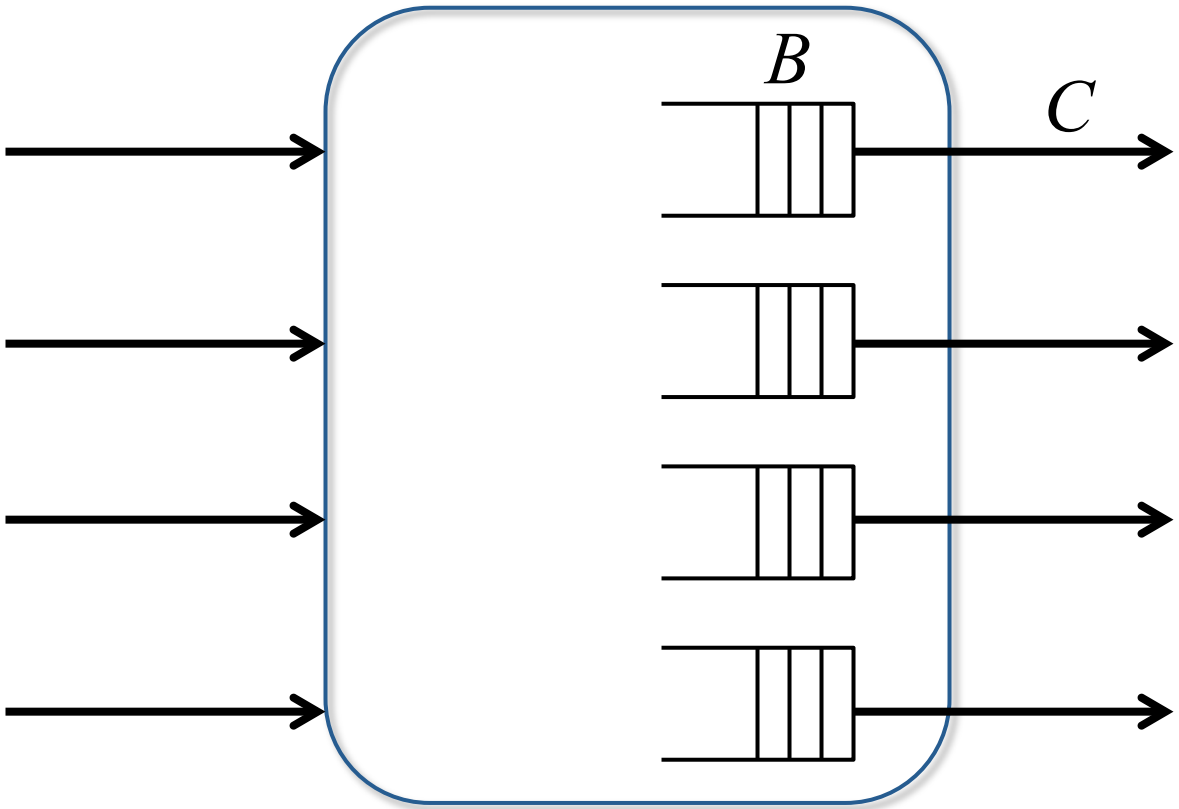
How SDNs will tame networks

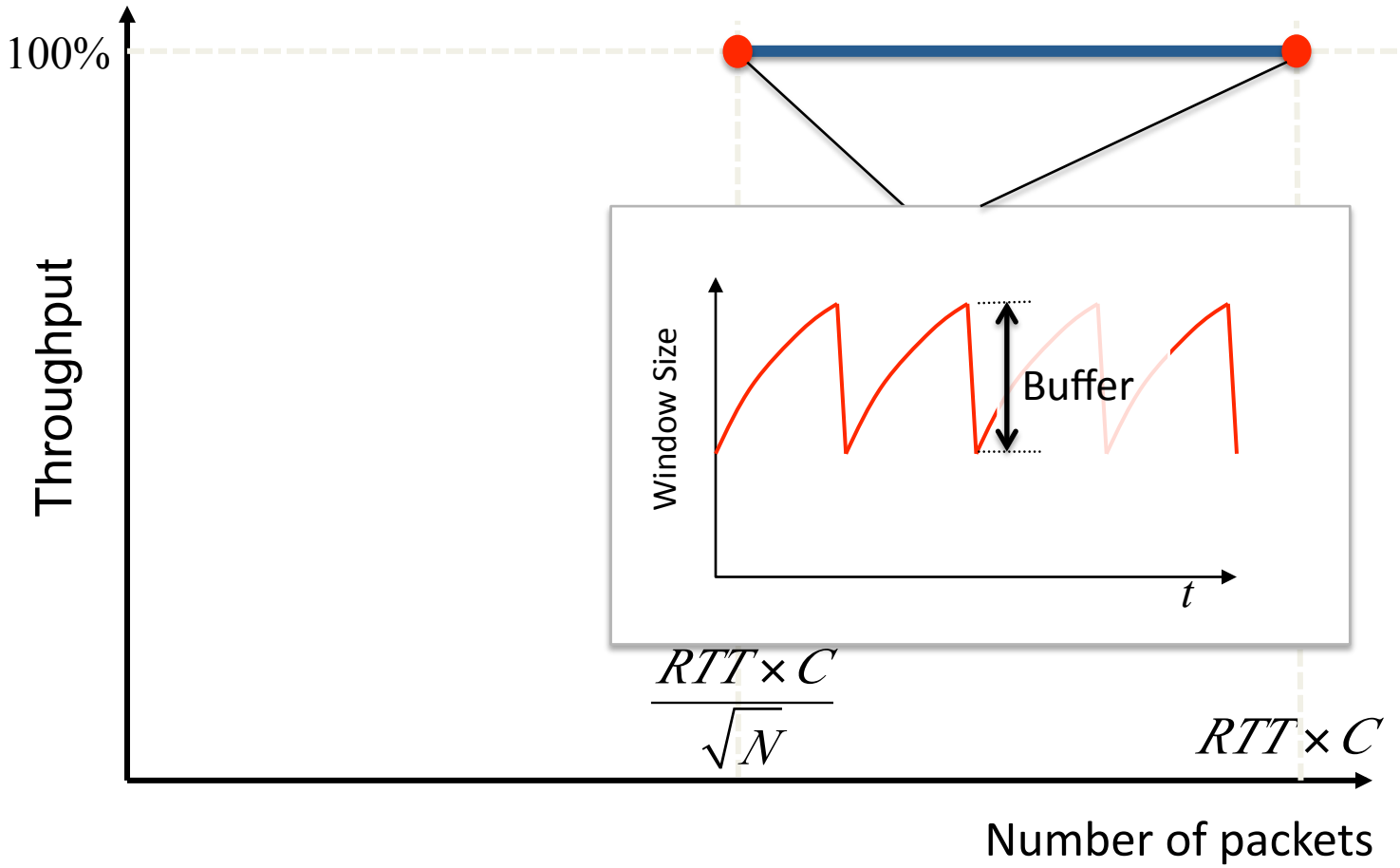
Nick McKeown
Stanford University

As I was saying…

How big to make a backbone router buffer?

