Introduction to Software-Defined Networking

UG3 Computer Communications & Networks (COMN)

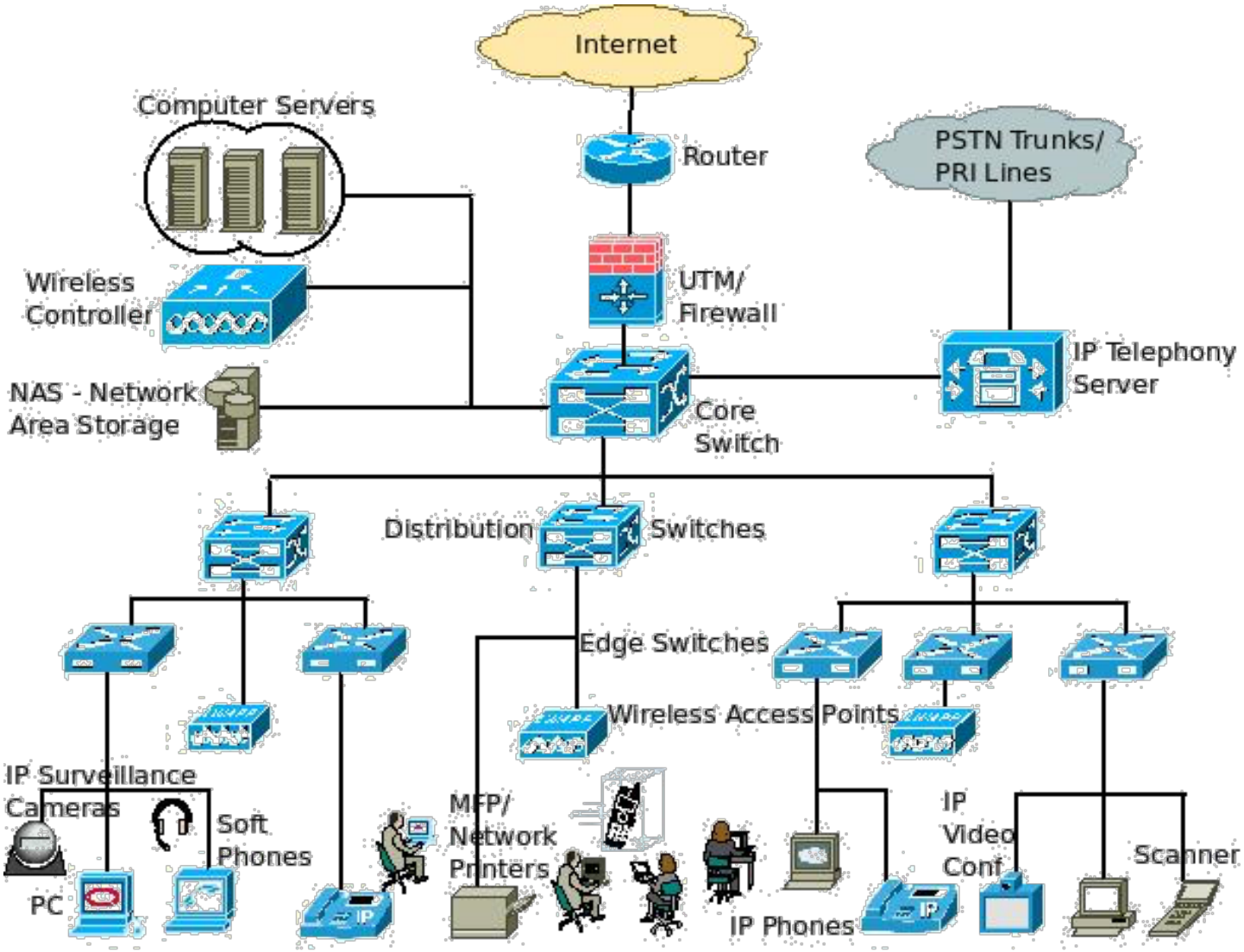
Myungjin Lee
myungjin.lee@ed.ac.uk

Courtesy note: Slides from **course CPS514** Spring 2013 at Duke University and Hot Interconnects Keynote by **Nick McKeown**, 2012

Outline

- **What is SDN?**
  - Limitations of current networks
  - The idea of Network OS
- **What is OpenFlow?**
  - How it helps SDN
- **Application: Network debugging with SDN**

Limitations of Current Networks



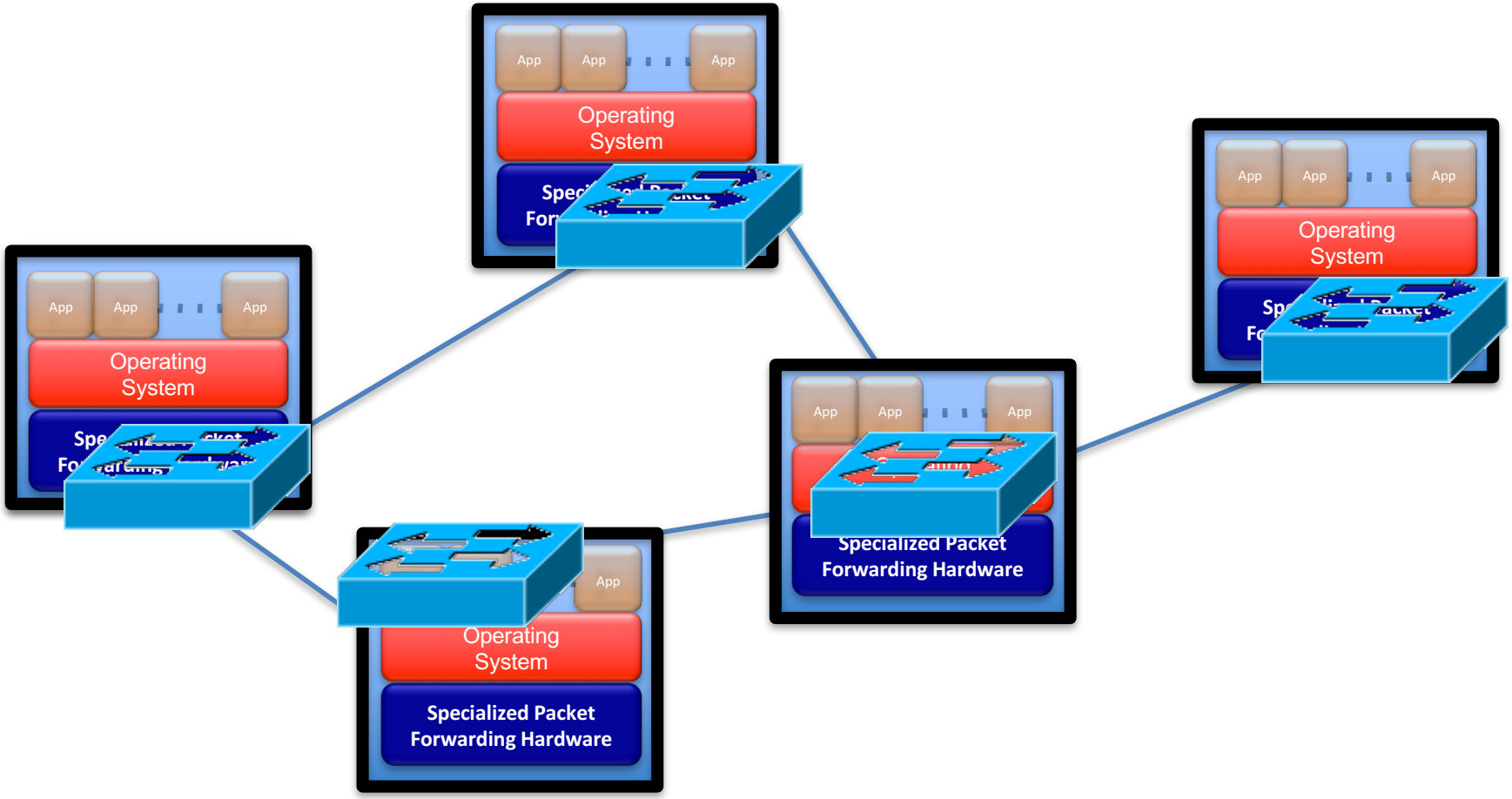
http://www.excitingip.net/27/a-basic-enterprise-lan-network-architecture-block-diagram-and-components/

Limitations of Current Networks

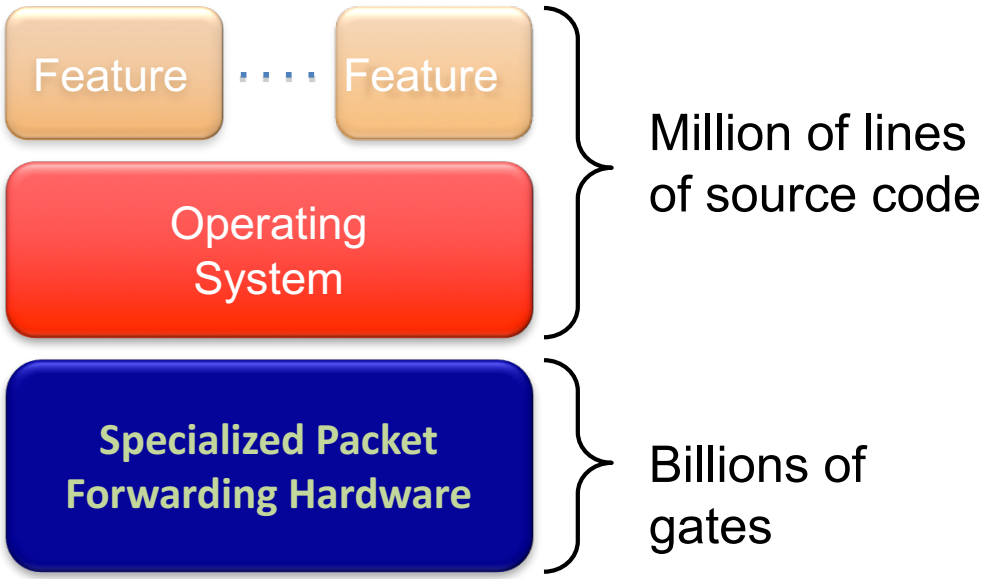
- **Enterprise networks are difficult to manage**
- **"New control requirements have arisen":**
  - Greater scale
  - Migration of VMs
- **How to easily configure huge networks?**

Limitations of Current Networks

- **Old ways to configure a network**



Limitations of Current Networks



Many complex functions baked into infrastructure

OSPF, BGP, multicast, differentiated services, Traffic Engineering, NAT, firewalls, …

