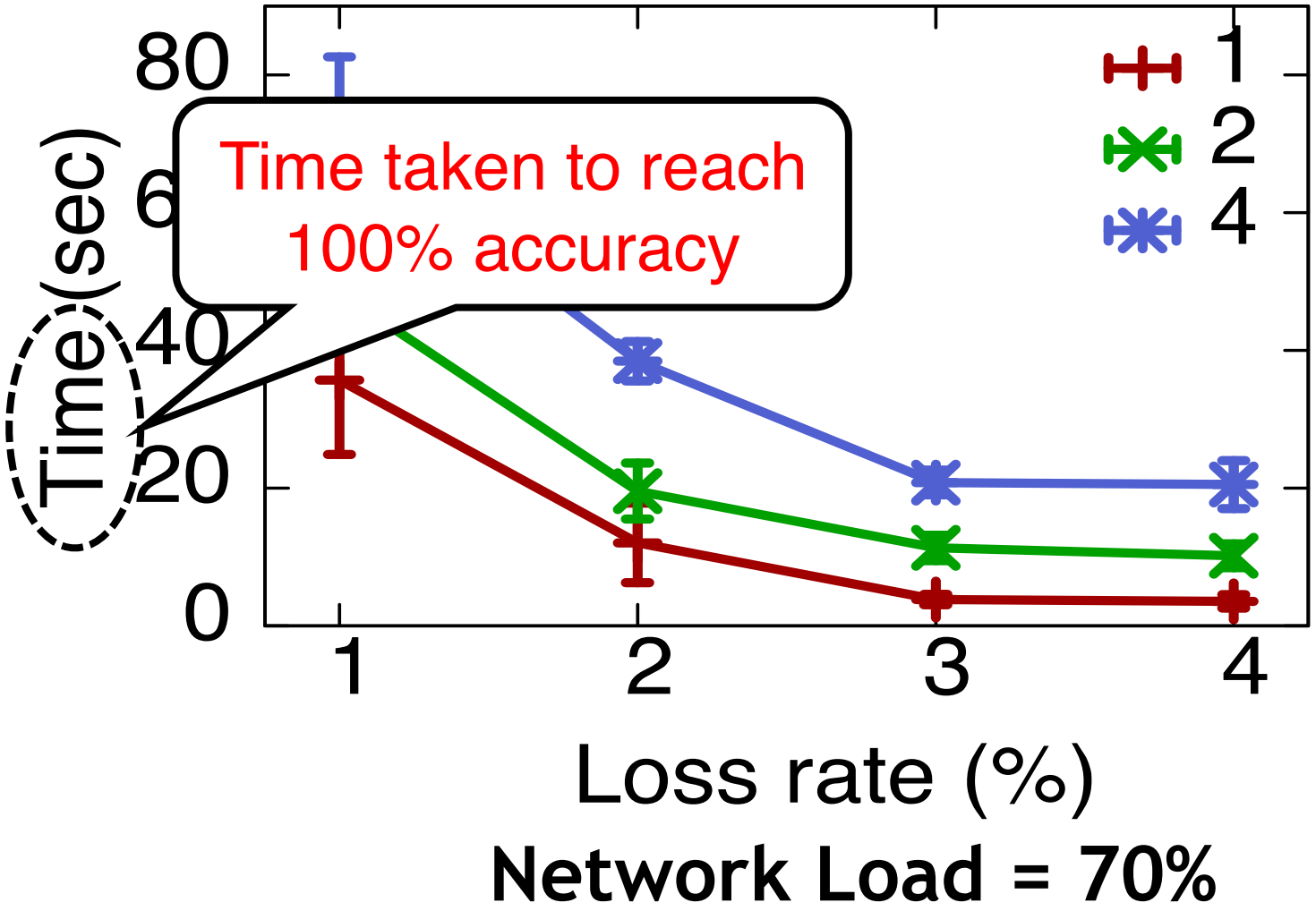
Example 2: Silent random packet drop diagnosis



Lab setup

- 4-ary fat-tree topology
- Web-traffic model

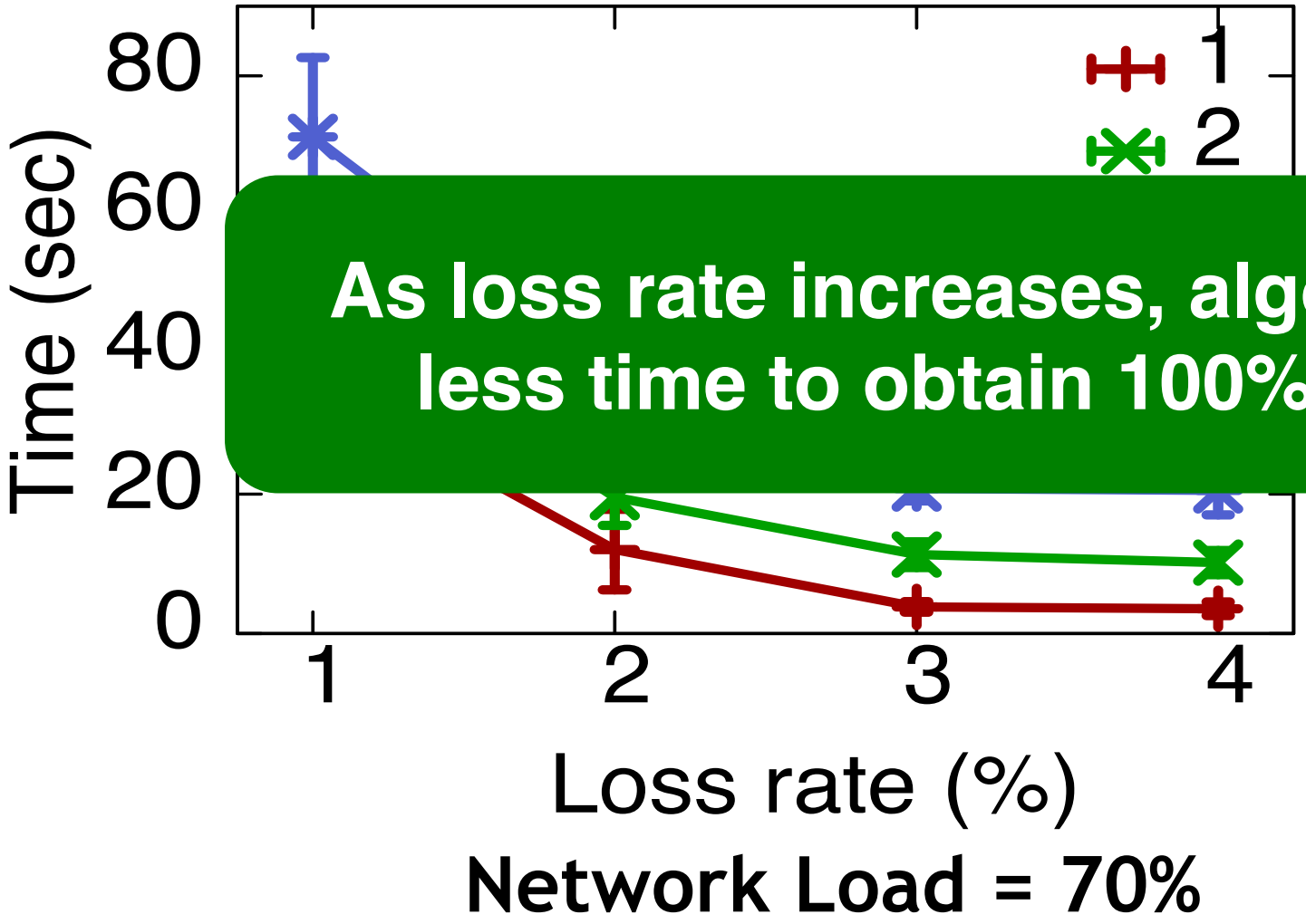
Example 2: Silent random packet drop diagnosis