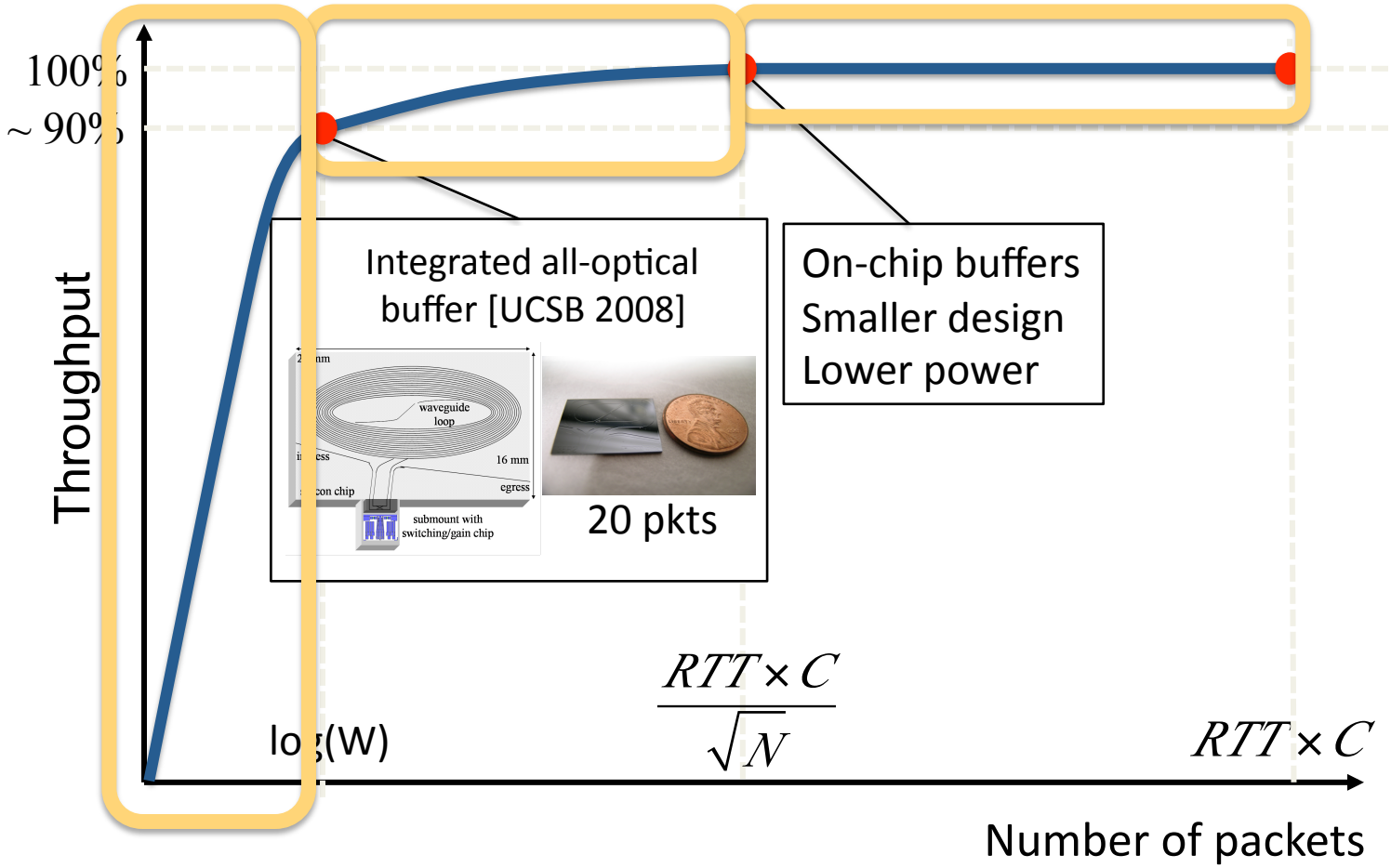


Buffer Size



10Gb/s WAN 25,000 2,500,000

Buffer Size



10Gb/s WAN	~50	25,000	2,500,000

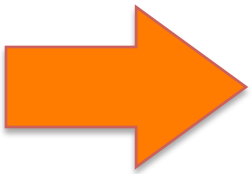
Software Defined Networks

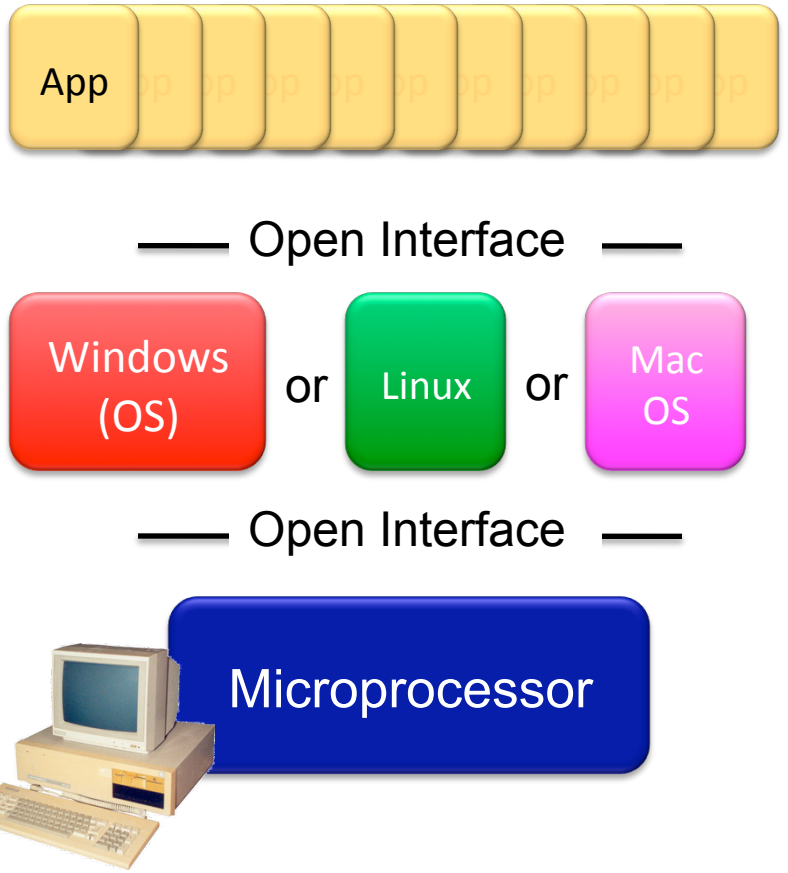
Scott Shenker
Teemu Koponen

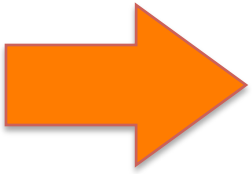
Martin Casado

Guru Parulkar
+ many (brave) students

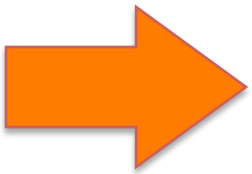
Vertically integrated
Closed, proprietary
Slow innovation
Small industry

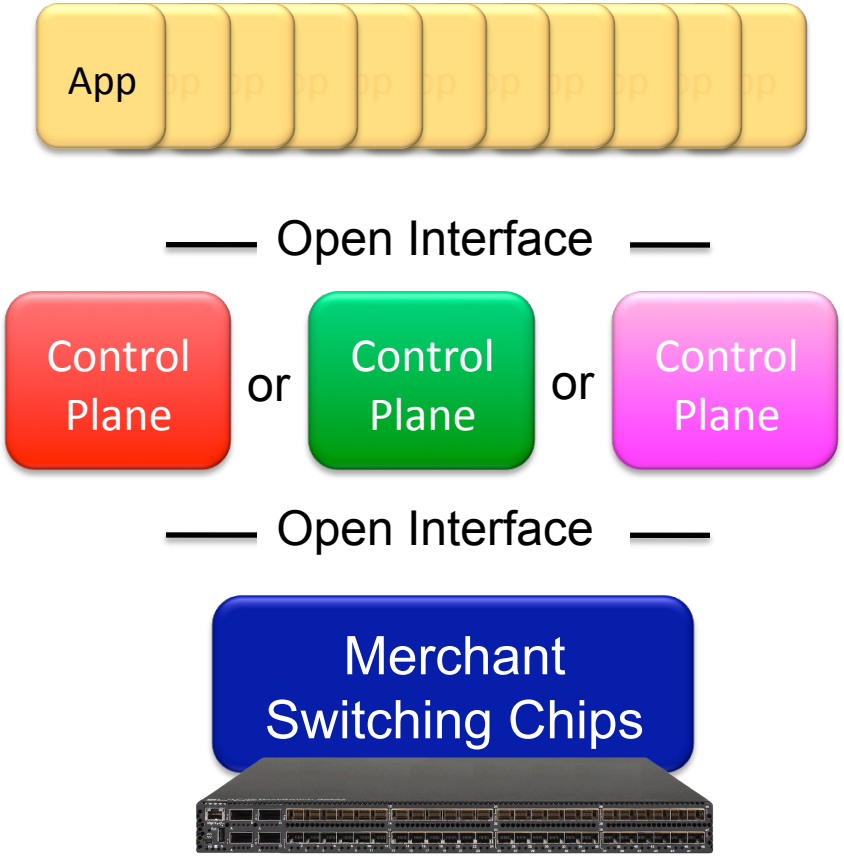




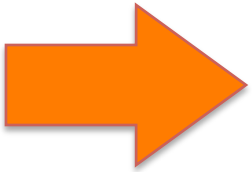


Horizontal
Open interfaces
Rapid innovation
Huge industry





Vertically integrated
Closed, proprietary
Slow innovation



Horizontal
Open interfaces
Rapid innovation

Where SDN will be deployed

1. Multi-tenant “virtualized” data centers
   - Public and private clouds
2. WANs
   - Google WAN
   - Eventually, public WANs
3. Enterprise networks
   - Greater control, fewer middleboxes

Where SDN will be deployed (2)

4. Home networks
   - Outsourced management

5. Cellular Networks
   - Separation of service from physical infrastructure

6. Research and Education Networks
   - National backbones
   - College campus networks

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”

Tool & Deployment Support

Open Networking Lab (ON.Lab)

- Independent non-profit lab
- Open source tools
- Help with deployments
- Based in Palo Alto
- Hiring…

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 ☺)

Switch ASICs

- Current ASICs work, but not optimized for OpenFlow
- Expect some OpenFlow-optimized ASICs in 1-2 years

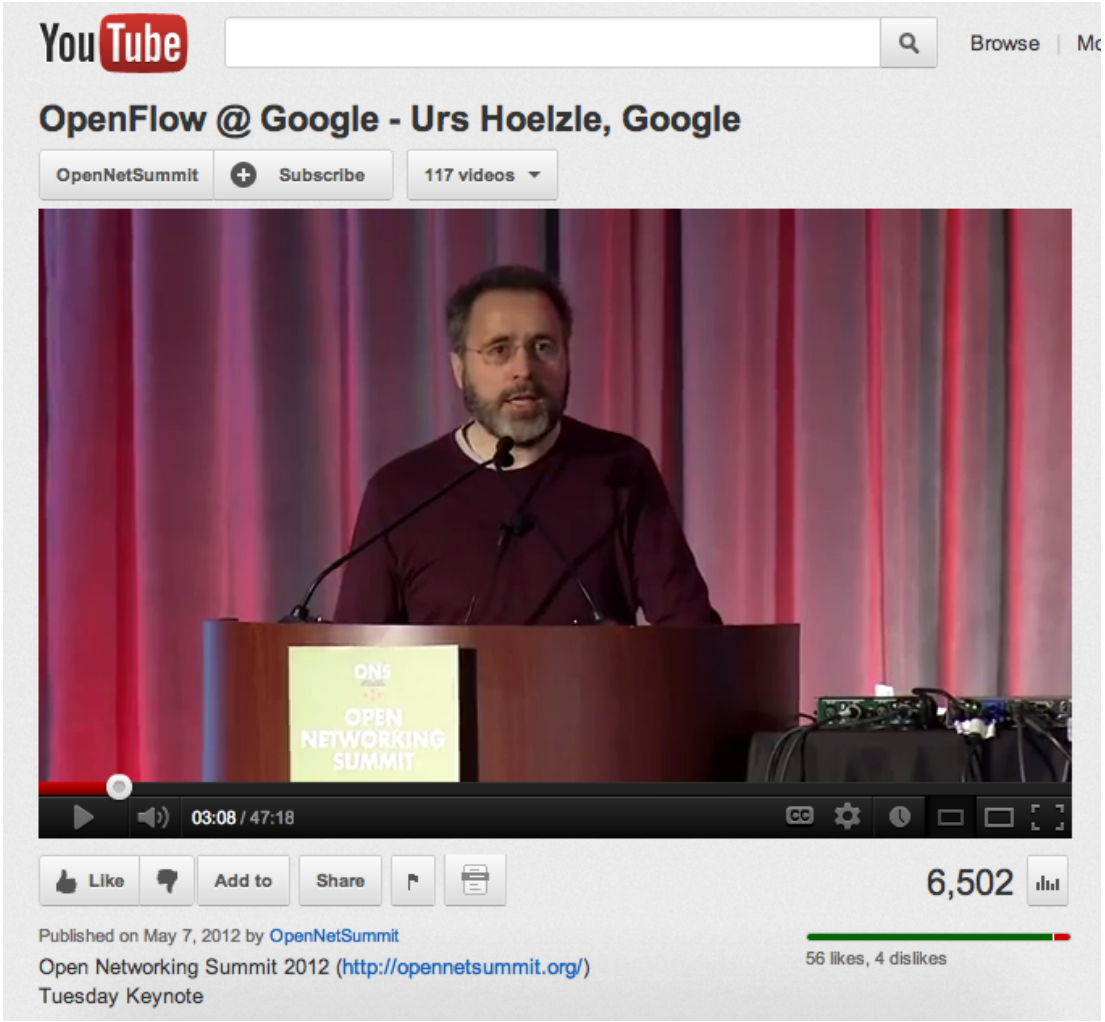
An example: What's possible in silicon

- Stanford/TI Labs collaboration
- 64 x 10Gb/s
- Multiple table support (>12 flexible stages)
- 64k TCAM entries (wide) for wildcards
- 128k hash table entries (wide) for exact matches
- >1k queues per port
- All OpenFlow counters
- On-chip ARM CPU
- Generic ALU-based action engine