Cannot dynamically change according to network conditions

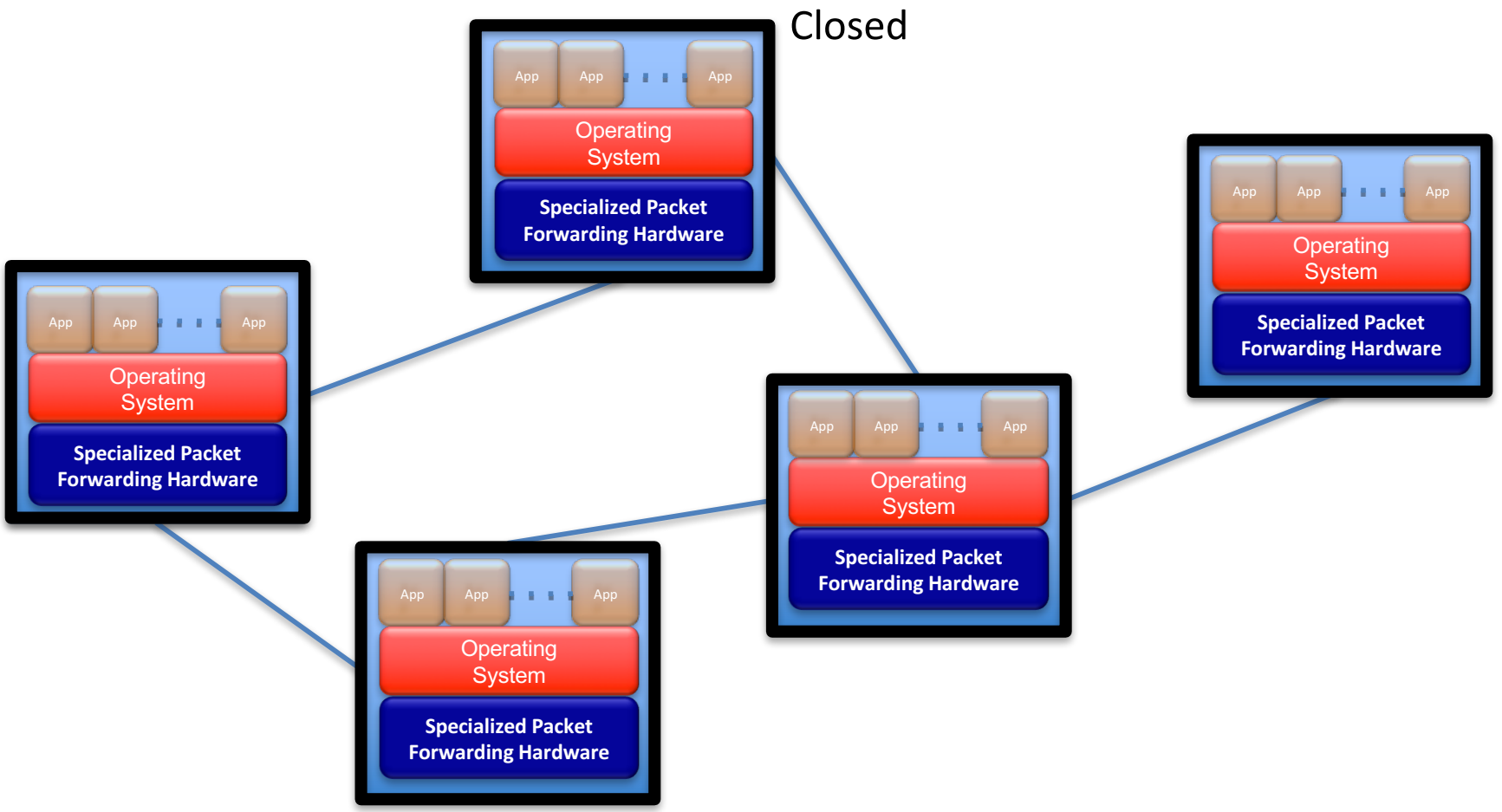
Limitations of Current Networks

- **No control plane abstraction for the whole network!**

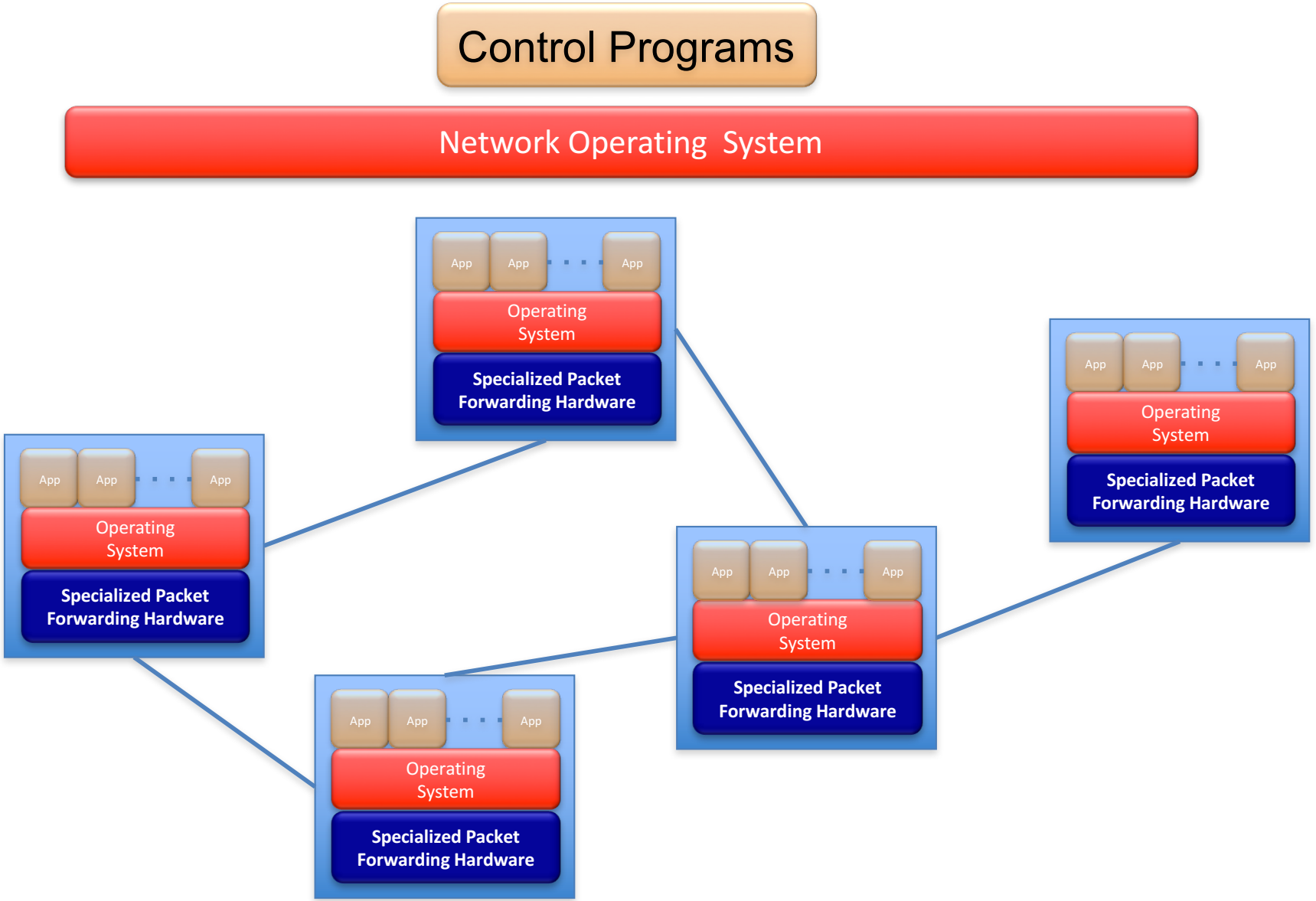
- **It's like old times – when there was no OS…**

Wilkes with the EDSAC, 1949

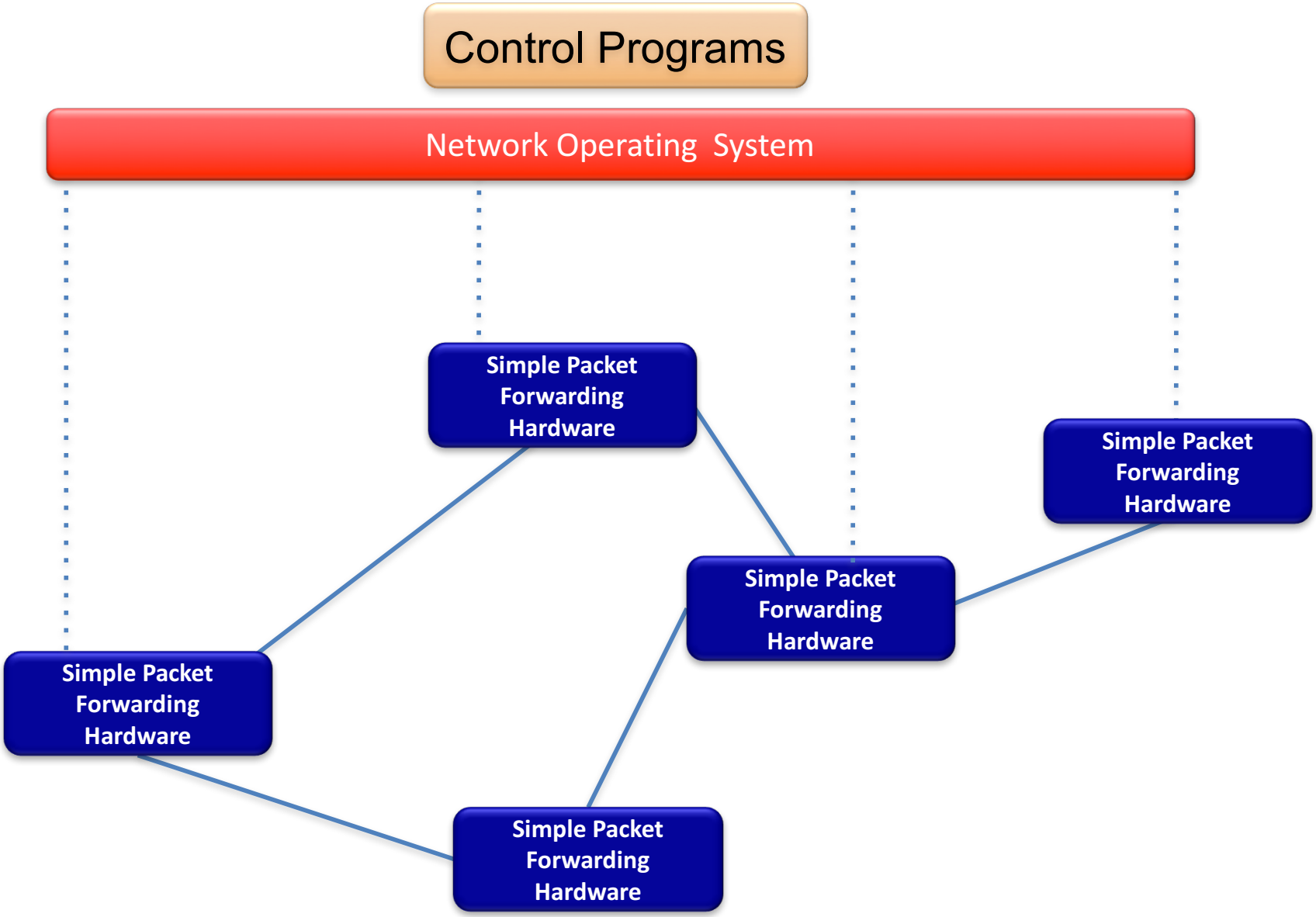
Idea: An OS for Networks



Idea: An OS for Networks



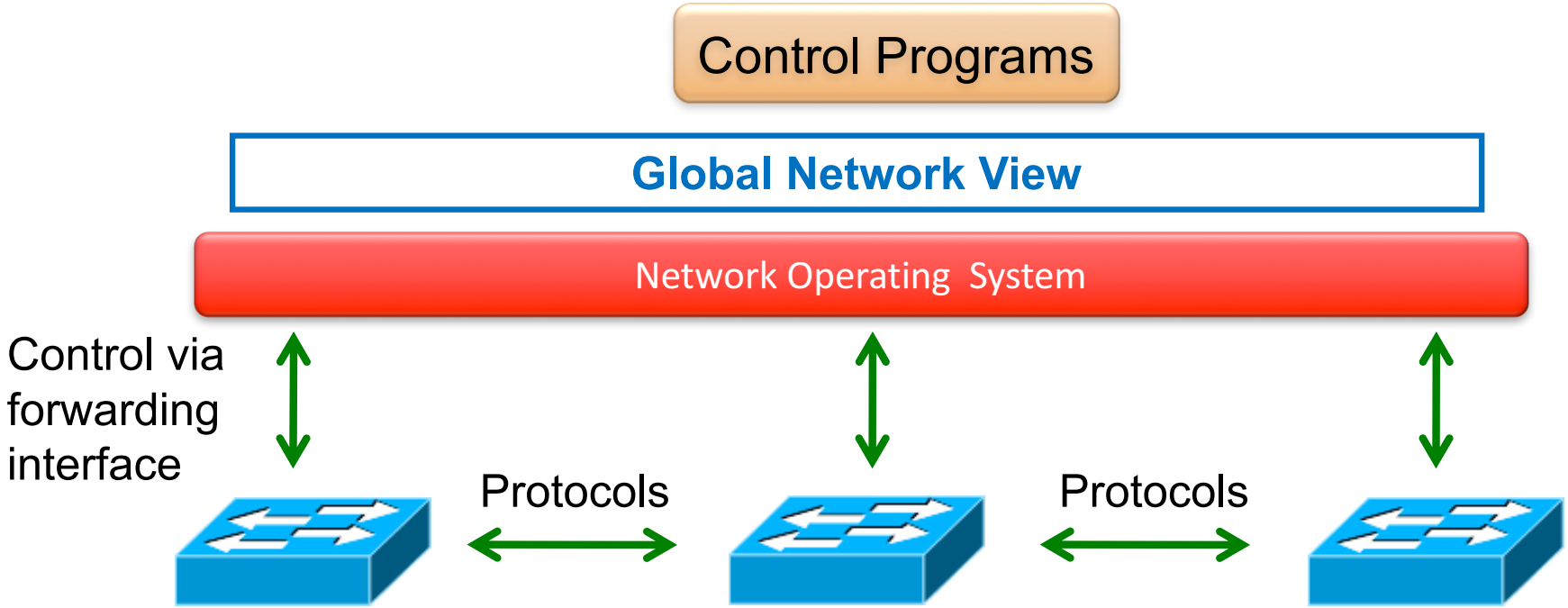
Idea: An OS for Networks



Idea: An OS for Networks

- **"NOX: Towards an Operating System for Networks"**

  Software-Defined Networking (SDN)



The Future of Networking, and the Past of Protocols, Scott Shenker, *with Martin Casado, Teemu Koponen, Nick McKeown*

Software Defined Networking

- **No longer designing distributed control protocols**
- **Much easier to write, verify, maintain, …**
  - An interface for programming
- **NOS serves as fundamental control block**
  - With a global view of network

Software Defined Networking

- **Questions:**
  - How to obtain global information?
  - What are the configurations?
  - How to implement?
  - How is the scalability?
  - How does it really work?

Outline

- **What is SDN?**
  - Limitations of current networks
  - The idea of Network OS
- **What is OpenFlow?**
  - How it helps SDN
- **Application: Network debugging with SDN**

OpenFlow

- **“OpenFlow: Enabling Innovation in Campus Networks”**

  http://ccr.sigcomm.org/online/files/p69-v38n2n-mckeown.pdf

- **Like hardware drivers**
  - interface between switches and Network OS

Getting Started

OpenFlow Tutorial

- search: “OpenFlow Tutorial”

Mininet

- Network emulator
- Designed for emulating SDN networks
- Easy to use
- High performance (100 nodes on a laptop)
- search: “Mininet”

OpenFlow Switches?