Lab setup

- 4-ary fat-tree topology
- Web-traffic model

Example 2: Silent random packet drop diagnosis



Network Load = 70%

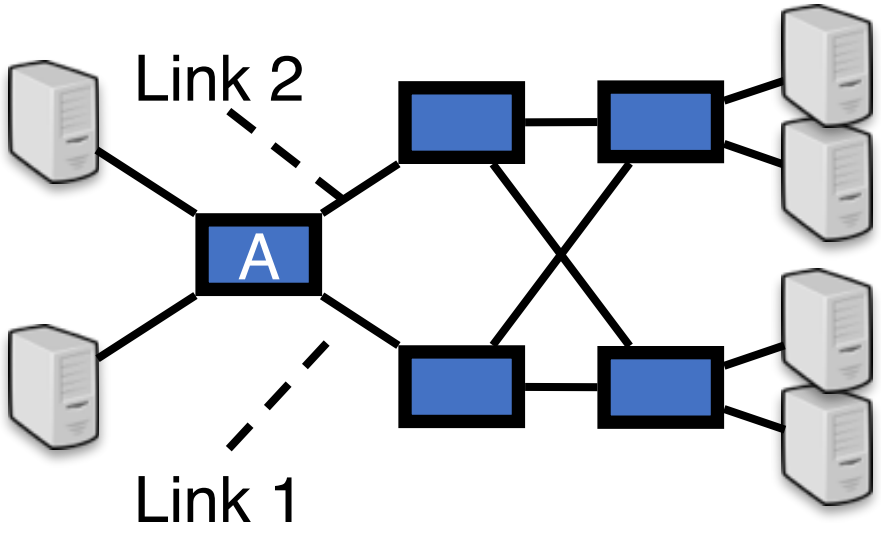
Lab setup

- 4-ary fat-tree topology
- del

As loss rate increases, algorithm takes less time to obtain 100% accuracy

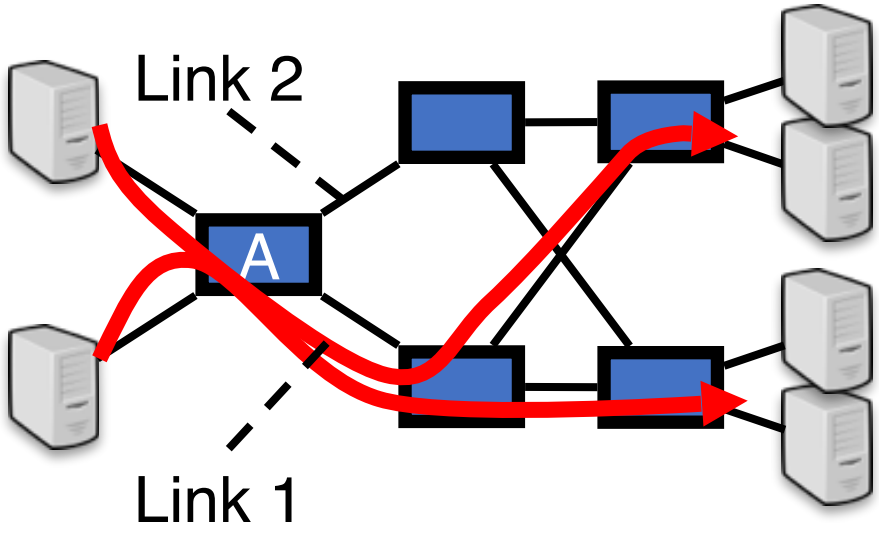
Example 3: Load imbalance diagnosis

Aggregator



Example 3: Load imbalance diagnosis

Aggregator

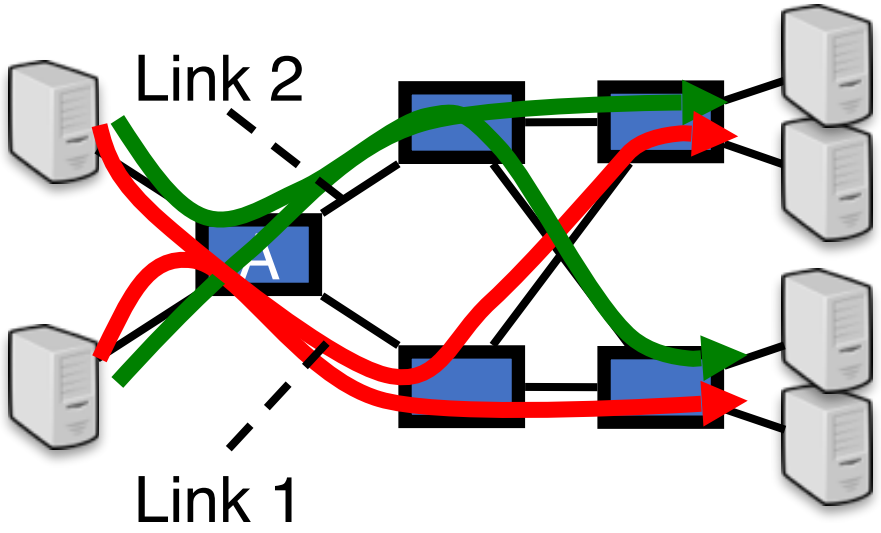


If flow >= 1 MB ➔ Link 1

Example 3: Load imbalance diagnosis

