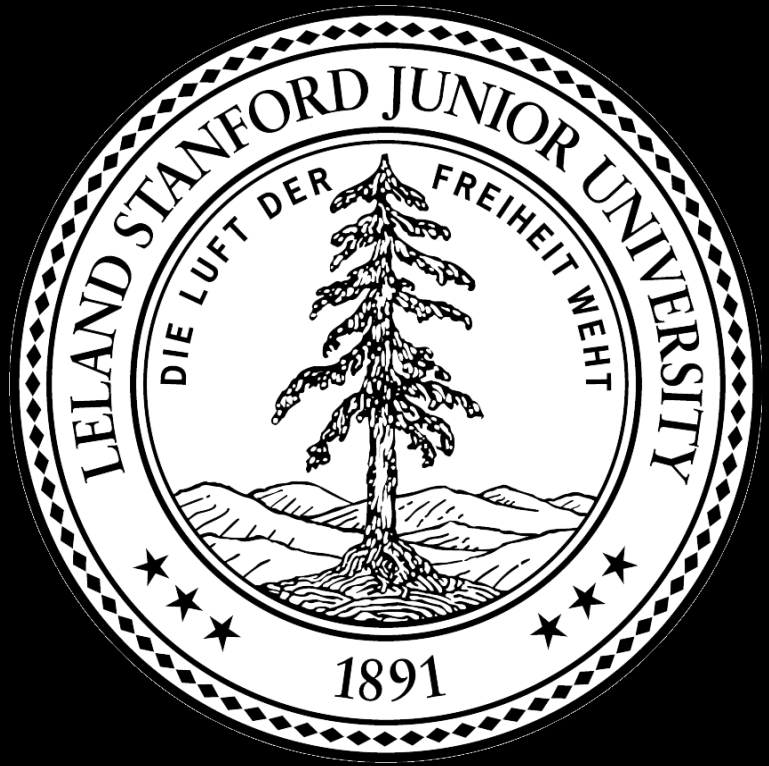
Welcome to CS244 Spring 2020!

Class will start shortly



CS244

Advanced Topics in Networking

Lecture 8: SDN (1)

Nick McKeown

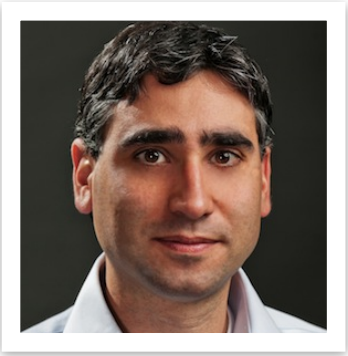
"Ethane: Taking Control of the Enterprise"
[Martin Casado et al, 2007]

"OpenFlow: Enabling Innovation in Campus Networks"
[A bunch of networking profs, 2008]

Spring 2020

Context

Martín Casado



- Previously at LLNL
- Stanford CS PhD 2007
- Founded Nicira
- GM for networking at VMware
- Now GP at A16Z
- Inventor: SDN
  and network virtualization (next Tuesday's class)

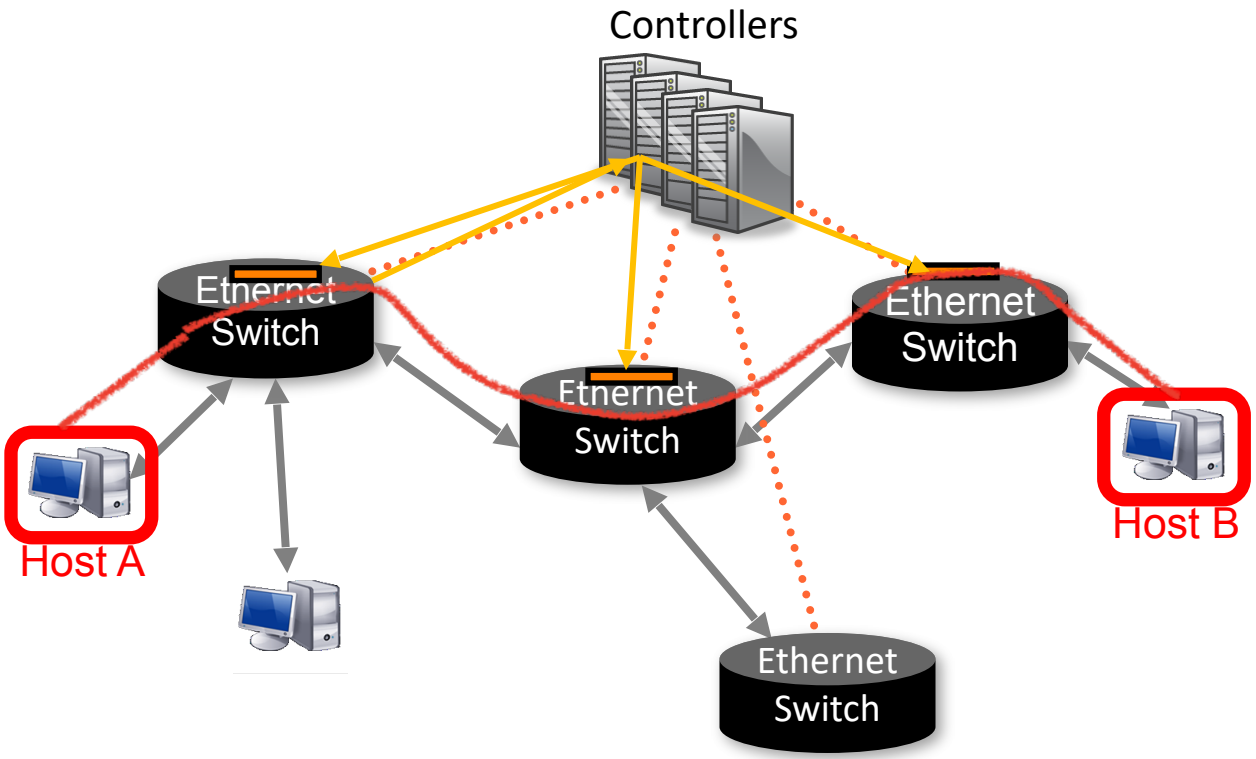
How difficult is it to define all network operations in software, outside the datapath?

2006

35,000 users
10,000 new flows/sec
137 network policies

2,000 switches
2,000 switch CPUs

Extreme thought experiment: What if software decides whether to accept each flow, and how to route it?



A question the team had:
How many $400 servers do we need
for 35,000 users?

Answer: less than one

If we can control the network centrally
then (eventually) we will.