With permission from Texas Instruments

If you are in any doubt about whether OpenFlow/SDN will be deployed in the WAN:

Urs Hoelzle (Google) at Open Networking Summit 2012



New Research Area

Making Networks Work

An intellectual framework for verifying, troubleshooting and debugging SDNs

With SDN we can:

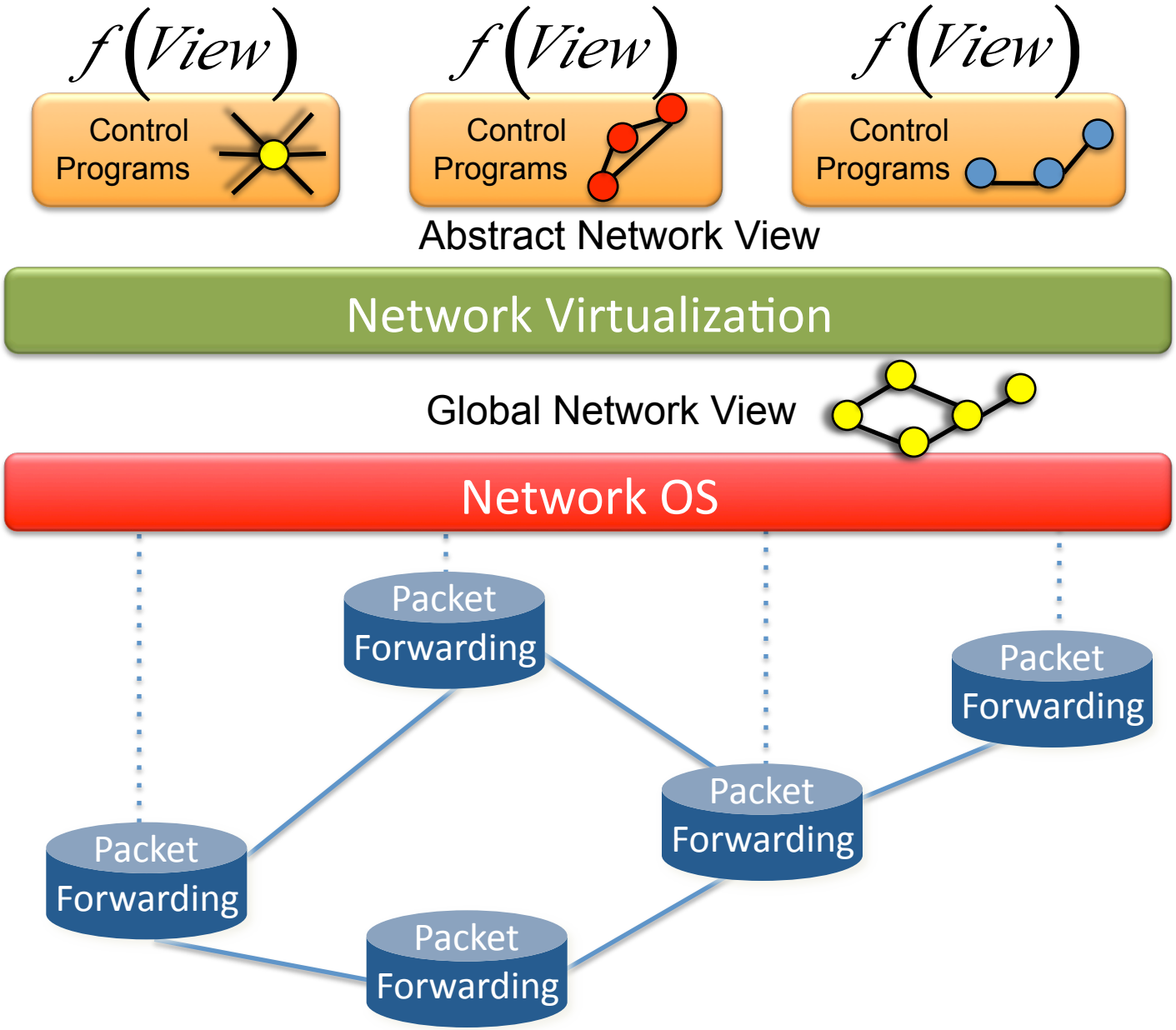
1. Formally verify that our networks are behaving correctly.
2. Identify bugs, then systematically track down their root cause.

- Ensuring correctness [Frenetic][HFT][Netcore]
  Nate Foster, Andrew Ferguson, Mike Freedman, Jen Rexford, Rob Harrison, Dave Walker, ++
- Software Fault Localization [W3]
  Scott Shenker, Colin Scott, Kyriakos Zarifis, Andreas Wundsam.
- Checking behavior [NICE]
  Marco Canini, Daniele Venzano, Peter Peresini, Dejan Kostic, Jen Rexford.
- Checking Invariants [VeriFlow]
  Ahmed Khurshid, Wenxuan Zhou, Matthew Caesar, P. Brighten Godfrey
- Consistent updates
  Mark Reitblatt, Rick McGeer, ++
- Troubleshooting [OFRewind]
  Andreas Wundsam, Dan Levin, Srini Seetharaman, Anja Feldman
- …

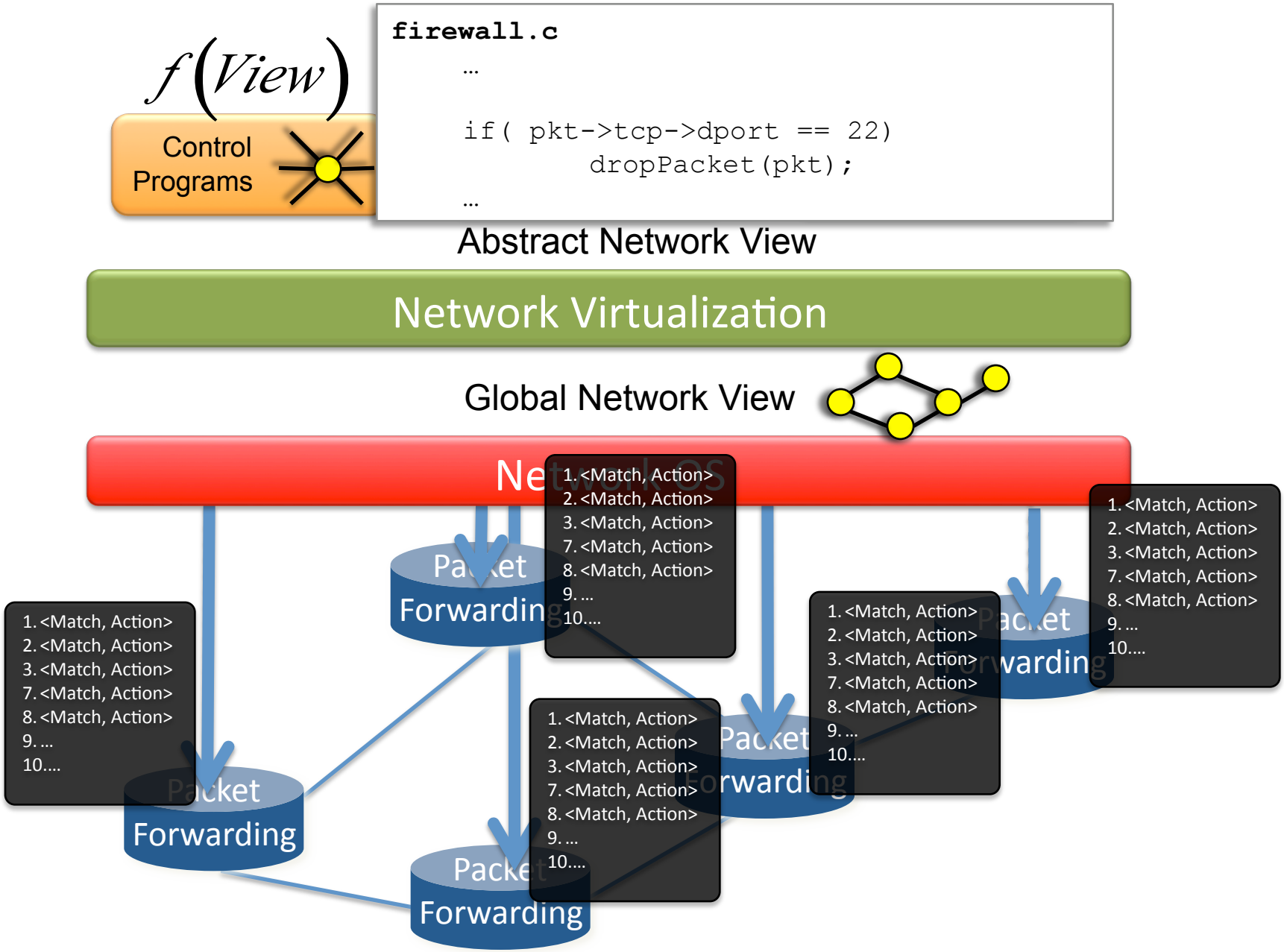
Scott Shenker at 1st ONS in 2011

"The Future of Networking and the Past of Protocols"

Software Defined Network (SDN)