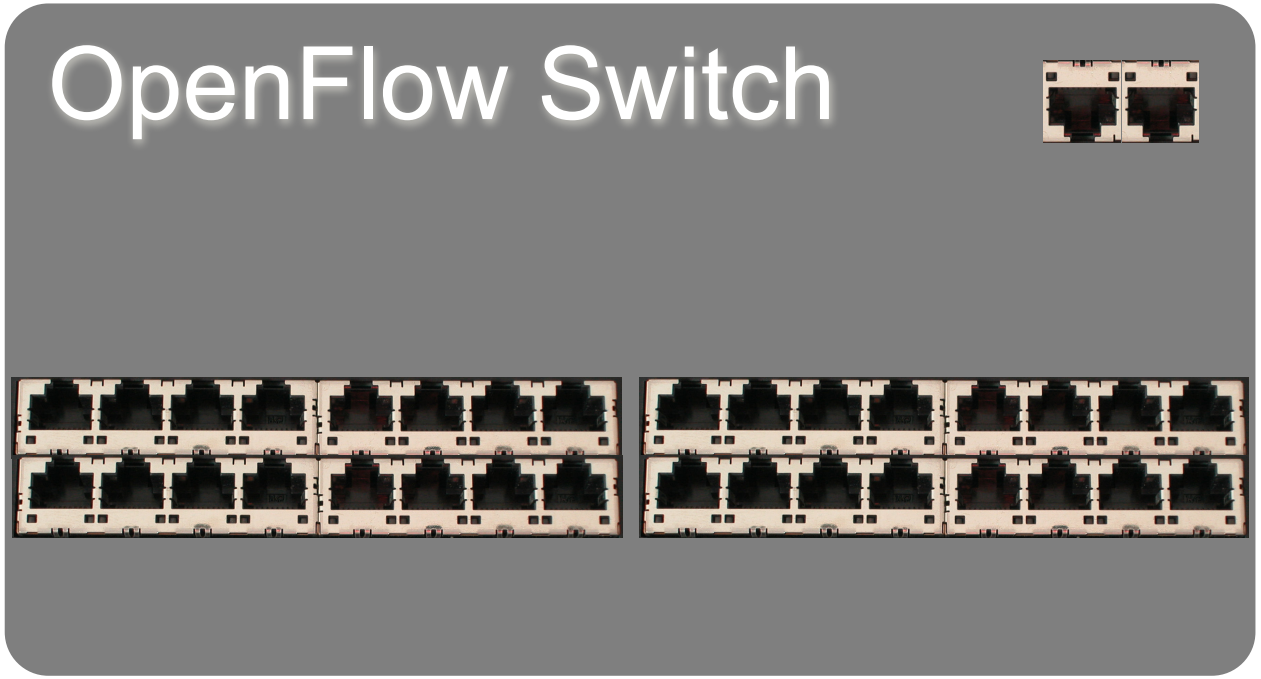
Software switch

- Open vSwitch (openvswitch.org)
- Now part of Linux distribution

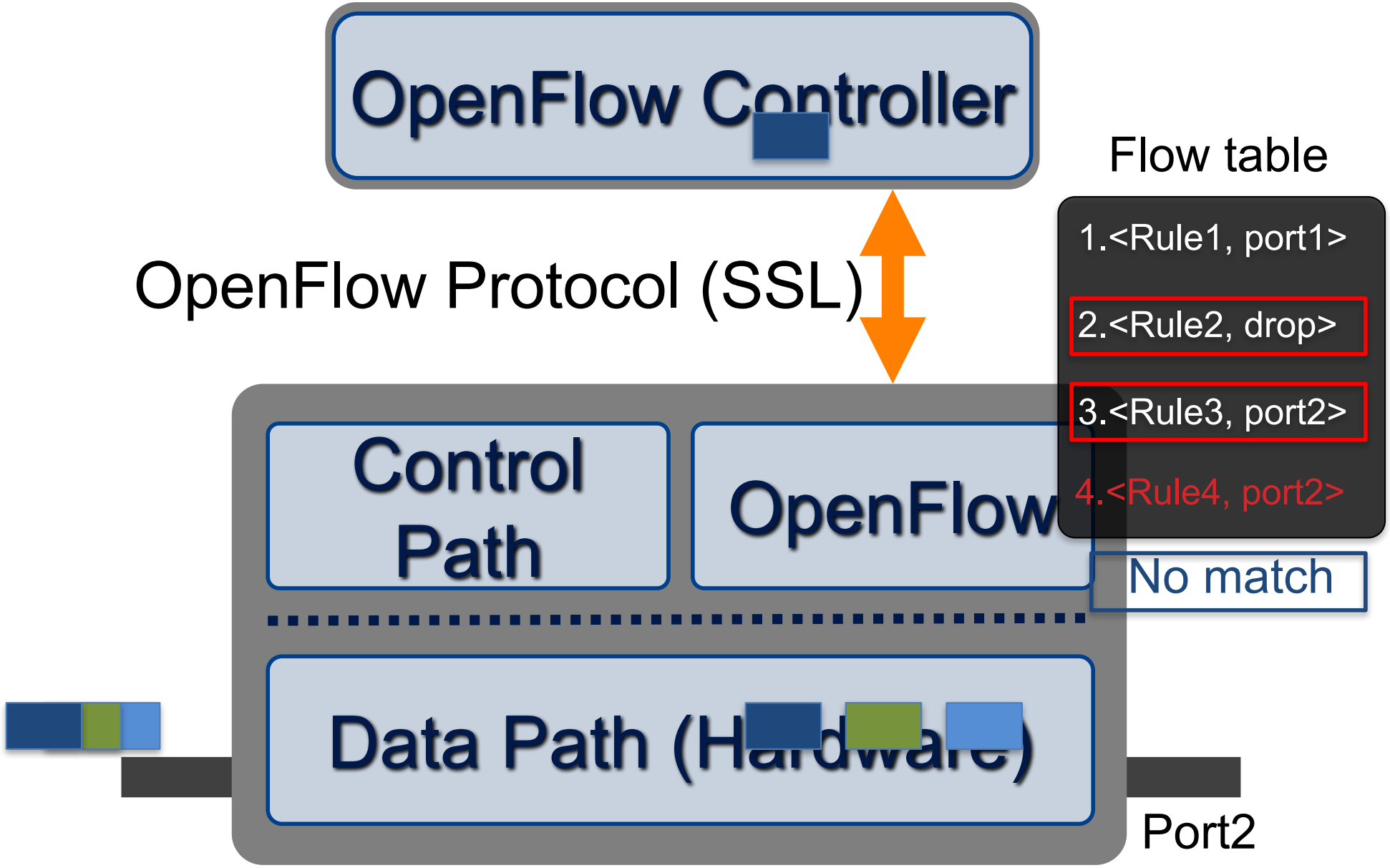
Hardware switches

- Announcements from several vendors
- HP, Brocade, NEC, …
- (You could ask Google for one of theirs ☺)