With replication for
fault-tolerance and performance scaling.

Q: Why might we want to control them centrally?
Q: How does this compare to how networks are controlled today?

You said (about controllers)

Margalit Glasgow

…there was not much discussion of the controller, which seems important for security reasons such that a user can only control specified traffic.

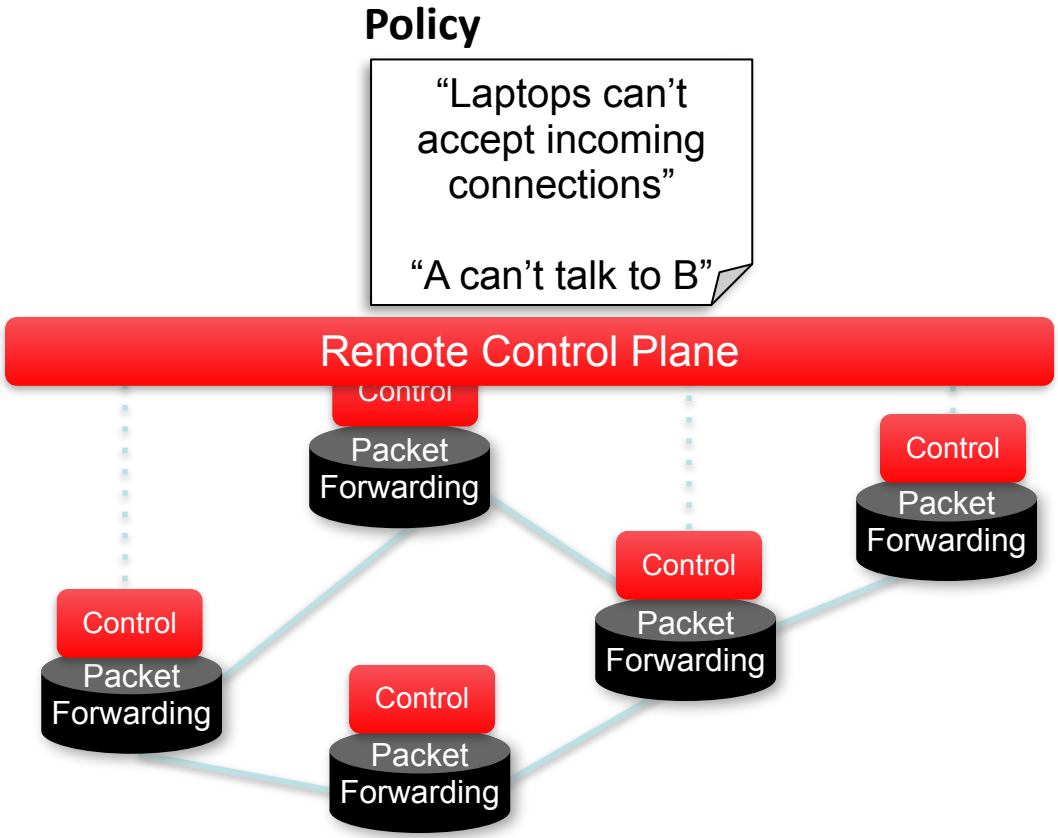
Sundararajan Renganathan

What is the time penalty paid by new flows (and therefore users) while they wait to get themselves registered at the controller and for the controller to subsequently update the flow tables at all the switches?

Wil Kautz

Aren't there major security issues with placing so much of the important decision-making power of the network in a single location? Doesn't a security vulnerability in that single location affect the entire network?

Ethane and Network Policy

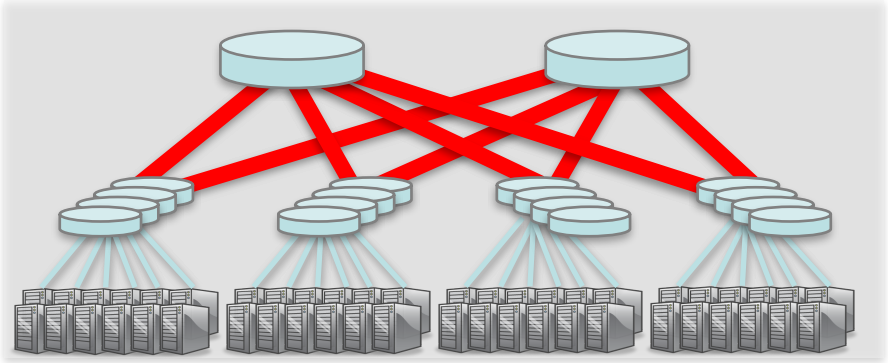


The approach was starting elsewhere…

1. Public WANs: Route reflectors decide routes centrally, and download to datapath
   - AT&T Backbone
2. WiFi: CAPWAP and Meraki; Ubiquiti
3. Cable TV: Docsis

4. Disaggregation: Datacenter owners were considering build their own networking equipment.

Example: Big Data Center



Cost

500,000 servers

25,000 switches

$10k per legacy switch = $250M

$2k disaggregated switch = $50M

Savings in 5 data centers = $1Bn

Control

Centralized remote control is easier

"Centralize if you can, distribute if you can't"