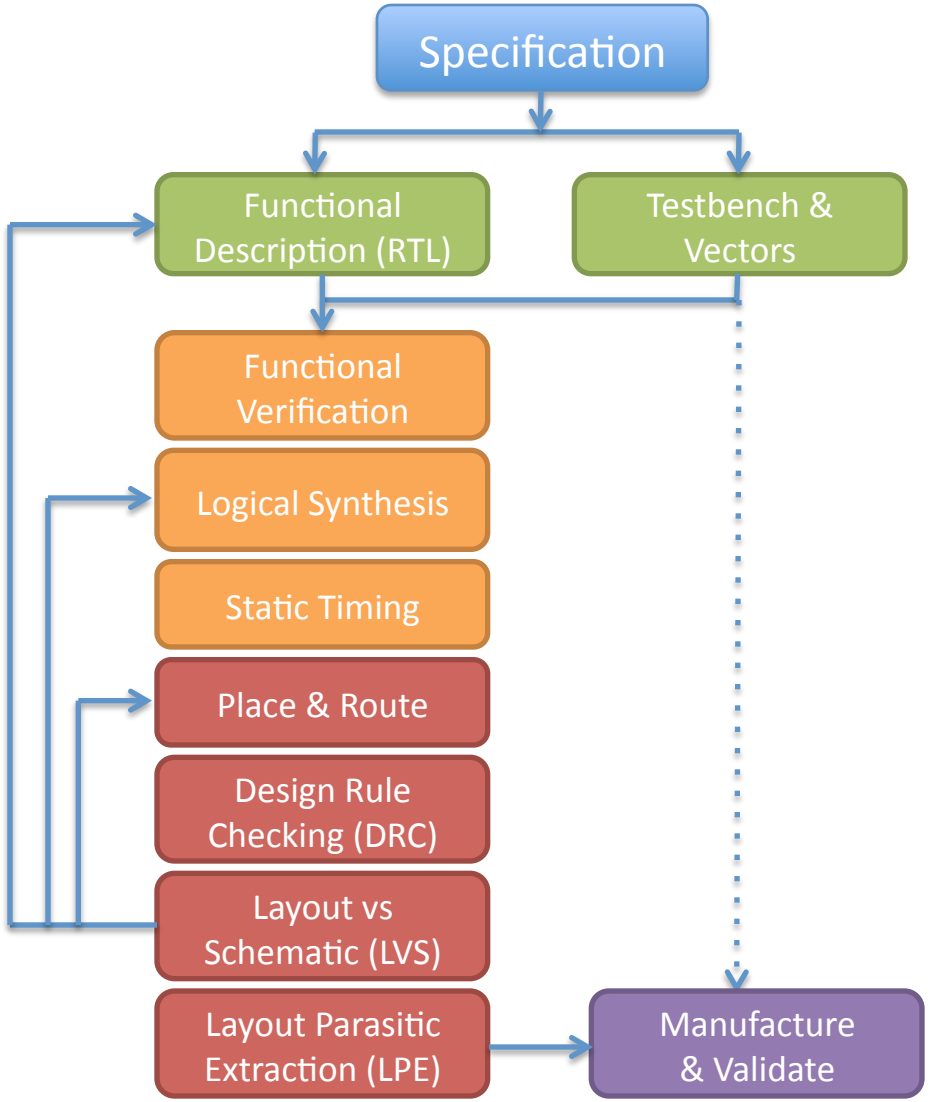


Software Defined Network (SDN)



How do other industries do it?

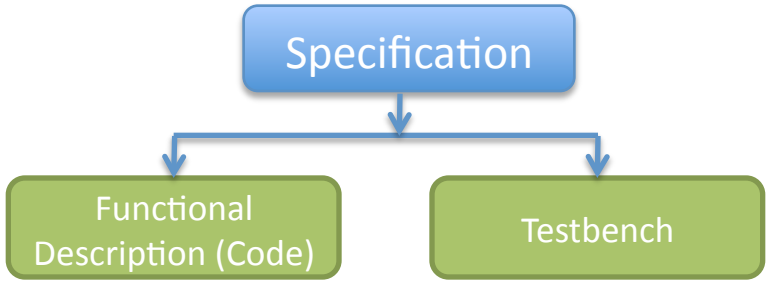
Making ASICs Work



$10B tool business supports a $250B chip industry

100s of Books
>10,000 Papers
10s of Classes

Making Software Work



$10B tool business supports a $300B S/W industry

Static Code Analysis

Invariant Checker

Model Checking

Run-time Checker

Interactive Debugger

100s of Books

>100,000 Papers

10s of Classes

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

Philosophy of Making Networks Work

YoYo

"You're On Your Own"

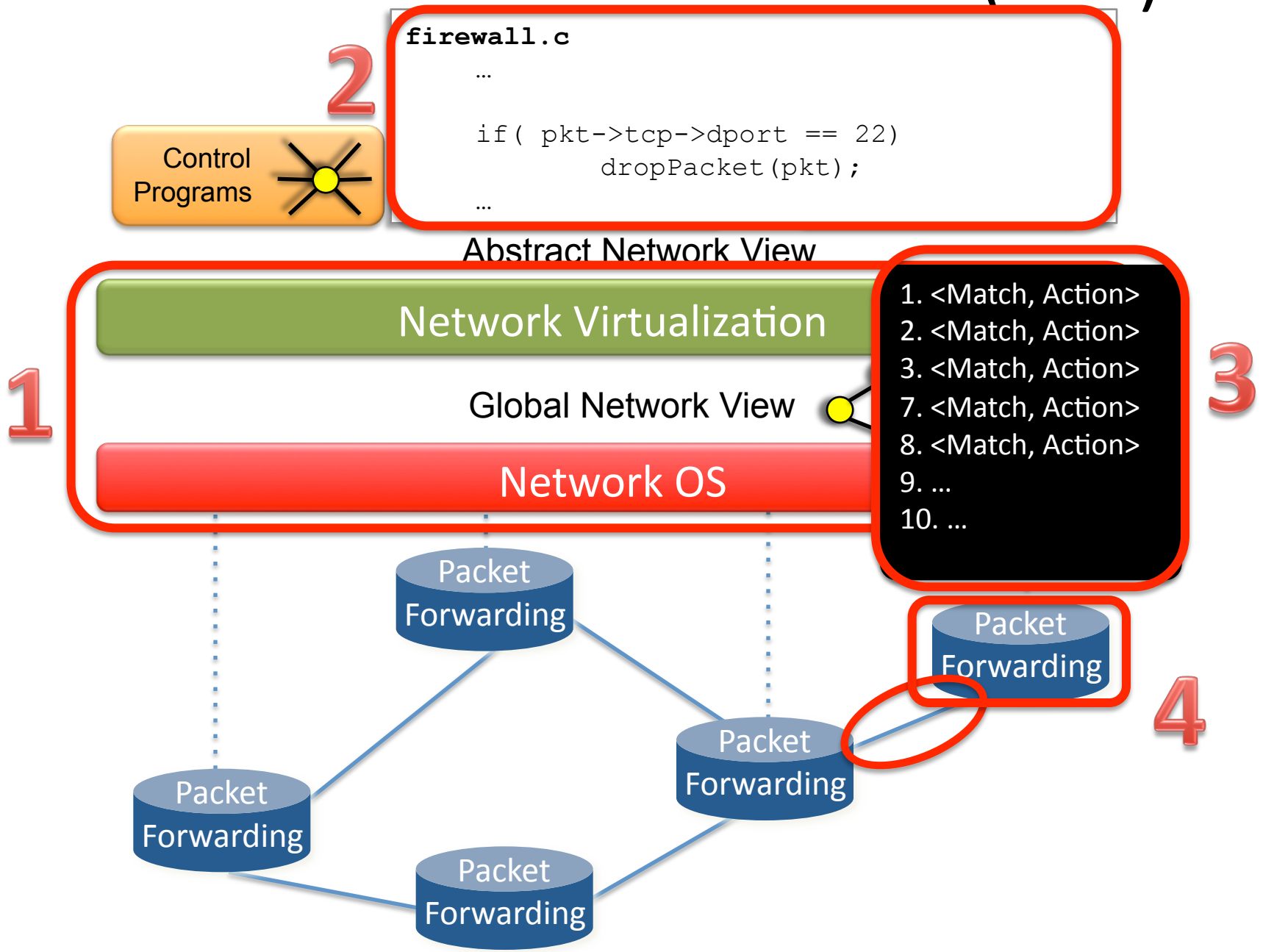
Yo-Yo Ma

"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)



Three of our projects

1. Static Checking [HSA]
   "Independently checking correctness"

2. Automatic Testing [ATPG]
   "Is the datapath behaving correctly?"

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

1. Static checking

Independently checking correctness

Peyman Kazemian

Hongyi 'James' Zeng

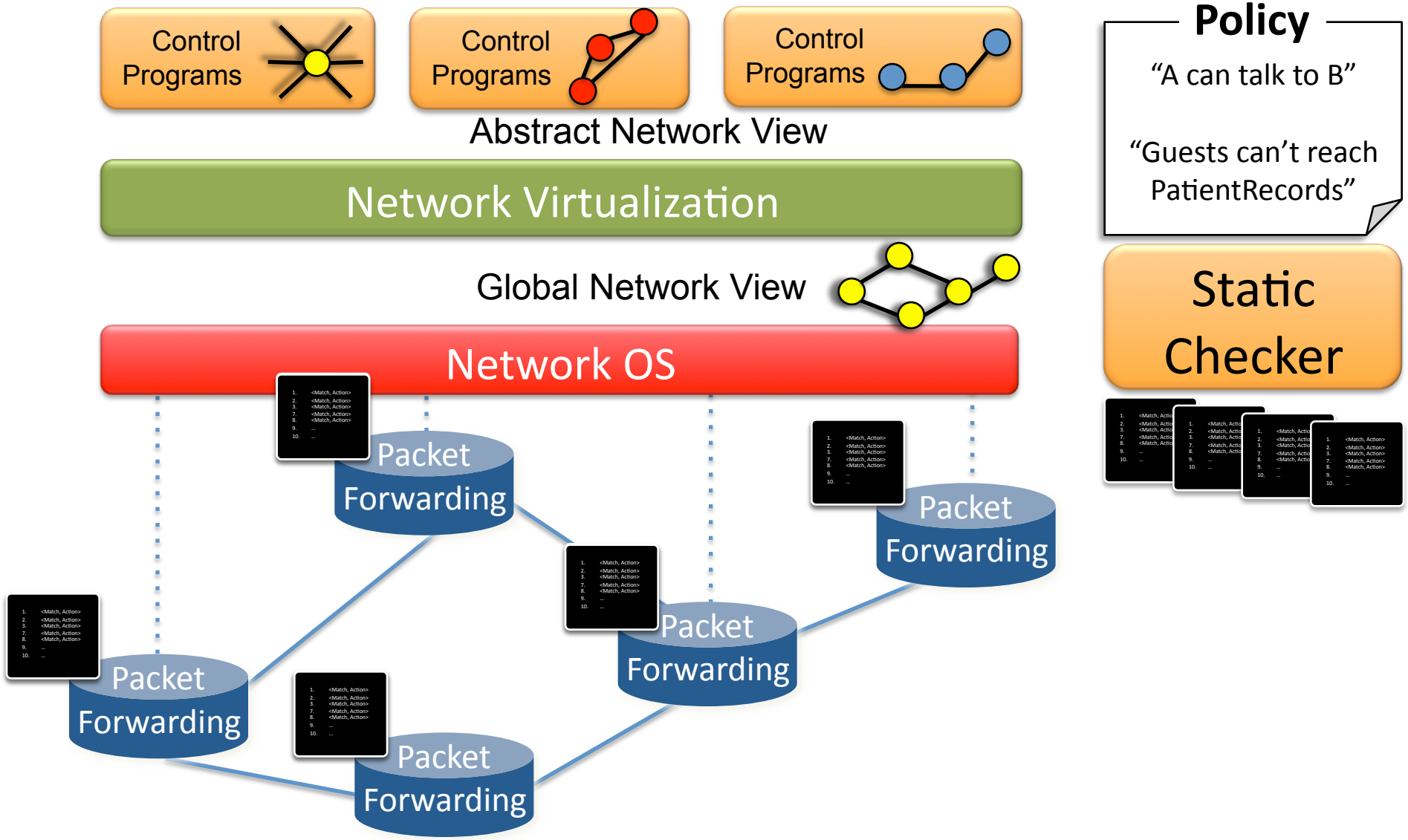
George Varghese (UCSD)