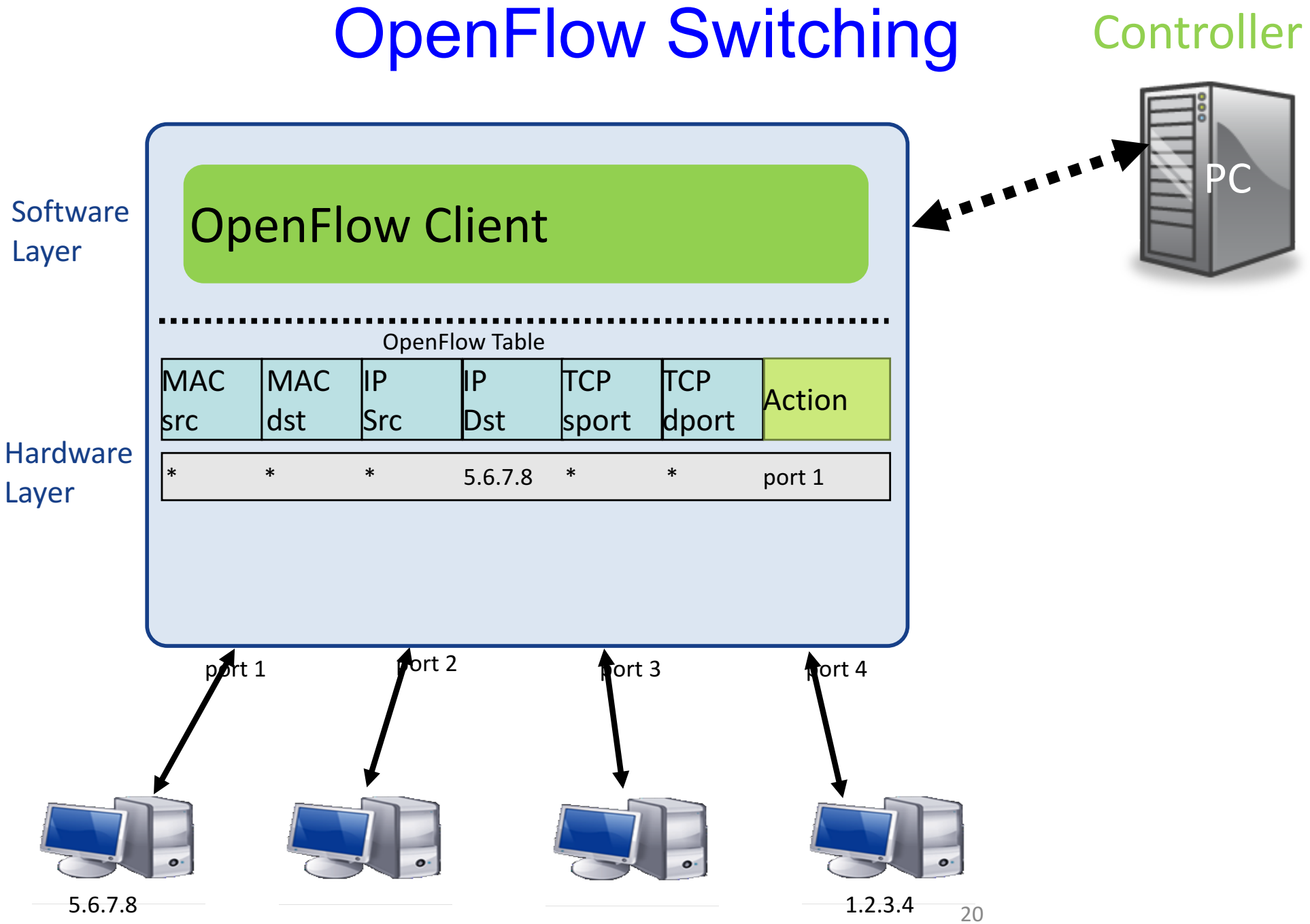
OpenFlow Basics



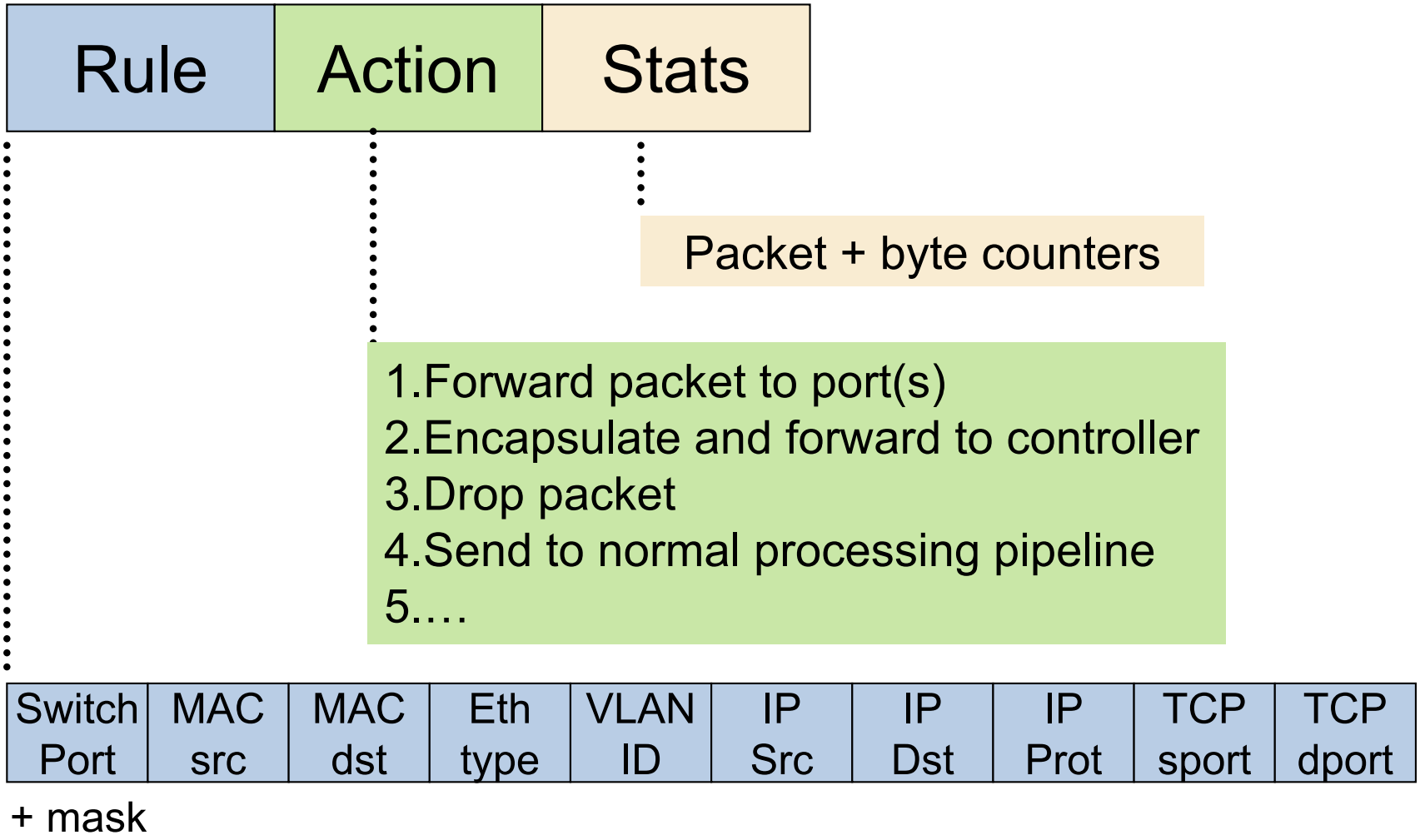
OpenFlow Basics



OpenFlow Switching



OpenFlow Table Entry



OpenFlow Examples

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

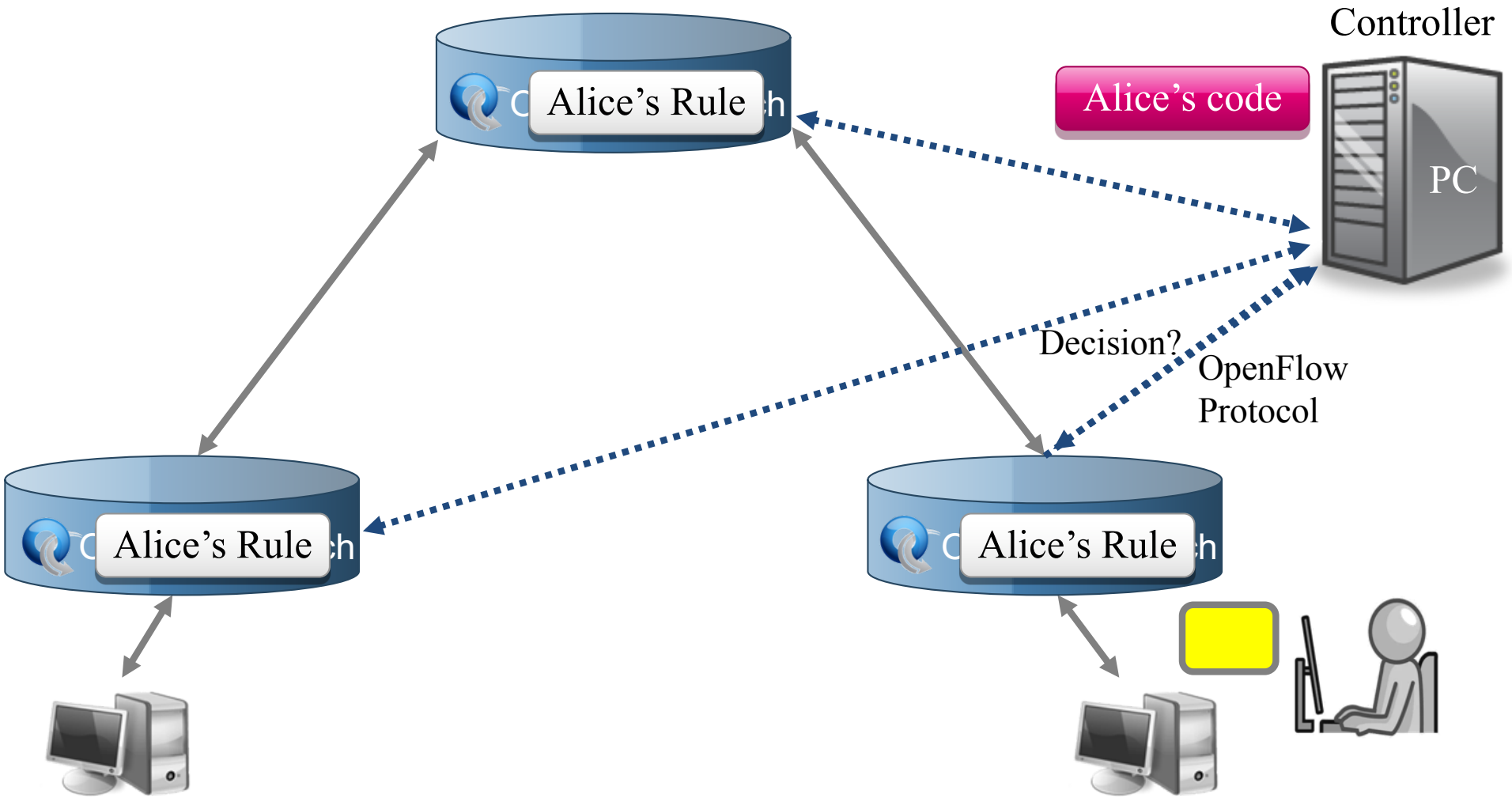
Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

Firewall

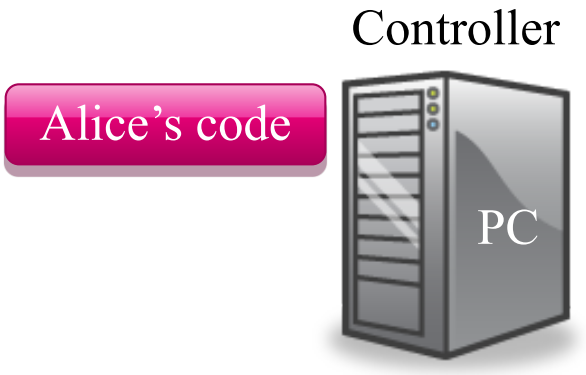
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

OpenFlow Usage



OpenFlow Usage

- Alice's code:
  - Simple learning switch
  - Per Flow switching
  - Network access control/firewall
  - Static "VLANs"
  - Her own new routing protocol: unicast, multicast, multipath
  - Home network manager
  - Packet processor (in controller)
  - IPvAlice

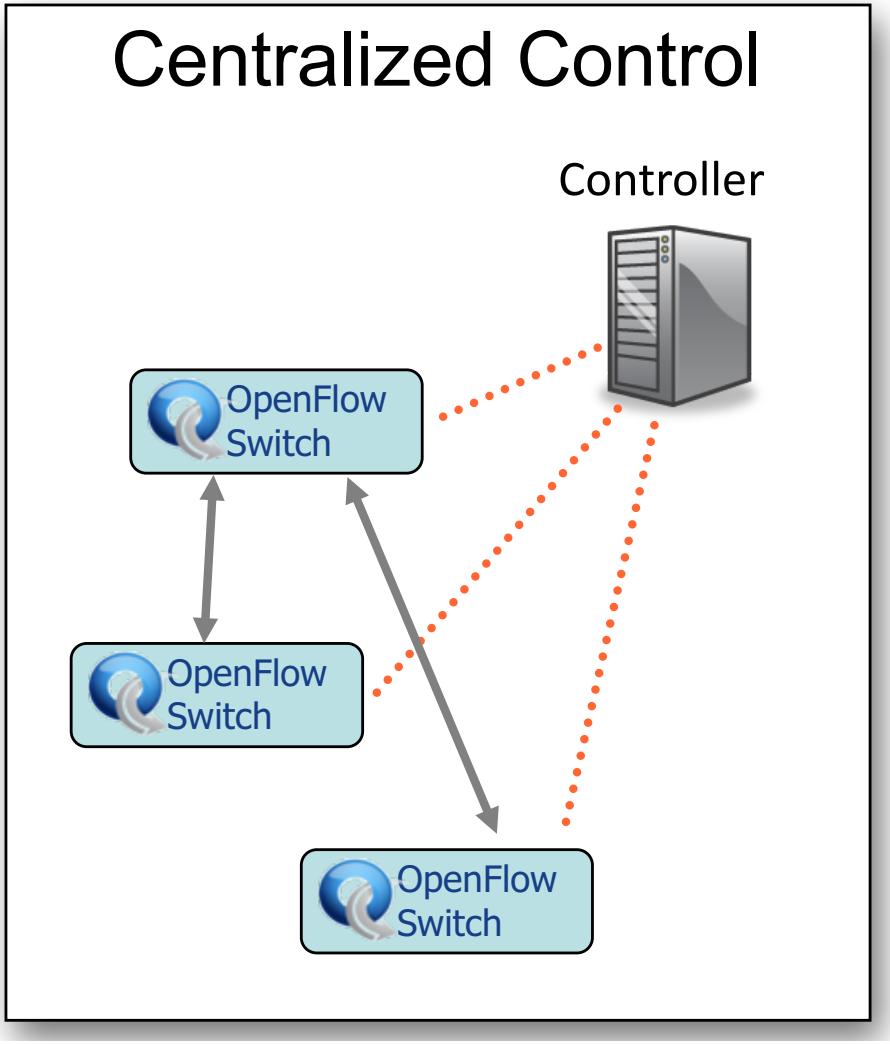


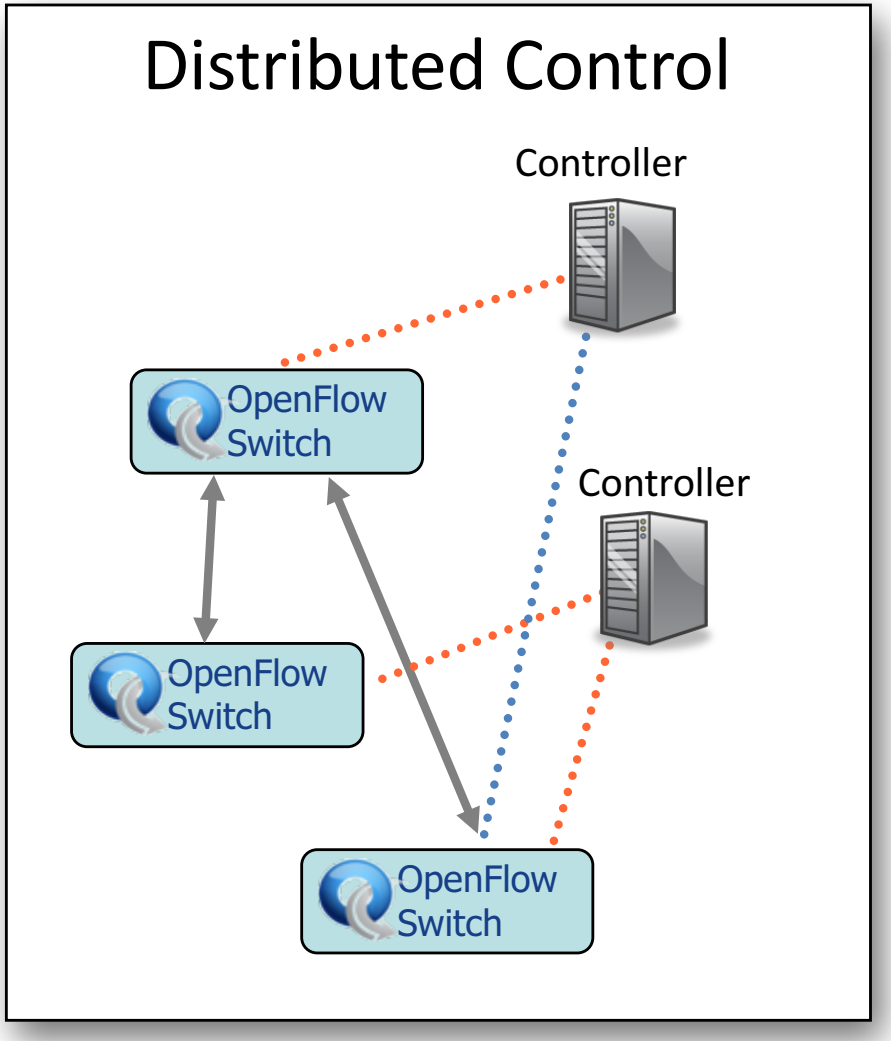
OpenFlow 

- Standard way to control flow-tables in commercial switches and routers
- Just need to update firmware
- Essential to the implementation of SDN

Centralized/Distributed Control

- **“Onix: A Distributed Control Platform for Large-scale Production Networks”**





Ongoing focuses of SDN

- New policies for security
- Programmable WLANs
- The placement of controllers (amount; location; centralized/distributed)
- Debugger for SDN

Outline

- **What is SDN?**
  - Limitations of current networks
  - The idea of Network OS
- **What is OpenFlow?**
  - How it helps SDN
- **Application: Network debugging with SDN**

Making Networks Work (Today)

`traceroute, ping, tcpdump, SNMP, Netflow`

…. er, that's about it.

Why debugging networks is hard

Complex interaction

- Between multiple protocols on a switch/router.
- Between state on different switches/routers.