Customized, differentiated network

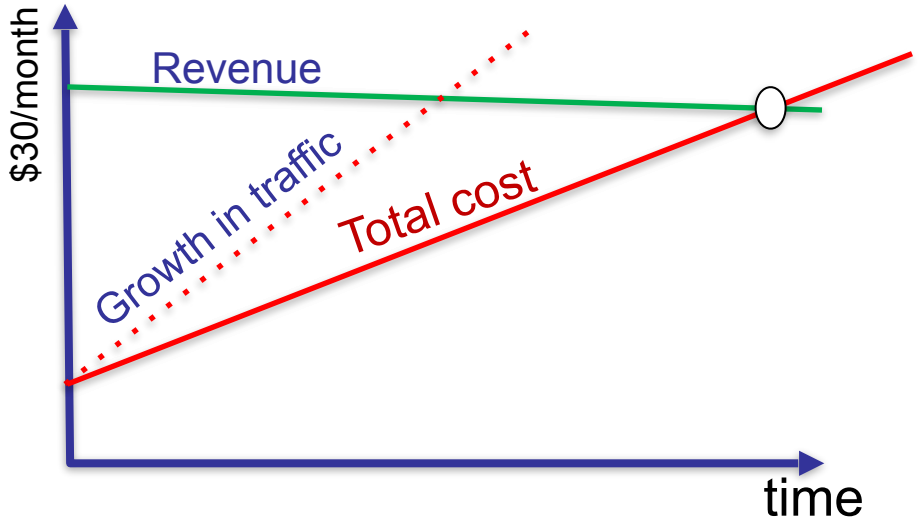
Home grown traffic engineering

50% utilization → 95% utilization

By 2008, Google and Amazon were starting to write their own software

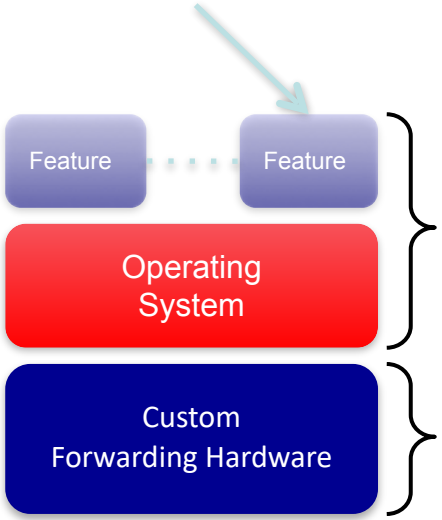
Internet Service Providers (ISPs)

- Global IP traffic growing 40-50% per year
- End-customer monthly bill unchanged
- Therefore, CAPEX and OPEX need to reduce 40-50% per Gb/s per year
- But in practice, reduces by ~20% per year



What a big Internet router looked like

Routing, management, mobility management, access control, VPNs, …



Million of lines of source code

7,000 Internet RFCs

Billions of gates

Bloated

Power Hungry

- Overly complex
- Mainframe mentality
- Too expensive

After Ethane: What was next?

Microsoft: “Come on in….”

Cisco: “It will never work…”

Raw nerve.

We must be onto something.

“The Future of Networking and the Past of Protocols”

Scott Shenker 2011

Networks today are run by

"Masters of Complexity"

Abstractions in computer systems

Virtual memory: Abstract illusion of infinite, private physical memory

File system: Uniform illusion of read/write data store.

Operating system: Shields user from CPU scheduling and peripheral sharing.

…

“Modularity based on abstraction is the way things are done!”

Barbara Liskov (MIT)
Turing Award Lecture 2009

SDN: An early definition

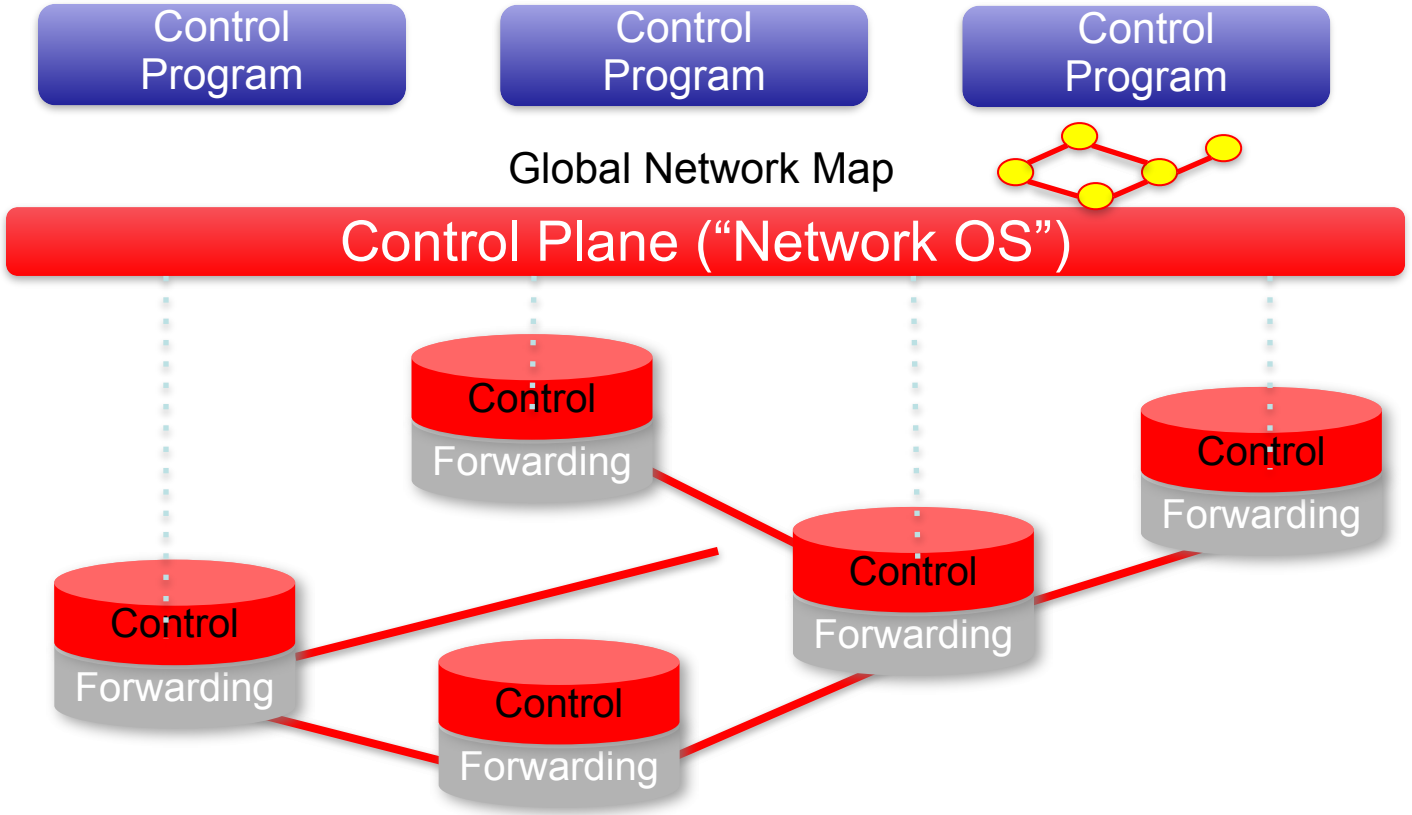
A network in which the control plane is physically separate from the forwarding plane.

and

A single control plane controls several forwarding devices.

(Evolved over time)

Software Defined Network (SDN)



OpenFlow

Motivation for OpenFlow

“Thus, the commercial solutions are too closed and inflexible, and the research solutions either have insufficient performance or fanout, or are too expensive. It seems unlikely that the research solutions, with their complete generality, can overcome their performance or cost limitations. A more promising approach is to compromise on generality and to seek a degree of switch flexibility that is:

1. Amenable to high-performance and low-cost implementations.
2. Capable of supporting a broad range of research.
3. Assured to isolate experimental traffic from production traffic.
4. Consistent with vendors’ need for closed platforms."