Motivations

In today's networks, simple questions are hard to answer:

- Can host A talk to host B?
- What are all the packet headers from A that can reach B?
- Are there any loops in the network?
- Is Group X provably isolated from Group Y?
- What happens if I remove a line in the config file?

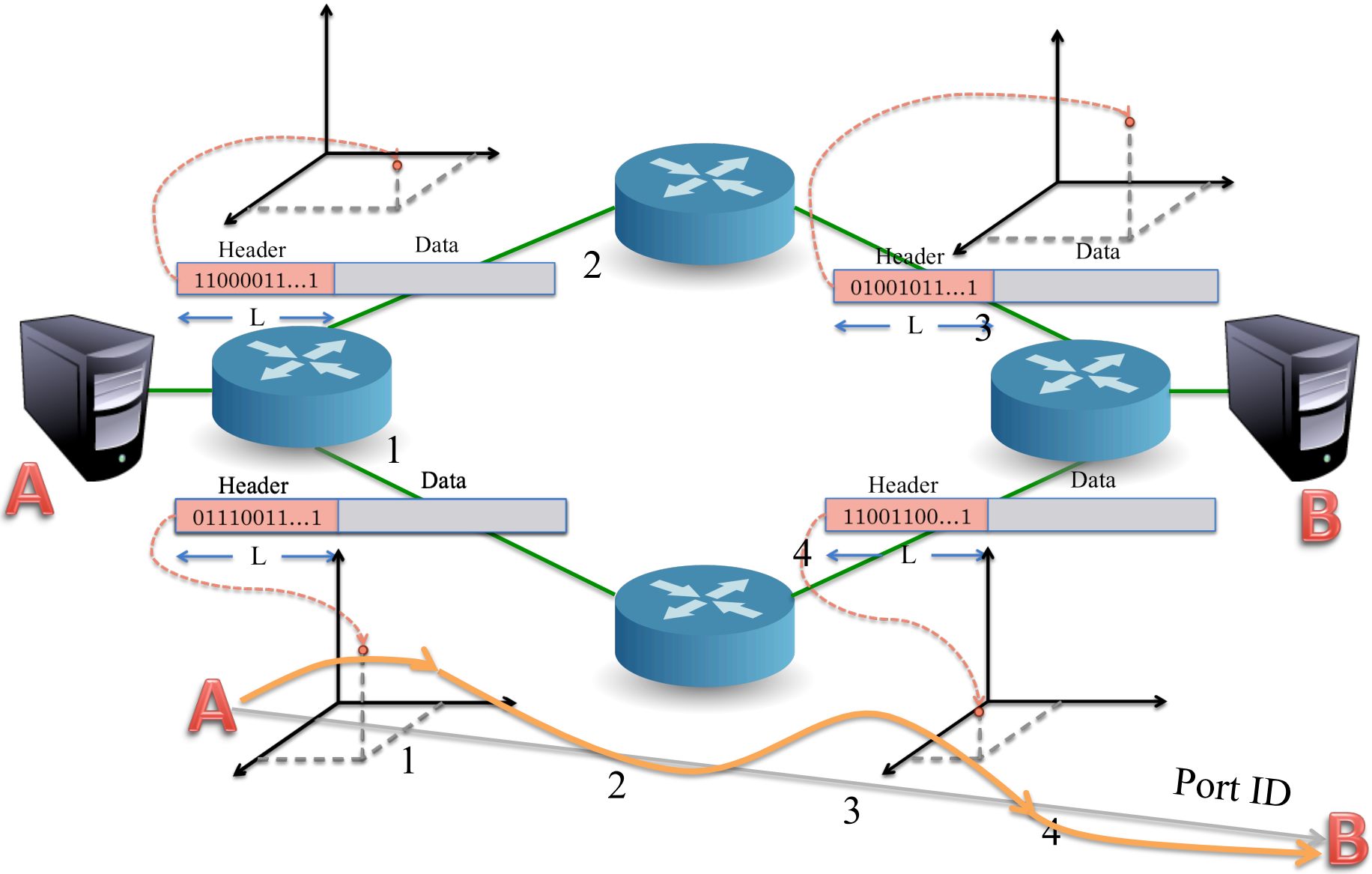
Software Defined Network (SDN)



