Welcome to

Networking Technologies for Cloud Computing

USTC-CYSC6402P

Instructor: Chi Zhang

Fall 2020

Today's agenda

- SDN & NFV
  - What is SDN? Why SDN?
  - History of SDN
  - SDN standardization
  - SDN deployment models
  - OpenFlow
  - SDN applications
  - SDN development tools
  - Future directions

Today's agenda

- SDN & NFV
  - What is SDN? Why SDN?
  - History of SDN
  - SDN standardization
  - SDN deployment models
  - OpenFlow
  - SDN applications
  - SDN development tools
  - Future directions

SDN market


BY THE NUMBERS:
Software-defined Networking
$3.7b
Total value of the SDN market, including associated network equipment, in 2016, up from being a negligible market in 2012 †
9
Number of the top 10 network security vendors that will support OpenFlow by 2014 ††
85%
Share of 315 companies surveyed who are currently researching SDN ‡
$30m
Potential savings large data centers can achieve with SDN, according to network virtualization company Nicira ‡‡
80%
Share of service providers who are including OpenFlow in their purchase considerations, based on interviews with 21 purchase-decision makers *
SOURCES
† Rohit Mehra, IDC
†† Neil MacDonald, "The Impact of Software-Defined Data Centers on Information Security," Gartner, October 16, 2012
‡ Zeus Kerravala, "SDNs provide a solution for Extreme Networks' XOS," Network World, November 14, 2012
‡‡ Julie Bort, "The Least Stealthy Startup In The Valley Has Officially Launched," Business Insider, February 6, 2012
* Jim Duffy, "Many Cisco carrier customers planning SDNs," NetworkWorld, July 26, 2012
©2012 ITworld

SDN Papers

ACM SIGCOMM 2007

Ethane: Taking Control of the Enterprise

Martìn Casado, Michael J. Freedman,
Justin Pettit, Jianying Luo,
and Nick McKeown
Stanford University

Scott Shenker
U.C. Berkeley and ICSI

ABSTRACT

This paper presents Ethane, a new netwo
enterprise. Ethane allows managers to d
wide fine-grain policy, and then enforces
ples extremely simple flow-based Ethernet
ized controller that manages the admittanc

ACM SIGCOMM CCR 2008

OpenFlow: Enabling Innovation in Campus Networks

Nick McKeown
Stanford University

Tom Anderson
University of Washington

Hari Balakrishnan
MIT

Guru Parulkar
Stanford University

Larry Peterson
Princeton University

Jennifer Rexford
Princeton University

Scott Shenker
University of California, Berkeley

Jonathan Turner
Washington University in St. Louis

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
Authors take full responsibility for this article's technical content.
Comments can be posted through CCR Online.

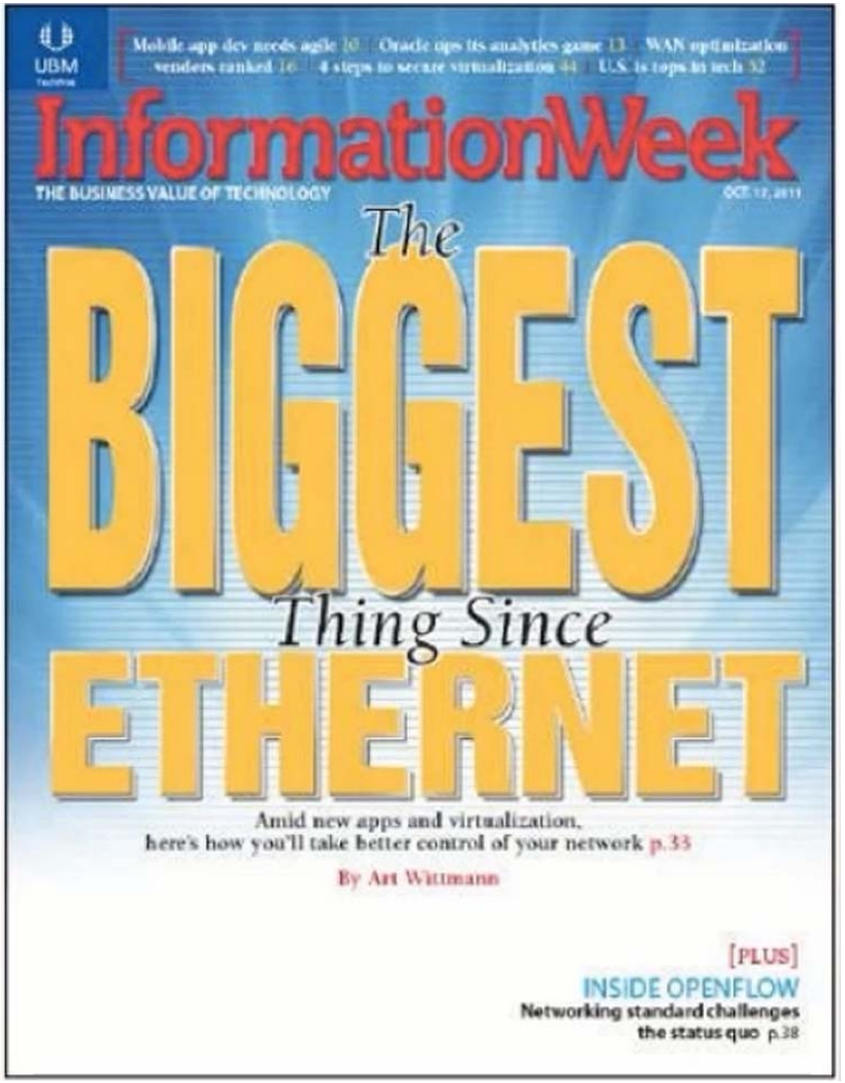
ABSTRACT

This whitepaper proposes OpenFlow: a way for researchers to run experimental protocols in the networks they use every day. OpenFlow is based on an Ethernet switch, with an internal flow-table, and a standardized interface to add and remove flow entries. Our goal is to encourage networking vendors to add OpenFlow to their switch products for deployment in college campus backbones and wiring closets.

to experiment with production traffic, which have created an exceedingly high barrier to entry for new ideas. Today, there is almost no practical way to experiment with new network protocols (e.g., new routing protocols, or alternatives to IP) in sufficiently realistic settings (e.g., at scale carrying real traffic) to gain the confidence needed for their widespread deployment. The result is that most new ideas from the networking research community go untried and untested; hence the commonly held belief that the network infrastructure has

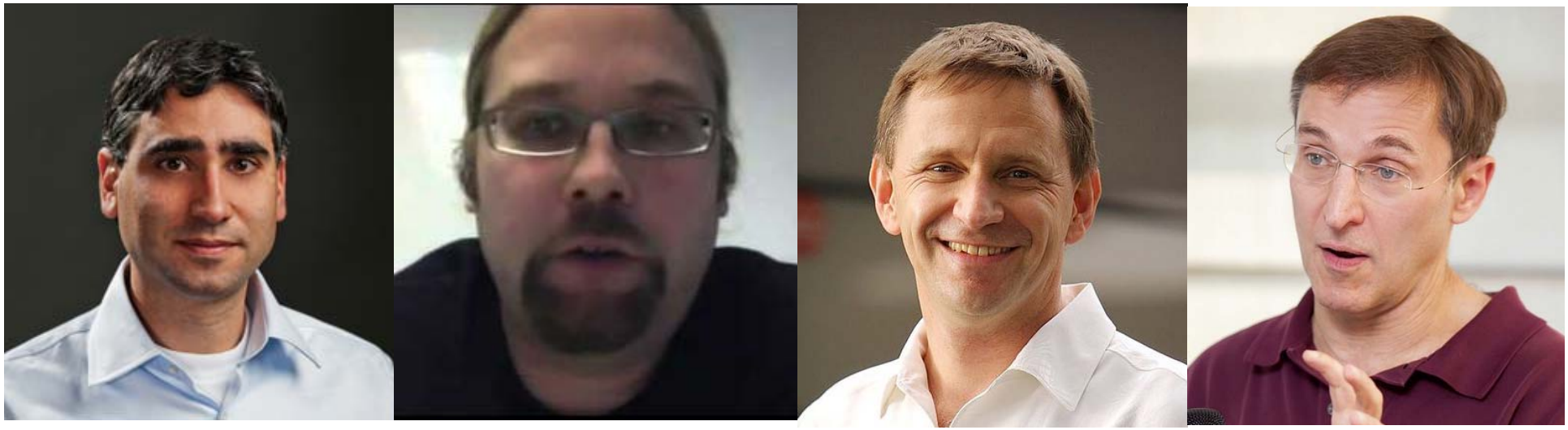
SDN: Future of Networking

MIT Technology Review 2009



InformationWeek 2011

Founders of SDN and Nicira



Martin Casado Teemu Koponen Nick Mckeown Scott Shenker

- Nicira is a company in SDN and network virtualization. It was founded in 2007 by Martin Casado, Nick McKeown and Scott Shenker.
- On July 23, 2012 VMware announced they intended to acquire Nicira for $1.26 billion.

My personal experience

Nick McKeown

2010

- **"How should the Internet evolve?"**
  Kailath Symposium, November 2010, Stanford University. ppt
  Load Balancing Video(mp4) OpenFlow Wireless Video(mp4)

2009

- **"Software Defined Networks"**
  FCC, October 2009, Washington DC. ppt
- **"Virtualization and OpenFlow"**
  Sigcomm VISA Workshop, August 2009, Barcelona
  ppt
- **"Software-defined Mobile Networks"**
  MobiHoc Keynote Talk, May 2009, New Orleans
  ppt
- **"Software-defined Networking"**
  Infocom Keynote Talk, April 2009, Rio de Janeiro, Brazil
  ppt, pdf

ALL CONFERENCE PANEL (Panel 1):
Tuesday, April 21, 11:00AM - 12:30PM

Clean Slate Architectures: Where Are We Today, And What Is The Path Forward?

Panelists:

Christophe Diot (Thomson Paris Research Center, France)
Joe Touch (Postel Center - Information Sciences Institute, US)
John Silvester (University of Southern California, US)
Nick McKeown (Stanford University, US)

Panel 3:
Thursday, April 23, 11:30AM - 1:00 PM

What Are The Hot Topics in Networking?

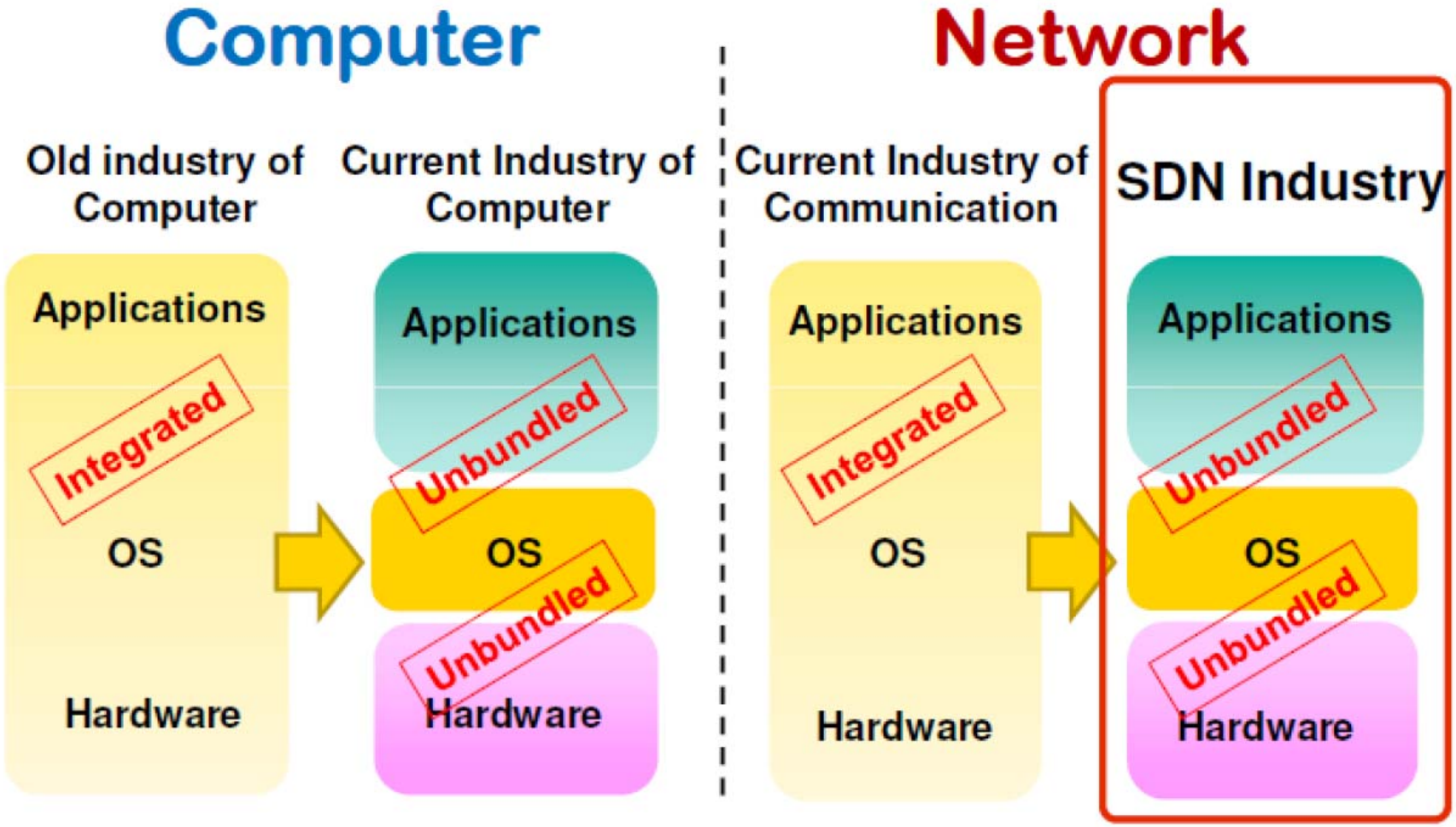
Panelists:

Keith Ross (Polytechnic Institute of NYU, US)
Ness B. Shroff (The Ohio State University, US)
Sajal Das (National Science Foundation and University Texas at Arlington, US)
Don Towsley (University of Massachusetts - Amherst, US)

What is SDN?

- Do you think SDN is
  - a *Protocol* (or a set of protocols) ?
  - an *Architecture* ?
  - a *Style* (meta-architecture) ?
  - an *Approach* expanding the design space?
  - or else
- A Swedish company www.tail-f.com did a survey and found out that
  - 87% of NA enterprises see SDN as more important than cloud or virtualization
  - yet only 51% know what SDN is!

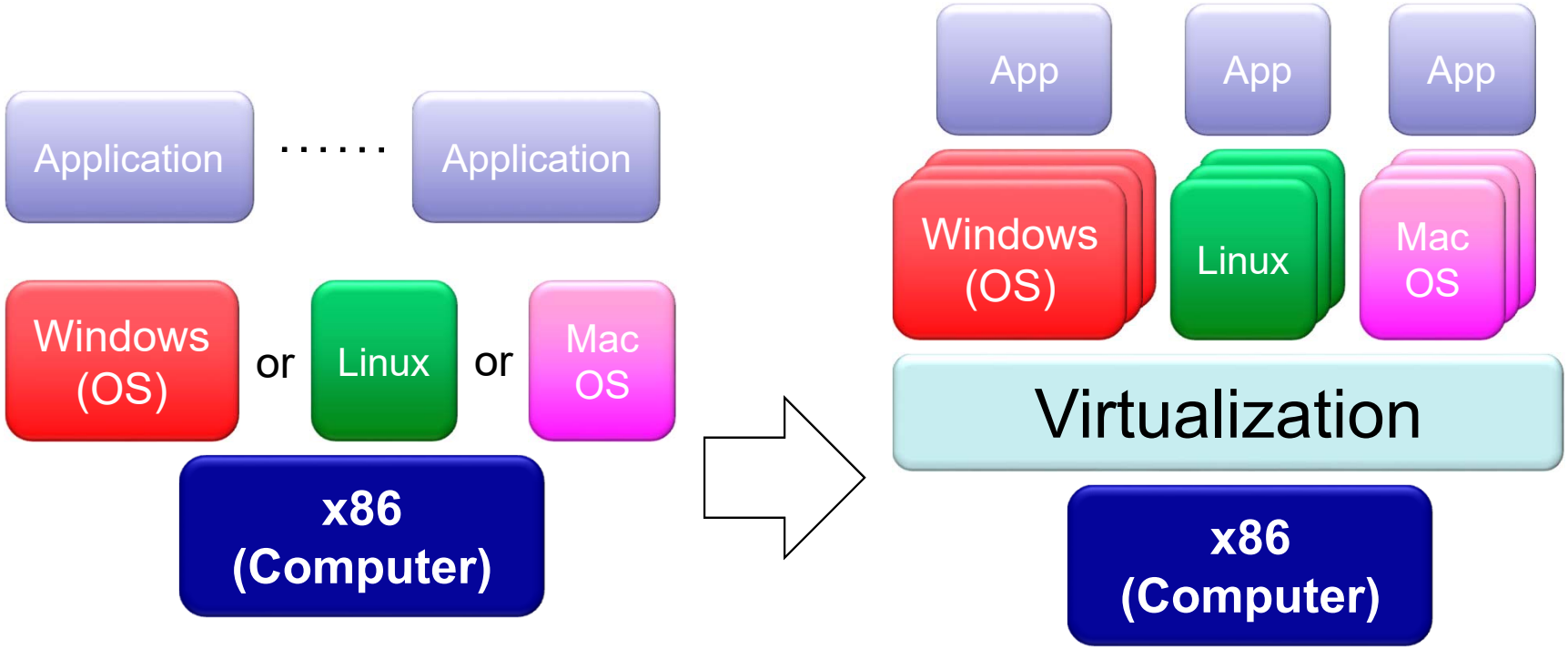
Past, current, and future of ICT



Innovation: computers vs. networks

- How difficult is it to create/modify a computer application?
- How difficult is it to create/modify a network feature?
- What is the difference?
- What are the tools available for each?

Innovation in computers



Simple, common, stable, hardware substrate below

+ Programmability
+ Strong isolation model
+ Competition above

→ Innovation in infrastructure

We have lost our way in networking

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

Feature | Feature

OS

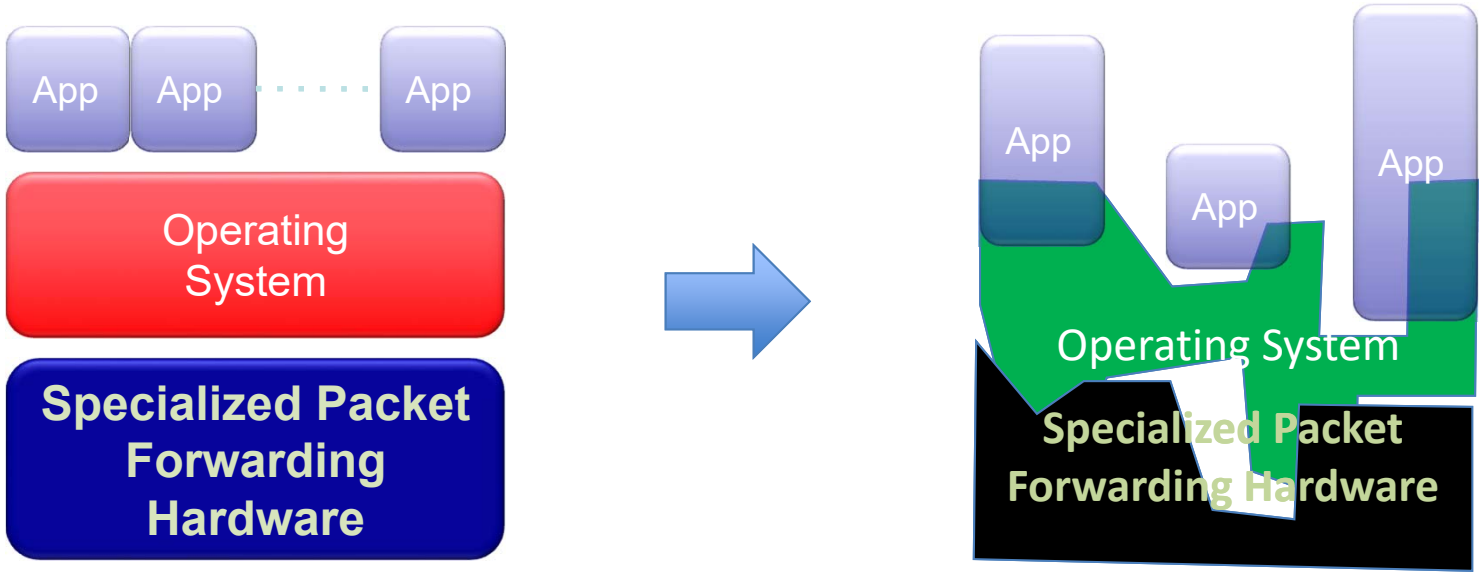
Custom Hardware

Million of lines of source code | 6,000 RFCs | Barrier to entry

Billions of gates | Bloated | Power Hungry

- Vertically integrated, complex, closed, proprietary
- Networking industry with “mainframe” mind-set

Reality is even worse



- Lack of competition means glacial innovation
- Closed architecture means blurry, closed interfaces