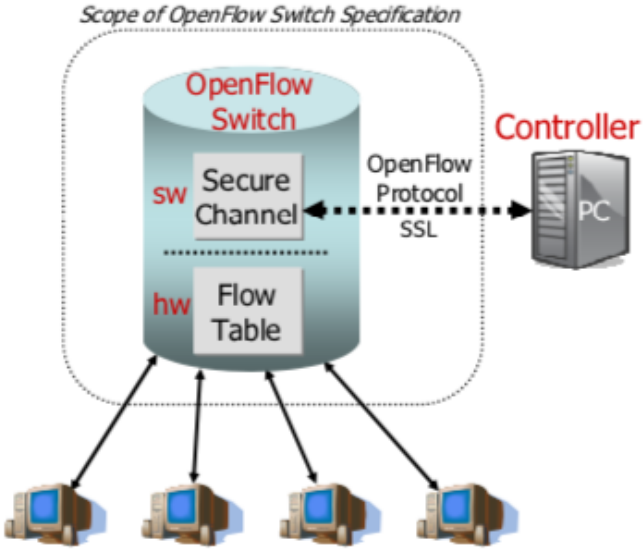


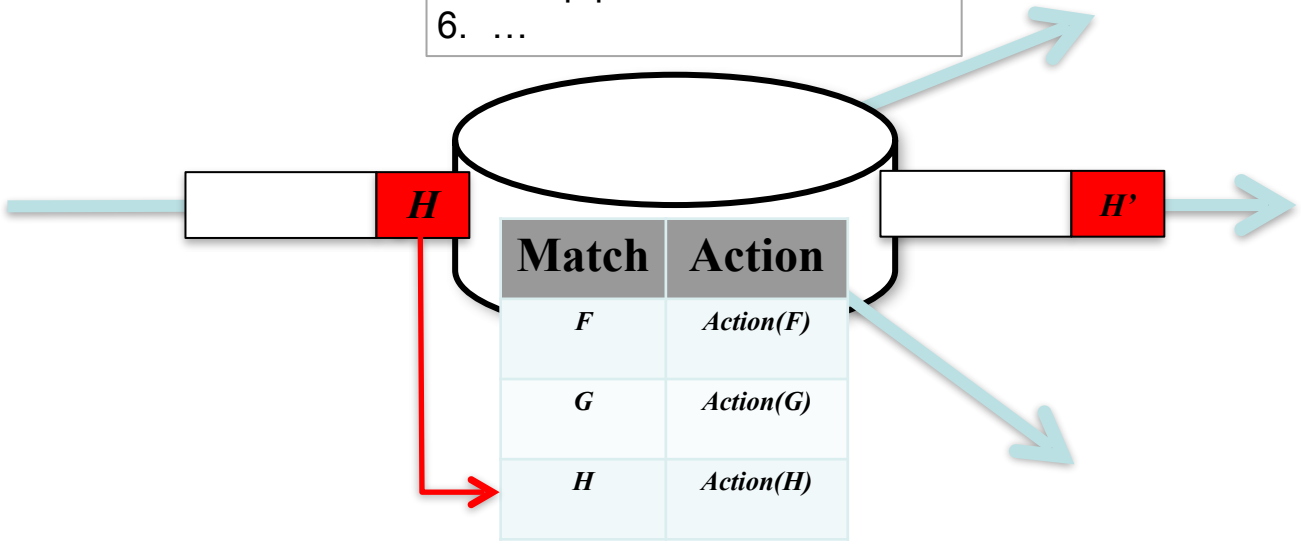
Figure 1: Idealized OpenFlow Switch. The Flow Table is controlled by a remote controller via the Secure Channel.

Match-Action Forwarding Abstraction

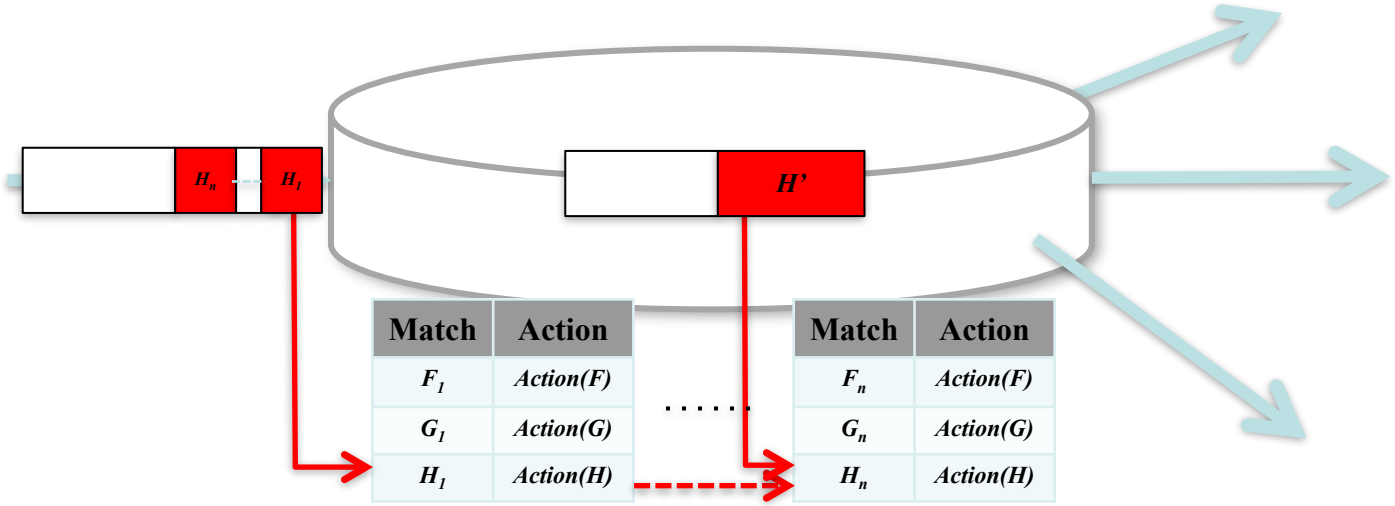
Action Primitives

1. “Forward to ports 4 & 5”
2. “Push header Y after bit 12”
3. “Pop header bits 8-12”
4. “Decrement bits 13-18”
5. “Drop packet”
6. …

“Plumbing primitives”



Multiple Table Match-Action



OpenFlow Goals

(as described at the time)

Short-term, backward compatability

Match: include well-known header fields.