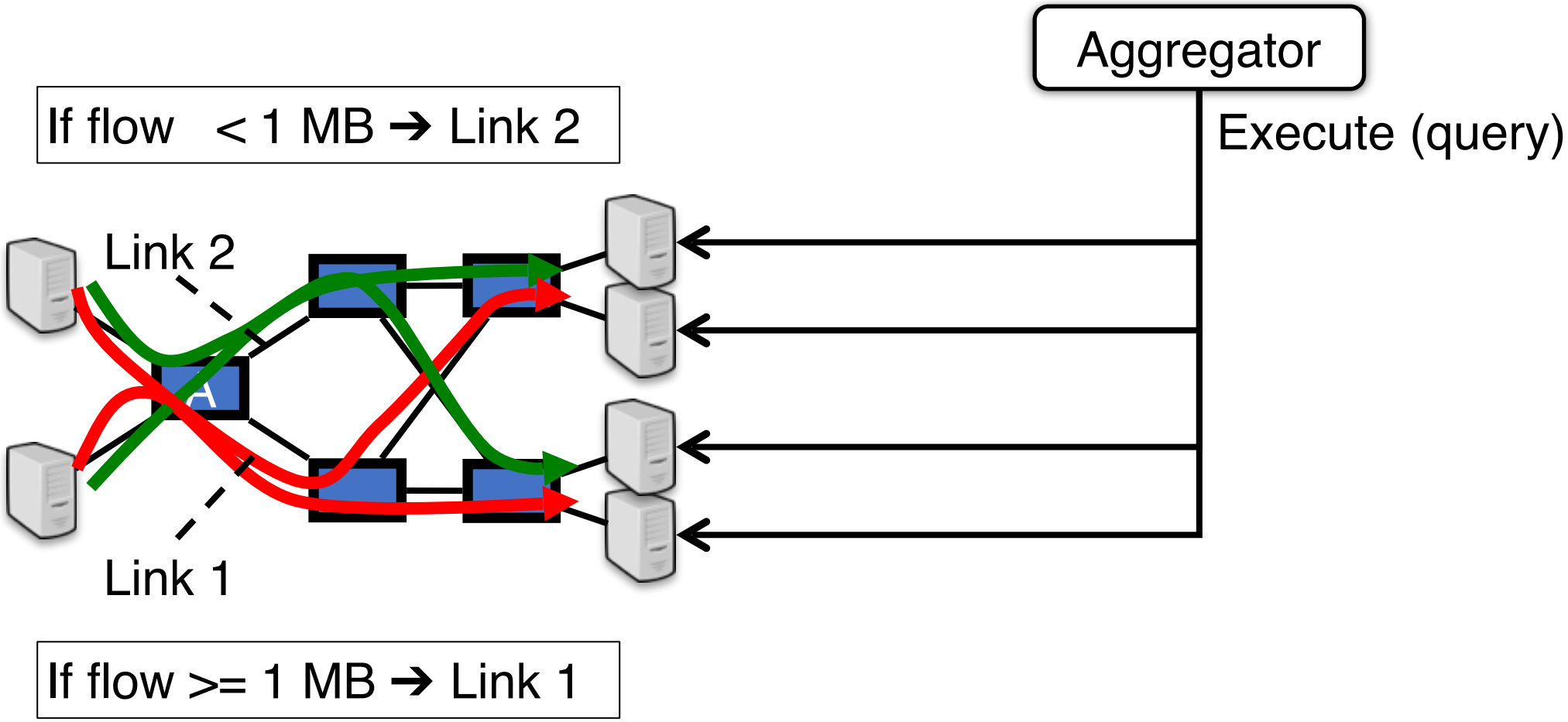
Aggregator

If flow < 1 MB ➔ Link 2



If flow >= 1 MB ➔ Link 1

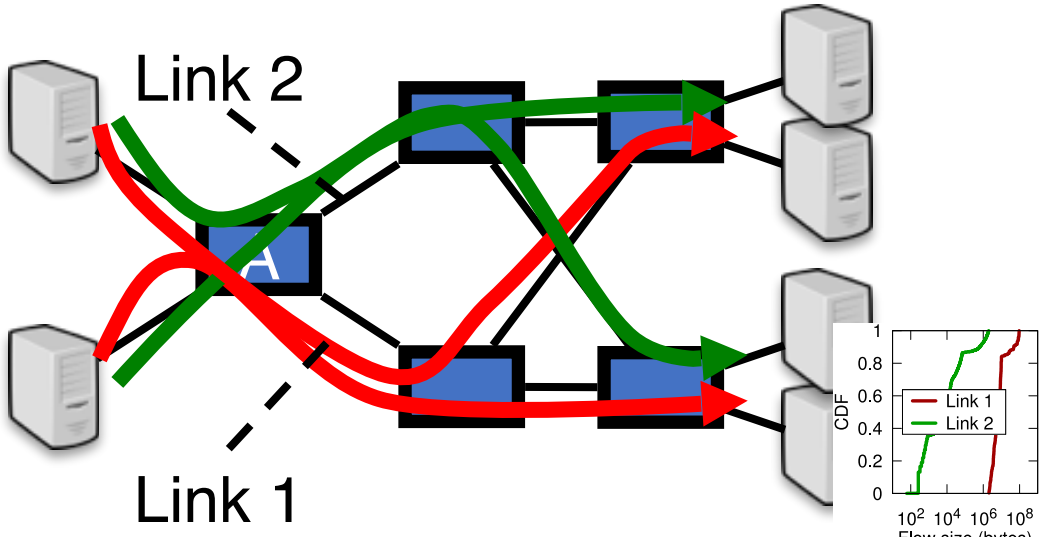
Example 3: Load imbalance diagnosis



Example 3: Load imbalance diagnosis

Aggregator

If flow < 1 MB ➔ Link 2

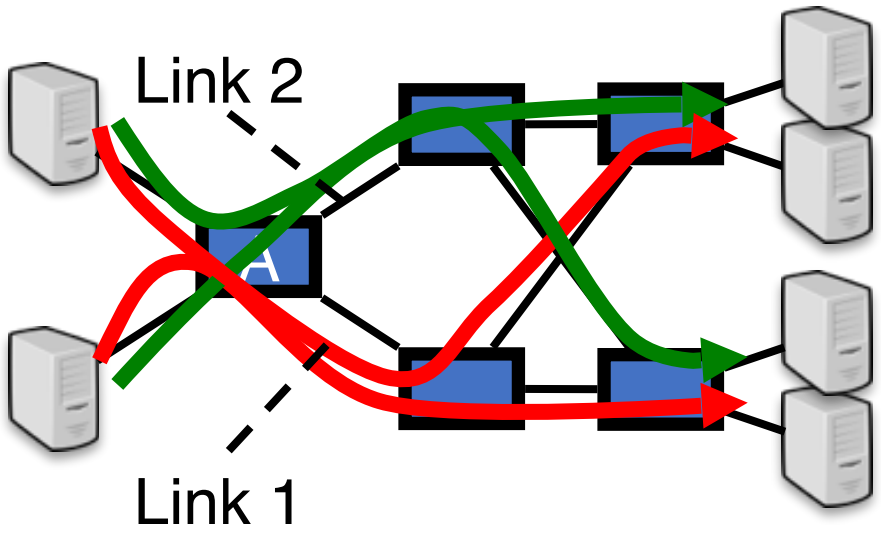


If flow >= 1 MB ➔ Link 1

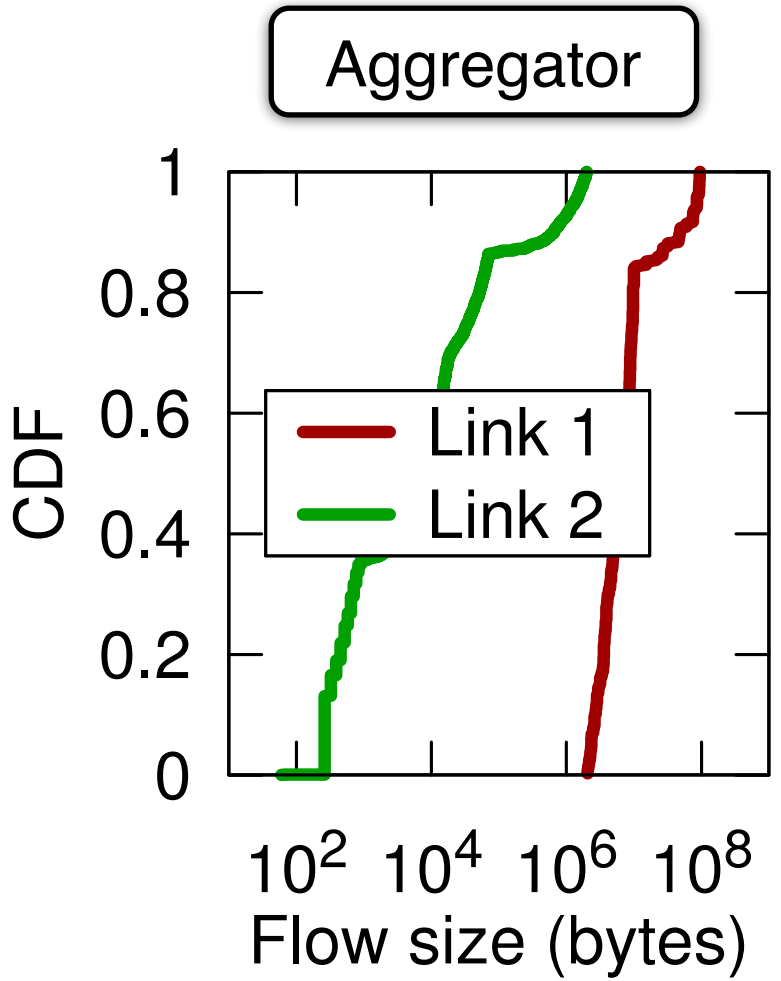
Local flow size distribution

Example 3: Load imbalance diagnosis

If flow < 1 MB ➔ Link 2

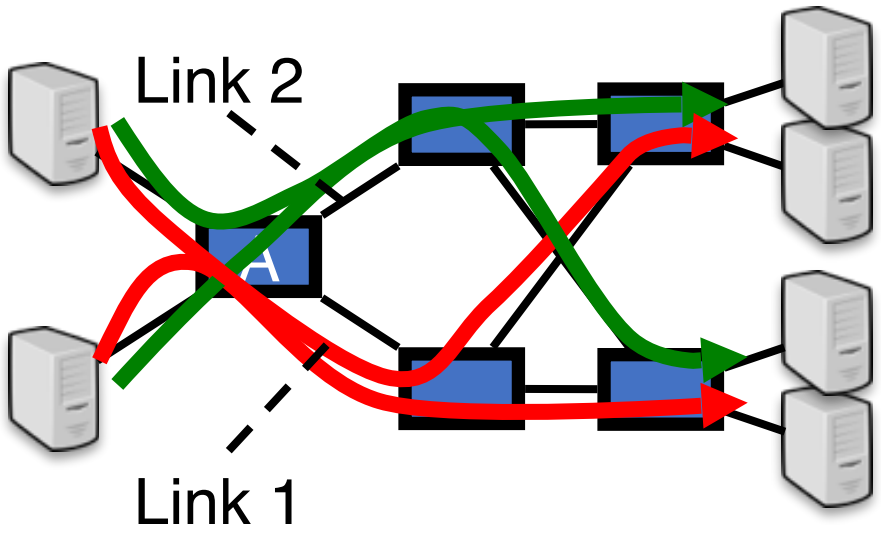


If flow >= 1 MB ➔ Link 1

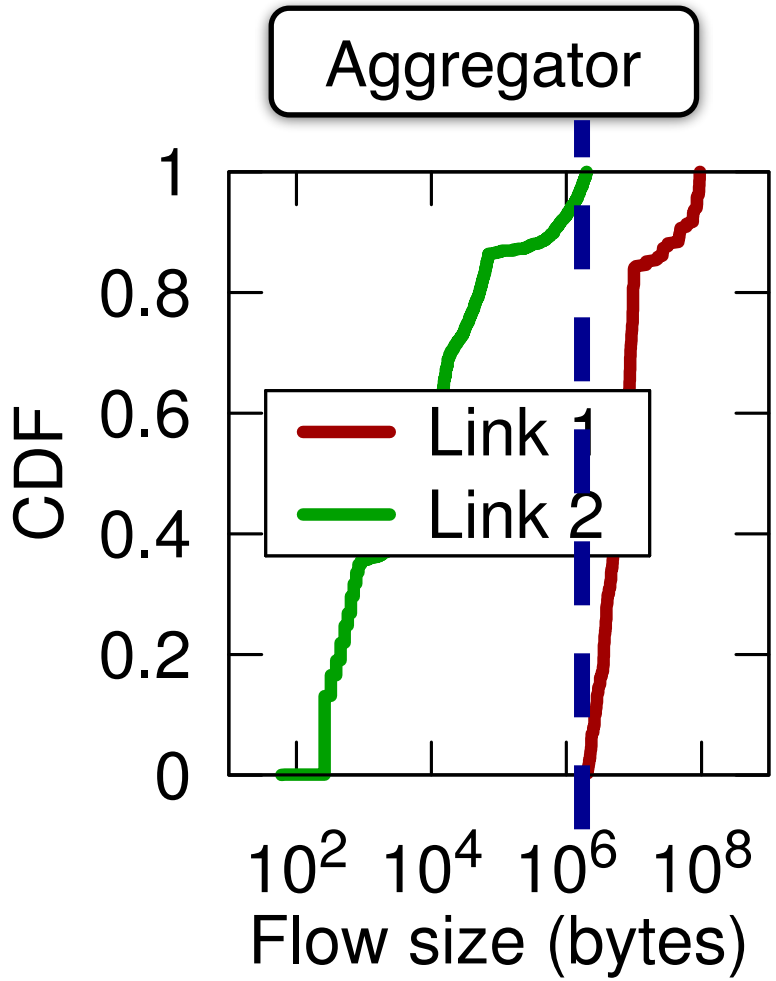