Multiple uncoordinated writers of state.

Operators can't…

- Observe all state.
- Control all state.

Networks are kept working by

“Masters of Complexity”

A handful of books

~~Almost no papers~~

No classes

Many papers since 2012 in the SDN context

Philosophy of Making Networks Work

YoYo

"You're On Your Own"

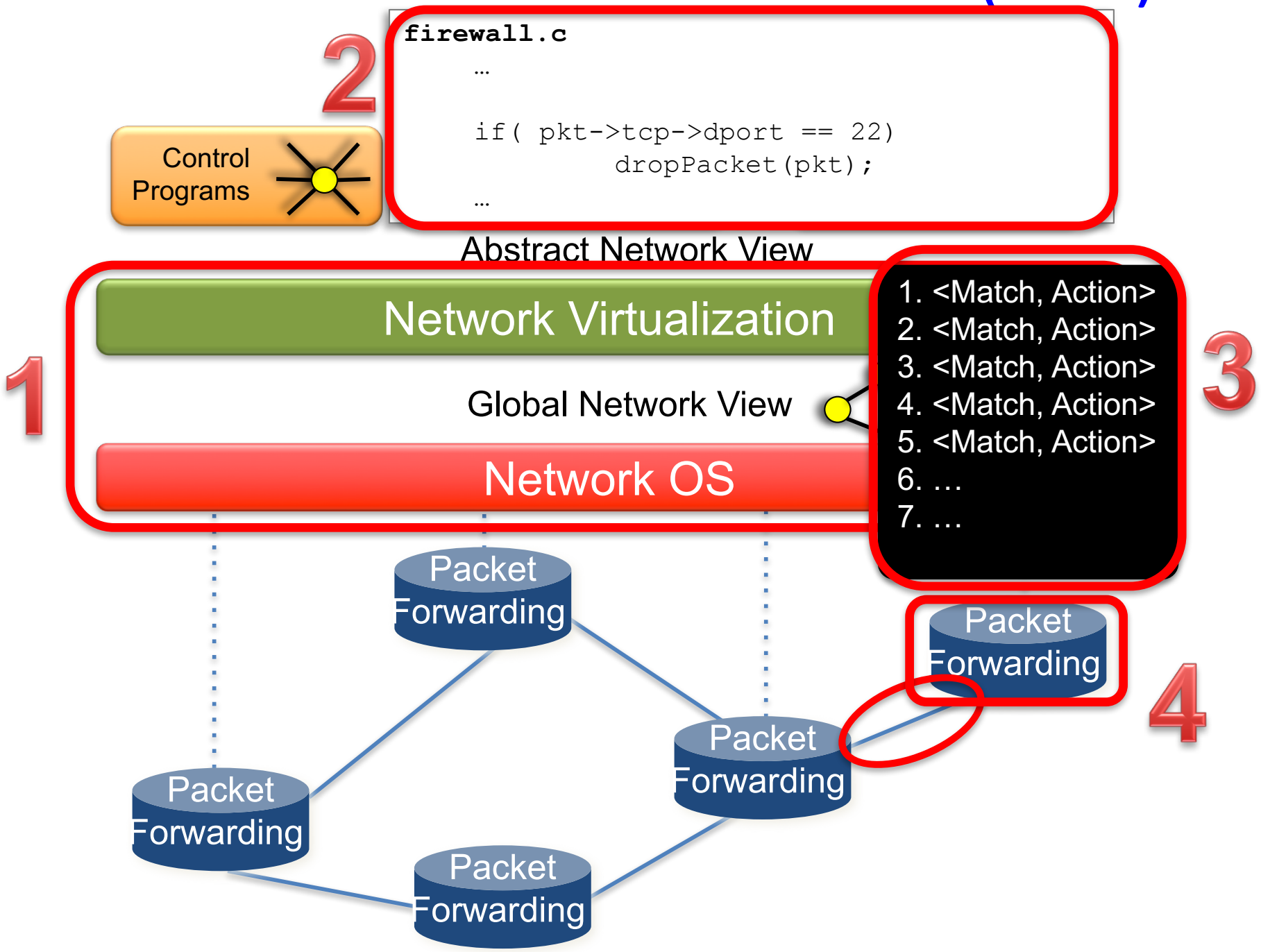
Yo-Yo Ma

"You're On Your Own, Mate"

With SDN we can:

1. Formally verify that our networks are behaving correctly
2. Identify bugs, then systematically track down their root cause

Software Defined Network (SDN)



Example debugging tools

1. Interactive Debugging [ndb]
   "Finding bugs, and their root cause, in an operational network"

2. PathDump: A path-tracing based debugger
   "Tracing paths of individual packets and debugging problems in a datacenter network"

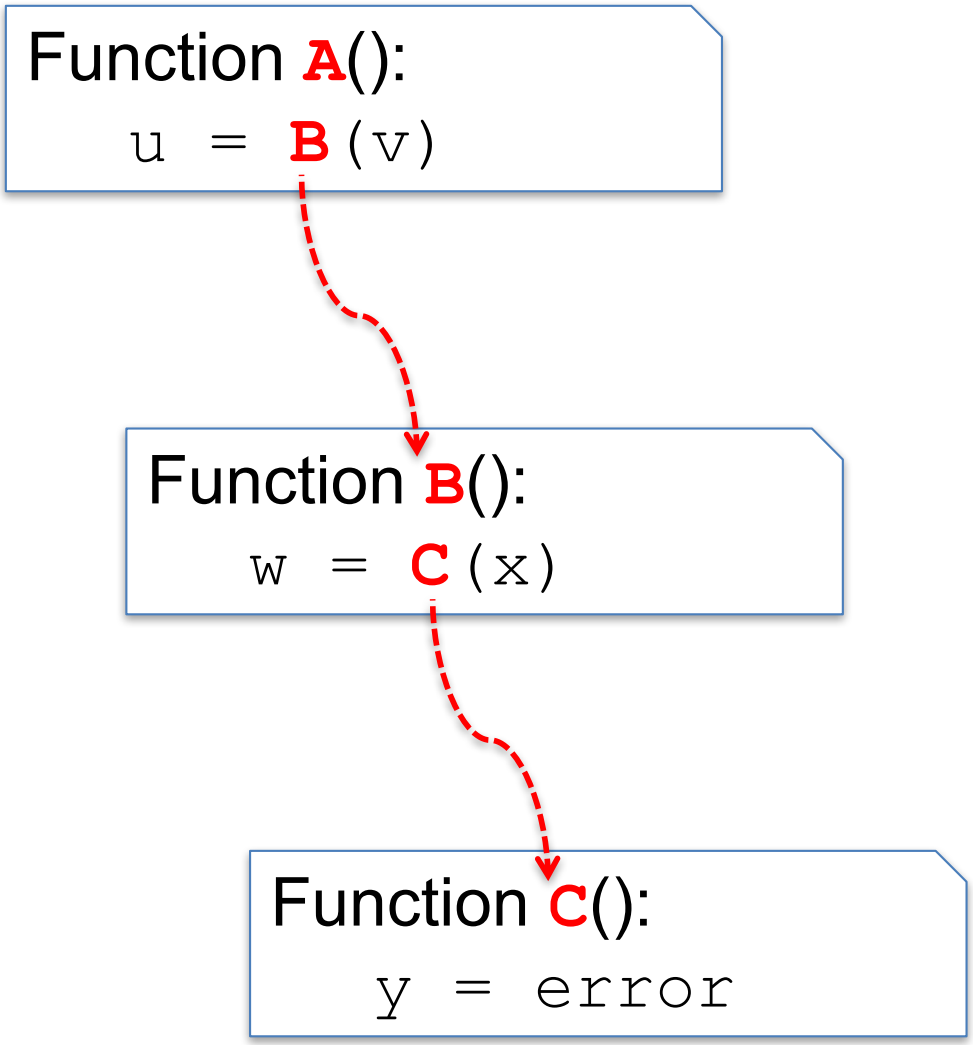
1. Interactive Debugging

Finding bugs, and their root cause, in an operational network

Nikhil Handigol
Brandon Heller
Vimal Jeyakumar
David Mazières

Stanford University

Backtrace: Software Programming



Breakpoint

"u == error"

Backtrace

File "A", line 10, Function A()
File "B", line 43, Function B()
File "C", line 21, Function C()