Action: necessary set for existing protocols.

- Support existing protocols on existing switch chips.

Q: How well was each goal met?

Long-term

Match: Very general, not protocol specific.

Action: Small instruction set, not protocol specific.

- Make it easy to add new headers and actions.
- Any network (packet, circuit, radio).

You said

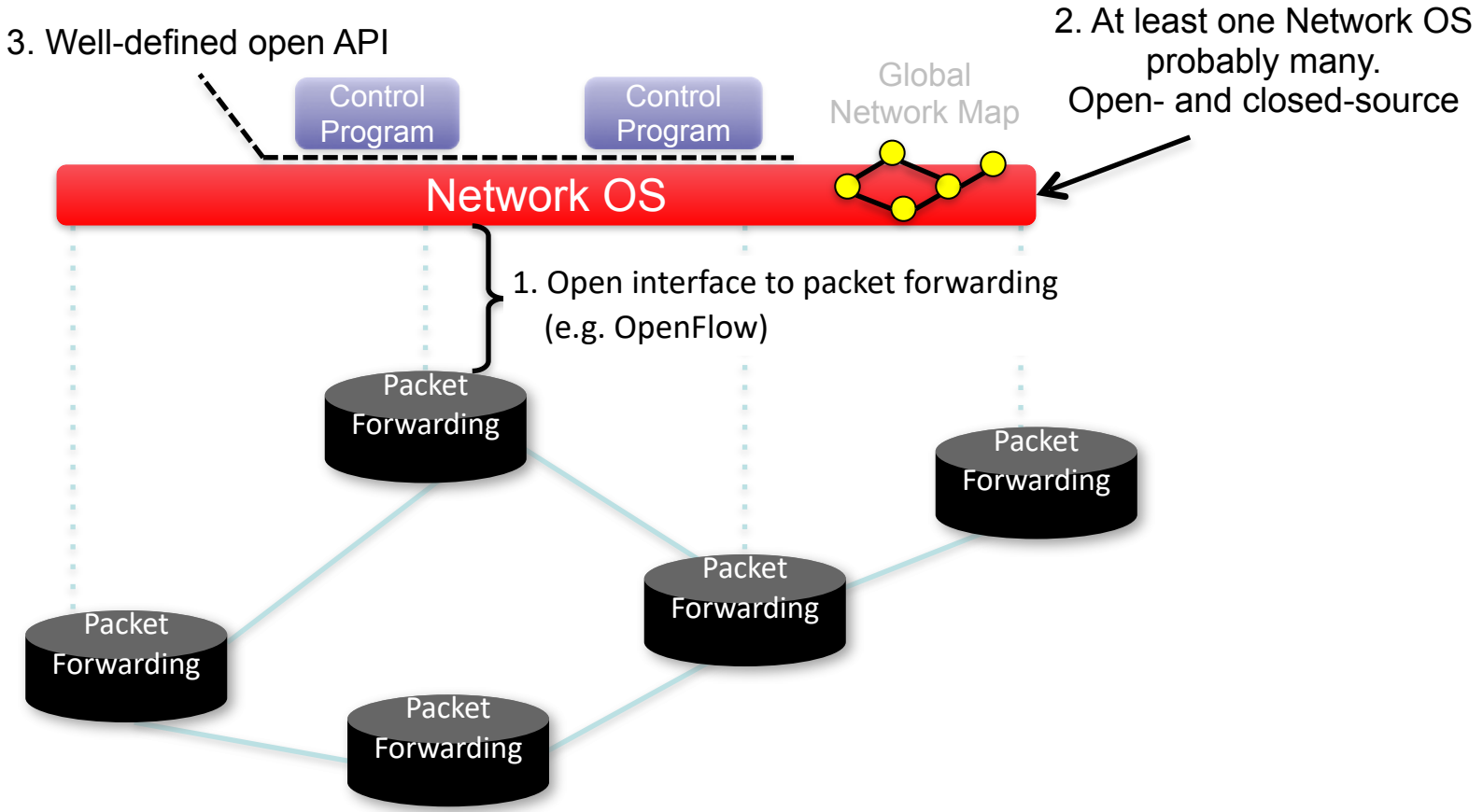
Kathryn Rydberg

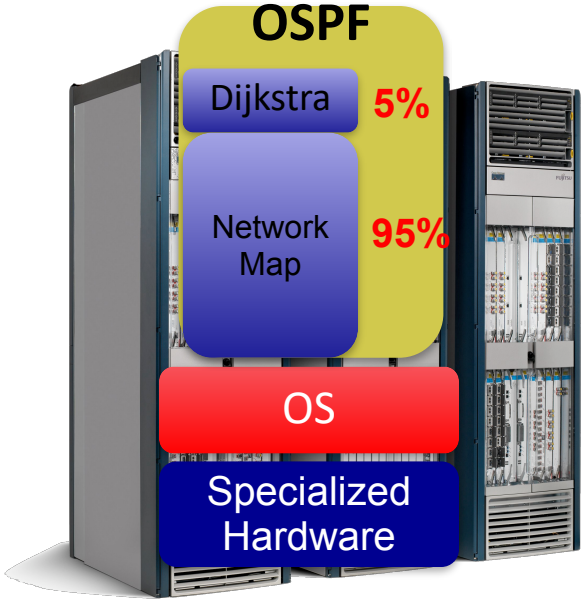
How did the experiment of using OpenFlow in a few Stanford buildings go? Does more of the Stanford network now use OpenFlow? Is it prevalent at other universities now? Does OpenFlow affect the performance of non-research traffic?