Example 3: Load imbalance diagnosis

If flow < 1 MB ➔ Link 2



If flow >= 1 MB ➔ Link 1

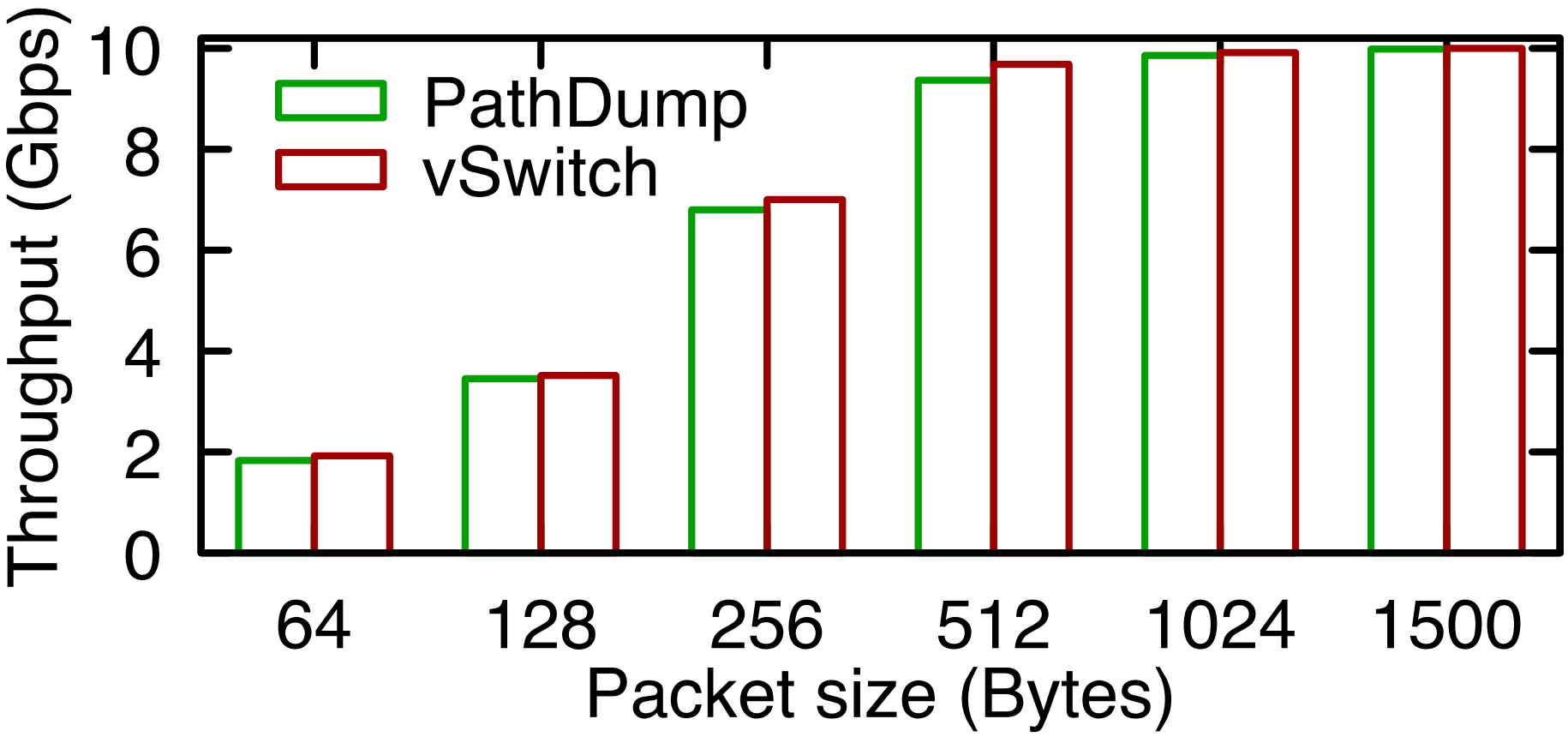


Other debugging applications

- Real-time routing loop detection
- Blackhole diagnosis
- TCP performance anomaly diagnosis
  - TCP incast and outcast
- Traffic measurement
  - Traffic matrix, heavy-hitter detection, etc.

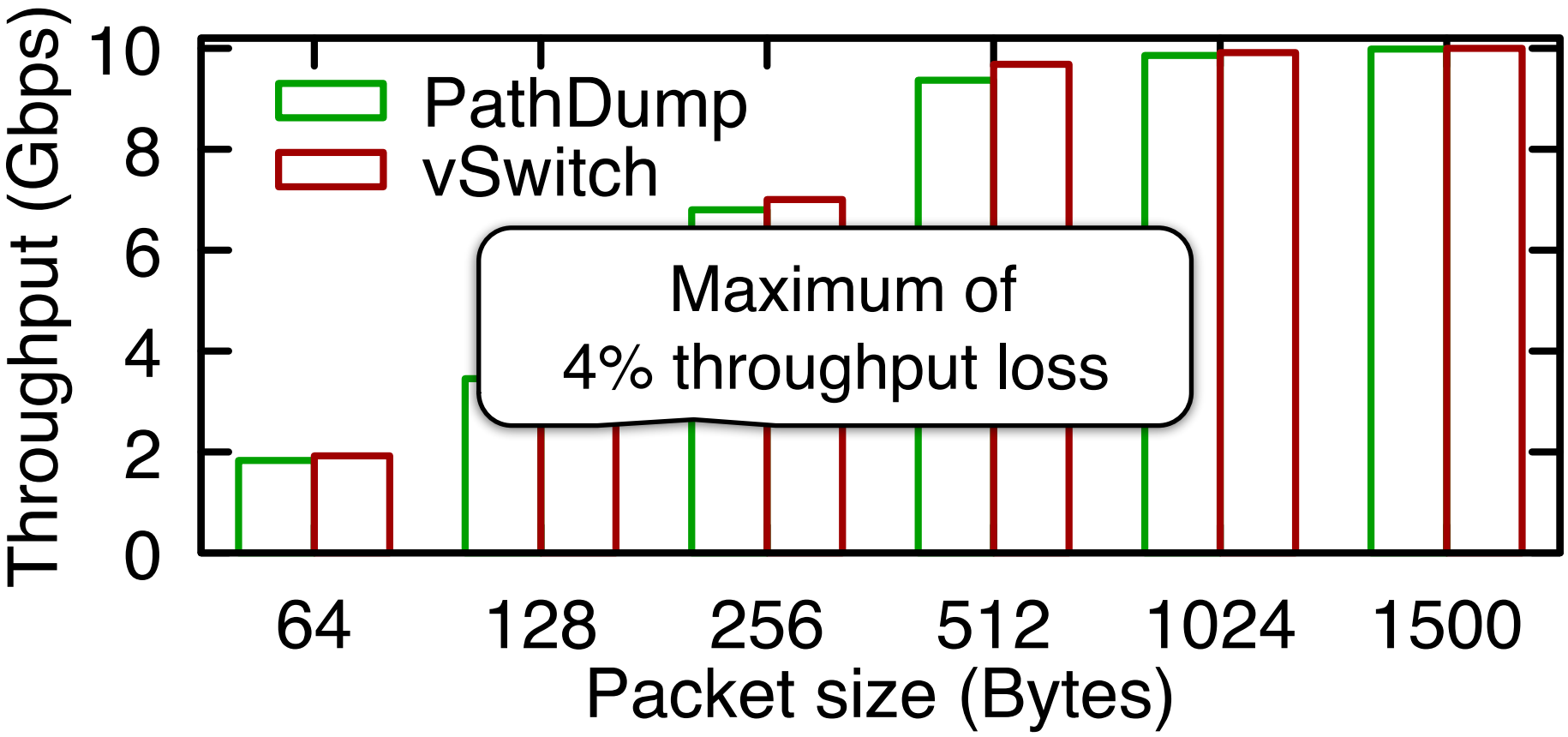
More details in our paper

Packet processing overhead at end-host



Packet processing overhead at end-host

Minimal packet processing overhead atop Open vSwitch



More details (In the paper)

- Distributed query mechanism
- Supported network debugging problems
- Implementation details
- Evaluation over real testbed(s)

Conclusion

- DCNs are complex; and their debuggers are even more complex
- Design and implement PathDump, a simple debugger
- Keeps network switches simple
  - No complex operations in network switches
- Executes debugging queries in a distributed manner
- Consumes small amount of data plane and end-host resources
- Debugs a large class of network problems