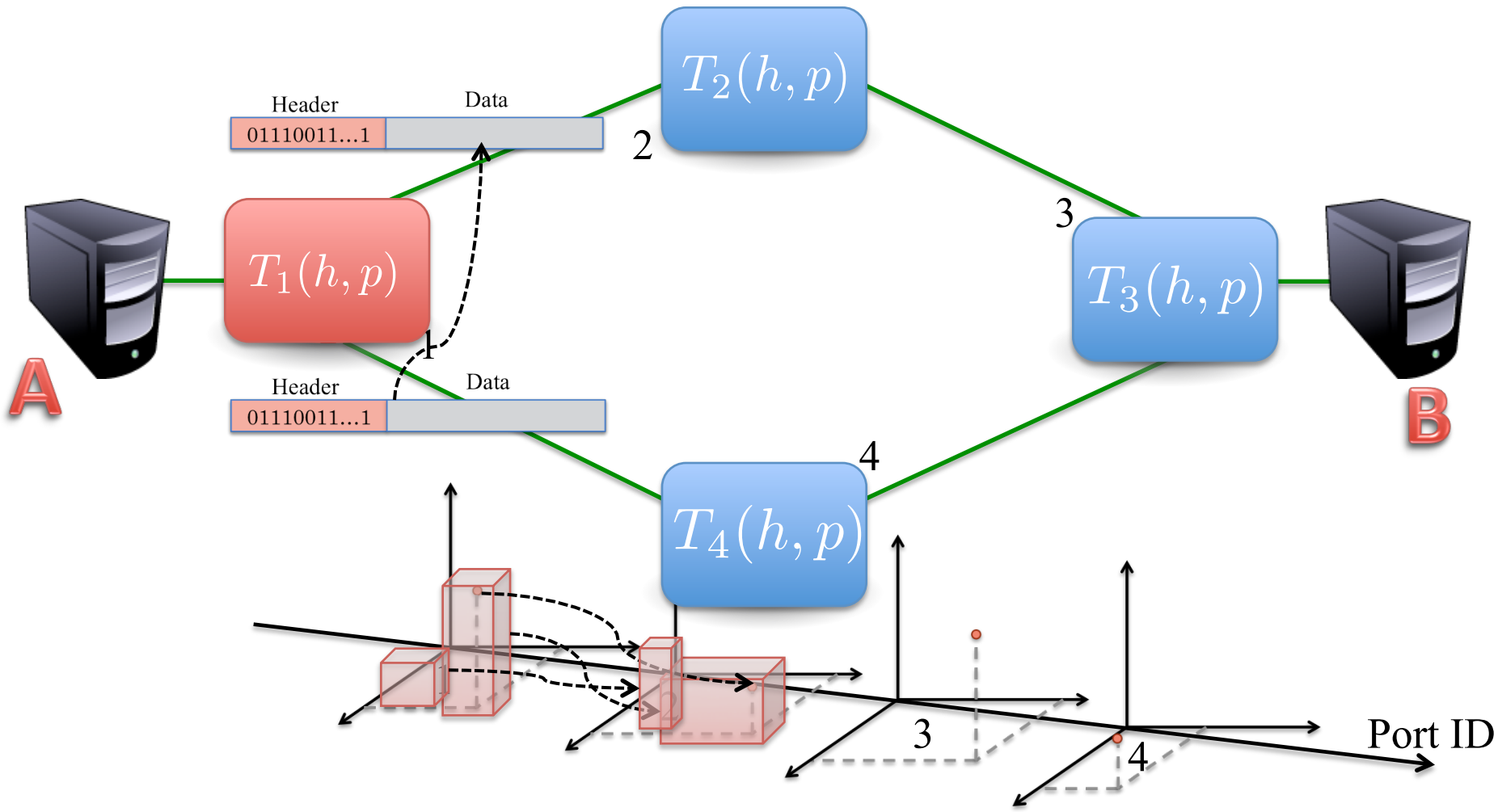
How it works

Header Space Analysis

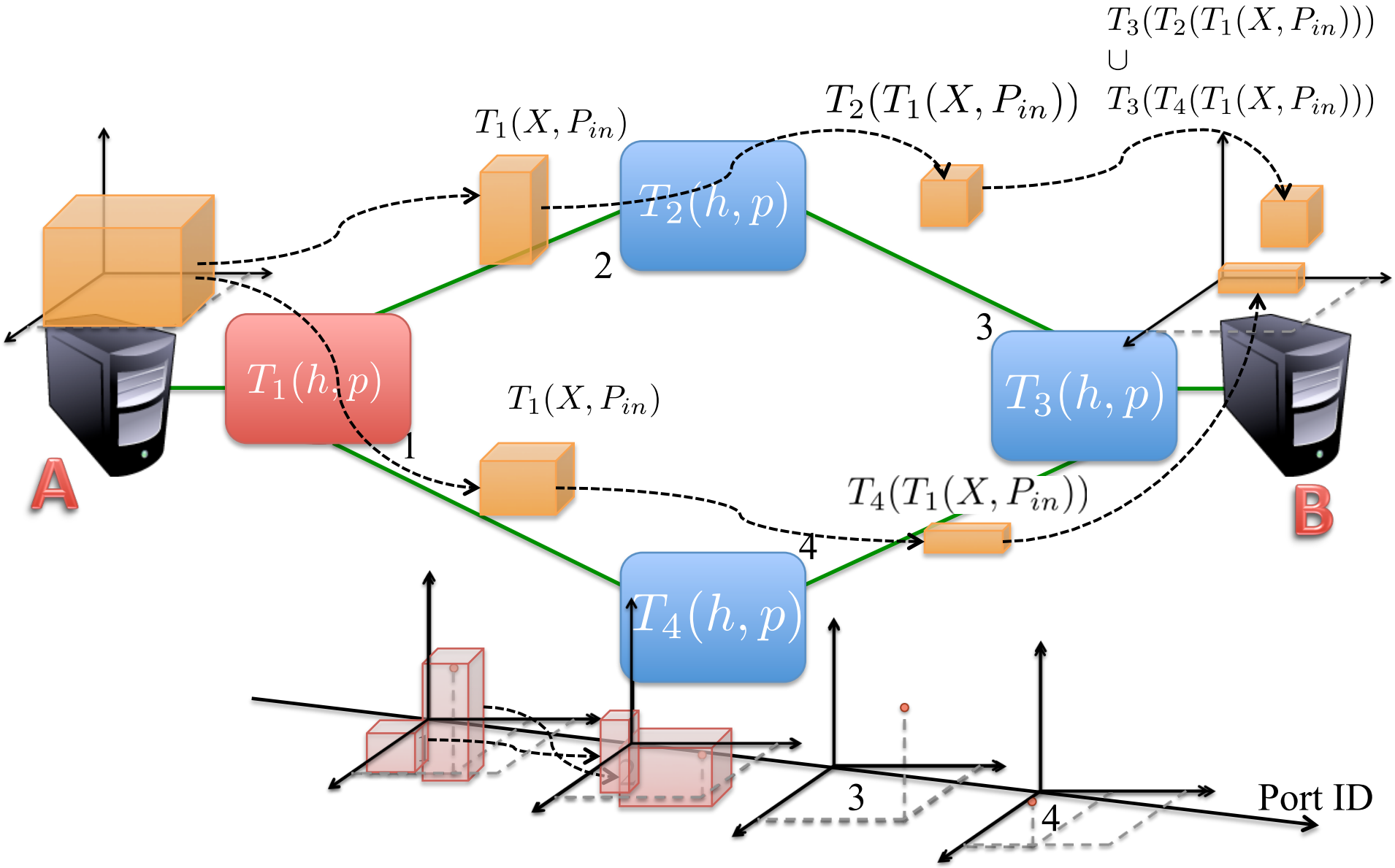
Header Space Analysis



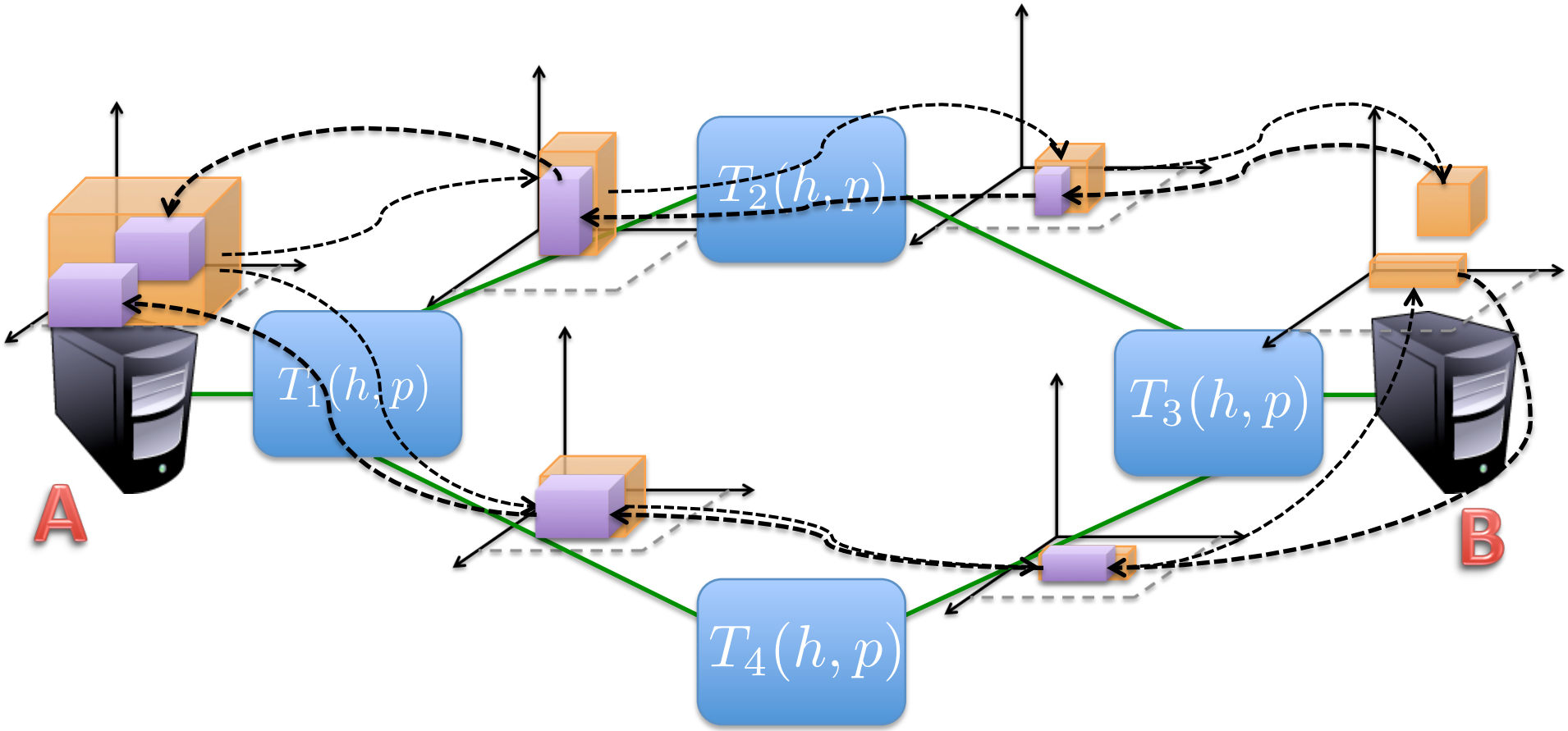
Header Space Analysis



Can A talk to B?



All packets from A that can reach B



Header Space Analysis

[Kazemian NSDI '12]

Consequences

- Abstract forwarding model; protocol independent
- Finds all packets from A that can reach B
- Find loops, regardless of protocol or layer
- Can prove that two groups are isolated

Can verify if network adheres to policy

HSA as a “foundation”

HAS enables many tools and methods

- Independent static checking
- In-line in-controller invariance checking
- Dynamic testing: Automatic test packet generation
- Dynamic testing: Automatic performance monitoring

Analogy to Boolean algebra for logic design

Software

Hassel tool

- Reads Cisco IOS Configuration
- Checks reachability, loops and isolation
- C: 60ms for Stanford Backbone
- Python: 10 mins for Stanford Backbone

Code

- http://bitbucket.org/peymank/hassel-public

Three of our projects

1. Static Checking [HSA]
   "Independently checking correctness"

2. Automatic Testing [ATPG]
   "Is the datapath behaving correctly?"

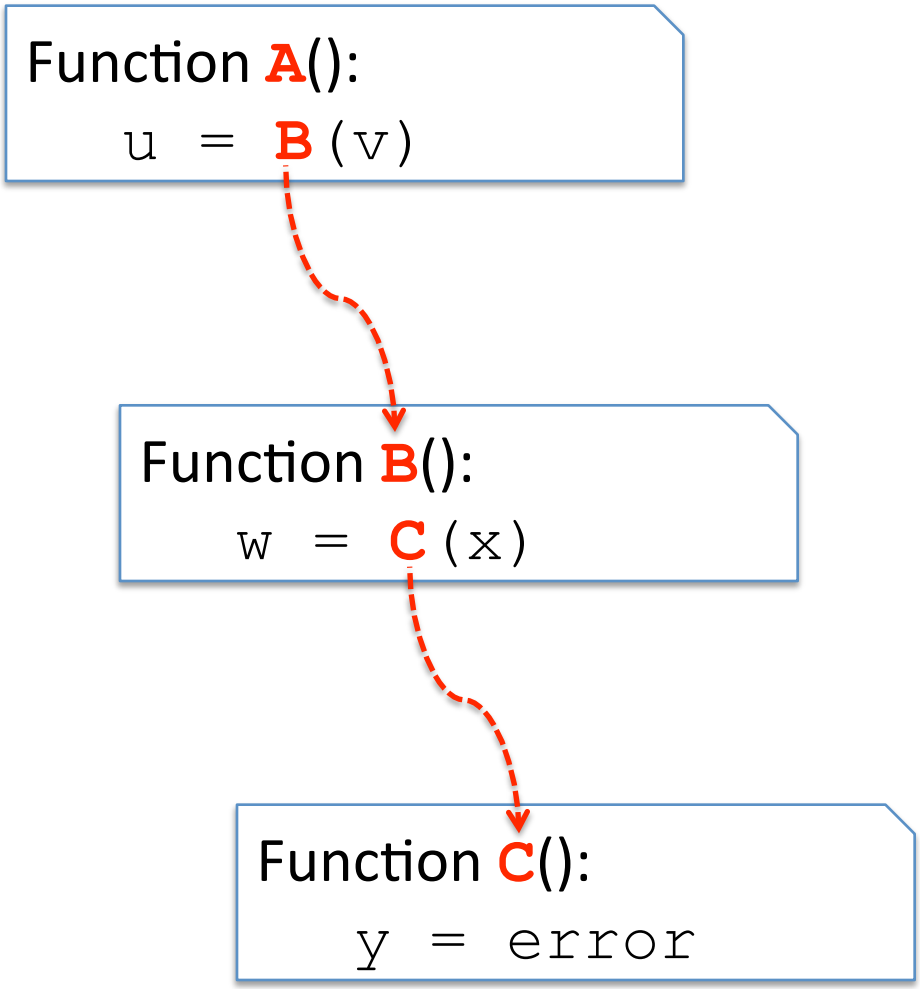
3. Interactive Debugging [ndb]
   "Finding bugs, and their root cause,
   in an operational network"