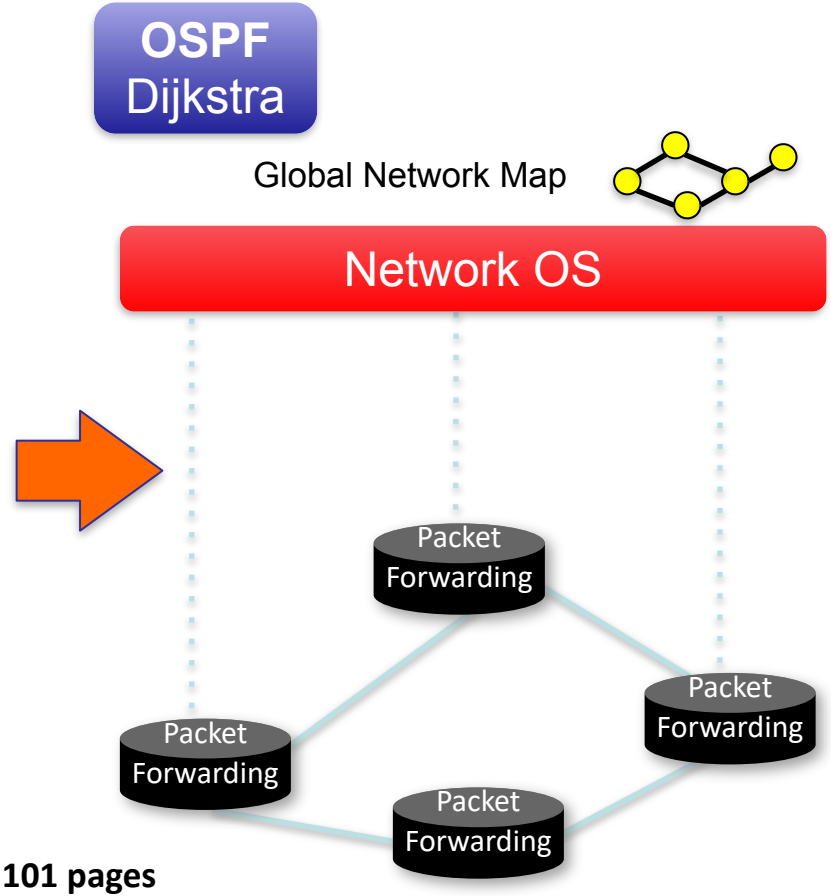
OpenFlow: Control Abstraction

1. Control plane can run on modern servers
2. Can adopt software engineering best-practices
3. Easier to add new control programs
4. …or customize locally
5. Solve distributed systems problem once, rather than for every protocol

SDN: Software Defined Networks







RFC 2328: **245 pages**

Distributed System

Builds consistent, up-to-date map of the network: **101 pages**

Dijkstra's Algorithm: **1 page**

OpenFlow: Forwarding Abstraction

1. Vendor-agnostic interface to forwarding plane
2. Simpler, lower-cost, lower-power hardware

Match + Action abstraction

Pros

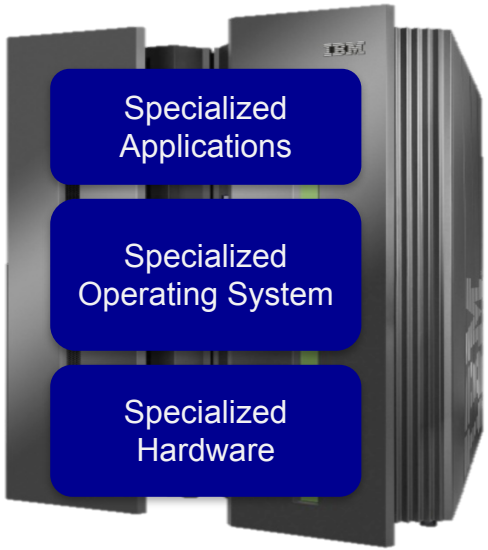
- Simple abstraction of stateless forwarding
  (e.g. Ethernet, IPv4, IPv6, VLAN, VPNs, …)
- Add/delete table entries: If a packet matches a field, then perform actions.
- Allows one API to control multiple protocols
- Enabled multiple controllers: NOX, POX, ONIX, Beacon, Floodlight, …
- Easy to add to existing switches or new disaggregated switches
  (hence Google adoption)

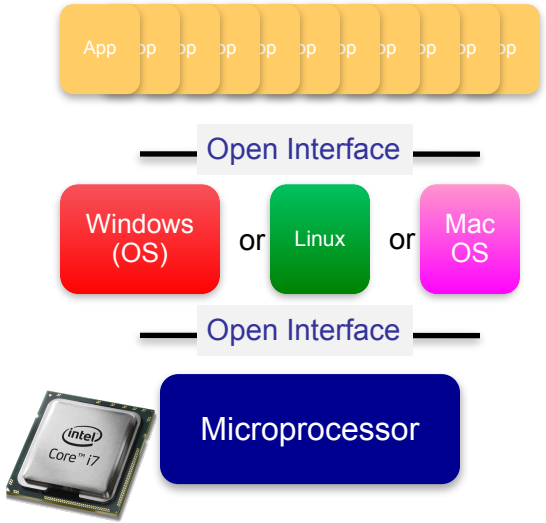
Cons

- Underlying functions were fixed, hard to add or evolve (hence P4 later)
- Hard to introduce new versions of API
- Switch vendors very reluctant to support

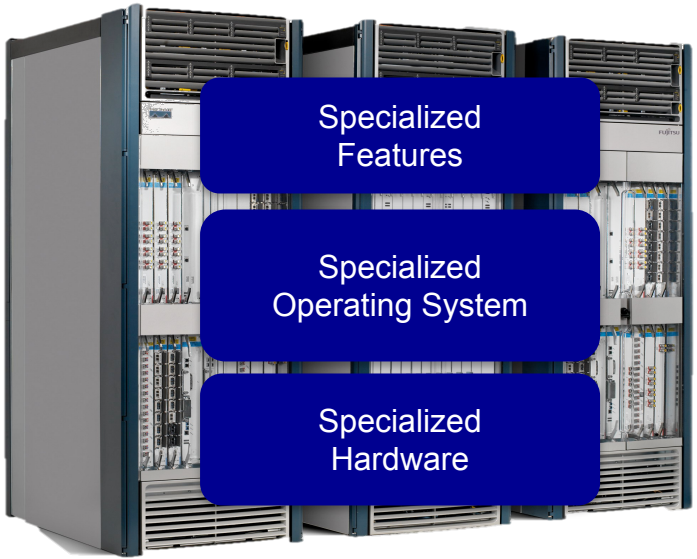
In the context of bigger networking industry changes