Interactive Debugging with `ndb`

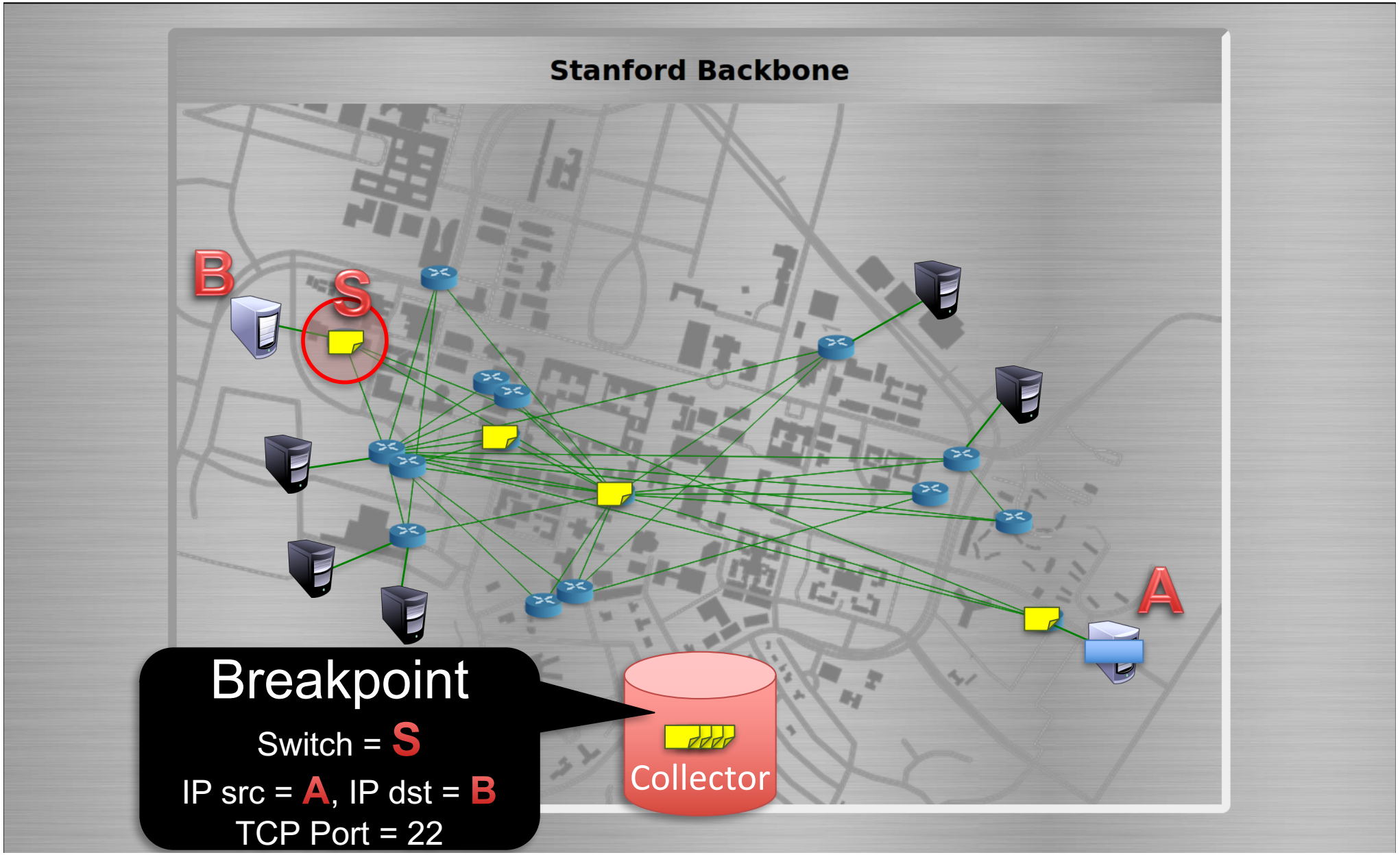
Problem

When an operational network misbehaves,
it is very hard to find the root cause

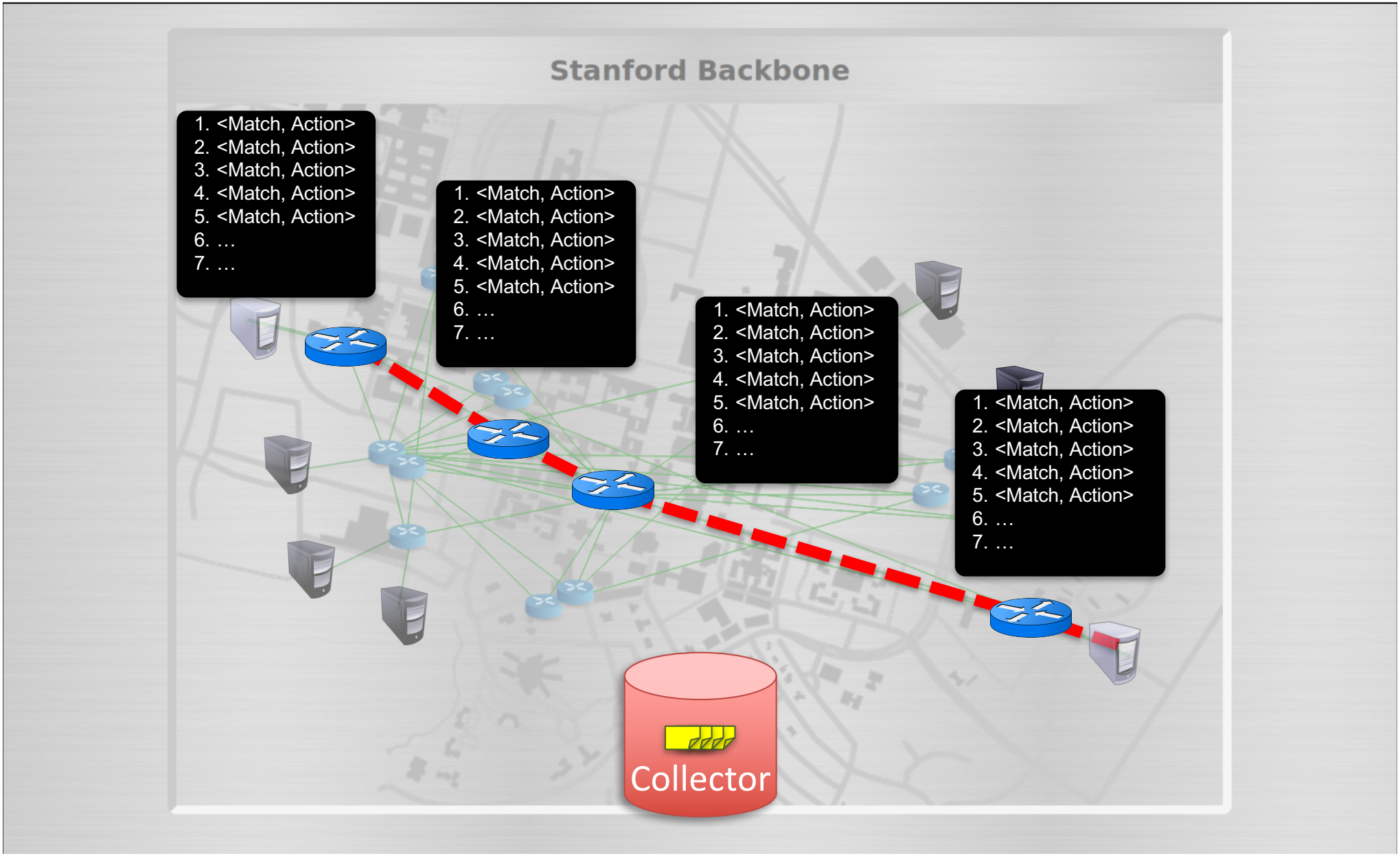
Goal

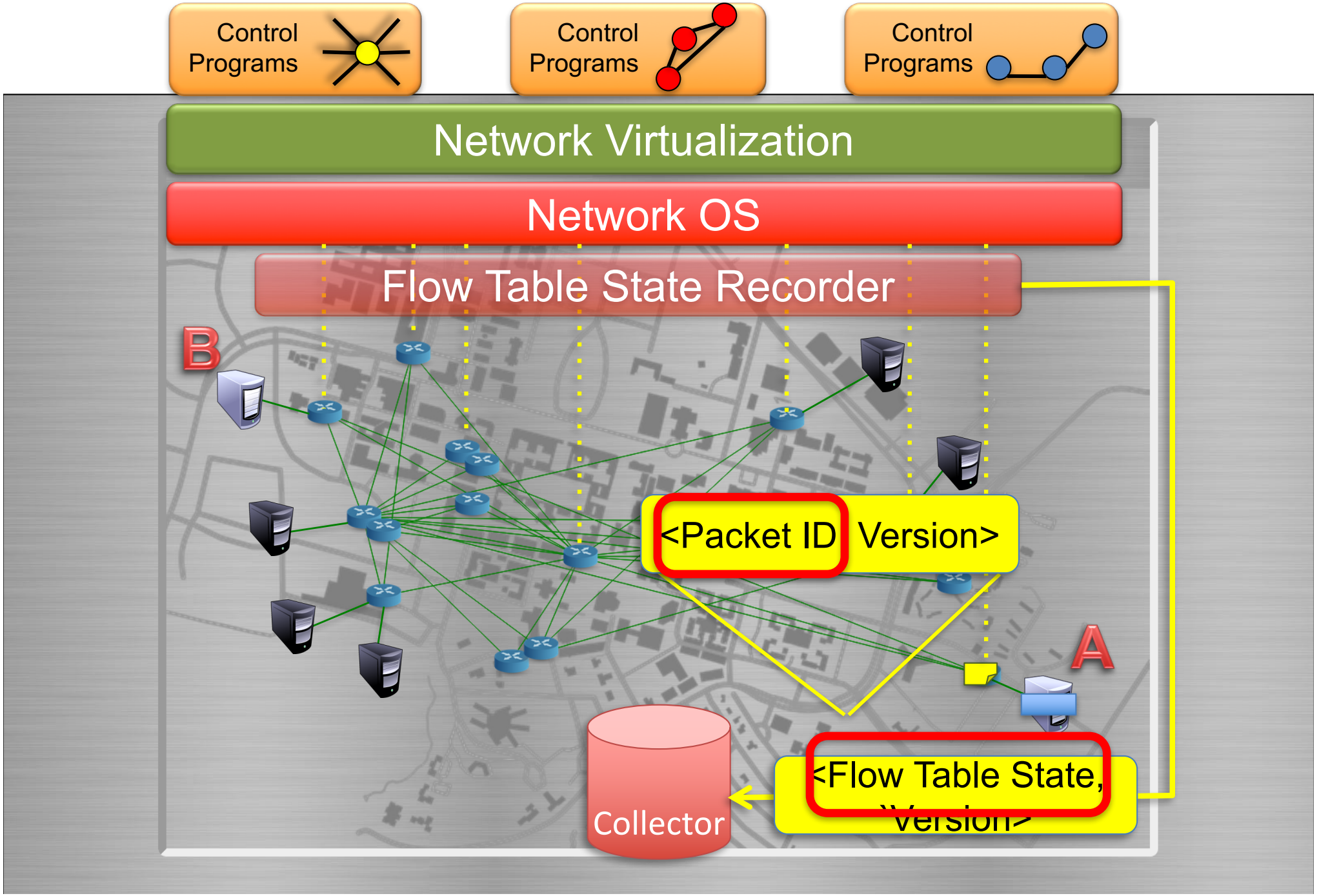
- Allow users to define a Network Breakpoint
- Capture and reconstruct the sequence of events leading to the breakpoint.

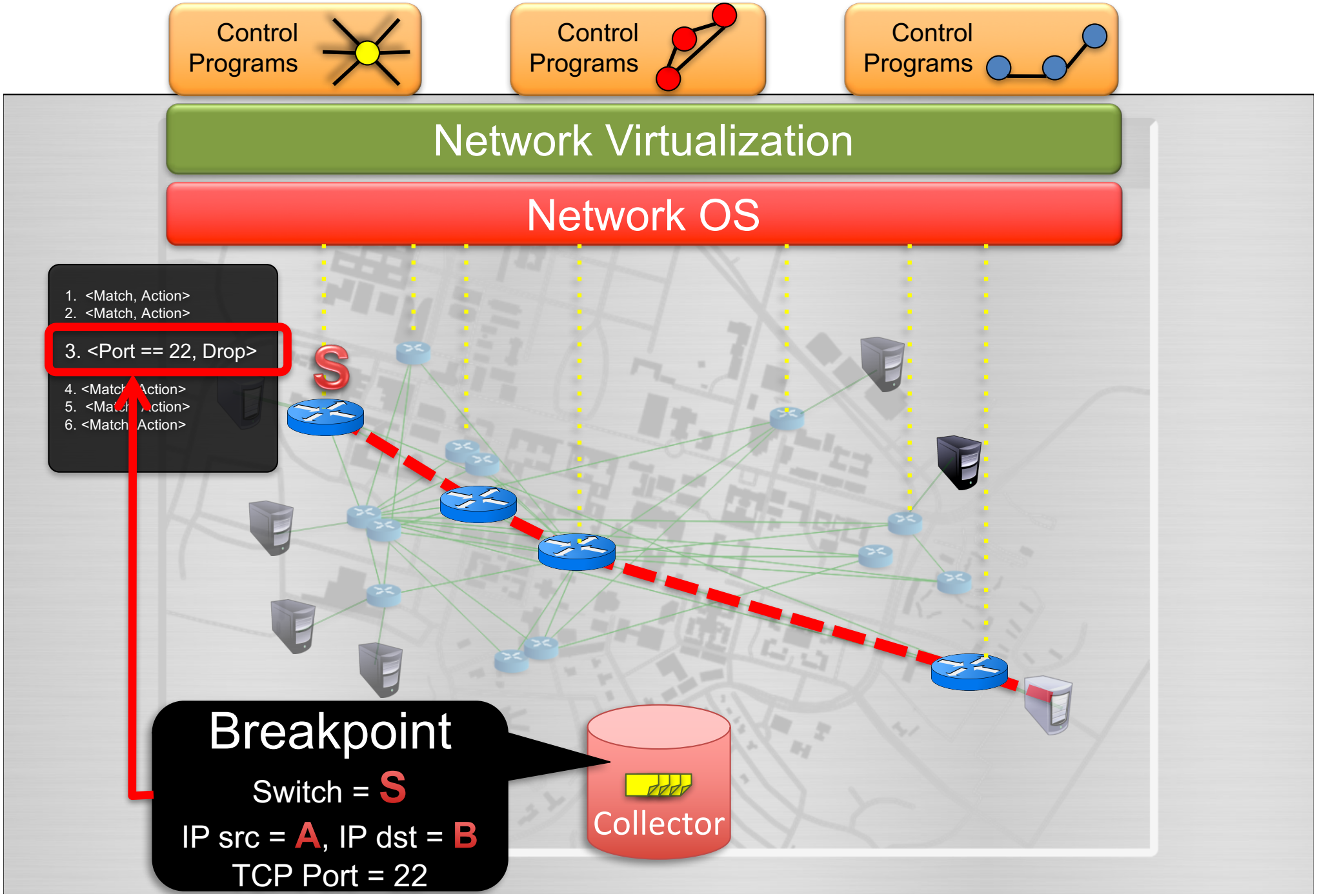
Network Debugger



Network Debugger




Control Programs
Control Programs
Control Programs
Network Virtualization
Network OS
Flow Table State Recorder
B
A
<Packet ID Version>
Collector
<Flow Table State,


Control Programs
Control Programs
Control Programs
Network Virtualization
Network OS
1. <Match, Action>
2. <Match, Action>
3. <Port == 22, Drop>
4. <Match, Action>
5. <Match, Action>
6. <Match, Action>
S
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Collector


firewall.c
…
if(pkt->tcp->dport == 22)
dropPacket(pkt);
…
Control Programs
Network Virtualization
Network OS
1. <Match, Action>
2. <Match, Action>
3. <Port == 22, Drop>
4. <Match, Action>
5. <Match, Action>
6. <Match, Action>
S
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Collector

Who benefits

Network developers

- Programmers debugging control programs

Network operators

- Find policy error
- Send error report to switch vendor
- Send error report to control program vendor

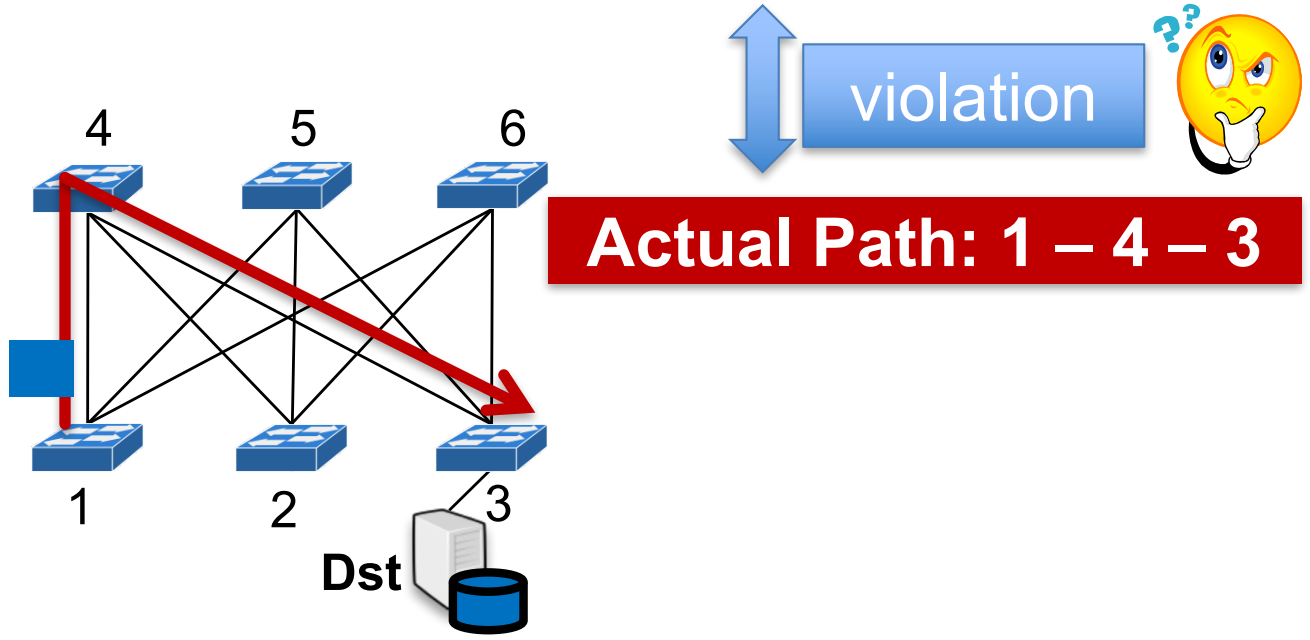
2. PathDump

Tracing paths of individual packets and debugging problems in a datacenter network

http://homepages.inf.ed.ac.uk/mlee23/paper/osdi16.pdf

Packet trajectory tracing

- "Tracing" the path taken by the packet
  - Scalability: Switch flow rules, packet header space
- Policy: **All packets from 1 to 3 must avoid 4**