https://github.com/PathDump

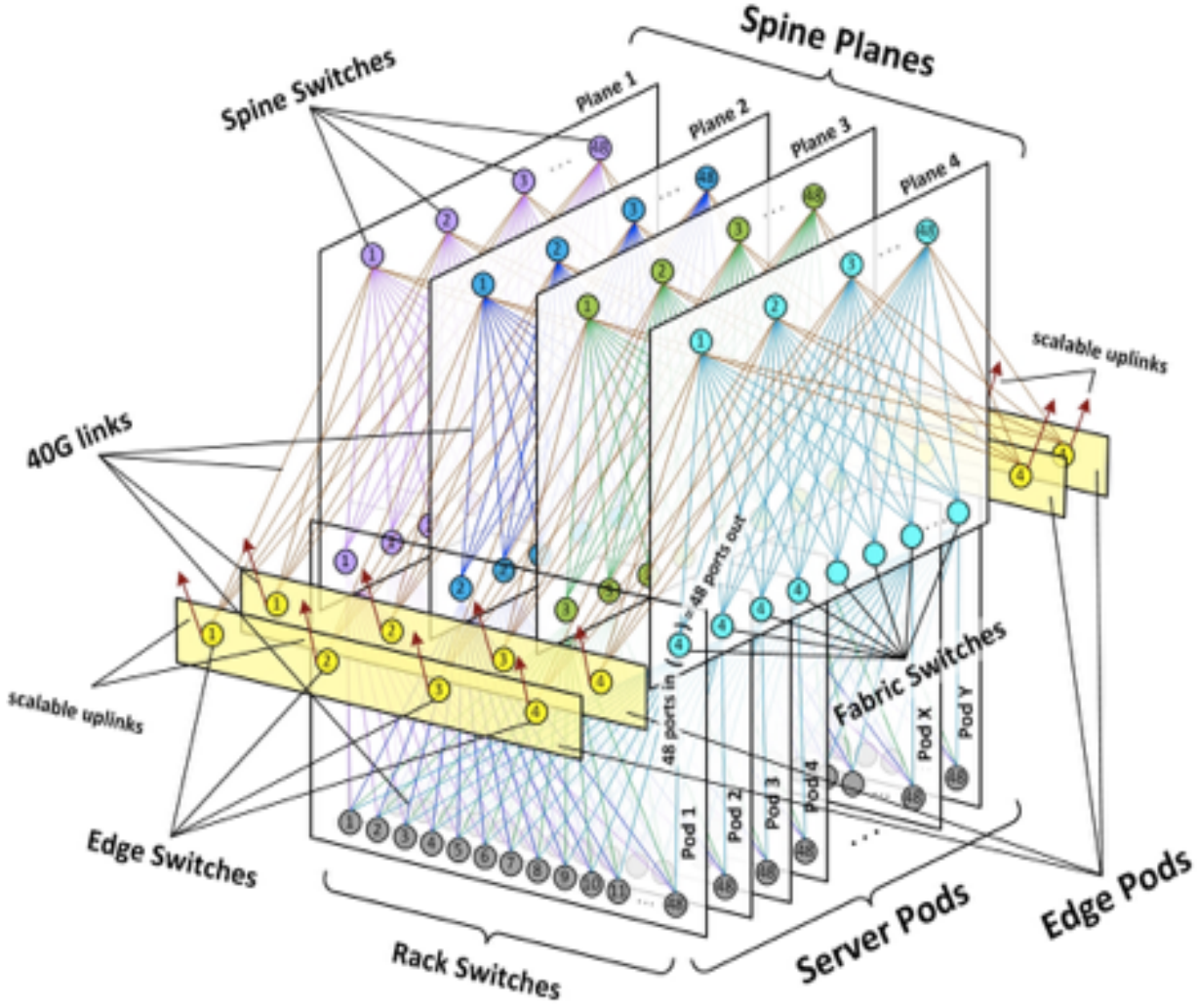
Datacenter networks are complex(*remove slide)

- Complexity due to need for
  - High availability
  - High performance

Latency matters. Amazon found every 100ms of latency cost them 1% in sales.

Google found an extra .5 seconds in search page generation time dropped traffic by 20%.

--source: The Gigaspaces blog



--source: TechRepublic.com

PathDump interface

Simple 9 APIs enables a variety of debugging applications

Host API

```
getFlows(linkID, timeRange)
getPaths(flowID, linkID, timeRange)
getCount(flow, timeRange)
getDuration(flow, timeRange)
getPoorTCPFlows(threshold)
Alarm(flowID, reason, paths)
```

Aggregator API

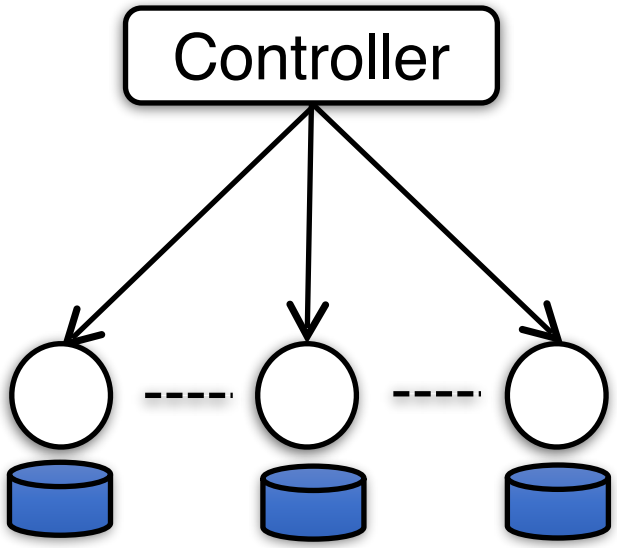
```
execute(list<hostID>, query)
install(list<hostID>, query, Period)
uninstall(list<hostID>, query)
```

✓ Write a query using host API

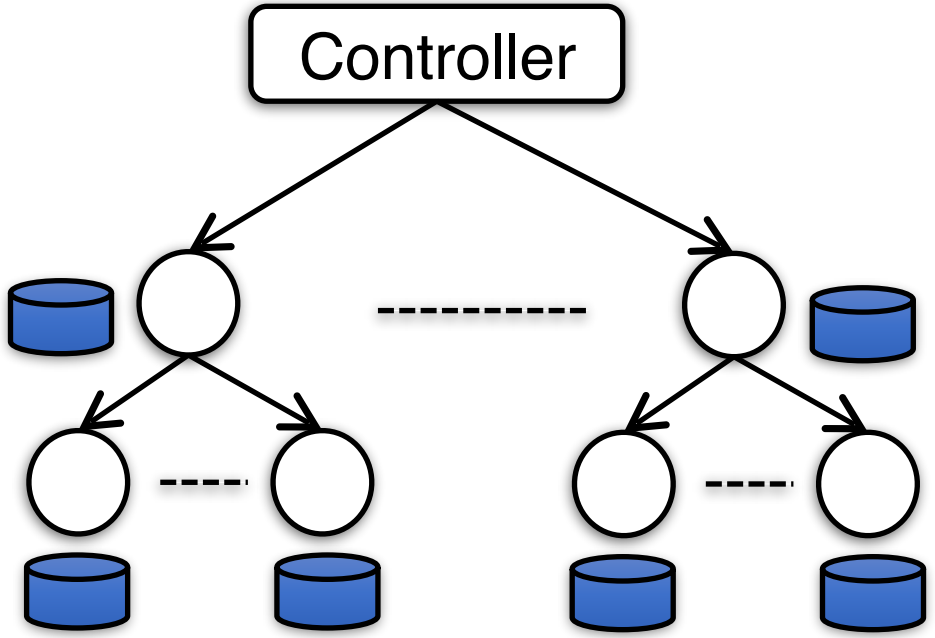
✓ Install(), execute() or uninstall() with Aggregator API