3. Interactive Debugging

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

Nikhil Handigol
Brandon Heller
Vimal Jeyakumar
David Mazières

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`

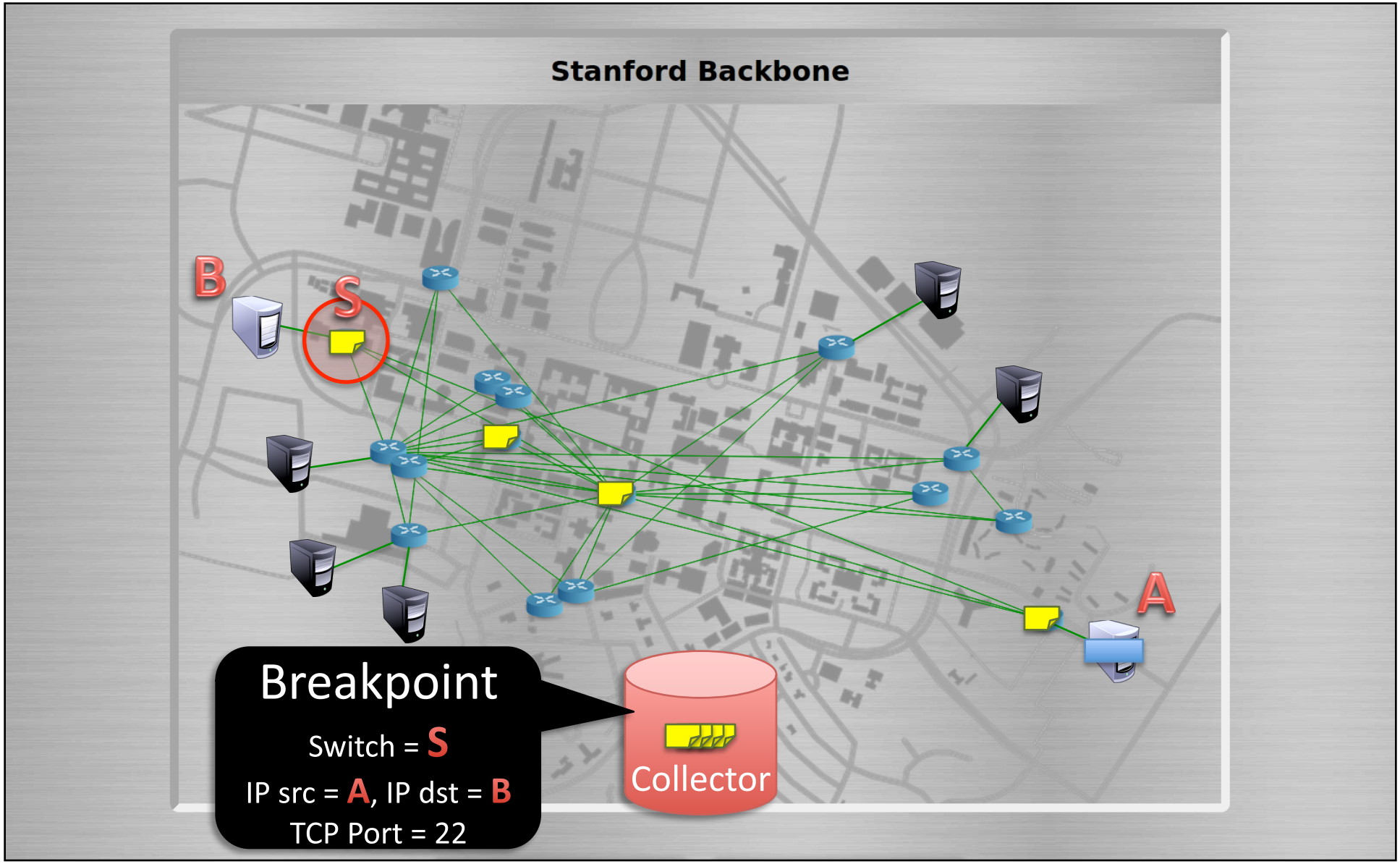
Problem

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

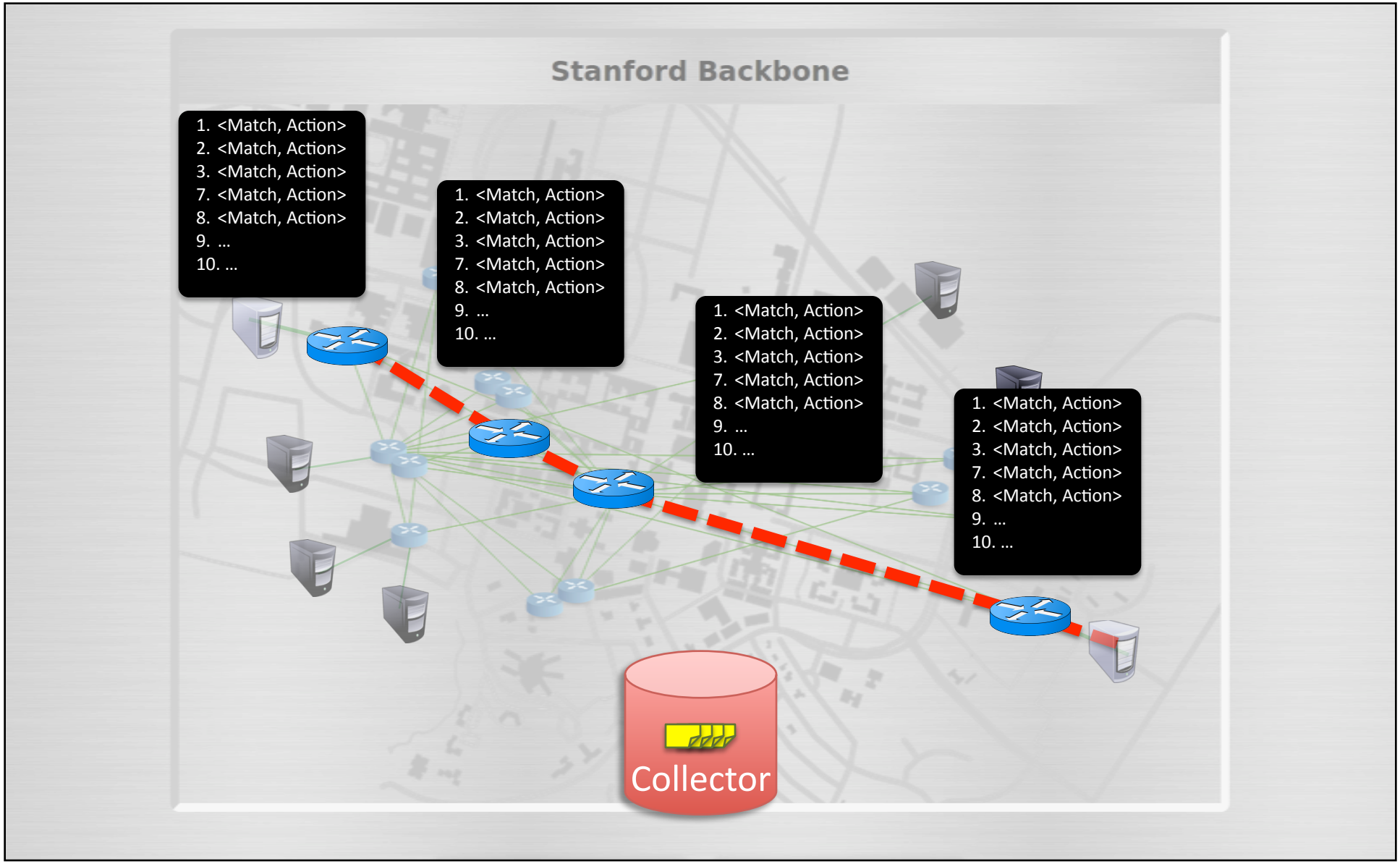
Goal

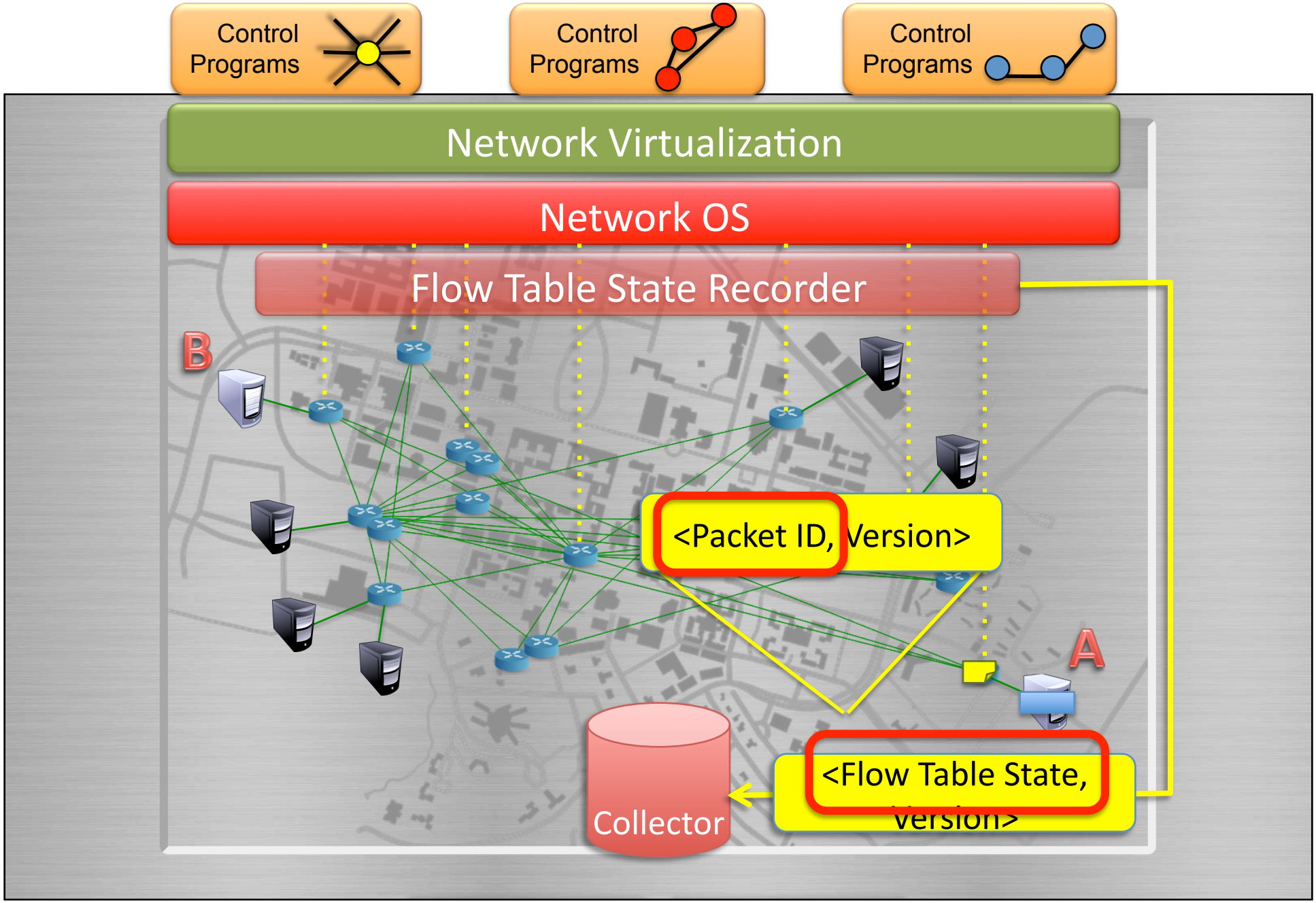
- Allow users to define a Network Breakpoint.
- Capture and reconstruct the sequence of events leading to the breakpoint.