Limitations of current networks

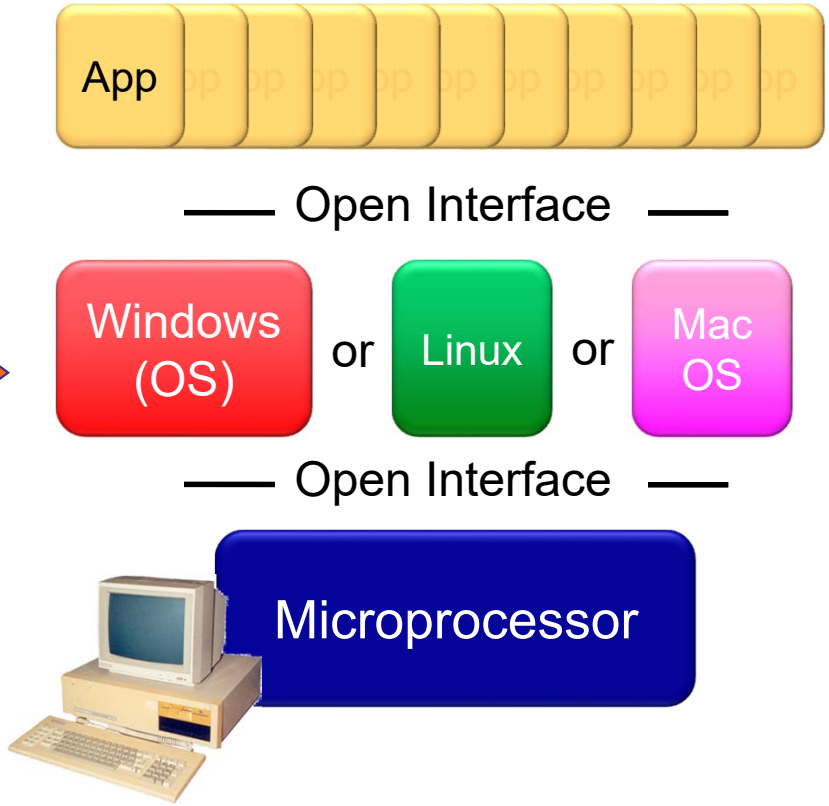
- Enterprise networks are difficult to manage
  - How to easily configure huge networks?
  - No control plane abstraction for the whole network!
- New control requirements have arisen
  - Greater scale
  - Migration of VMS
- It's like old times – when there was no OS…

Wilkes with the EDSAC, 1949

Call for change

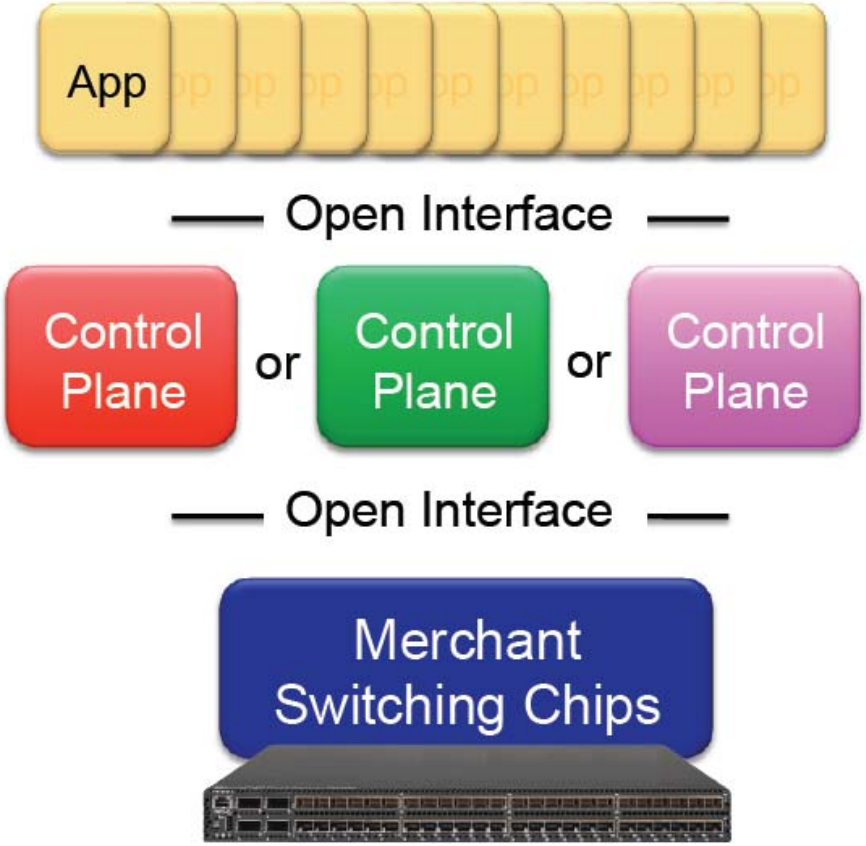
Evolution of computer architectures

Vertically integrated
Closed, proprietary
Slow innovation
Small industry



Horizontal
Open interfaces
Rapid innovation
Huge industry

Evolution of network systems



Vertically integrated
Closed, proprietary
Slow innovation

Horizontal
Open interfaces
Rapid innovation

So why SDN?

- Networks are hard to *manage*
- Networks are hard to *evolve*
- Network design not based on *formal principles*
- SDN is not really a *"technological" advance*
  - It is merely a way of *organizing network functionality*
- But that's all the Internet architecture is…
  - Not clever, but the right design
- So what is research here?
  - Why is it a right design?

Architecture and academia don't mix

Academics get paid for being clever,
not for being right.

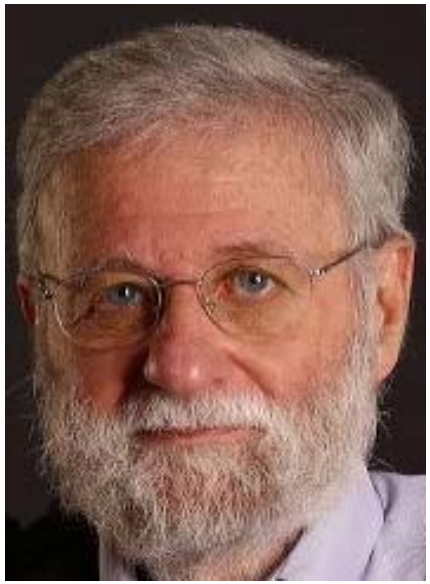
— Don Norman

The man who knows how will always
have a job. The man who also knows
why will always be his boss.

— Ralph Waldo Emerson

A story about mastering complexity

- 1985: Don Norman visits Xerox PARC
- At start of his talk on UI design he asks:
  - **"Who in the audience drives a stick shift?"**
- After most of the audience raises their hands, he looks sternly out over the crowd and says:
  - **"None of you should ever design a user interface"**



What was his point?

- The ability to *master complexity* is not the same as the ability to *extract simplicity*
- When first getting systems to work….
  - Focus on mastering complexity
- When making system easy to use and understand
  - Focus on extracting simplicity
- You will never succeed in extracting simplicity
  - If don't recognize it is different from mastering complexity

What is my point?

- Networking has never made the distinction…
- And therefore has never made the transition
  - Still focused on mastering complexity
  - Little emphasis on extracting simplicity from control plane
- Extracting simplicity builds intellectual foundations
  - **Necessary for creating a discipline….**

Programming analogy

- What if programmers had to:
  - Specify where each bit was stored
  - Explicitly deal with all internal communication errors
  - Within a programming language with limited expressability
- Programmers would redefine problem:
  - Define a higher level abstraction for memory
  - Build on reliable communication abstractions
  - Use a more general language
- Abstractions divide problem into tractable pieces
  - Make task of control program easier…

How programming made the transition

- Machine languages: no abstractions
  - Mastering complexity was crucial
- Higher-level languages: OS and other abstractions
  - File system, virtual memory, abstract data types, ...
- Modern languages: even more abstractions
  - Object orientation, garbage collection,…

Abstractions simplify programming

Easier to write, maintain, reason about programs

The power of abstraction

"Modularity based on abstraction is the way things get done"

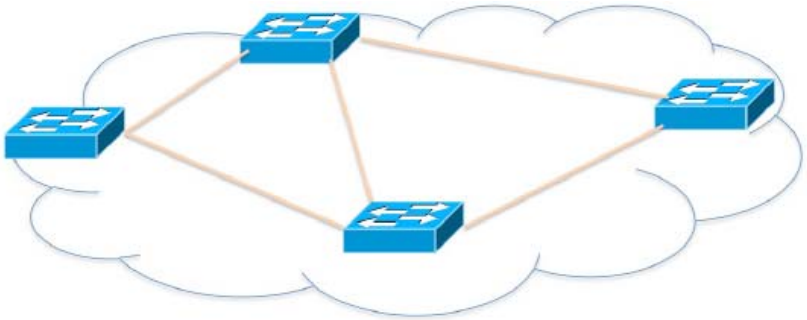
– Barbara Liskov

Abstractions → Interfaces → Modularity

- Modularity provides:
  - Code reuse
  - Flexibility of implementation
  - Conceptual separation of concerns

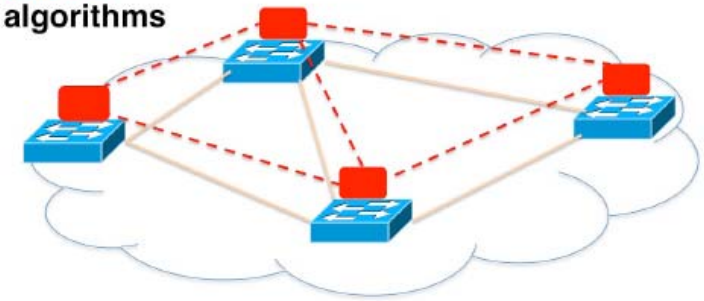
What abstractions have been applied to networking?

Data plane:
Packet streaming



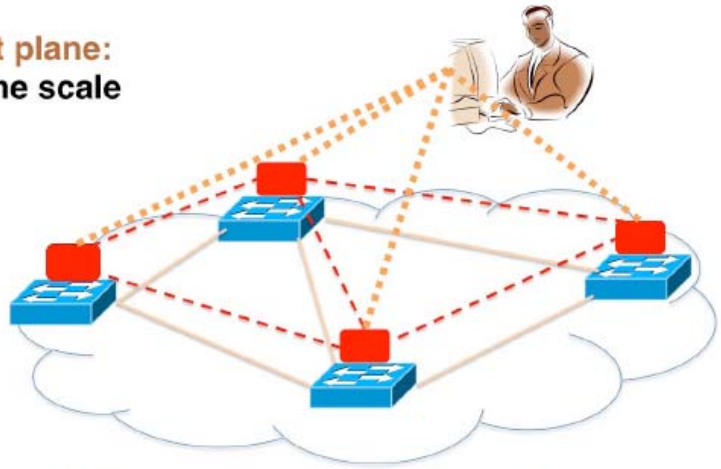
Forward, filter, buffer, mark, rate-limit, and measure packets

Control plane:
Distributed algorithms

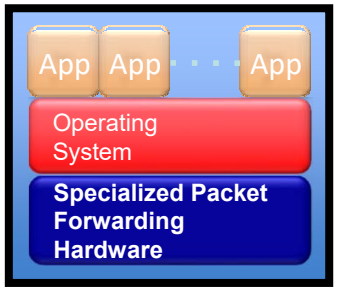


Track topology changes, compute routes, install forwarding rules

Management plane:
Human time scale



Collect measurements and configure the equipment



Closed network node

The two networking "planes"

- Data plane: process packets with local fwding state
  - Fwding state + packet header → forwarding decision
  - Examples: IP forwarding, layer 2 switching
- Control plane: compute the forwarding state
  - Manual configuration (and scripting)
  - Examples: routing protocols, network middlebox configuration
- These different planes require different abstractions

Data plane abstractions: layers

Applications

…built on…

Reliable (or unreliable) transport

…built on…

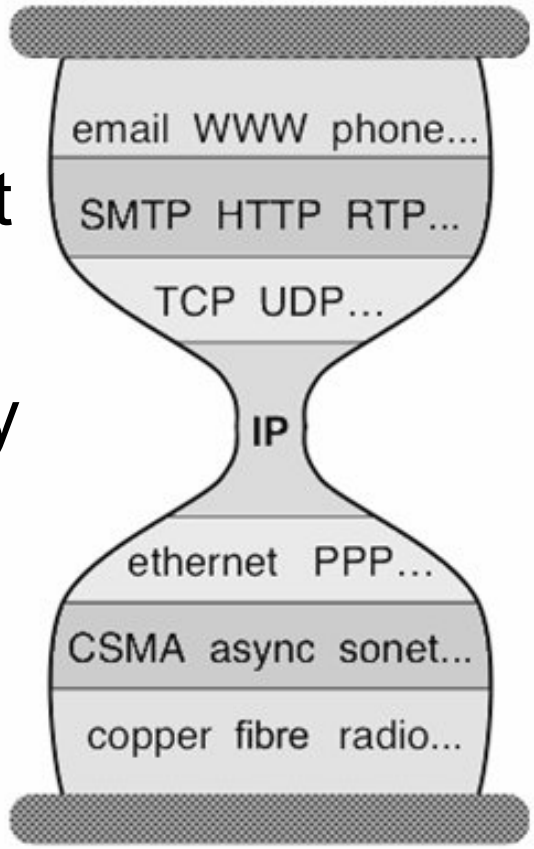
Best-effort global packet delivery

…built on…

Best-effort local packet delivery

…built on…

Physical transfer of bits



Control plane abstractions: ???

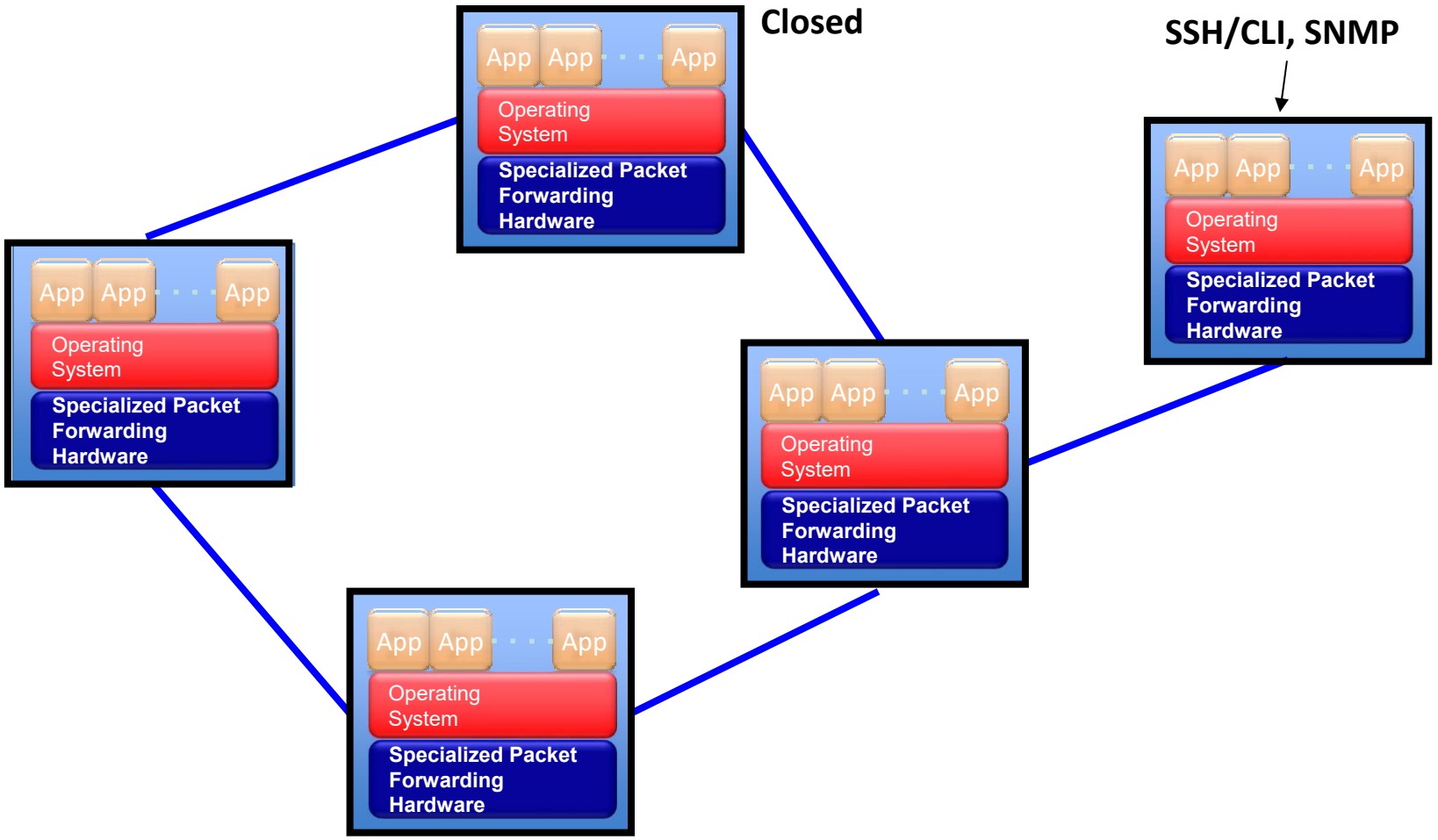
- Too many control plane mechanisms with variety of goals:
  - Routing: distributed routing algorithms
  - Isolation: ACLs, VLANs, Firewalls,…
  - Traffic engineering: adjusting weights, MPLS,…
- No modularity, limited functionality
- **Control plane: mechanism without abstraction**
  - *Too many mechanisms, not enough functionality*

3 fundamental principles of SDN

- **Control and forwarding planes**
  - **Traditional: Control and forwarding planes are co-located in network elements.**
  - **SDN: Control plane is separated from forwarding plane and moved to SDN controller.**
- **Control intelligence**
  - **Traditional: Control intelligence is distributed in each network element.**
  - **SDN: Control intelligence is logically centralized at the SDN controller.**
- **Network programmability by applications**
  - **Traditional: Network cannot be programmed by applications. Each network element must be separately configured.**
  - **SDN: Network can be programmed by applications. Controller can expose application interfaces to manipulate network.**