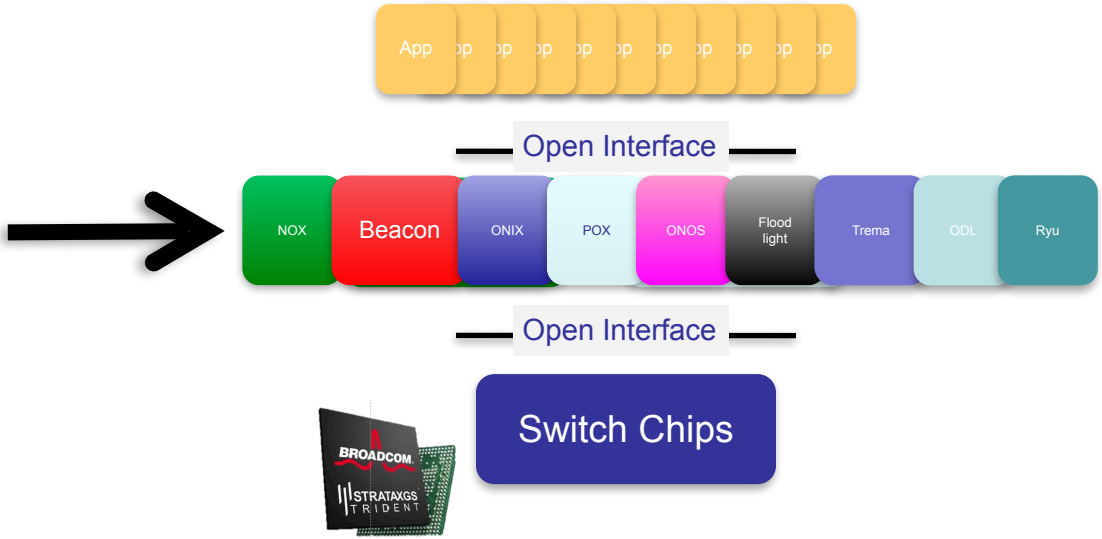
Computer Industry





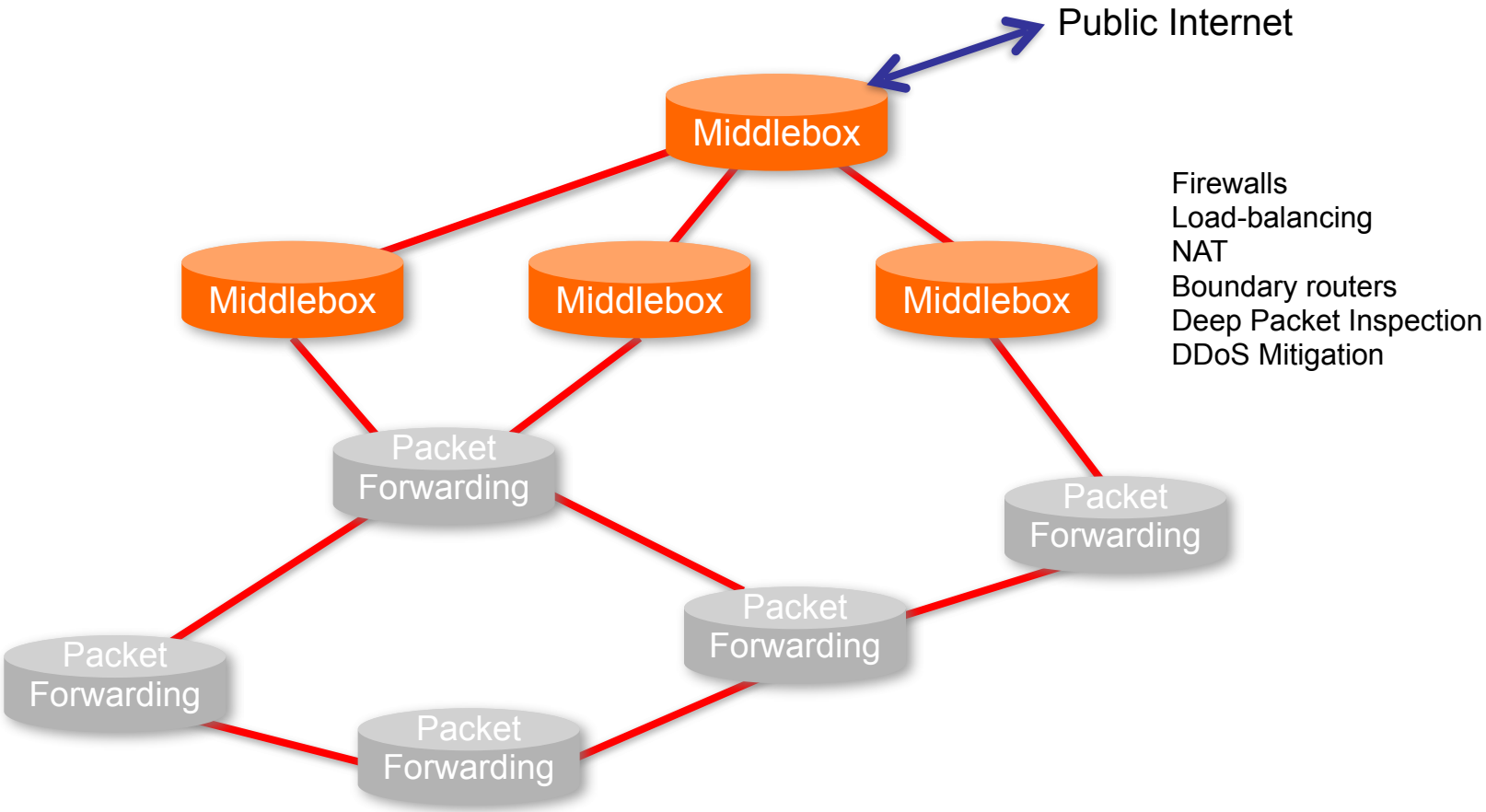
Networking Industry



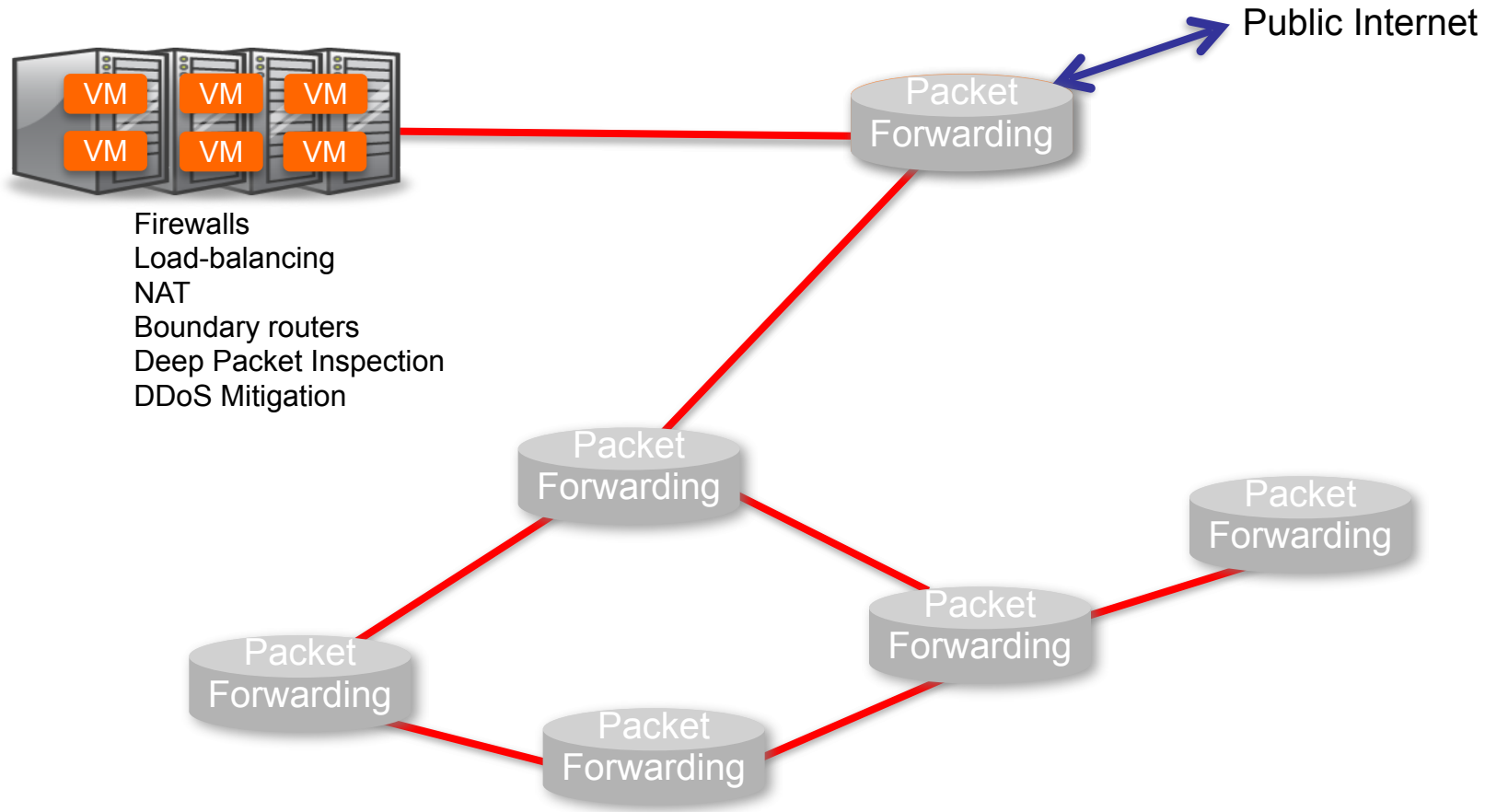


"Software is eating the world (of networking)"

Network Function Virtualization (NFV)



Network Function Virtualization (NFV)



With hindsight, Disaggregation, SDN and NFV were probably inevitable

Part of a bigger trend towards the owners and operators of networks taking control of how they work

Inevitable because…