Query processing

Direct query

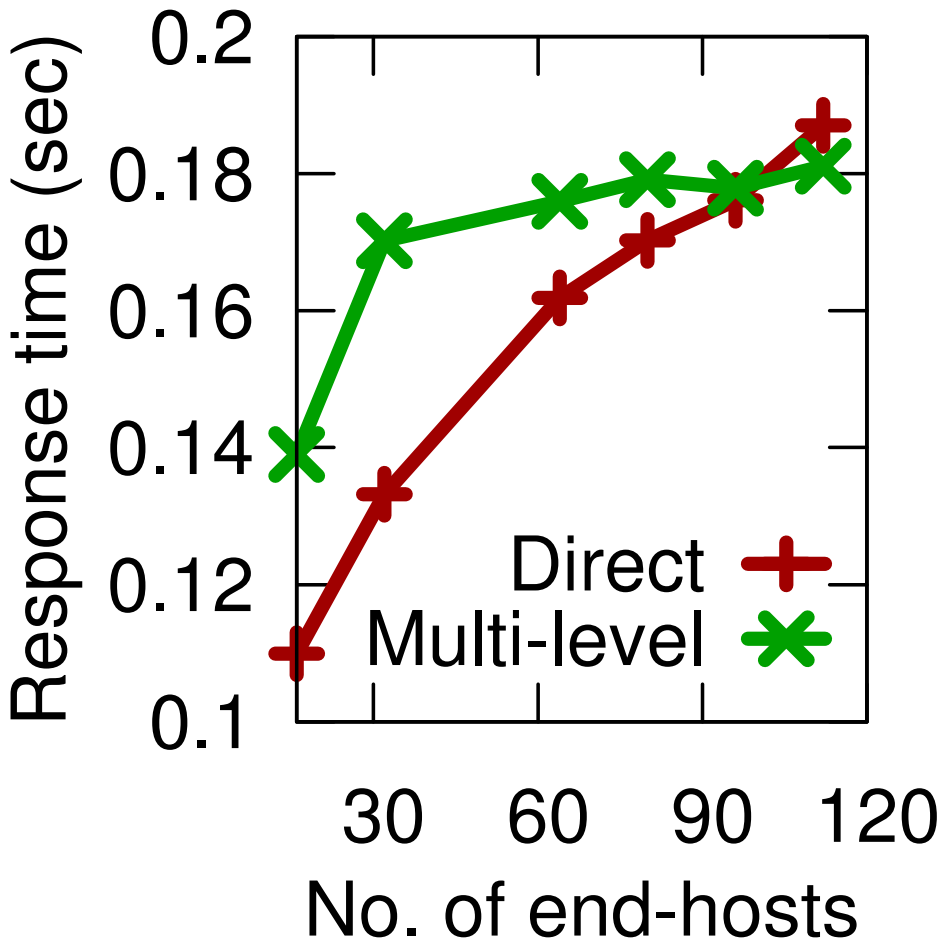


Multi level query

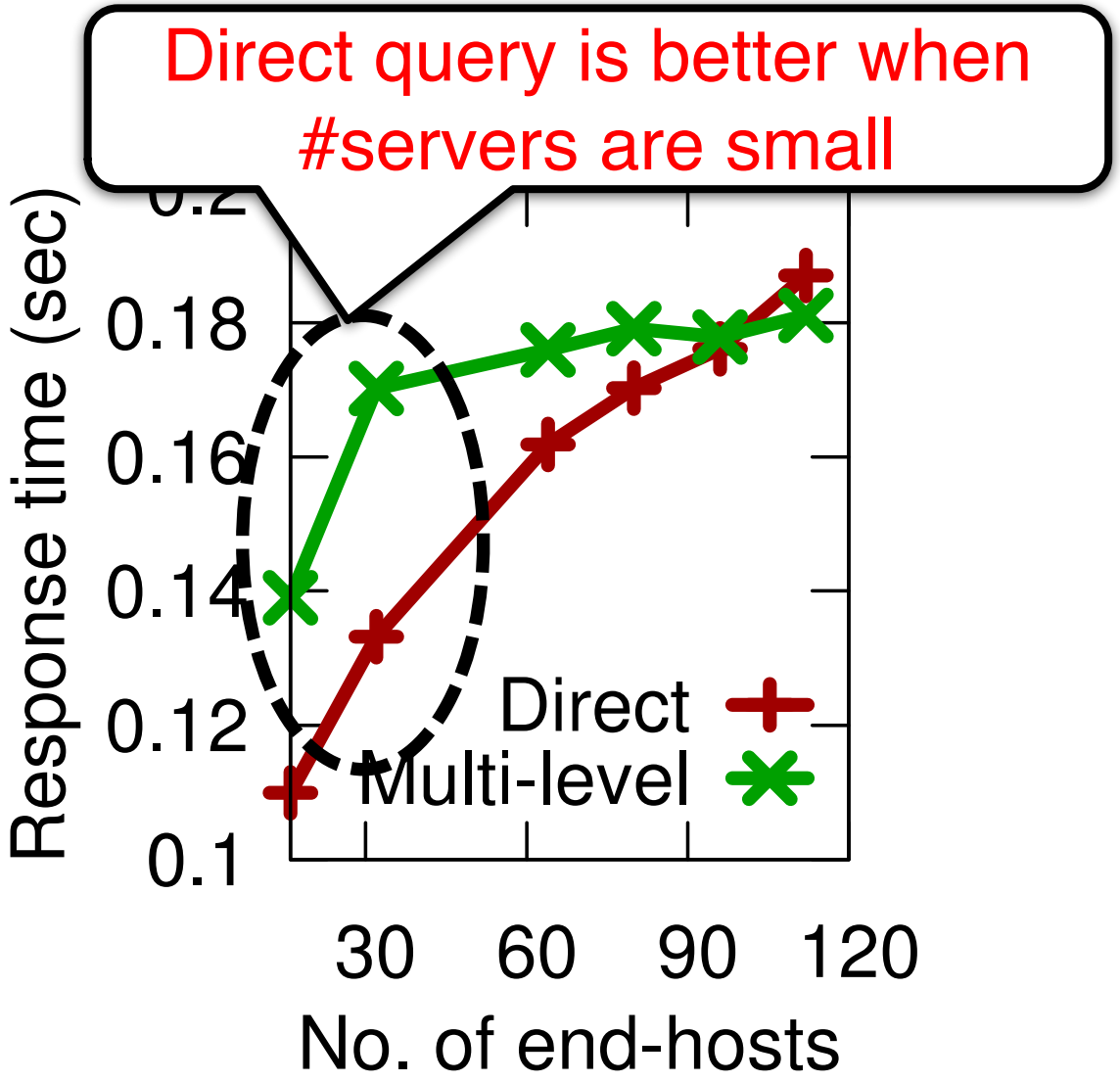


Query processing

Flow size distribution query

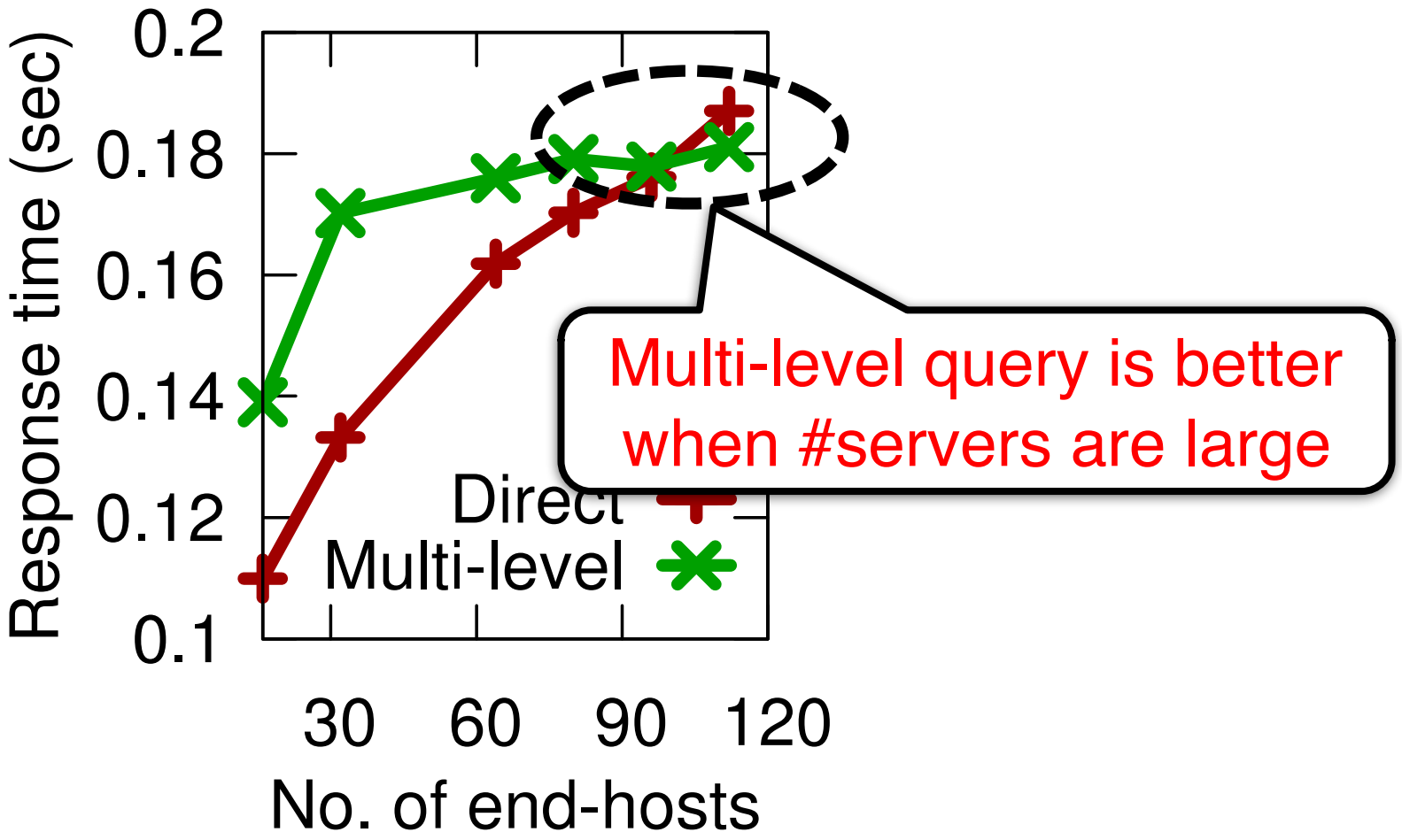


Query processing