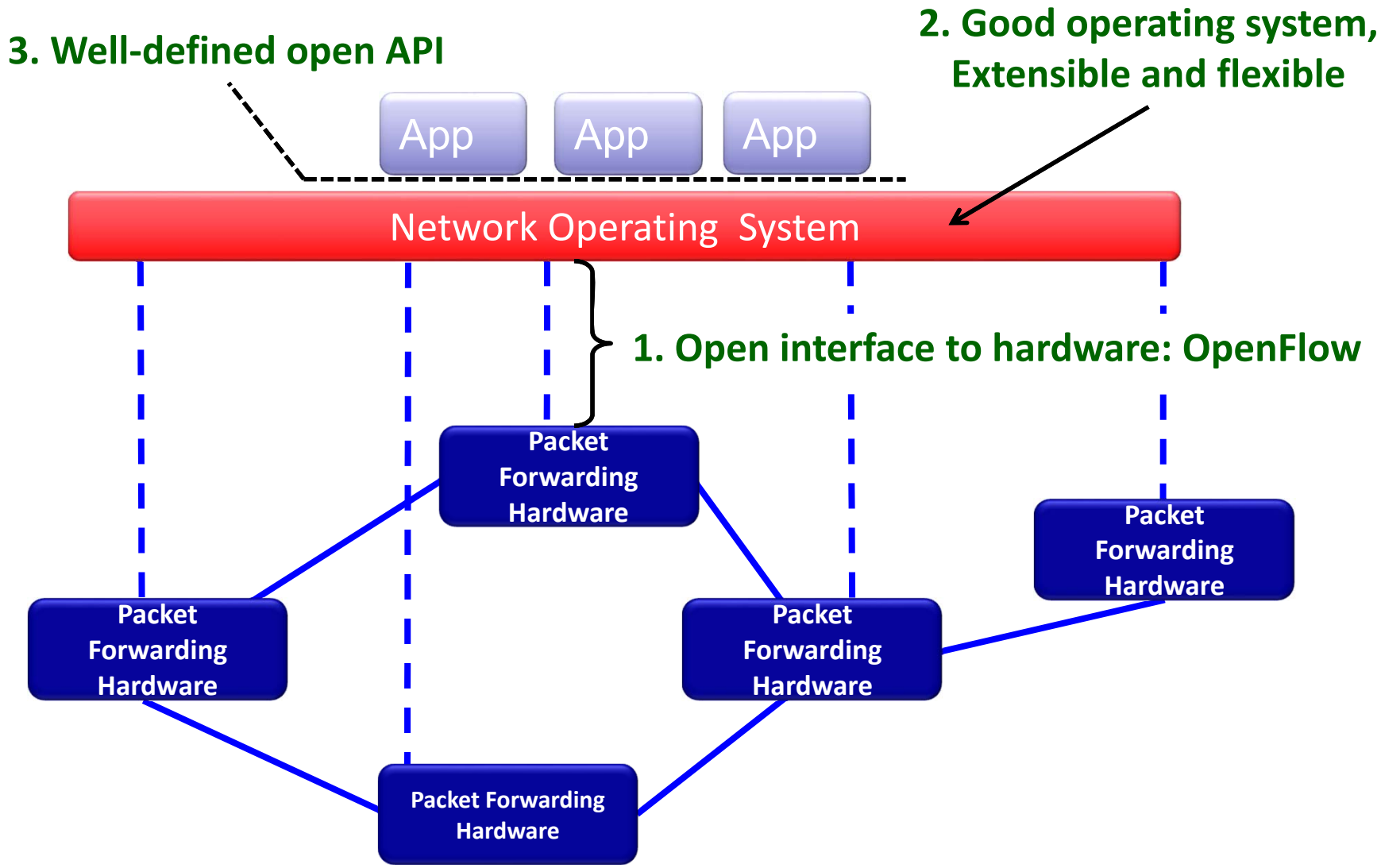
Control & data plane separation

- Current network



Control & data plane separation

- SDN tomorrow



Control & data plane separation

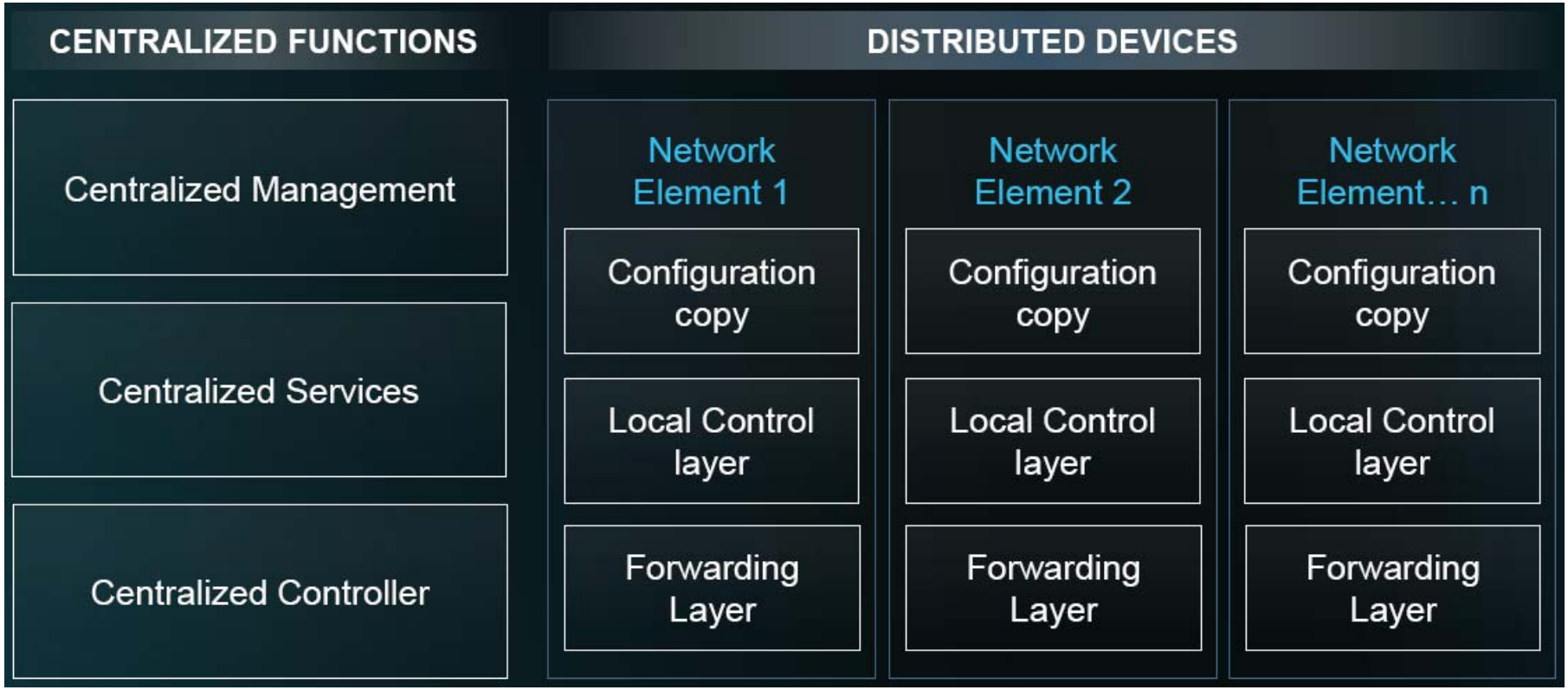
- Benefits
  - Independent evolution and development
    - The software control of the network can evolve independently of the hardware
    - Turns network hardware into a commodity
    - Make switches simpler, faster
  - Control from high-level software program
    - Control behavior using higher-order programs
    - Debug/check behavior more easily
  - Standardized APIs
    - Provide opportunity for rapid innovation in networking

Control & data plane separation

- Challenges
  - "OpenFlow is too limited"
    - How can you solve all networking problems with such a narrow set of primitives?
    - All solutions will require lots of network services outside of OpenFlow in order to function, so does the "openness story" really hang together?
  - "You cannot replace all the traditional switch/routing functions"
    - Need to maintain Controller connectivity across a network
    - Local processing required for HA/Fast failover
    - So will the switches really be any cheaper/simpler, or does OpenFlow support become yet another switch feature?

Logically centralized control

- ARPAnet: both the data and control planes were totally distributed
- SDN: enables logically centralized control

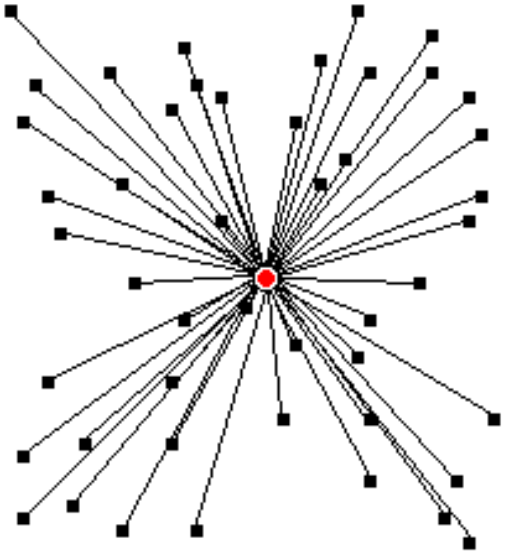


Logically centralized control

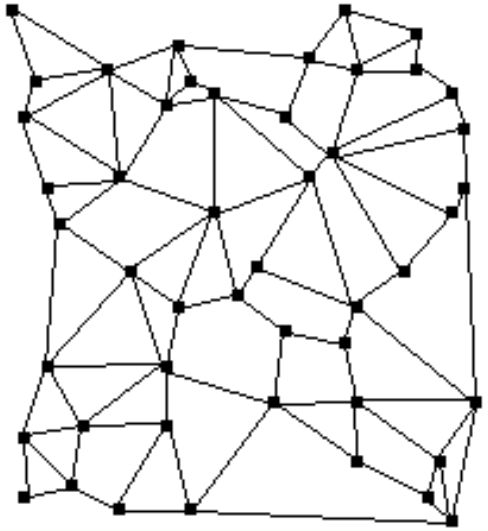
- Benefits
  - Offers deeper integration
    - End-to-end control is possible
    - QoS is possible
  - Allows global optimization and planning
    - Global information is available
    - Global policy enforcement is easy
  - Makes algorithms simpler
    - Centralized algorithms are always simpler than distributed ones
  - Provides APIs for APPs
    - Provide opportunity for rapid innovation in networking

Why not at the beginning?

- History:
  - ARPAnet-distributed vs. centralized network



Centralized Network



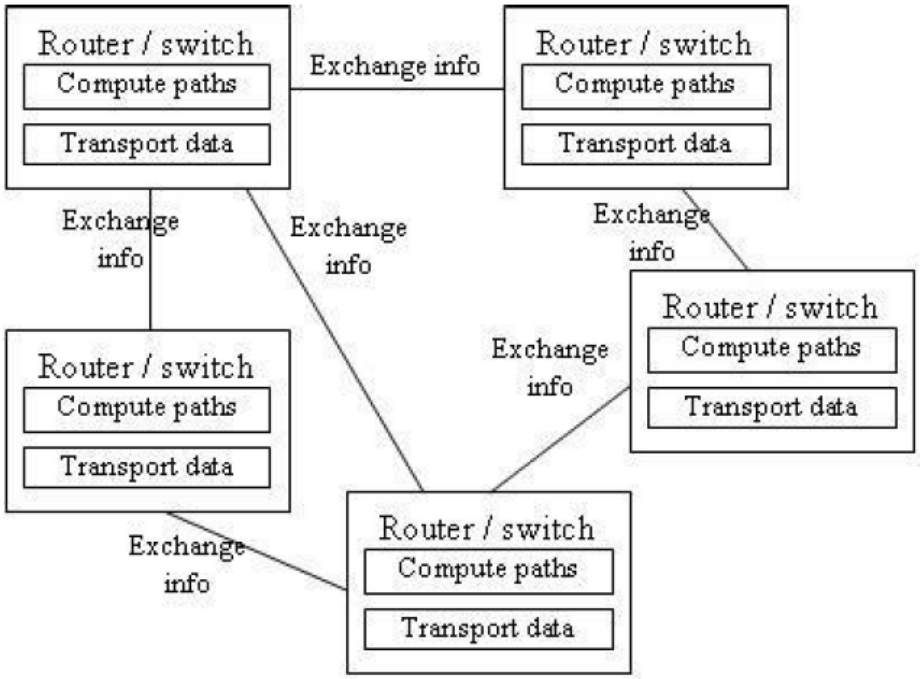
Distributed Network

  - Key point: centralized control, not star like topology

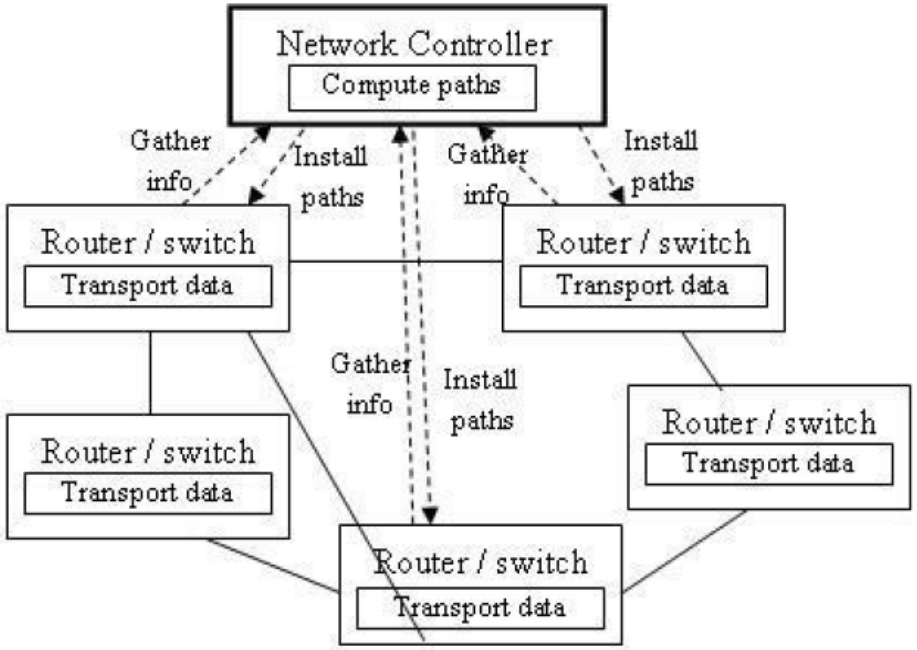
Why routing is difficult?

- Incomplete information
  - Routers are given clues and must infer what they need to know
- Duplicated calculation
  - Routers do the route calculations independently
  - Even if the network topology means two nearby routers will calculate similar results, they do not share the results of the overlapping calculations.

Path calculation comparison



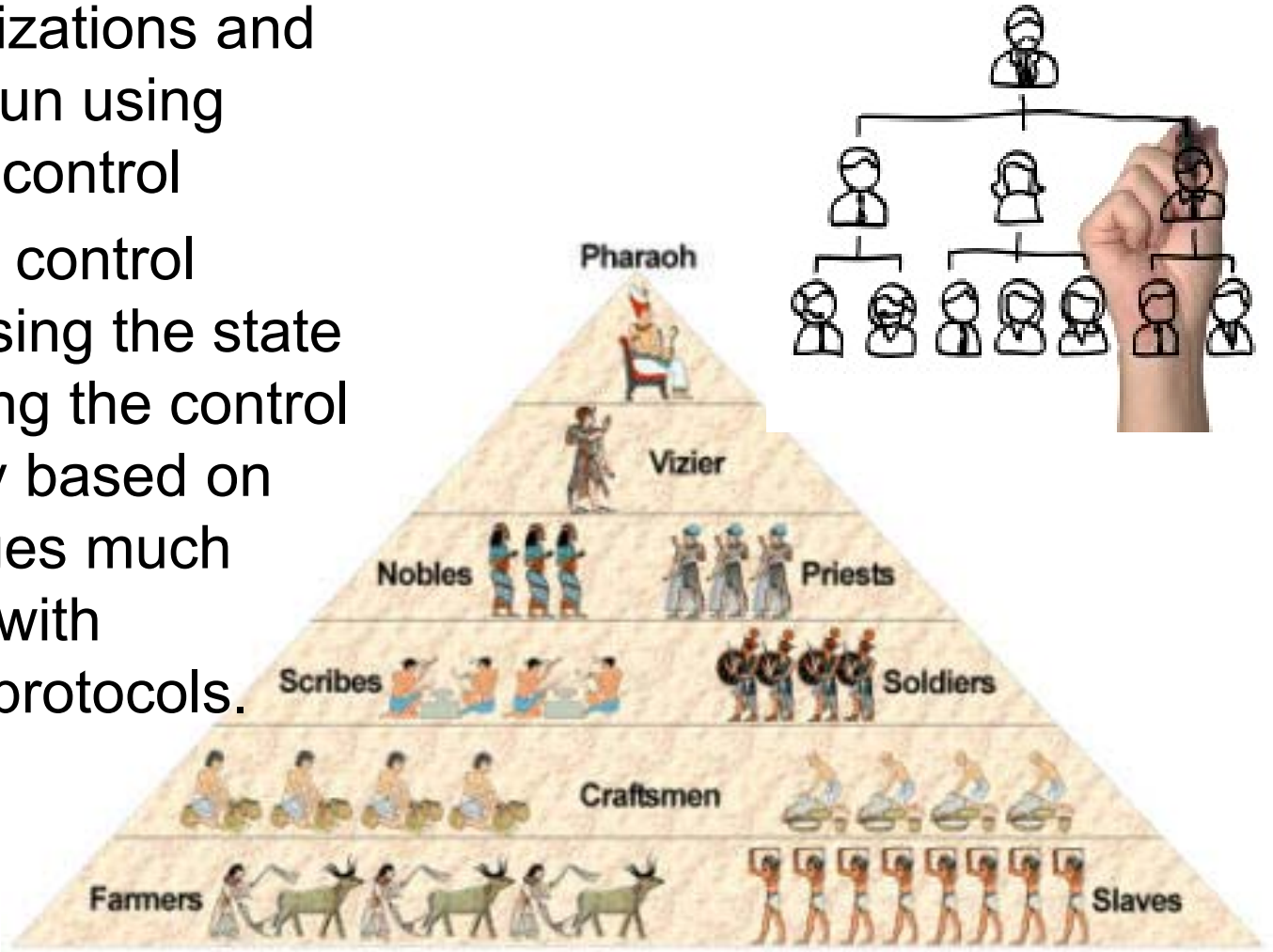
Traditional network



SDN

Centralized control is more efficient

- Most organizations and teams are run using centralized control
- Centralized control makes sensing the state and adjusting the control dynamically based on state changes much faster than with distributed protocols.

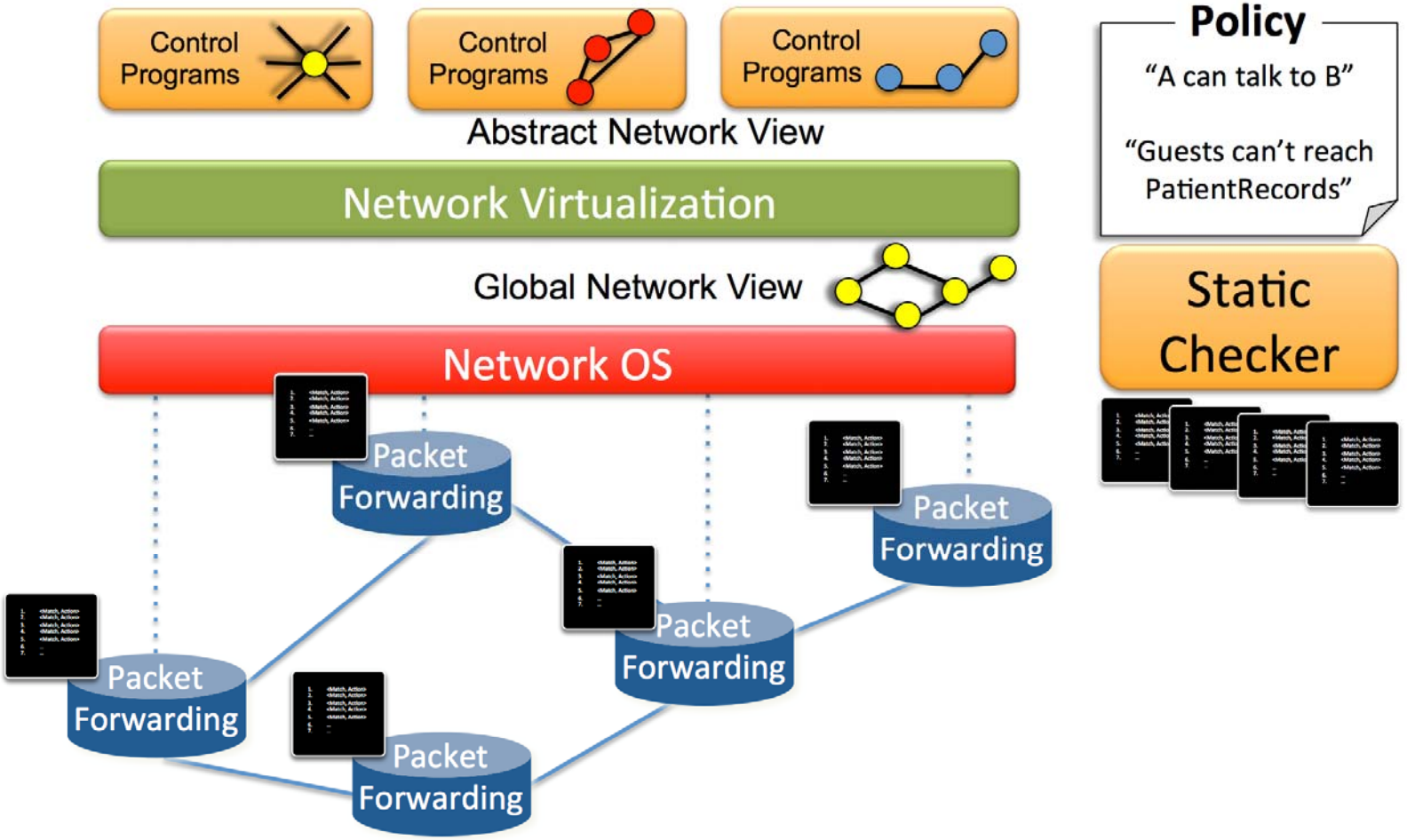


Logically centralized control

- Challenges
  - Scalability: Routing decisions for many routers
  - Reliability: Correct operation under failure
  - Consistency: Ensuring consistency across multiple control replicas
  - Security: How to handle attacks on centralized controller
- Note that
  - distributed methods also have the same issues!

Programmable network

- Well-defined layer structure for control plane
- Lets you write APPs



APPs change cellular industry

App Store, 2008

Network APPs

Classical Network Appliance Approach


Message Router


CDN


Session Border Controller


WAN Acceleration


DPI


Firewall


Carrier Grade NAT


Tester/QoE monitor


SGSN/GGSN


PE Router


BRAS


Radio/Fixed Access Network Nodes

- Fragmented, purpose-built hardware.
- Physical install per appliance per site.
- Hardware development large barrier to entry for new vendors, constraining innovation & competition.