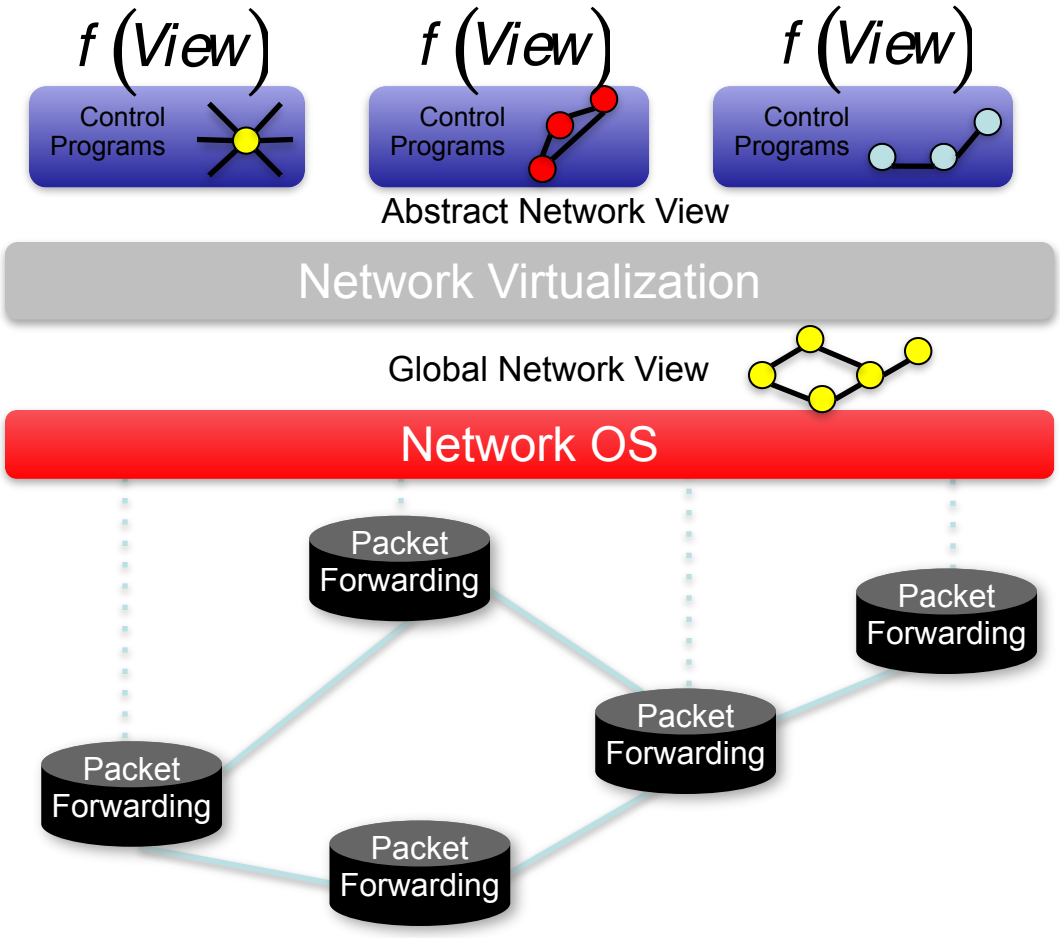
1. Rise of Linux.
2. Rise of baremetal servers and data centers.
3. SDN: Rise of merchant switching silicon.
4. NFV: Rise of computer virtualization.

Today

Most networking equipment is disaggregating

- Intra- and inter-datacenter networks
- ISP routers and switches
- WiFi APs
- Cellular basestations (4G, 5G…)
- Optical and Metro Transport
- Residential broadband access
- Enterprise network equipment: switch, router, firewall

Network Virtualization (next time)



End.