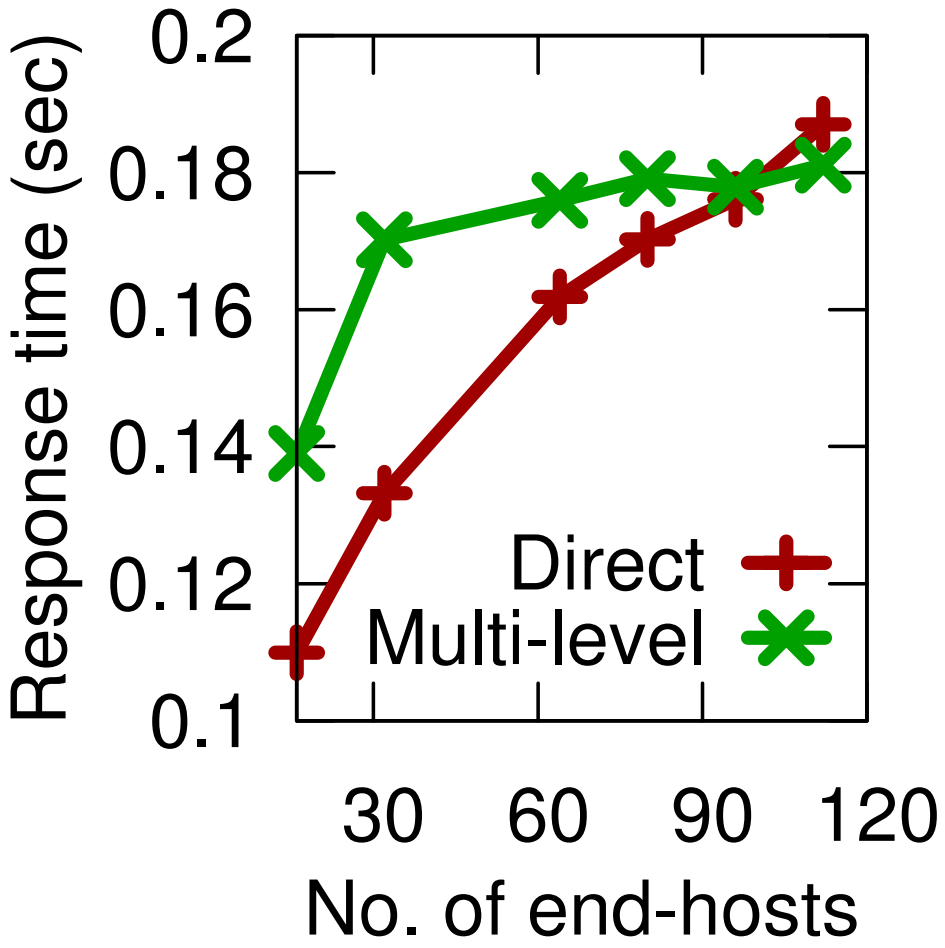
Query processing

Flow size distribution query

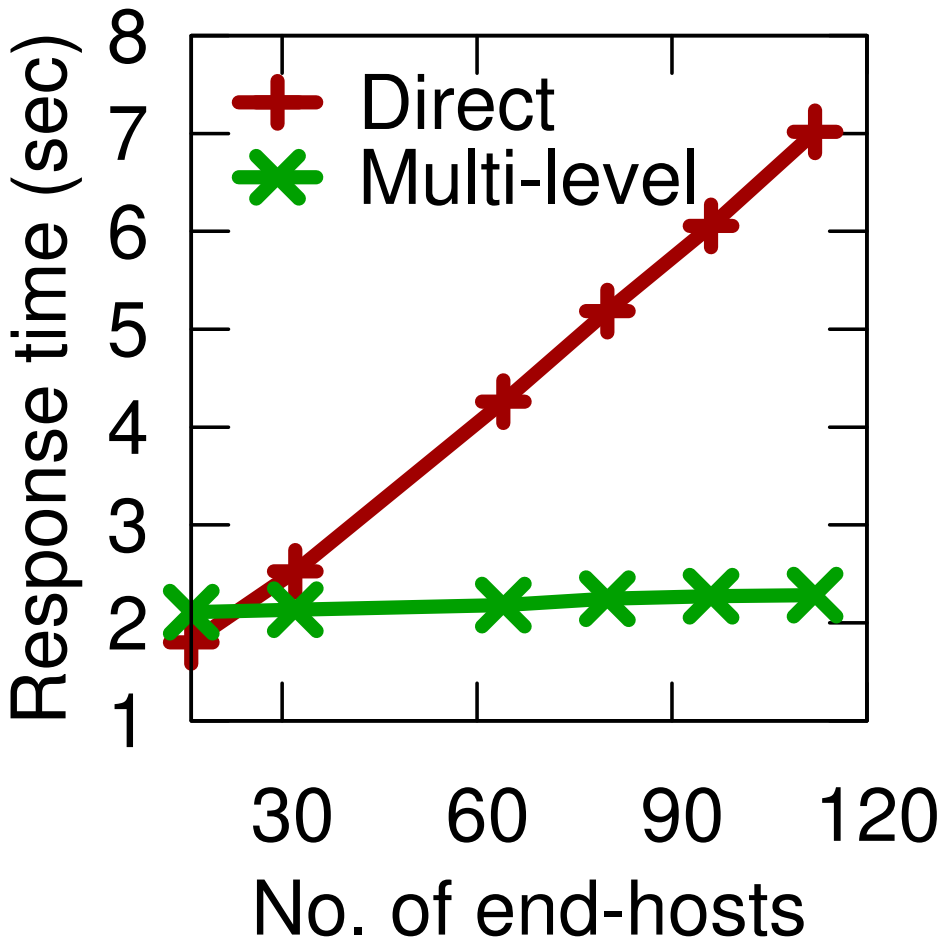


Query processing

Flow size distribution query



Top-k flows query



Query processing

Flow size distribution query

Top-k flows query