Network Functions Virtualization Approach

Open Ecosystem

Competitive & Innovative

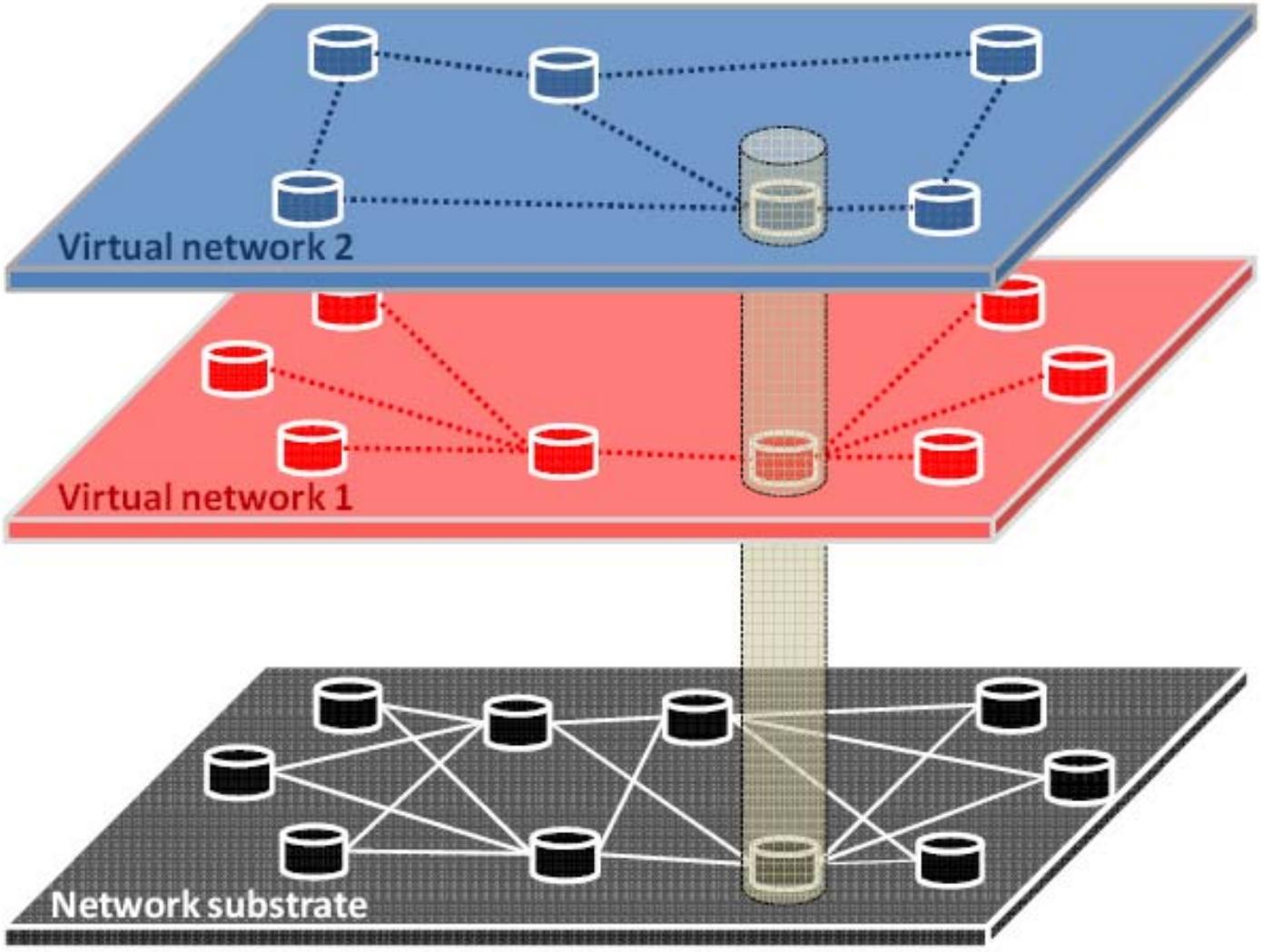

Orchestrated, automatic & remote install.


High volume standard servers

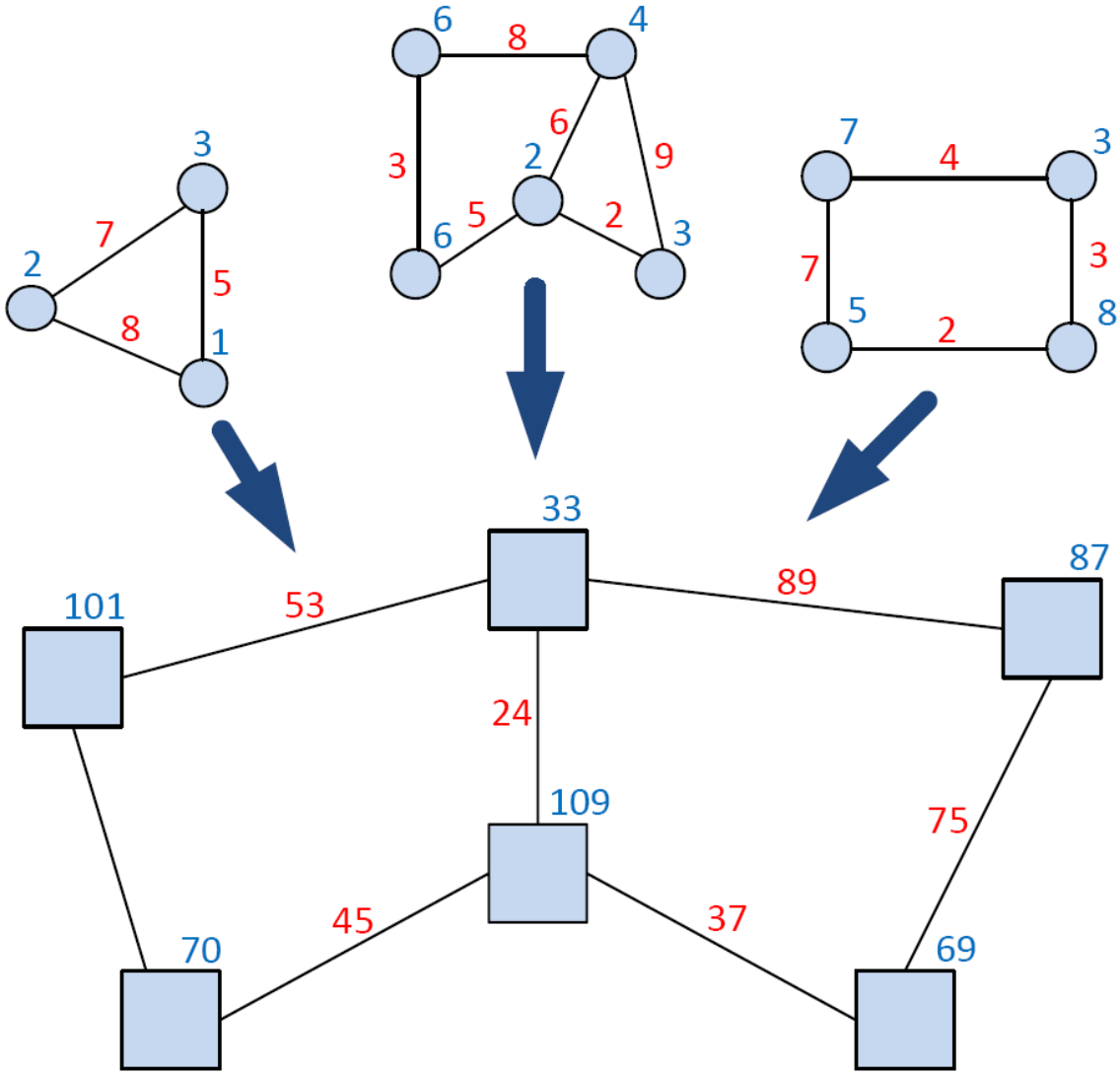

High volume standard storage


High volume Ethernet switches

Network virtualization: killer APP?

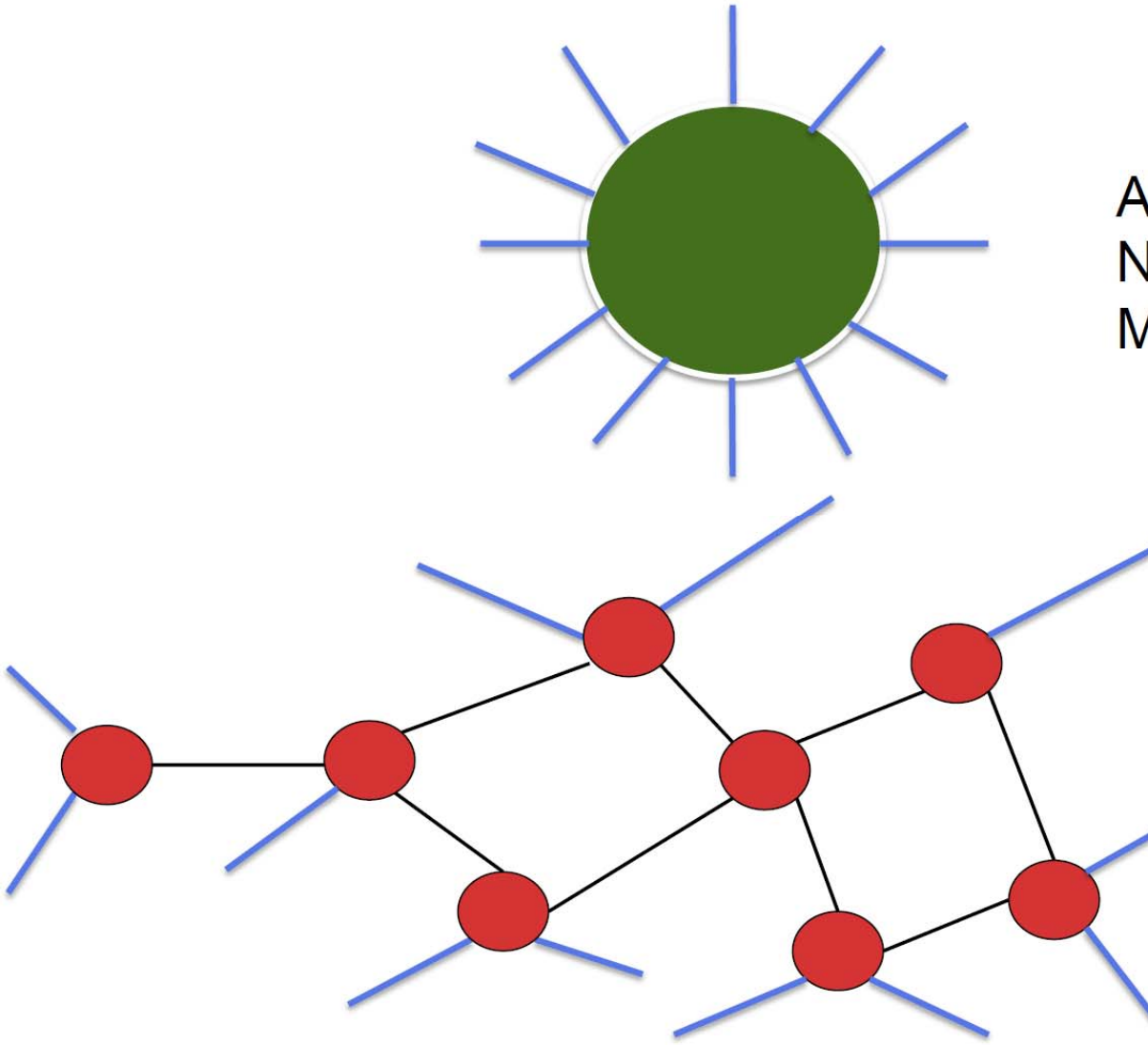


Network virtualization: killer APP?



Physical (substrate) network

Simple Example: Access Control



Layers are Great Abstractions, but

- Layers only deal with the data plane
- We have no powerful control plane abstractions!
- How do we find those abstractions?
- Define our problem, and then decompose it.

The Network Control Problem

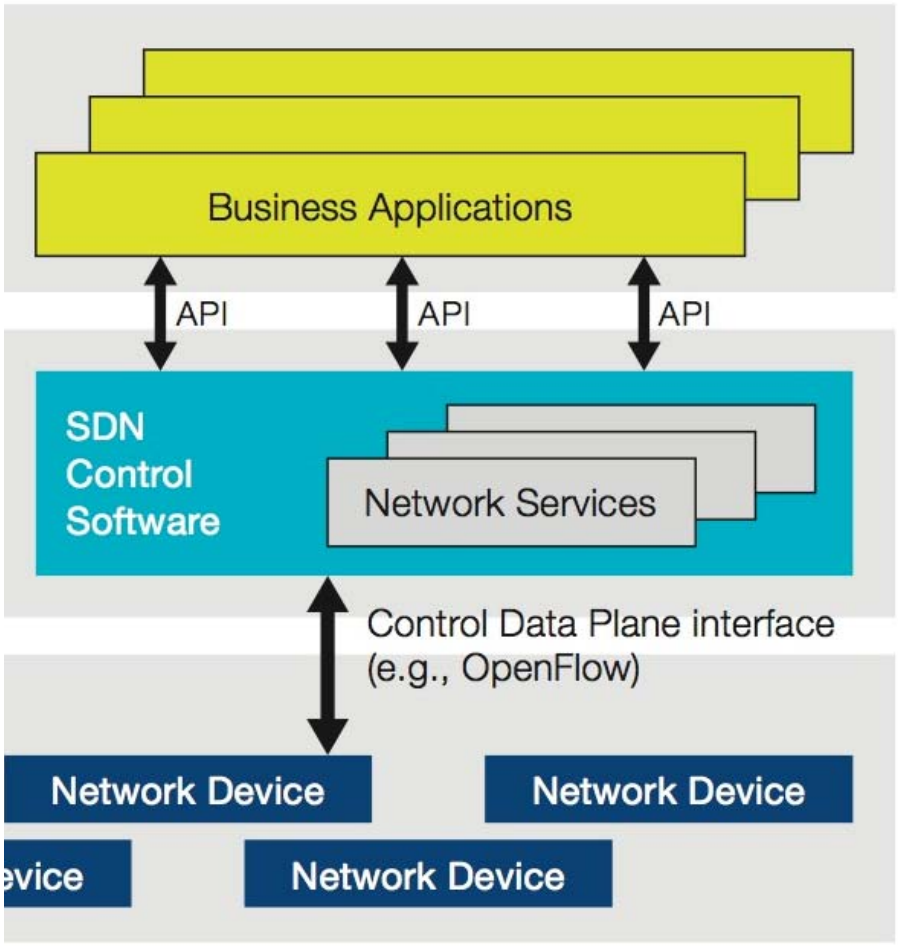
- Compute the configuration of each physical device
  - E.g., Forwarding tables, ACLs,…
- Operate without communication guarantees
- Operate within given network-level protocol

Only people who love complexity would find this a reasonable request

From Requirements to Abstractions

- Operate without communication guarantees
  - Need an abstraction for distributed state
  - Natural abstraction: global network view
  - No longer a distributed protocol, now just a graph algorithm
- Compute the configuration of each physical device
  - Need an abstraction that simplifies configuration
  - Simple model only enough detail to specify goals
- Operate within given network-level protocol
  - Need an abstraction for general forwarding model
  - It should hide details of underlying hardware

Summary: What is SDN



- Providing programmability of networks
- Logically centralized control
- Control and data plane separation

Key SDN principles (from ONF)

Abstraction

- Decouple business applications
- Decouples control plane
- Decouple Virtualized configuration

Programmability

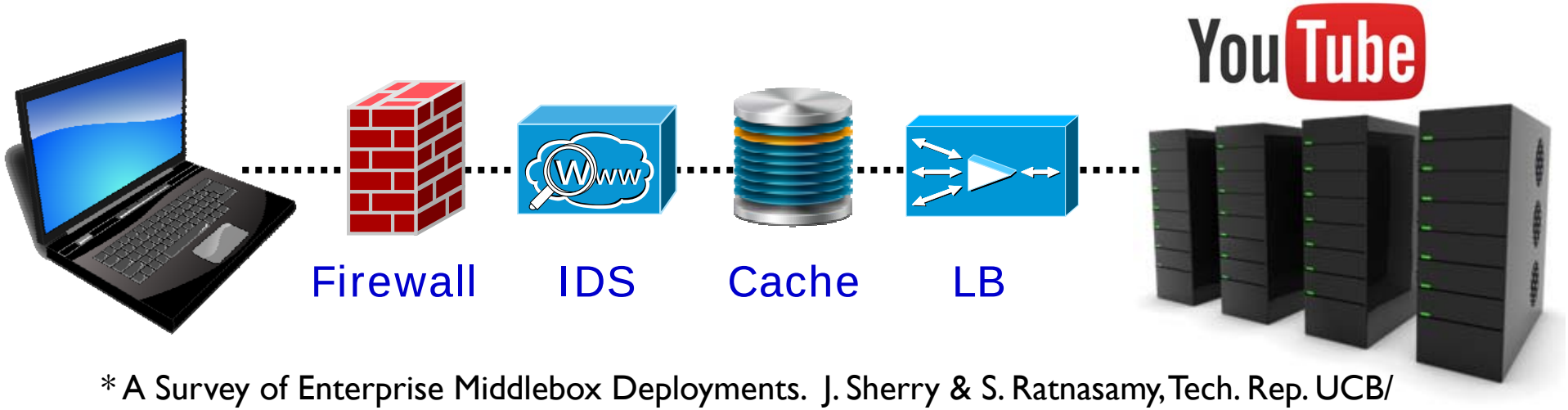
- Enable innovation/differentiation
- Accelerates new feature and service introduction

Centralized Intelligence

- Simplify provisioning
- Optimize performance
- Granular policy management

Network Functions, or Middleboxes

- Middlebox (MB): a special purpose device that may inspect, transform, or modify packets to improve performance and security
- A survey* shows that roughly one in three devices in an enterprise network is a middlebox



* A Survey of Enterprise Middlebox Deployments. J. Sherry & S. Ratnasamy, Tech. Rep. UCB/EECS-2012-24, EECS Department, University of California, Berkeley, 2012.

Traditional Middleboxes

- Monolithic closed black-boxes
  - ✘ High cost
  - ✘ Limited provisioning and scalability

Firewall

Load Balancer

Intrusion Prevention System

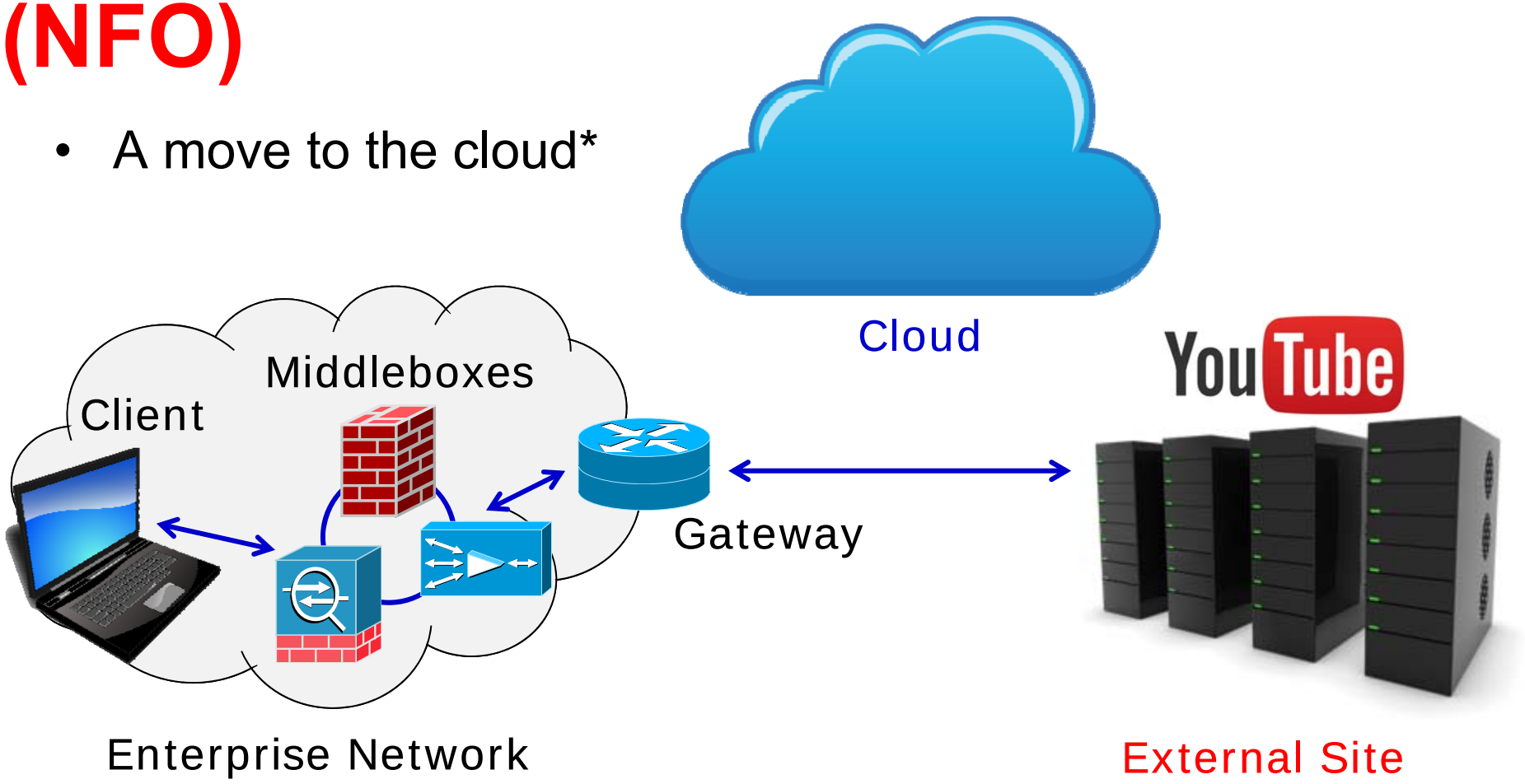
Network Function Virtualization

- Monolithic closed black-boxes
  - ✘ High cost
  - ✘ Limited provisioning and scalability

- Network Function Virtualization (NFV):
  - ✔ Reduce cost (by moving to software)
  - ✔ Improve provisioning and scalability (by virtualizing software NFs)

Network Function Outsourcing (NFO)