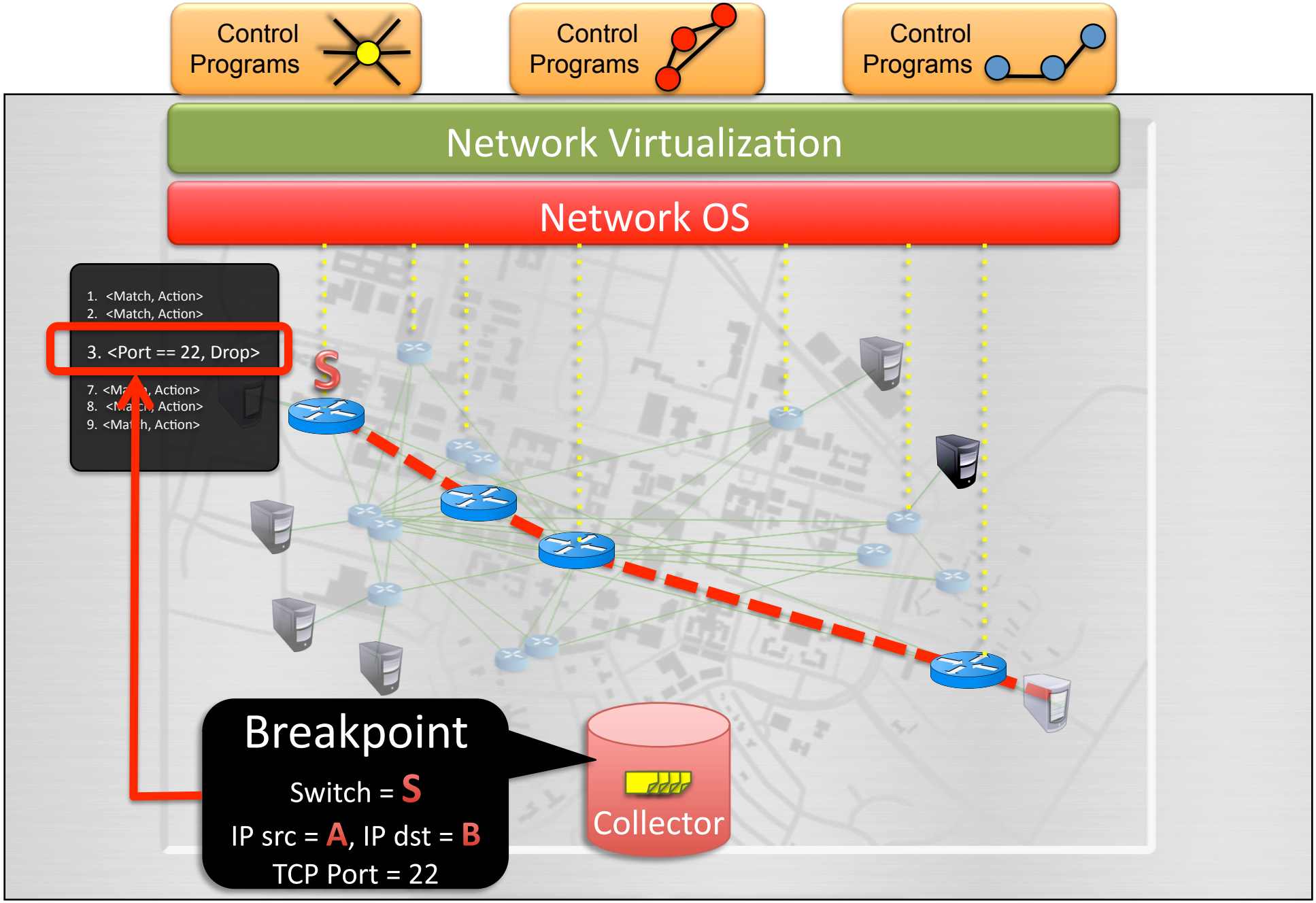
Network Debugger



Network Debugger




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


Control Programs
Control Programs
Control Programs
Network Virtualization
Network OS
1. <Match, Action>
2. <Match, Action>
3. <Port == 22, Drop>
7. <Match, Action>
8. <Match, Action>
9. <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

Status

First working prototype of `ndb`

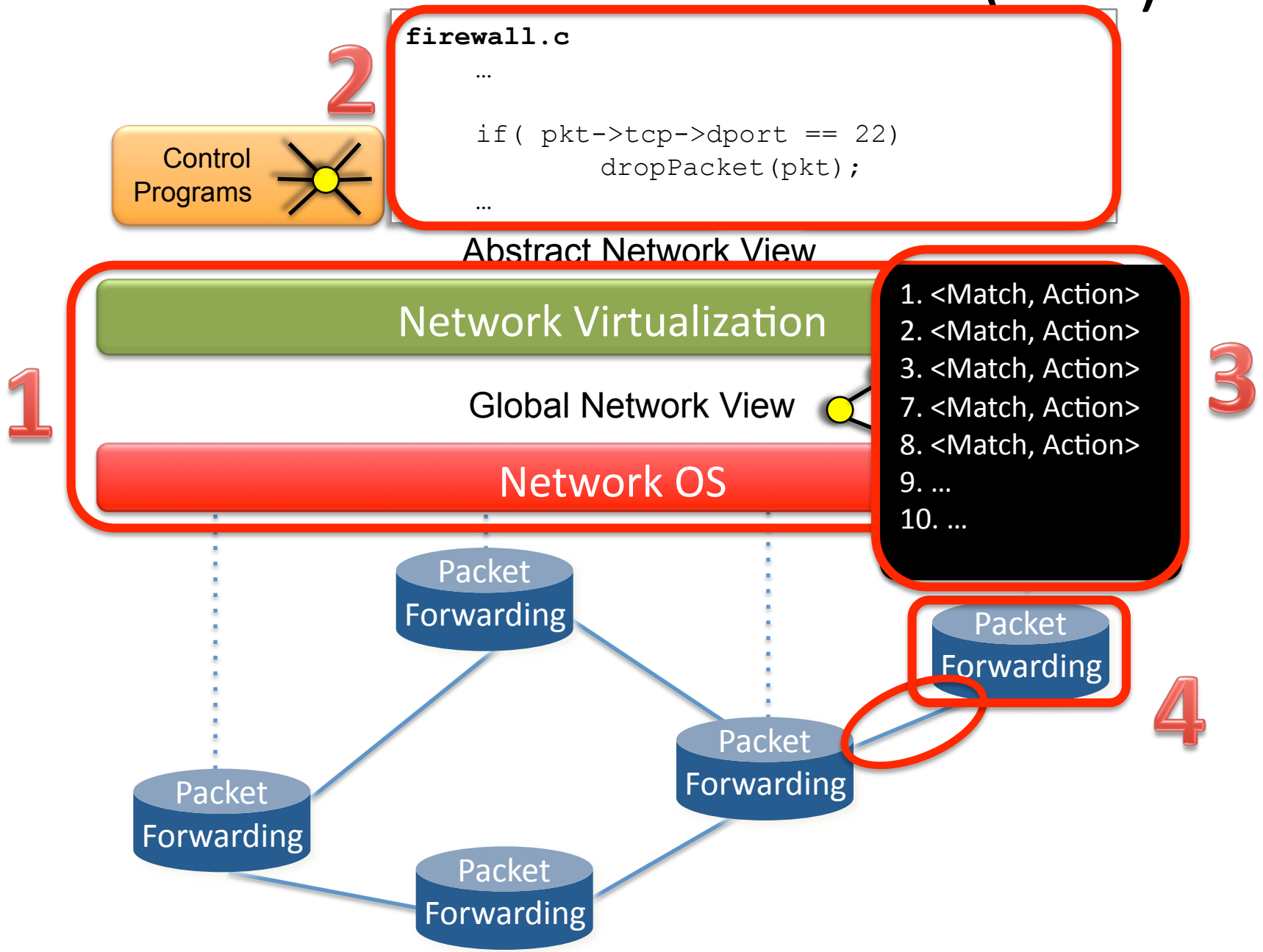
- Works without change to OpenFlow

Performance on Stanford backbone

- Collector could be just one server


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>
7. <Match, Action>
8. <Match, Action>
9. <Match, Action>
S
Breakpoint
Switch = S
IP src = A, IP dst = B
TCP Port = 22
Collector

Software Defined Network (SDN)



With SDN we will:

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

Software Defined Networks

- Allows a stronger intellectual foundation to networking
- Allows us to define the right abstractions
- Will allow us to transfer technology much faster, in both directions
- Is already closing the gap with industry

The End