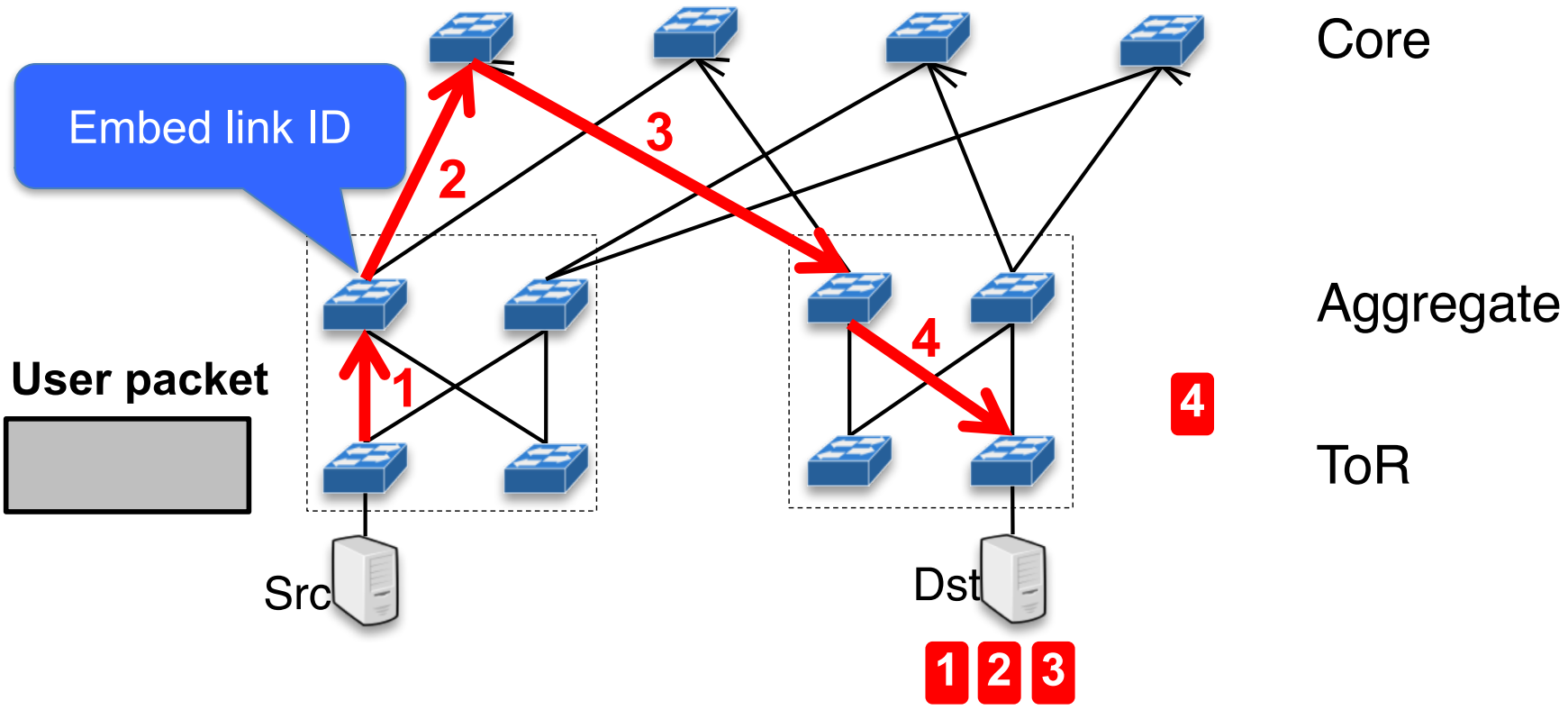


Packet trajectory tracing

- Checks whether packet followed route as defined in policy
- Helps to localize network problems
  Ex: misconfiguration, failures