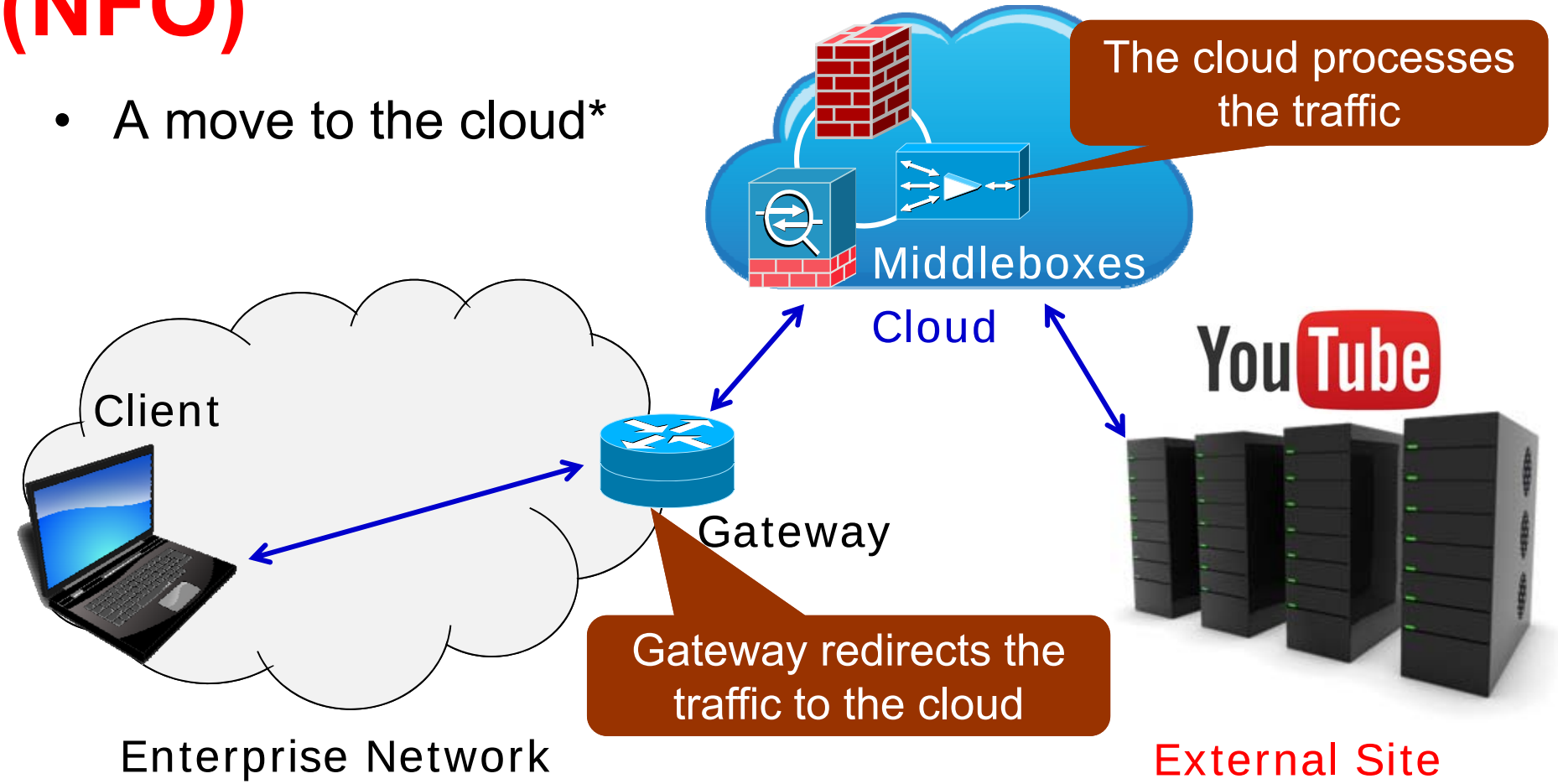
- A move to the cloud*



* Making Middleboxes Someone Else's Problem: Network Processing as a Cloud Service. J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, V. Sekar. ACM SIGCOMM 2012.

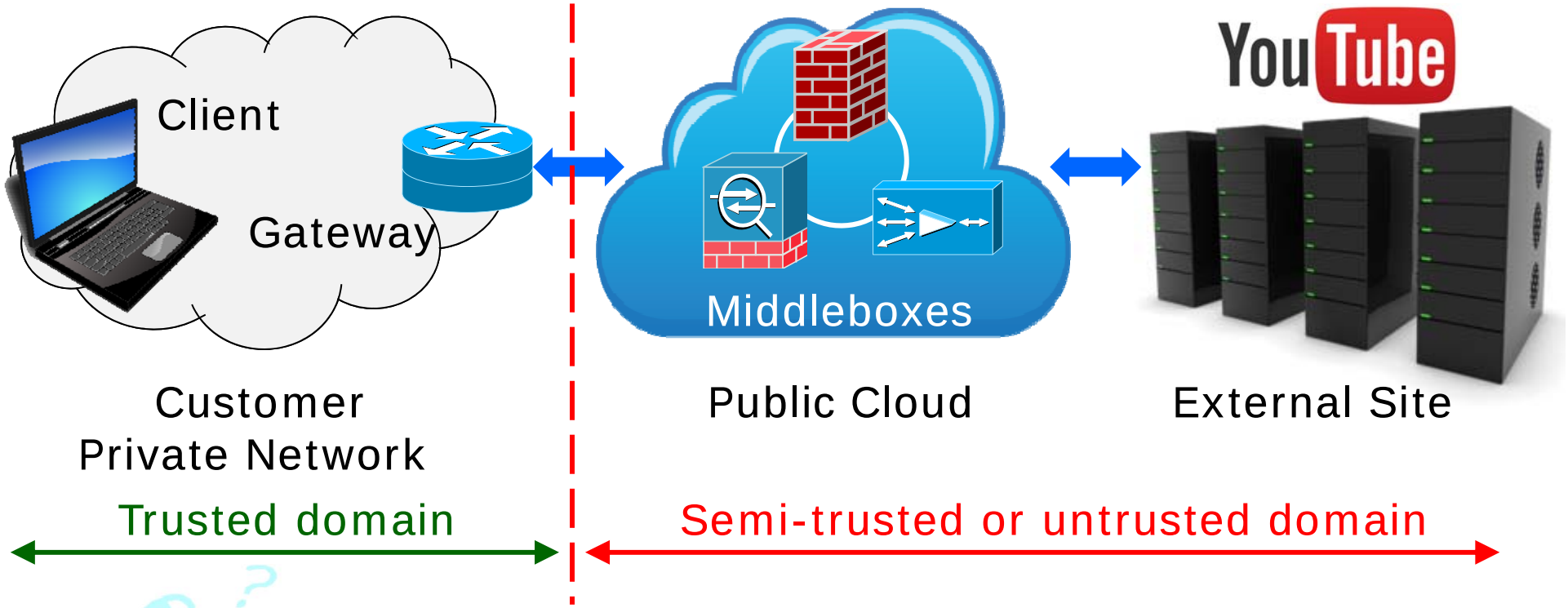
Network Function Outsourcing (NFO)

- A move to the cloud*



- Network Function as a Service (MB Outsourcing)
  - e.g., Firewall as a Service (FaaS)

Security Challenges for NFO

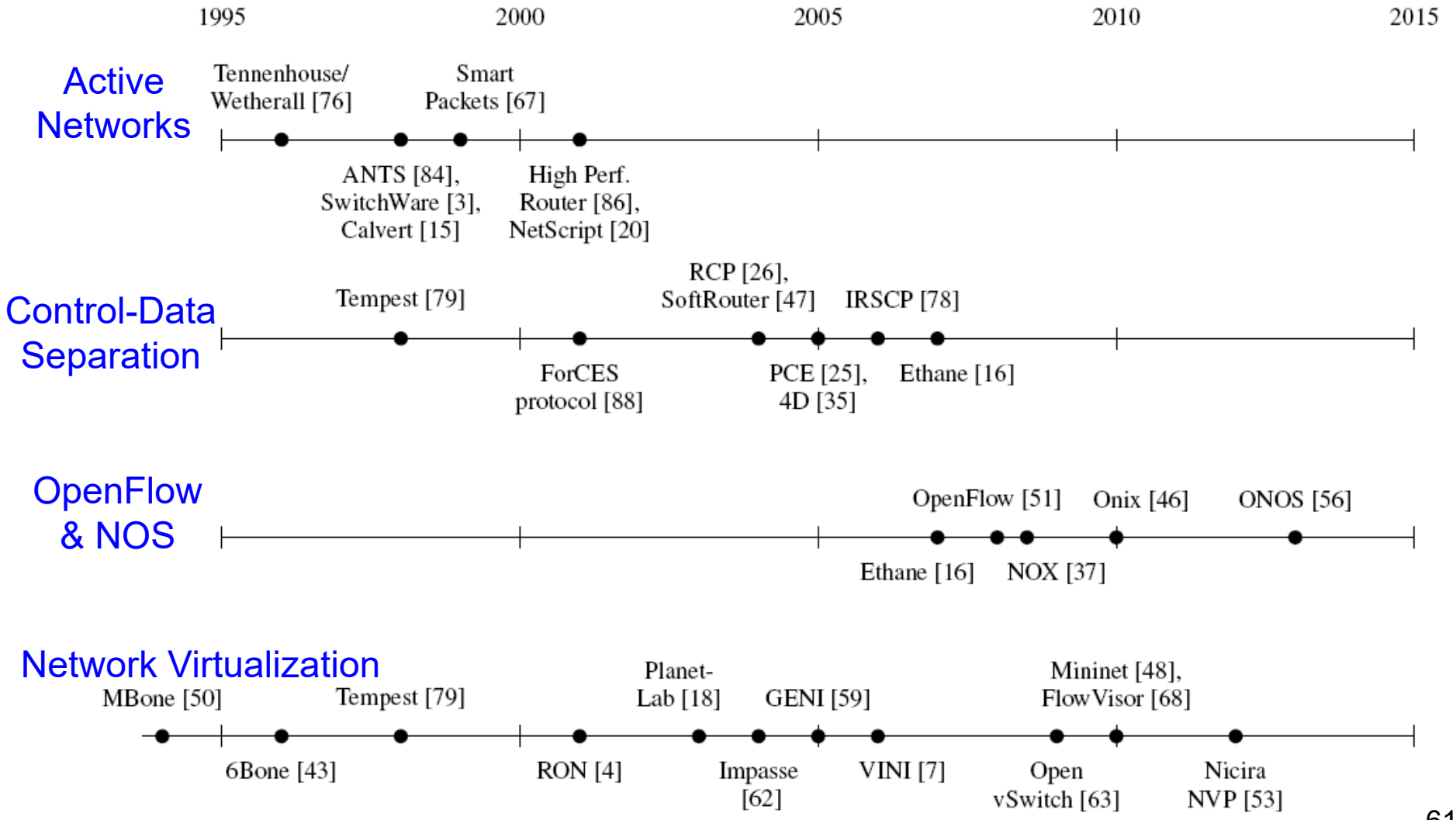


- Are packets being modified or incorrectly processed ?
  - Verifiable policy enforcement problem
- Does the cloud learn my network policies?
  - Policy confidentiality problem

Today's agenda

- Introduction to Software Defined Networking
  - What is SDN? Why SDN?
  - History of SDN
  - SDN standardization
  - SDN deployment models
  - OpenFlow
  - SDN applications
  - SDN development tools
  - Future directions

History of SDN



Origins of active networks

- DARPA research community (1994-1995)
- Identified problems with today's networks
  - Difficulty of integrating new technology
  - Poor performance due to redundant operations at several protocol layers
  - Difficulty accommodating new services

Motivation for active networks

- Accelerating innovation
  - Internet innovation relies on consensus
  - Takes ten years from prototype to deployment (standardization, procurement, deployment)
- Active nodes allow routers to download new services into the infrastructure
  - User-driven innovation

Two different approaches

- Capsules (“integrated”)
  - Every message is a program. Active nodes evaluate content carried in packets.
  - Code dispatched to execution environment
- Programmable Switches (“discrete”)
  - Custom processing functions run on the routers
  - Packets are routed through programmable nodes
  - Program depends on the packet header

What happened?

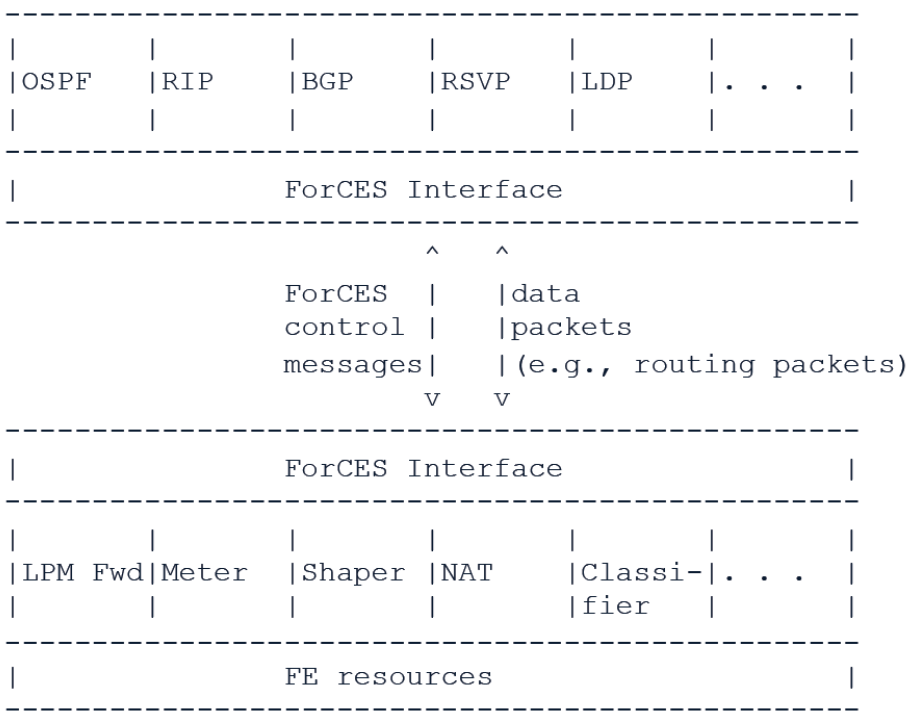
- Timing was off
  - No clear application (pre-data center/cloud)
  - Hardware support wasn't cheap -- everyone was using ASICs, whereas now TCAMs, FPGAs, NPUs.
- Some missteps
  - Security, special languages for safe code, packets carrying code
  - End user as programmer (vs. network operator)
  - Interoperability
- In contrast: OpenFlow did a good job grappling with backwards compatible with switch hardware.
  - Simple firmware upgrade.
  - Switch hardware already supported the basics.

Background of separation

- 1990s,
  - the link speeds in backbone networks grew rapidly, leading equipment vendors to implement packet-forwarding logic directly in hardware, separate from the control-plane software.
  - ISPs were struggling to manage the increasing size and scope of their networks, and the demands for greater reliability and new services
  - The rapid advances in commodity computing platforms meant that servers often had substantially more memory and processing resources than the control-plane processor of a router deployed just one or two years earlier.

IETF ForCES

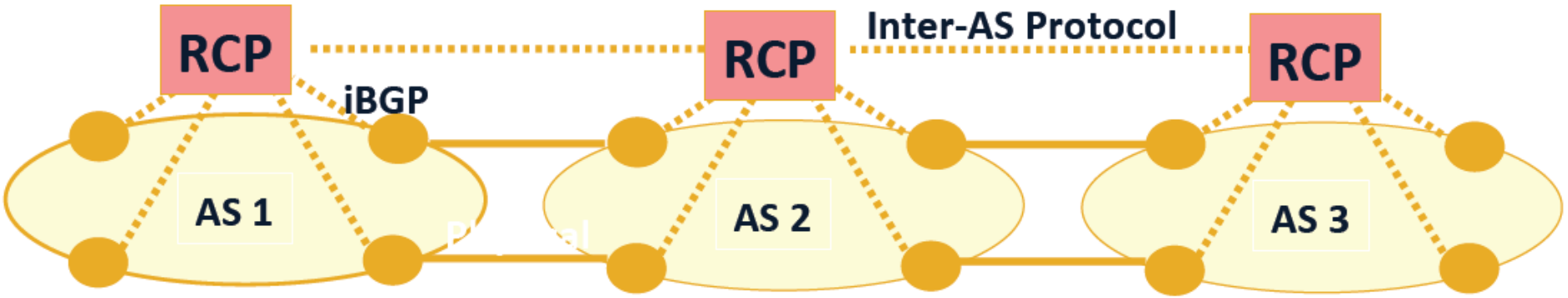
- First RFC in 2003, three implementations
- Protocols for multiple control elements (CE) and forwarding elements (FE)



Problem: Requires standardization, adoption, deployment of new hardware (same problem observed by previous work!)

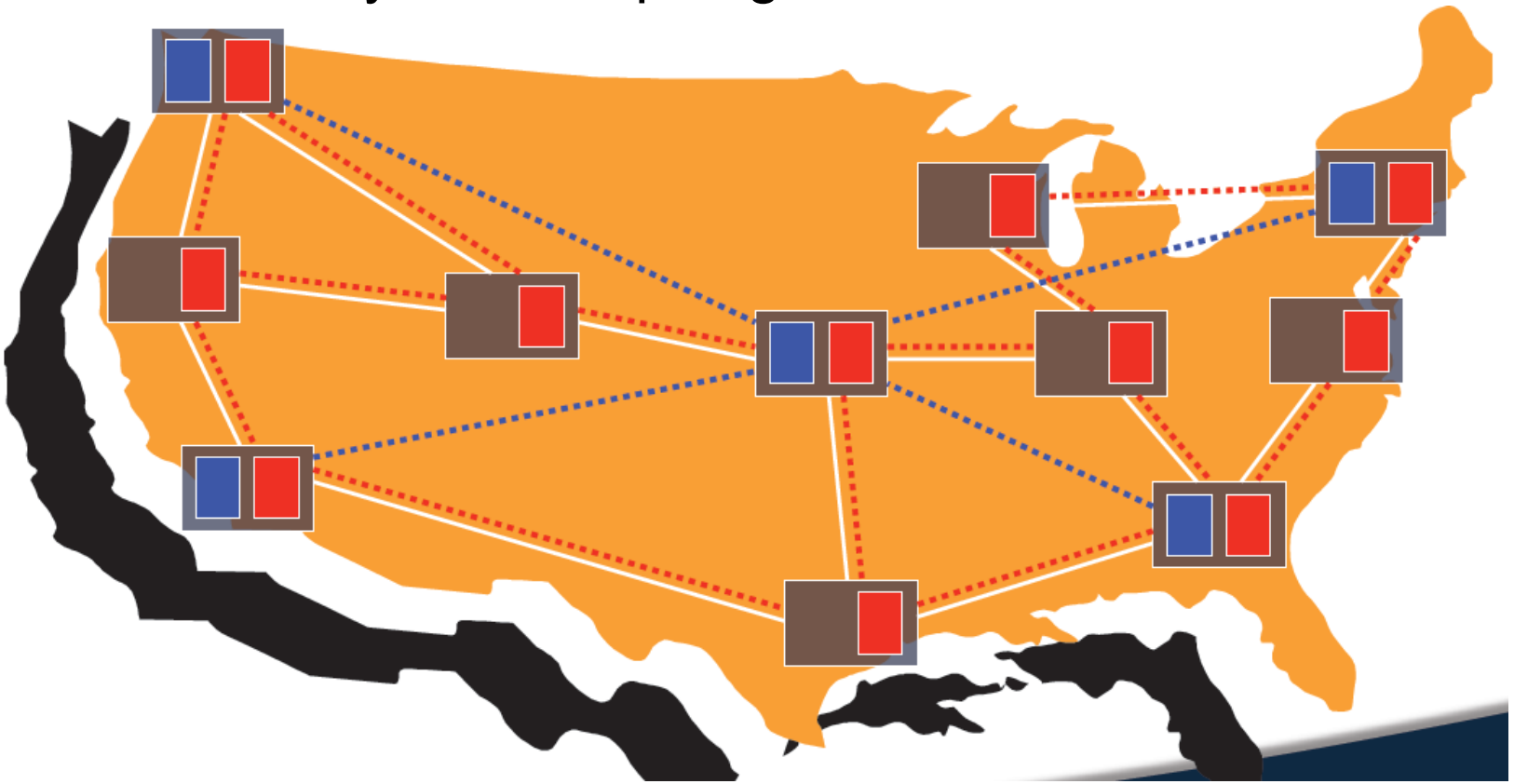
Routing control platform

- First proposed in 2004
- Computes routes on behalf of routers
- Uses existing routing protocol (BGP) to communicate routes to routers



Network virtualization motivation

- Sharing fixed physical infrastructure
- by multiple parties
- with arbitrary virtual topologies



Network virtualization before SDN

- Virtual networks
  - VLAN
  - Virtual private networks
- Overlay networks
  - Peer-to-Peer file-sharing App
  - Resilient overlay networks
- Experimental networks
  - PlanetLab
  - GENI

Today's agenda

- Introduction to Software Defined Networking
  - What is SDN? Why SDN?
  - History of SDN
  - SDN standardization
  - SDN deployment models
  - OpenFlow
  - SDN applications
  - SDN development tools
  - Future directions

SDN standardization

- Open Networking Foundation (ONF)
  - 2011.3
  - OpenFlow, OF-Config

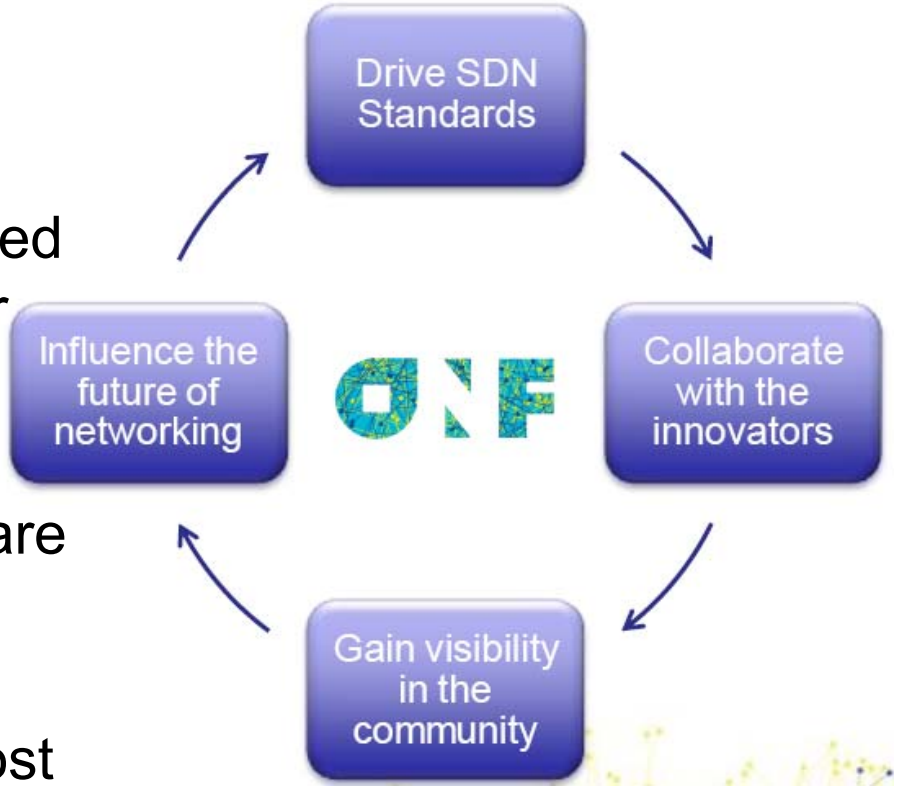
- European Telecommunication Standards Institute (ETSI)-Network Function Virtualization (NFV)
  - 2012.11
  - NFV

- OpenDaylight
  - 2013.4
  - OpenDaylight API, Controller

ONF

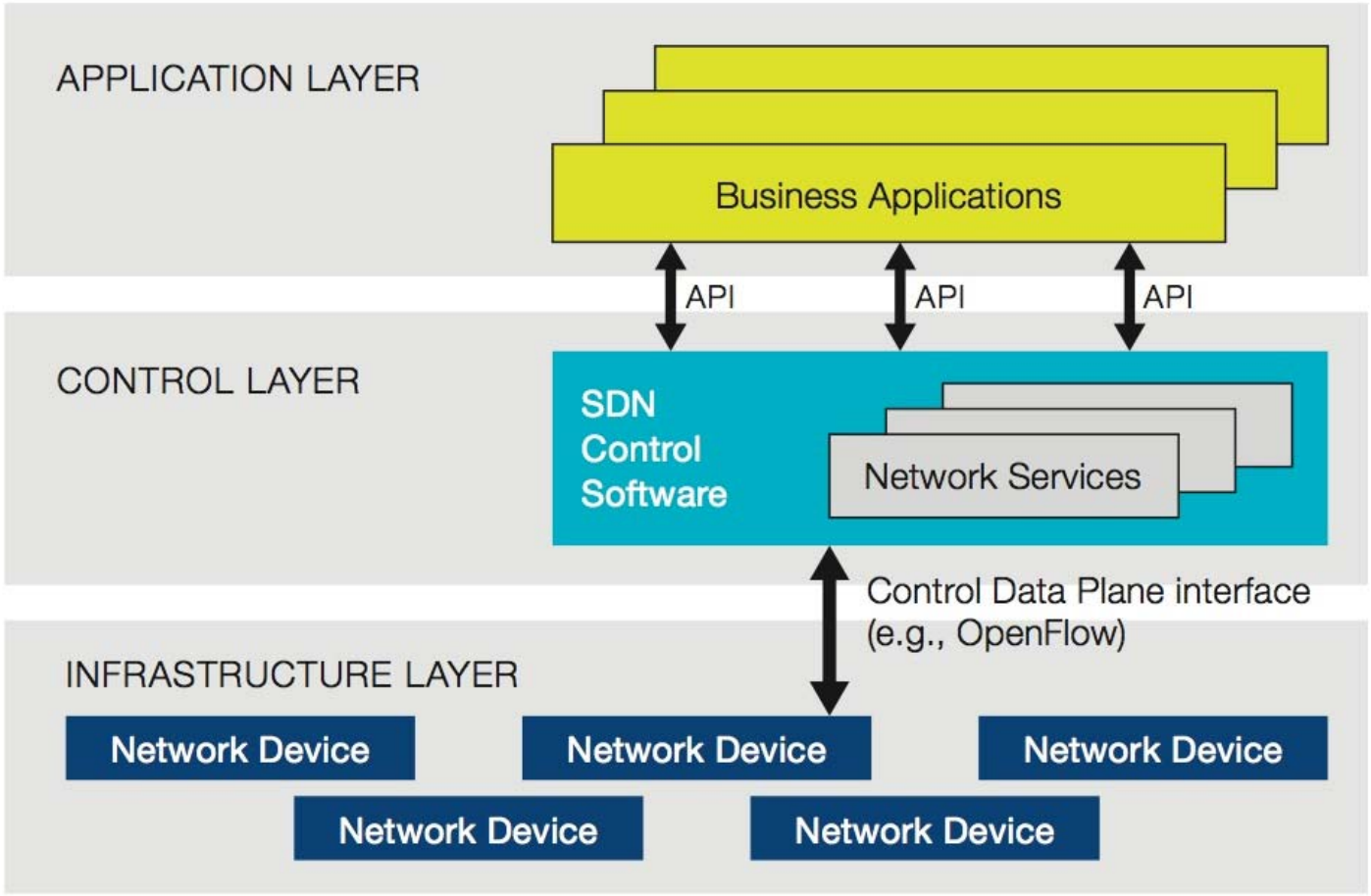
OPEN NETWORKING FOUNDATION

- A non-profit industry consortium
  - Founded: March 2011
  - Nearly 100 members
  - Vision: Make Software-Defined Networking the new norm for networks
  - ONF wants to transform networking industry to software industry through open SDN standards
  - ONF strives to create the most relevant SDN standards

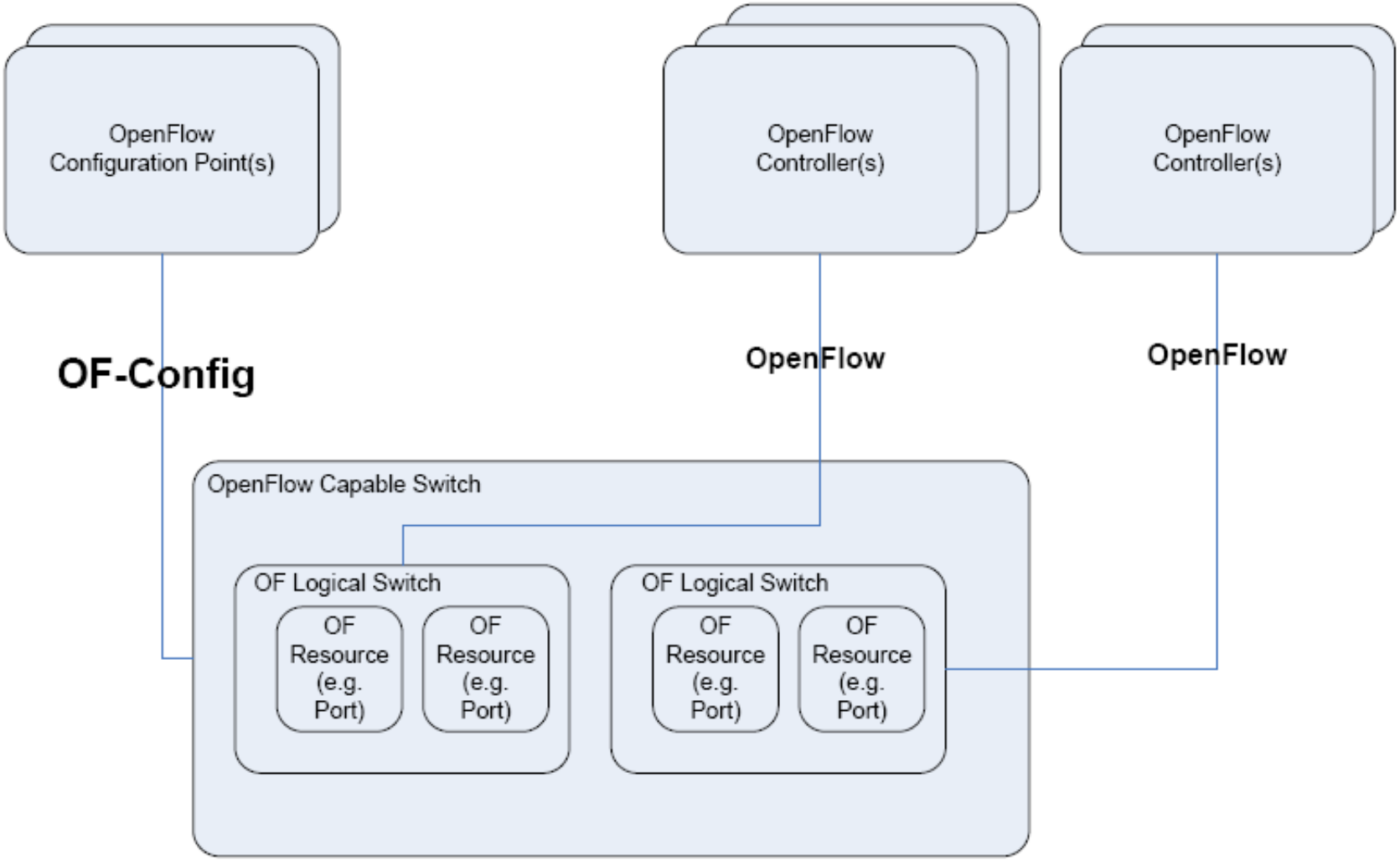


ONF's SDN

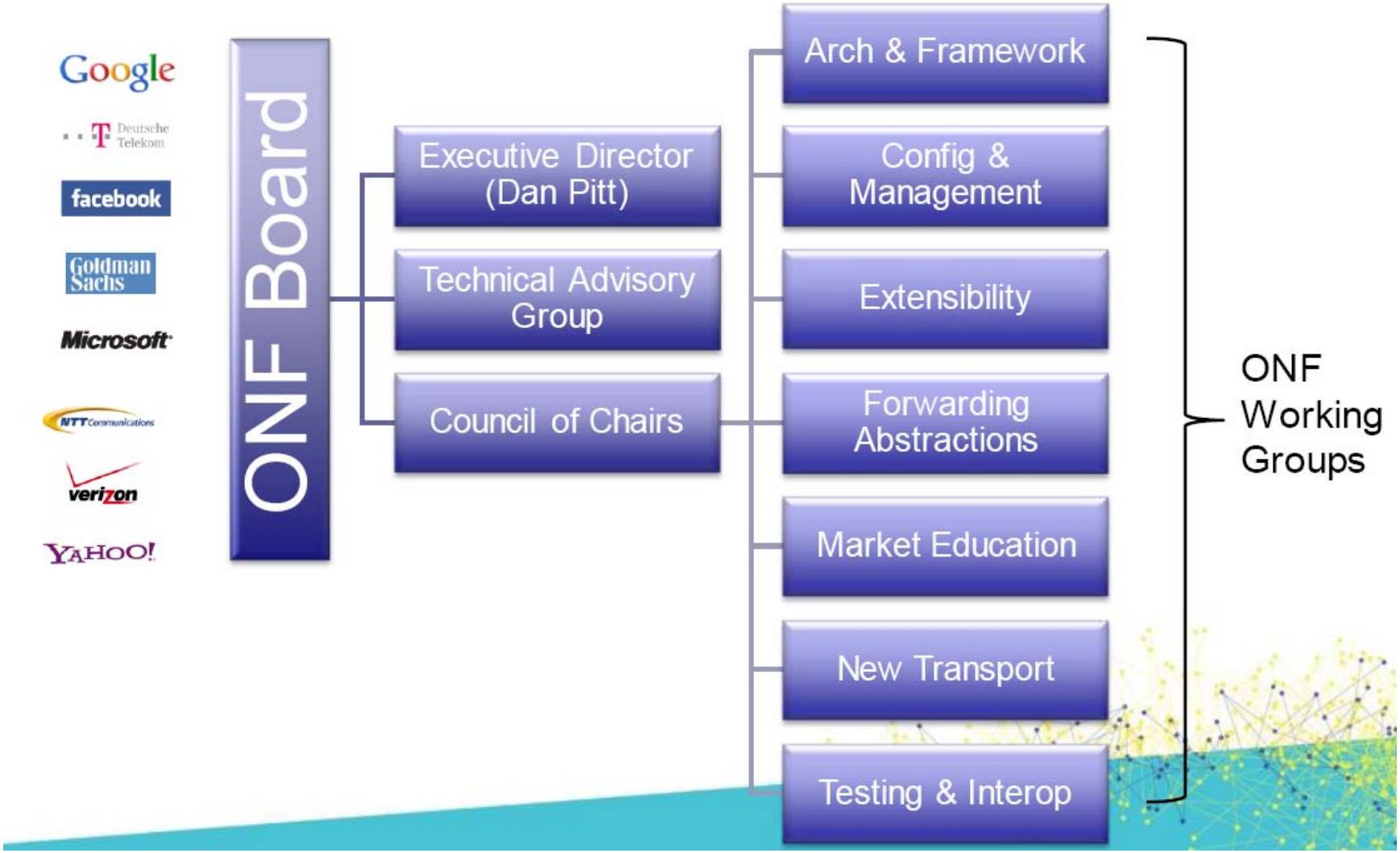
- ONF's SDN definition
  - From user's viewpoint



OpenFlow & OF-Config



ONF organiztion



ONF's working groups

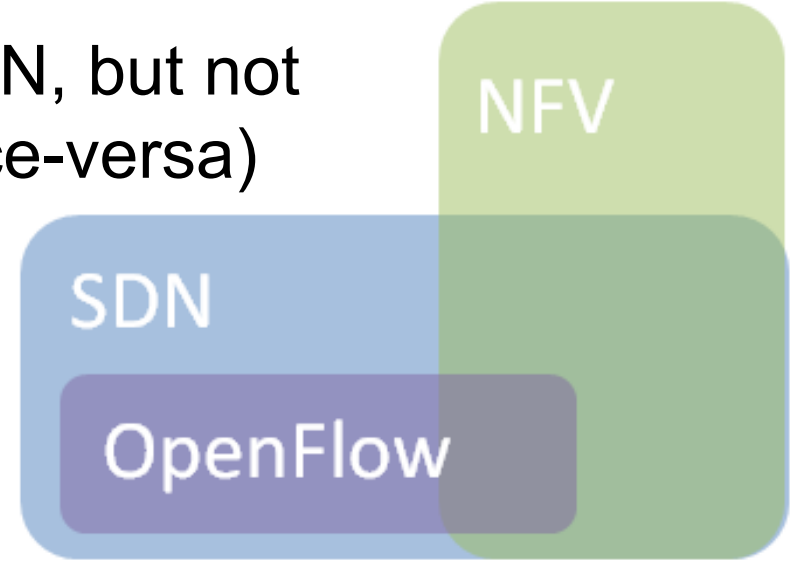
ETSI-NFV

Network Functions Virtualisation ISG (NFV)

ETSI

World Class Standards

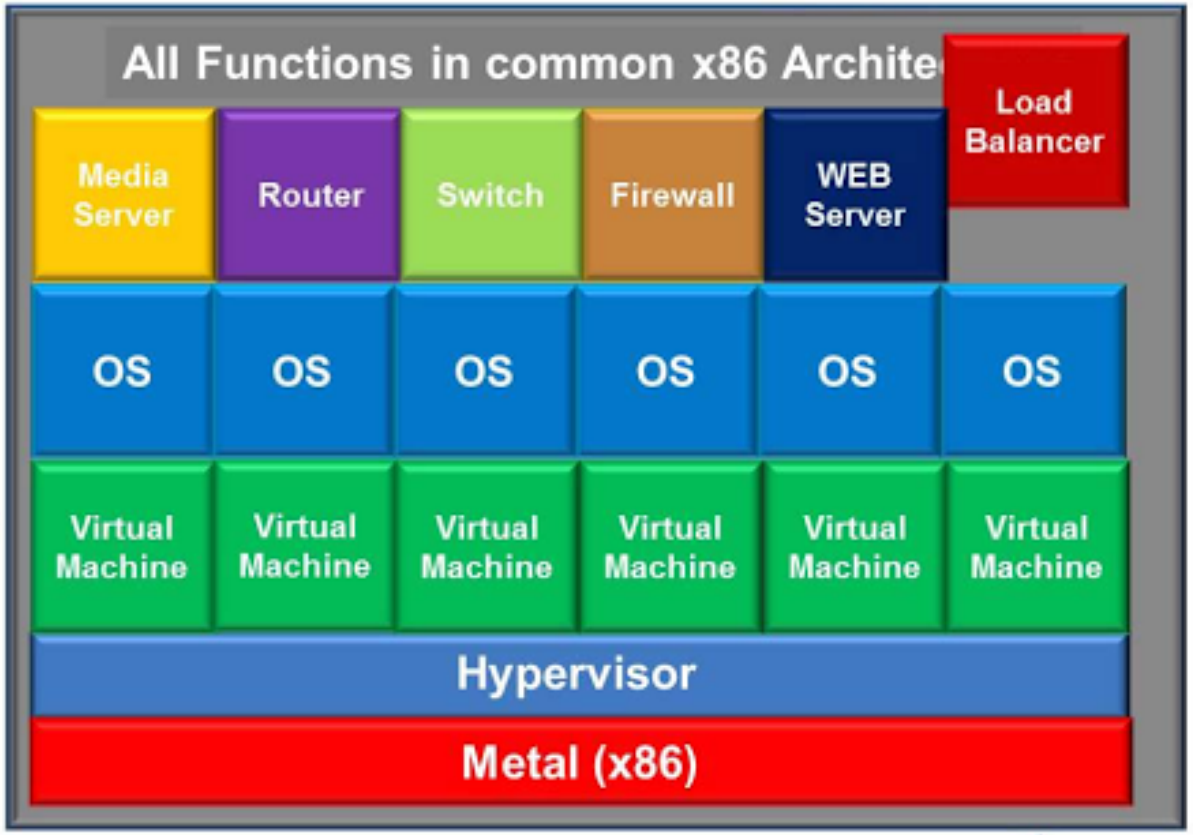
- Goal: Transform the way that operators architect networks by consolidating many network equipment types onto industry standard high volume servers, switches , storage
- Complementary to SDN, but not dependent on it (or vice-versa)



Network Functions Virtualization (NFV)

- Standard Hardware
- Less Complex
- Very Flexible
- Reduced Power
- Lower CapEx
- Lower OpEx
- Test new apps
- Low risk
- Reduced TTM
- Open Market to Software suppliers