PathDump architecture

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

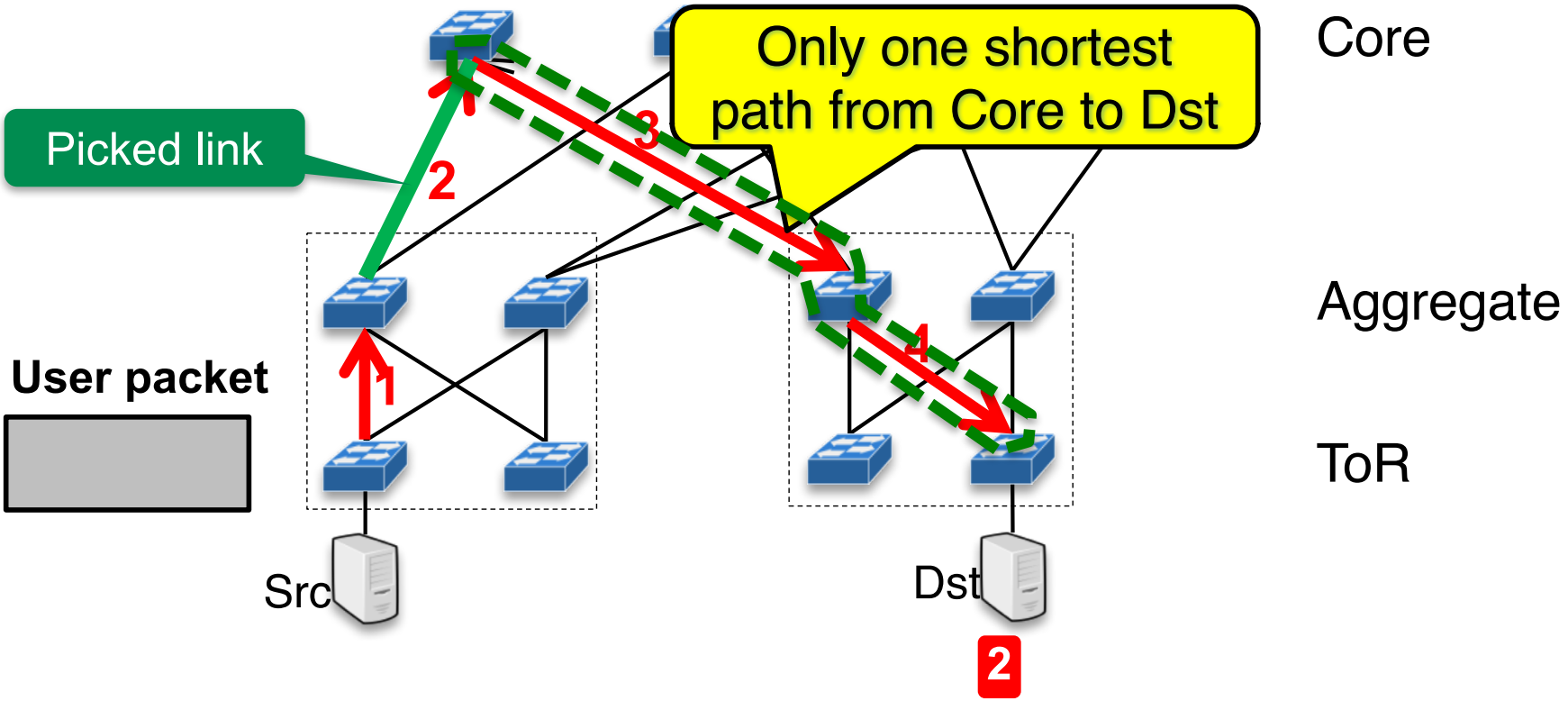


PathDump architecture

- Packet header space limitation
- Cherrypick [SOSR'15] for current deployments

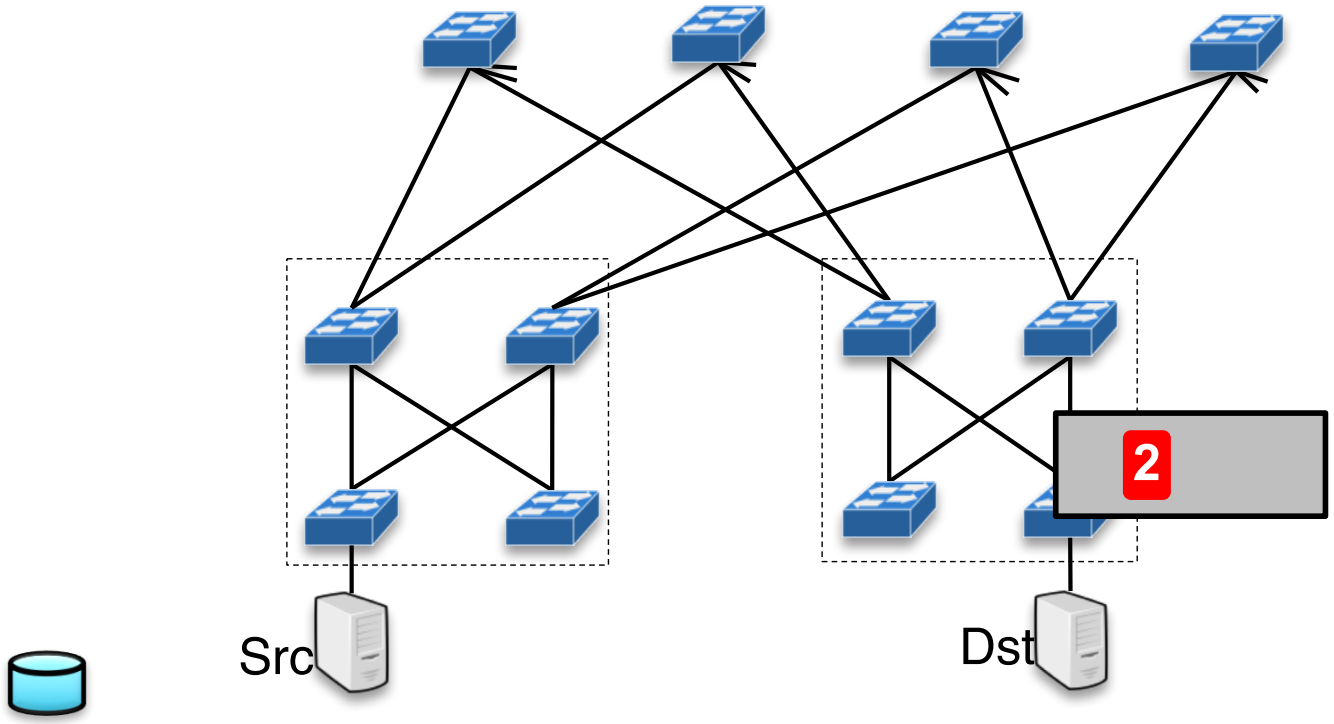
PathDump architecture

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



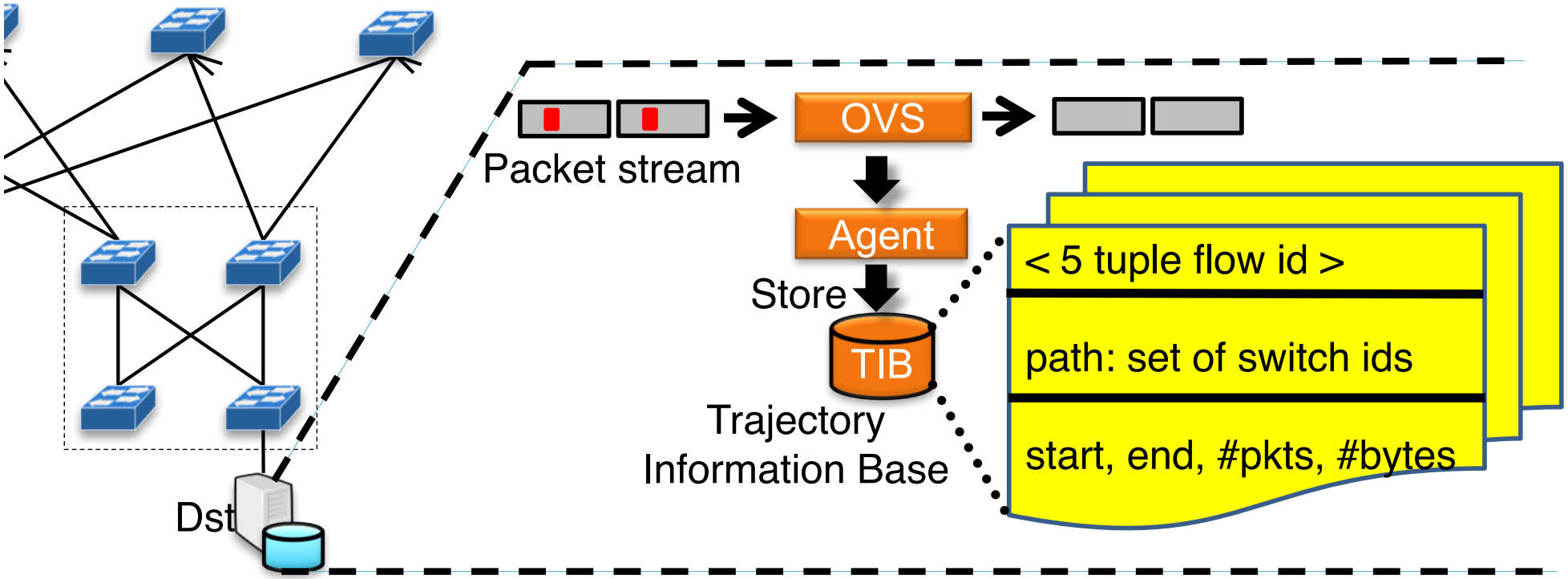
PathDump architecture

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



PathDump architecture

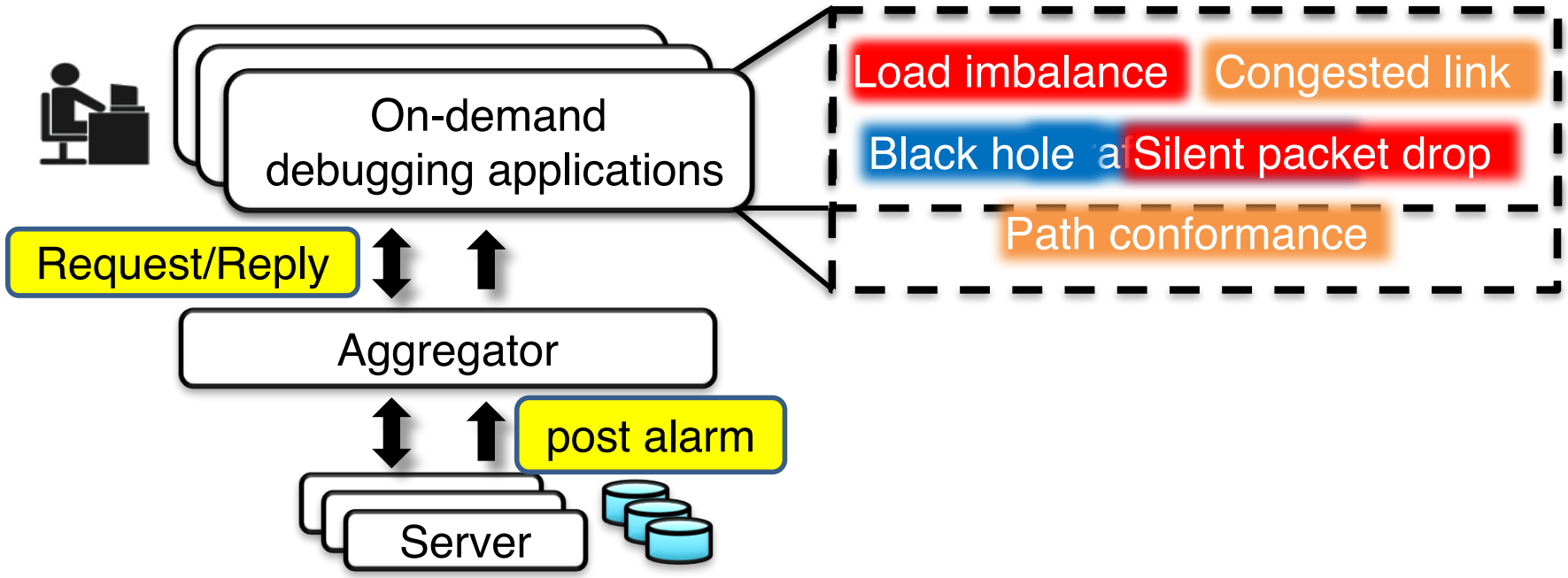
2. End-host captures packet path and updates flow-level statistics



PathDump architecture

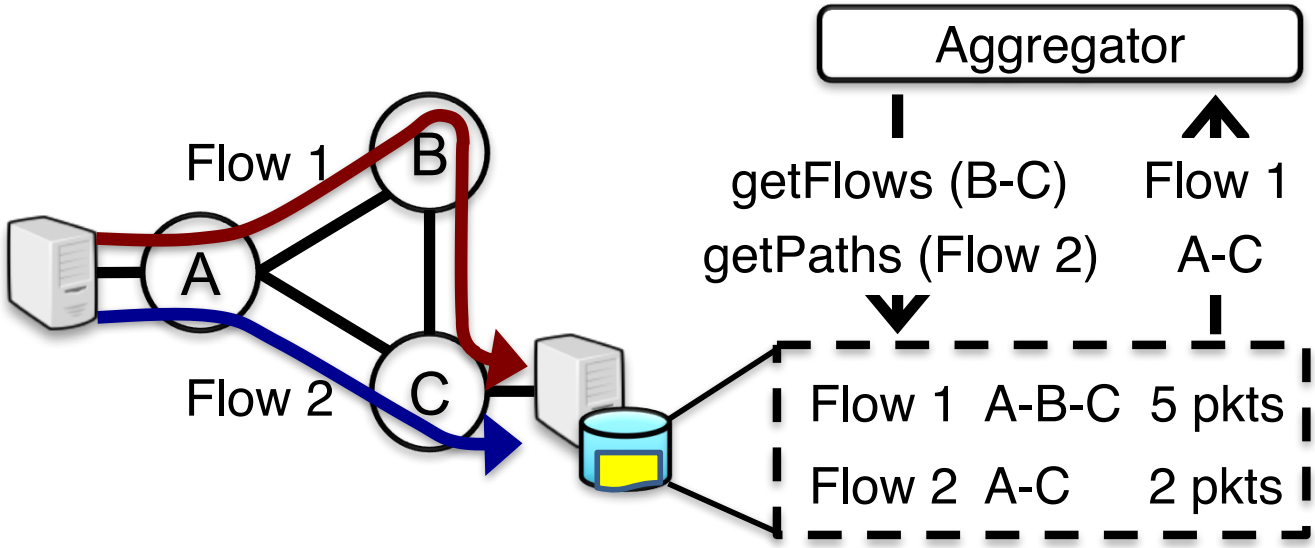
3. Aggregator runs debugging applications

On-demand vs. Event-driven



PathDump interface

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

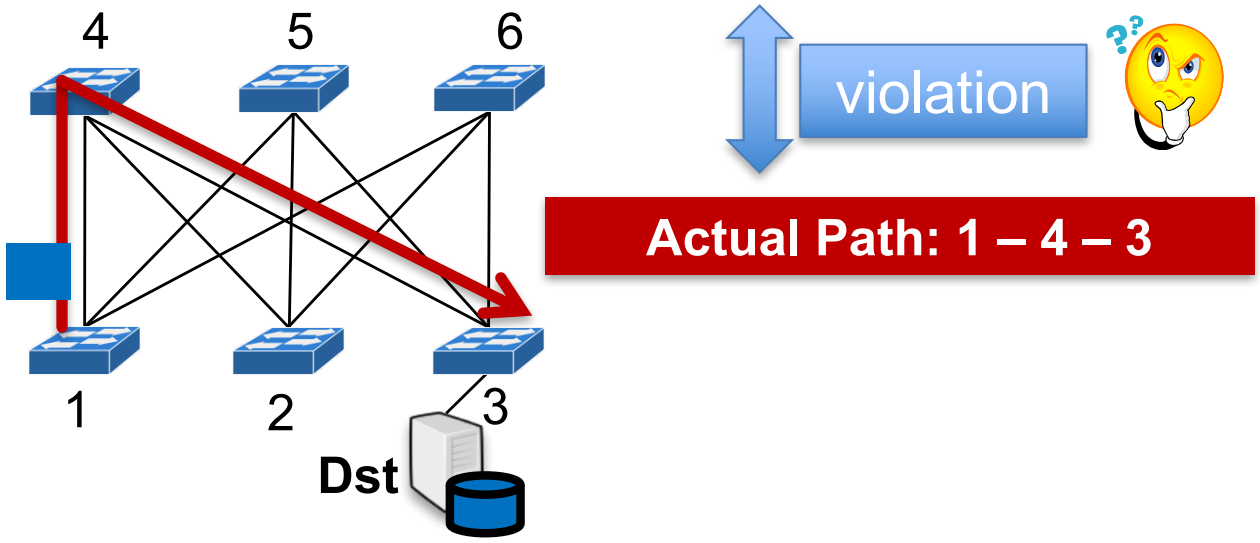


- Other end-host APIs: getCount(), getPoorTCPFlows(), Alarm(), etc.
- Aggregator APIs: Install(), execute() and uninstall()

Example 1: Path conformance

- Check if actual forwarding path != network policy
  - May occur due to switch faults or network state change

Policy: Packet must avoid switch 4



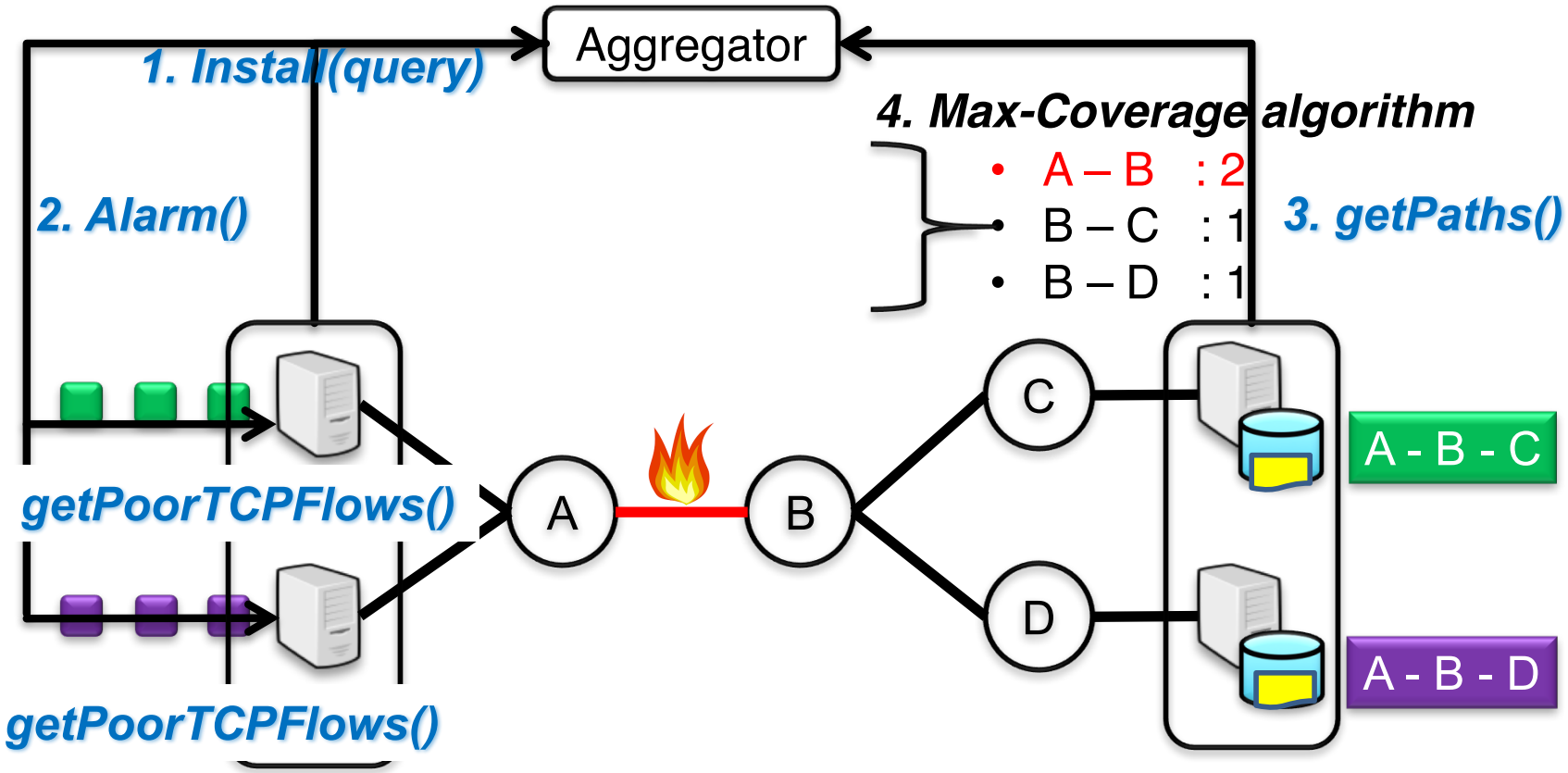
Example 1: Path conformance

- Check if actual forwarding path != network policy
  - May occur due to switch faults or network state change

```
# 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



- No packet drop hint
- Software/Hardware bug