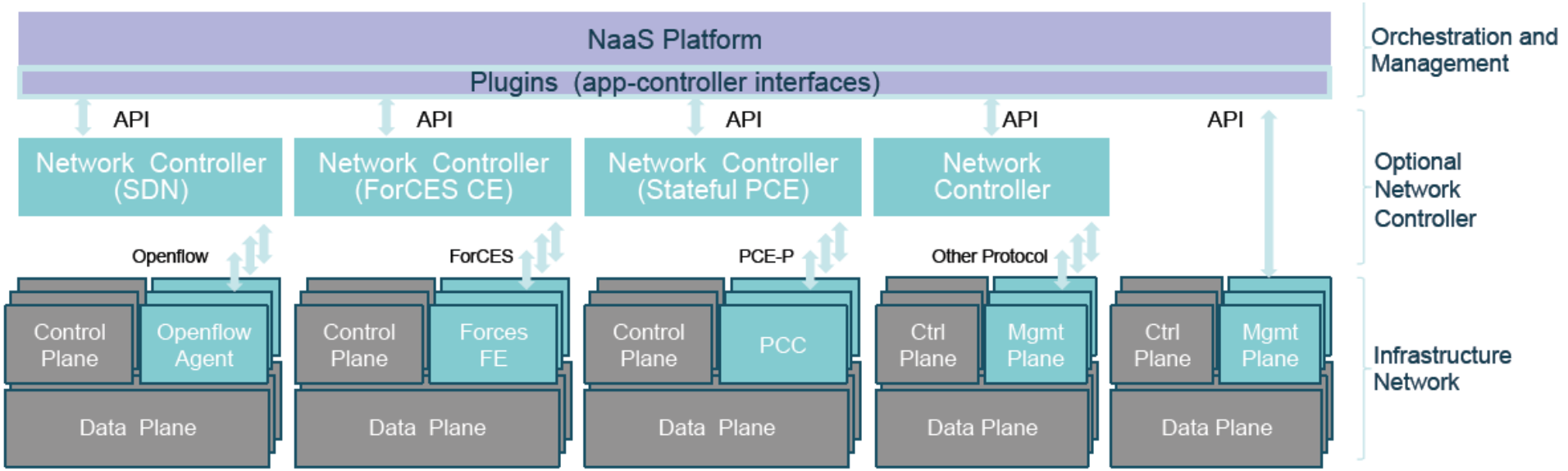
Using Virtualization



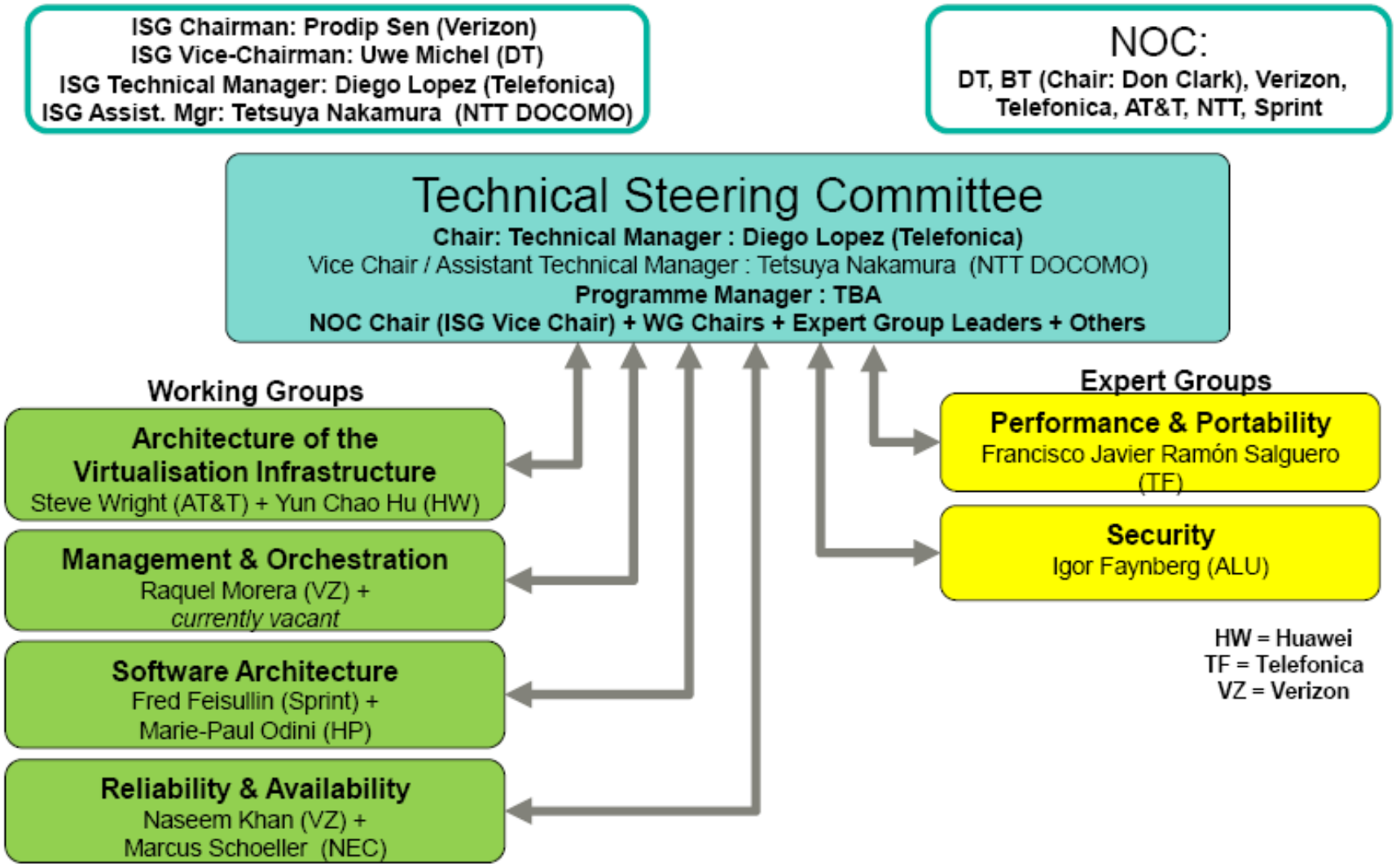
SPIRENT

ETSI-NFV's SDN

- ONF's SDN definition
  - From network operator's viewpoint



ETSI-NFV groups

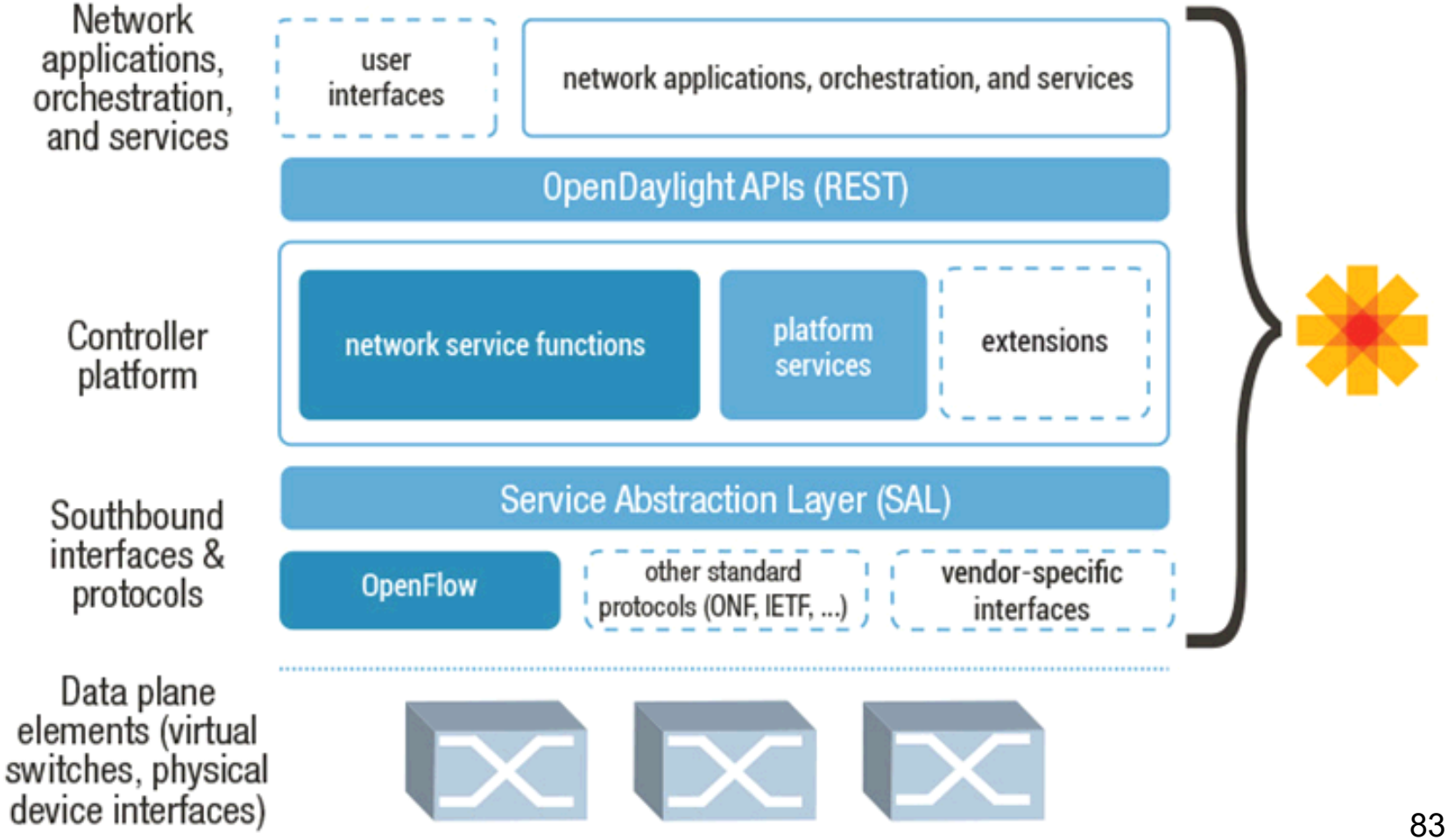


OpenDaylight

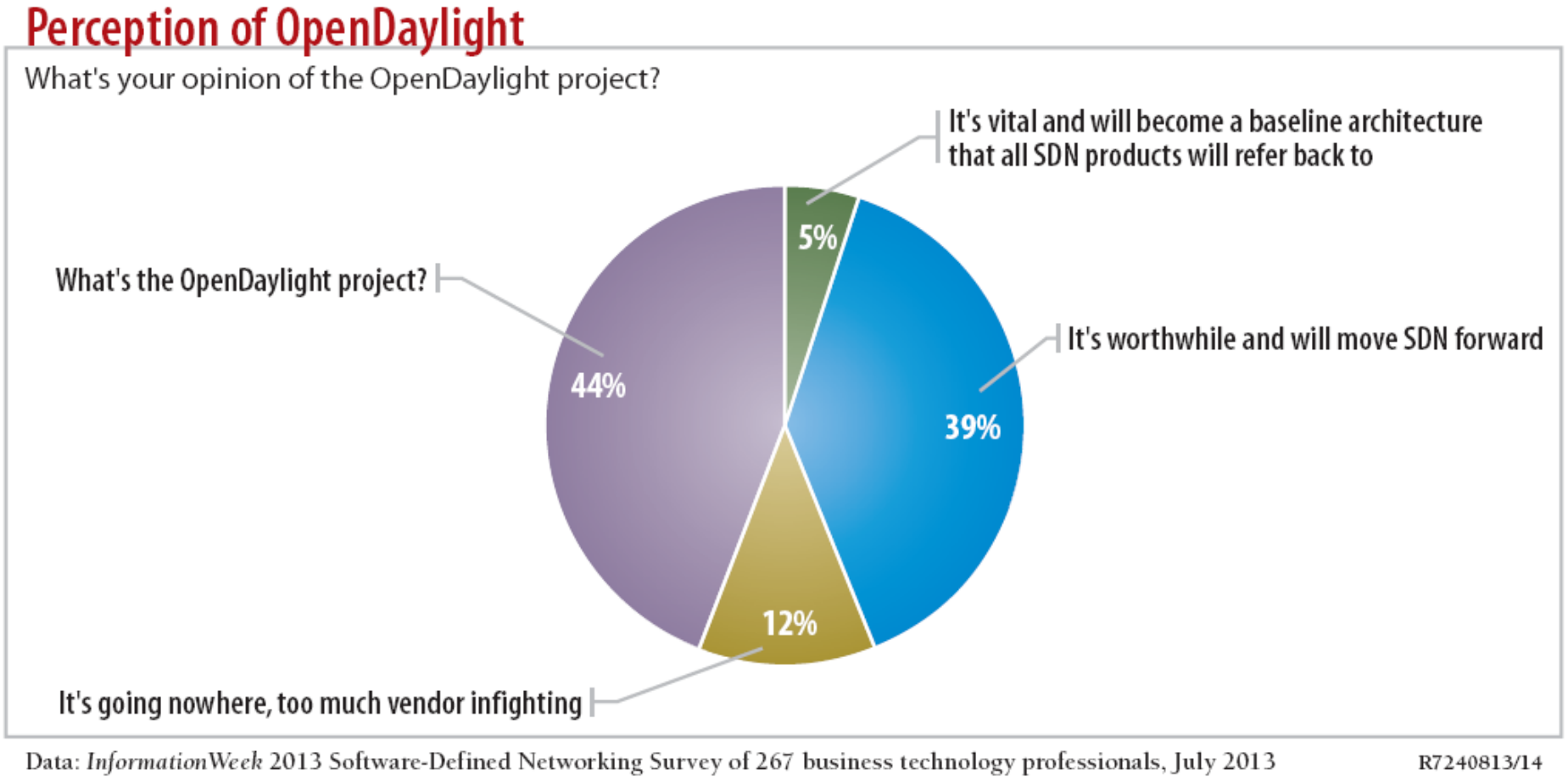
- A Linux Foundation Collaborative Project
- Facilitate a community-led, industry-supported open source framework, including code and architecture, to accelerate and advance a common, robust SDN platform

OpenDaylight's SDN

- OpenDaylight's SDN definition



Why OpenDaylight ?

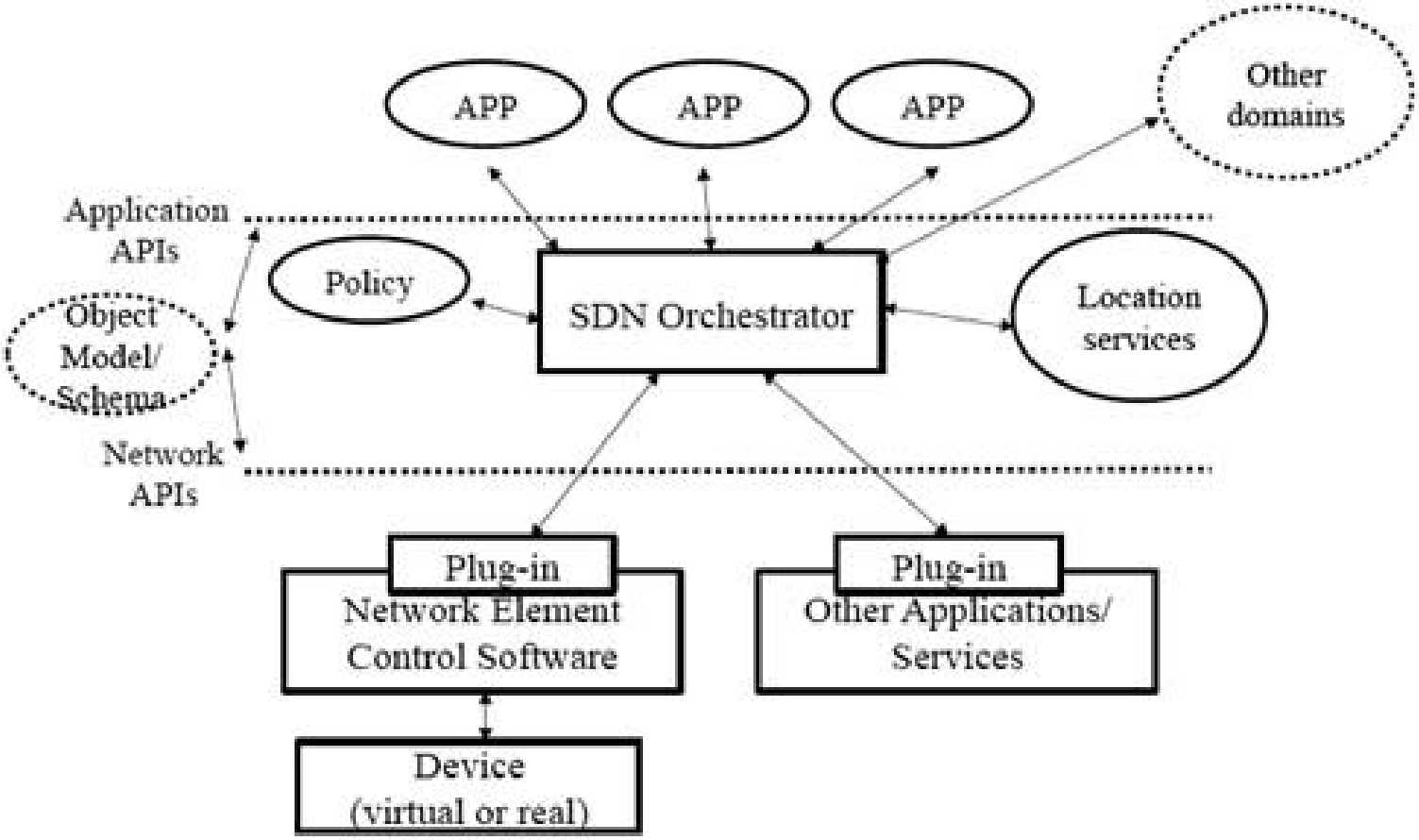


ONF vs. OpenDaylight

组织	派别	成立时间	目的	成员
开放网络基金会(ONF,Open Netwok Foundation)	用户派	2011 年 3 月	OpenFlow 标准和规范的维护和发展，以及推进 SDN 的商业化	德国电信、日本 NTT、Facebook、谷歌、微软、Verizon、雅虎和高盛等 8 家董事会成员和包括华为、腾讯、中国移动等在内的 80 多个会员
OpenDaylight	厂商派	2013 年 4 月	以产业联盟的形式推动 SDN 的产业化	思科、思杰、戴尔、爱立信、富士通、IBM、英特尔、Juniper、微软、NEC、Arista Networks、Big Switch Networks、博科、Nuage Networks、PLUMgrid、红帽和 VMware

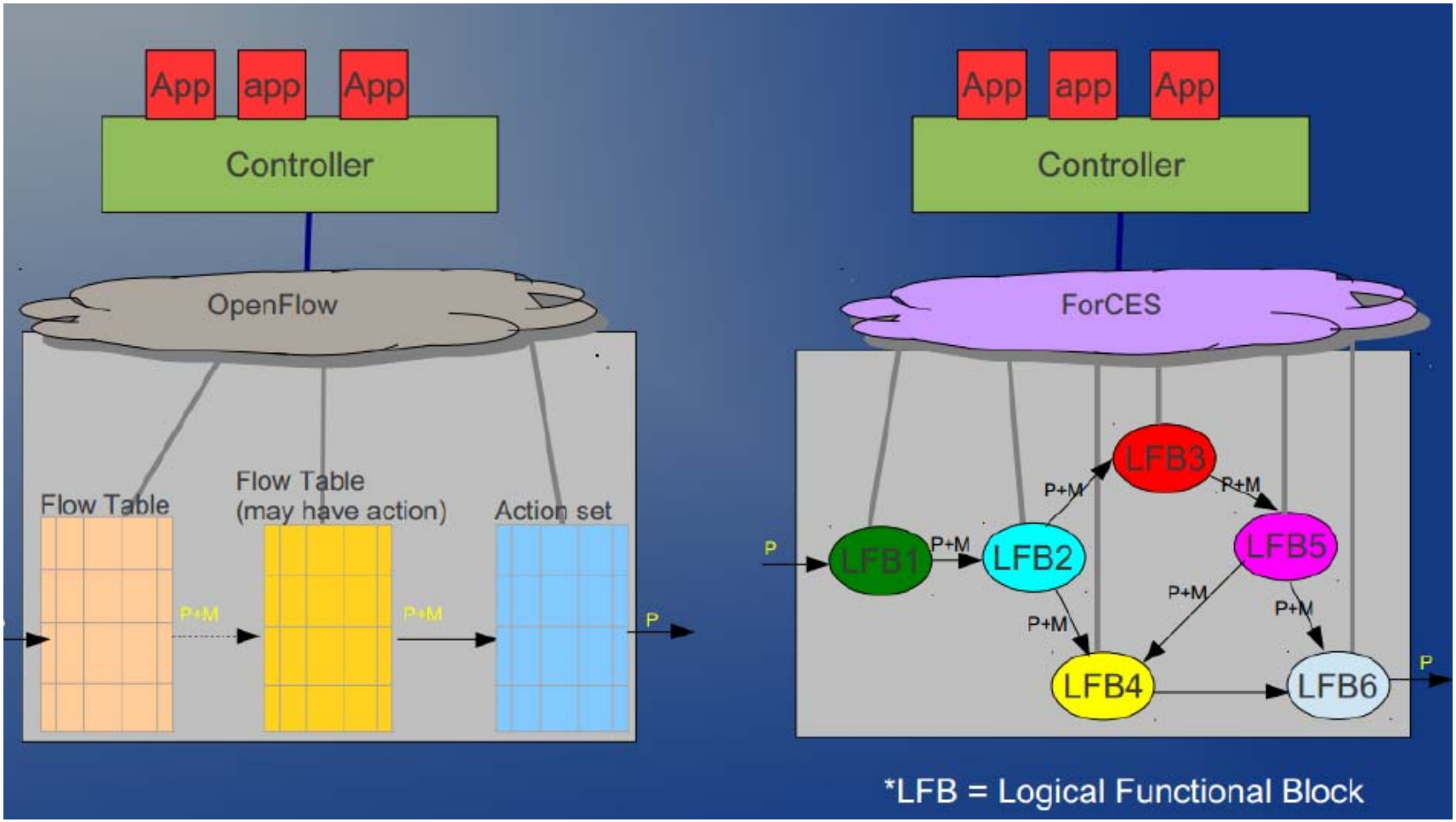
- OpenDaylight 的进展显然没有这么顺利， 其创始人成员Big Switch 对于组织“采用思科控制器作为基本代码库启动这个计划”不满而发出抗议，并声称要退出该组织。
- Big Switch Networks 创始人Appenzeller 称，OpenDaylight 采取的包含思科控制器的方法不是向前发展的正确方法。

IETF-I2RS

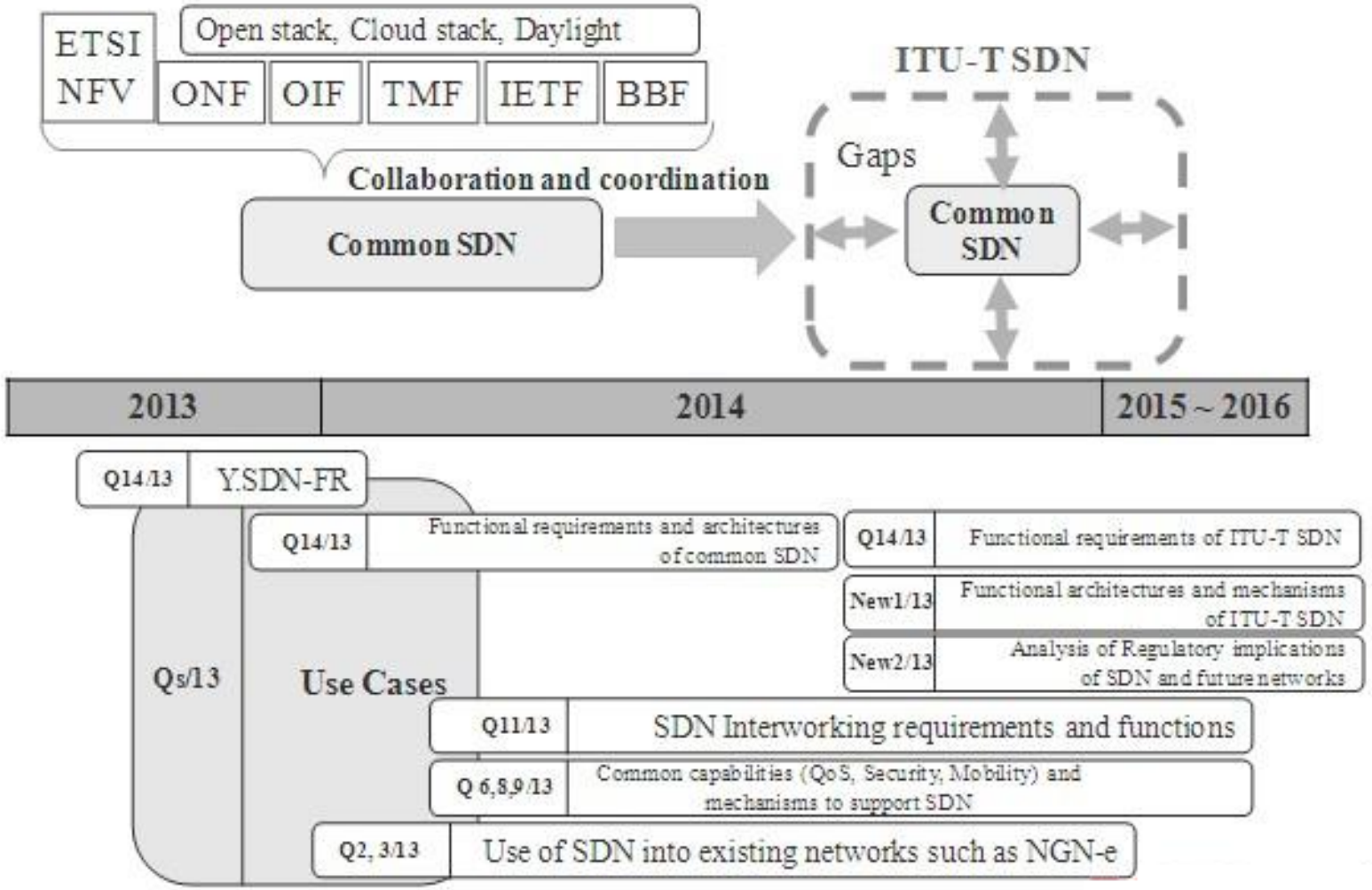

APP
APP
APP
Other domains
Application APIs
Policy
SDN Orchestrator
Location services
Object Model/ Schema
Network APIs
Plug-in
Network Element Control Software
Plug-in
Other Applications/ Services
Device
(virtual or real)

IETF: Other SDN-related activities

- ForCES – Forwarding and Control Element Separation



ITU-T SG13

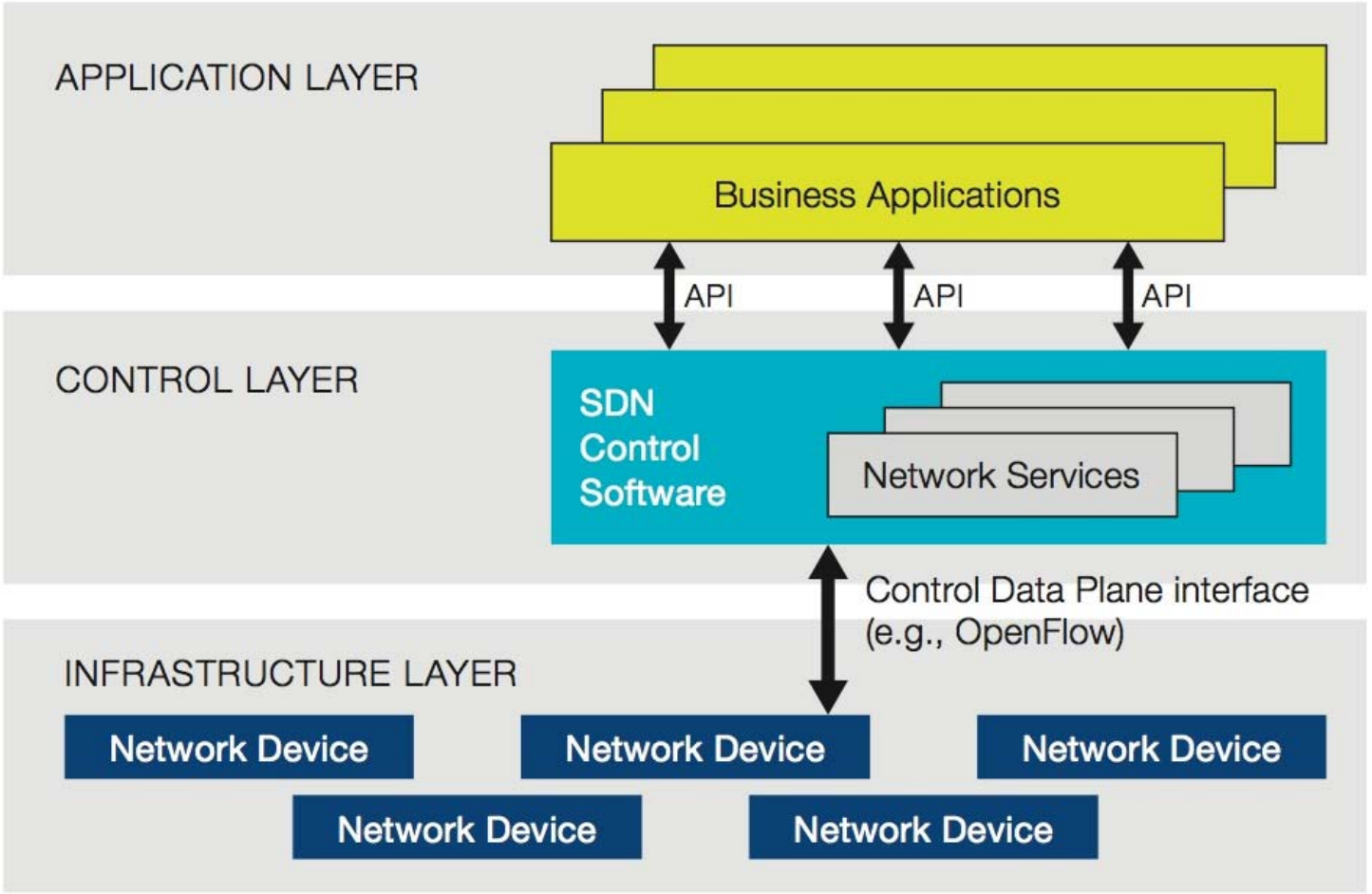

ETSI NFV
Open stack, Cloud stack, Daylight
ONF
OIF
TMF
IETF
BBF
Collaboration and coordination
Common SDN
ITU-T SDN
Gaps
Common SDN
2013
2014
2015 ~ 2016
Q14/13
Y.SDN-FR
Q14/13
Functional requirements and architectures of common SDN
Q14/13
Functional requirements of ITU-T SDN
New1/13
Functional architectures and mechanisms of ITU-T SDN
New2/13
Analysis of Regulatory implications of SDN and future networks
Qs/13
Use Cases
Q11/13
SDN Interworking requirements and functions
Q 6,8,9/13
Common capabilities (QoS, Security, Mobility) and mechanisms to support SDN
Q2, 3/13
Use of SDN into existing networks such as NGN-e

Today's agenda

- Introduction to Software Defined Networking
  - What is SDN? Why SDN?
  - History of SDN
  - SDN standardization
  - SDN deployment models
  - OpenFlow
  - SDN applications
  - SDN development tools
  - Future directions

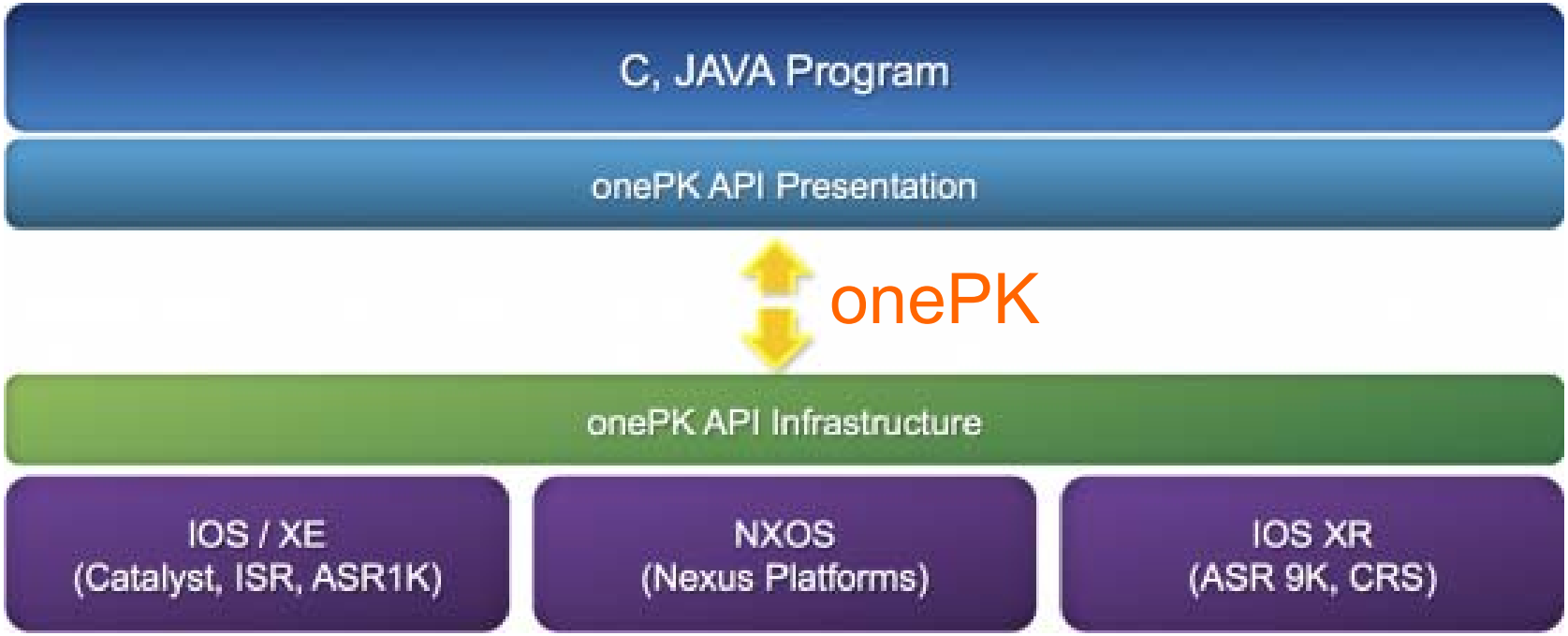
Open API based solution

- Example: OpenFlow



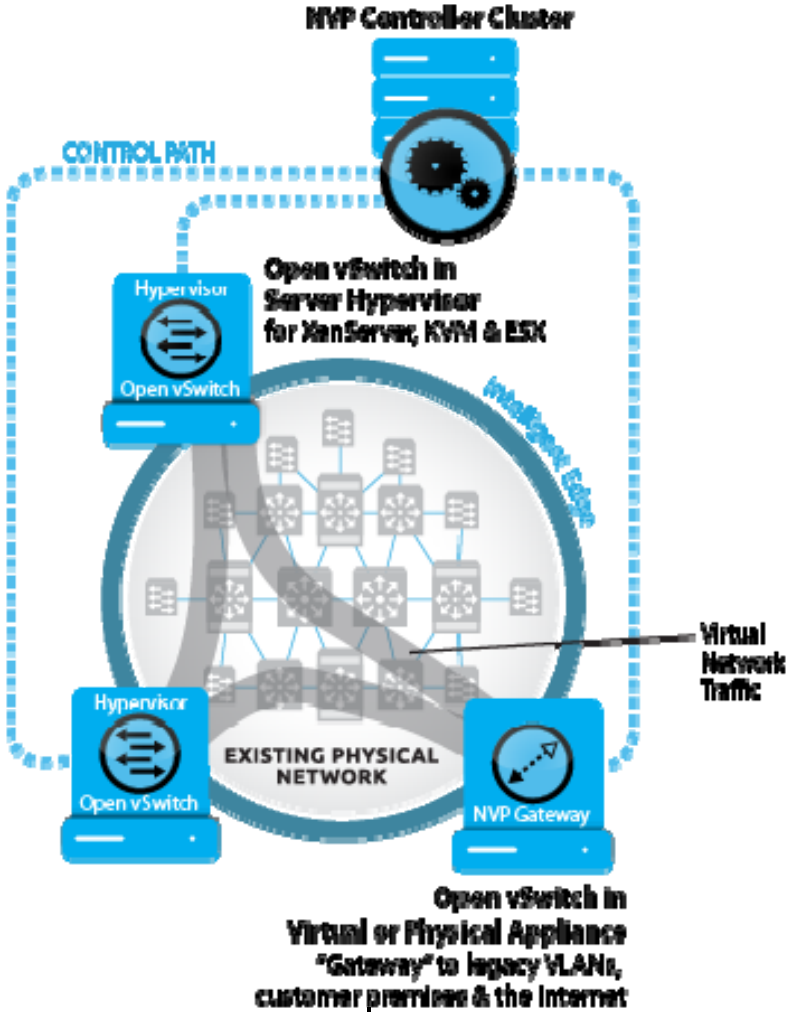
Closed API based solution

- Example: Cisco's onePK (ONE Platform Kit)

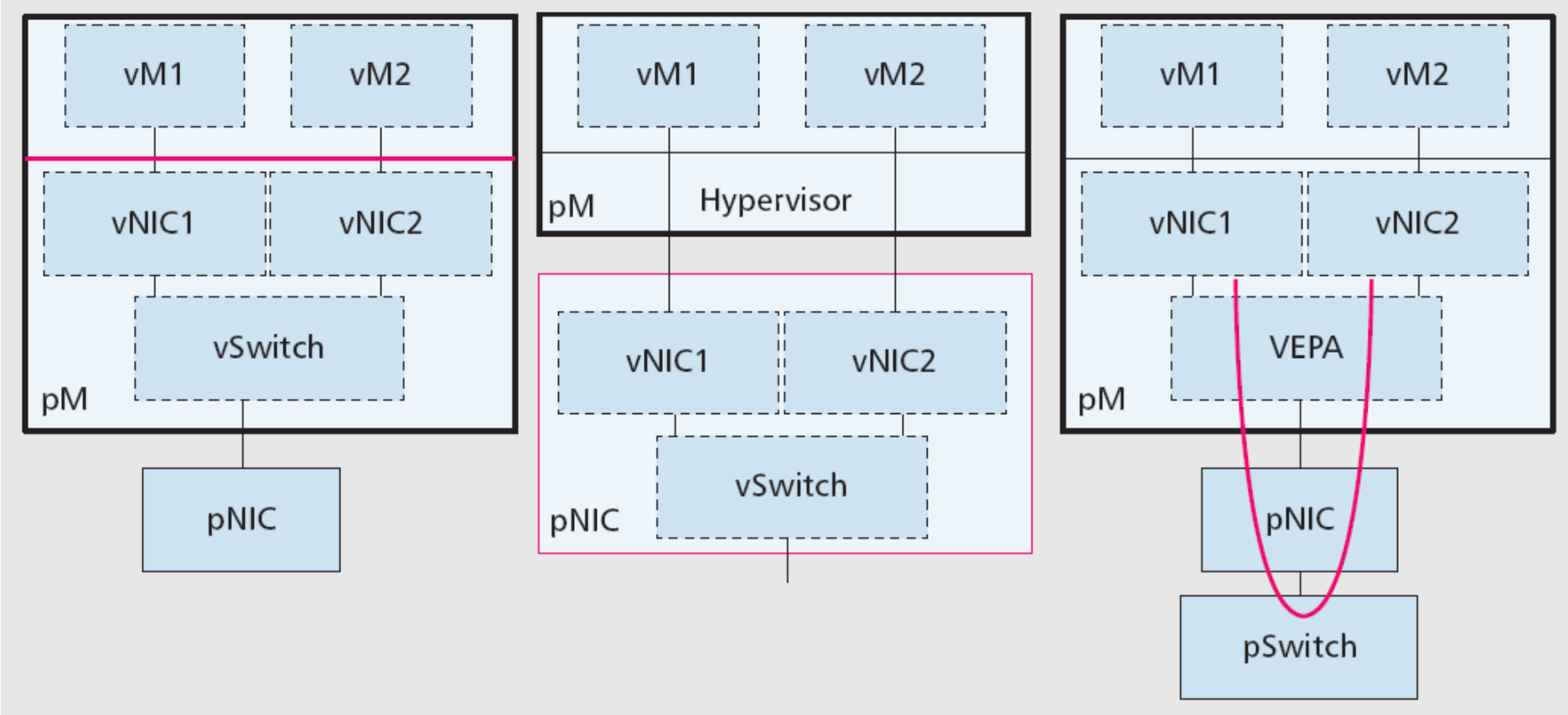


Overlay based solution

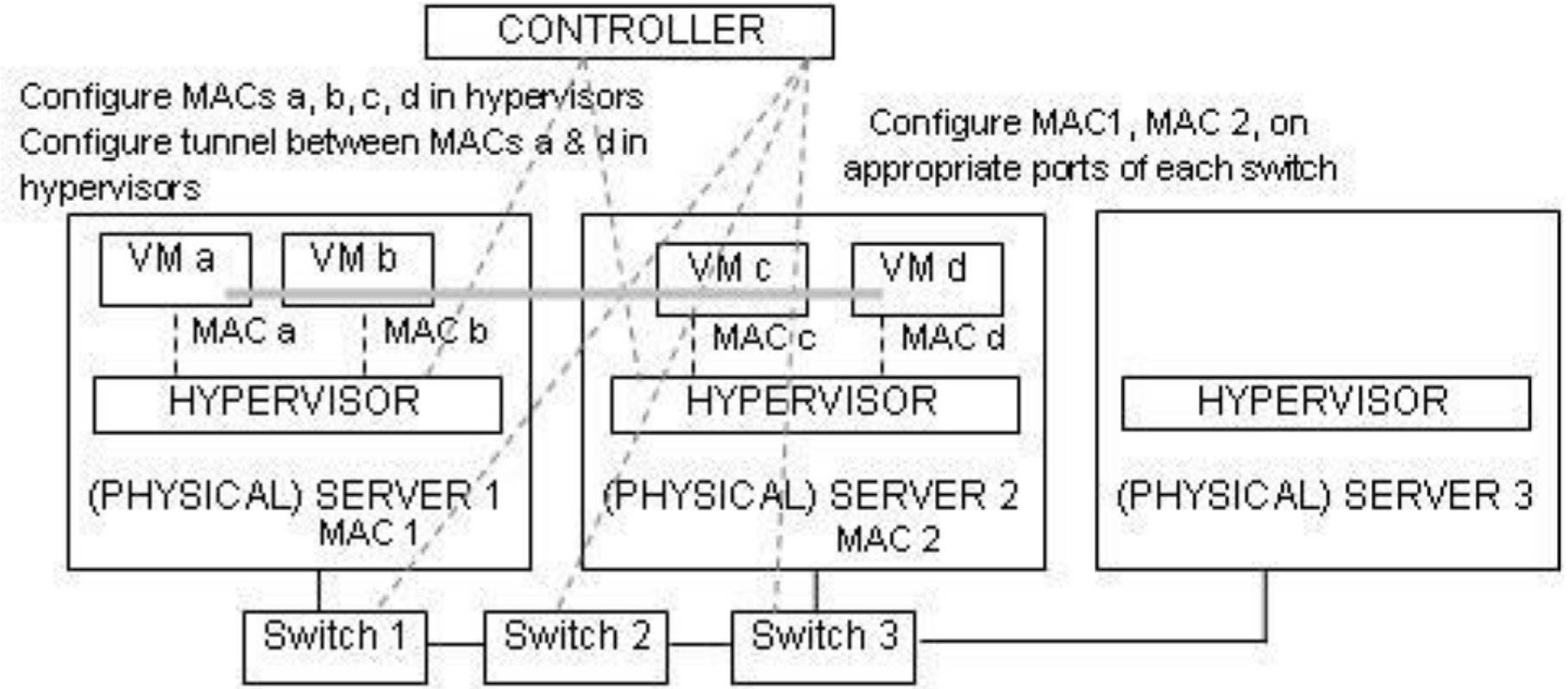
- Example: Nicira's NVP (Network Virtualization Platform)



3 approaches for NIC virtualization



Overlay based solution



Overlay based solution