OpenFlow & History of Software Defined Networks (SDN)

Alexander Raistrick, Beining Han, Karhan Kayan, Xindi Wu

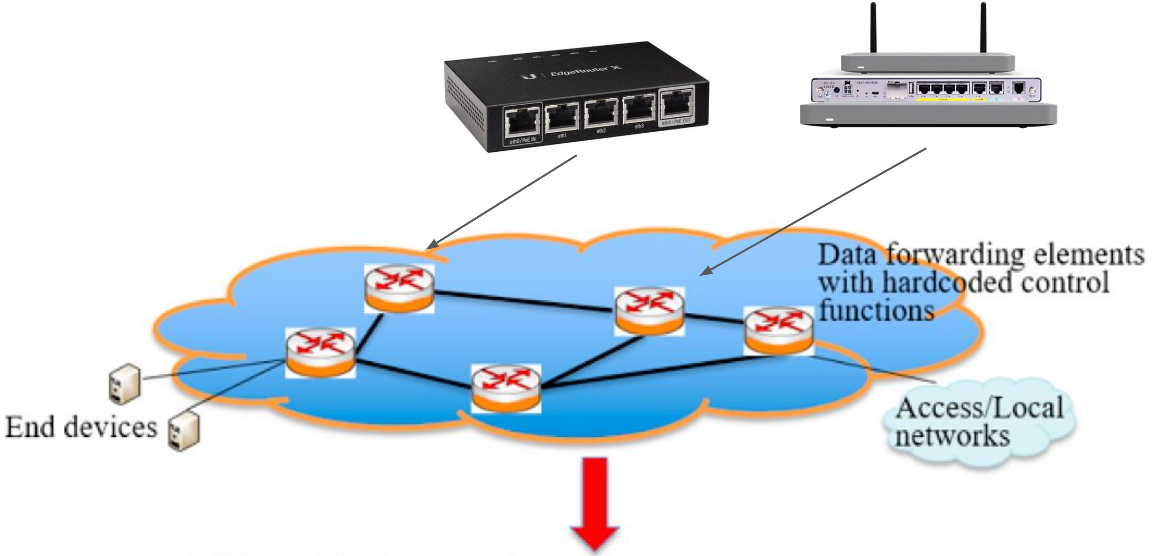
COS 561 Paper Presentation - Mar 3 2023

Outline

- Intro to SDN
- Active Networks
  - Capsule based
  - Router based
- Separating Control & Data Planes
  - Ethane
- OpenFlow
  - Motivation, why companies aren't going to help, what is the flow table, how is it used
  - OpenFlow Deployment
- Network Virtualization
  - GENI and PlanetLab

Intro: Network before SDN

- Different softwares/configuration interfaces for routers/switches
- Costly and complex to update the protocol.
- Headache for network administrators.

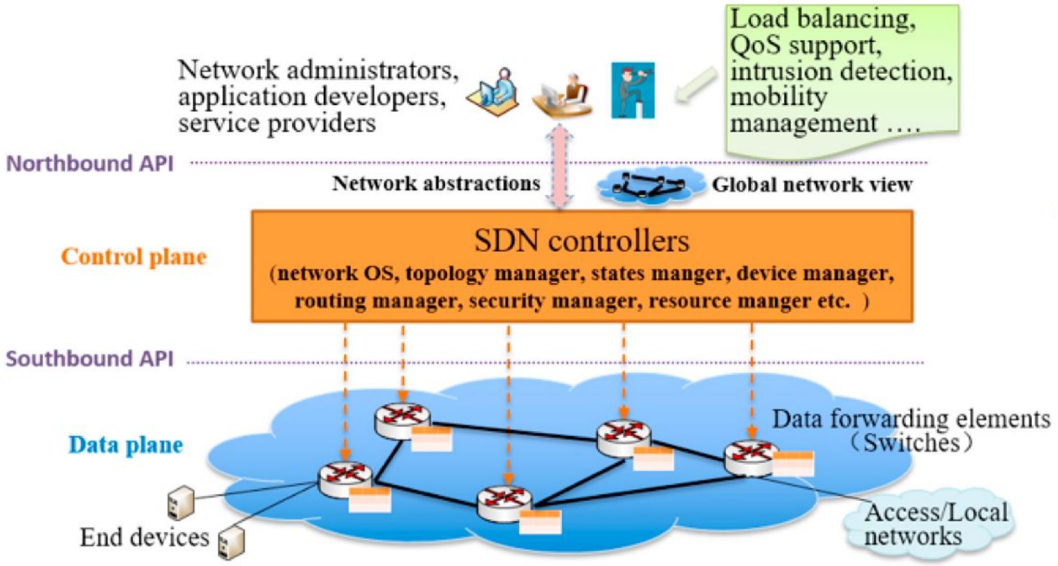


(a) Non-SDN network

https://www.mdpi.com/1424-8220/20/12/3459

Intro: Software-Defined Networking

- Separate Data Plane & Control Plane.

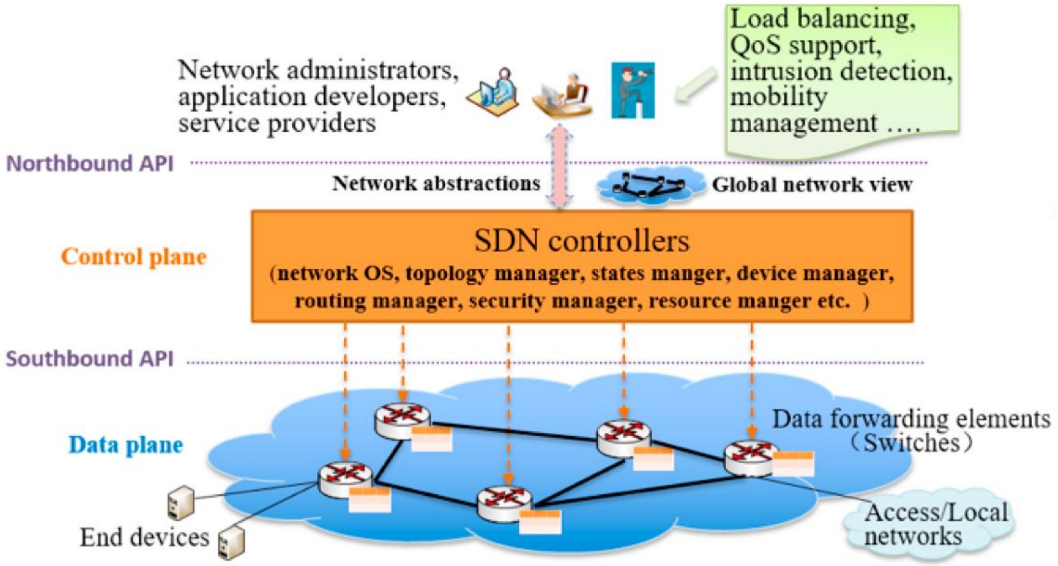


(b) SDN network

Intro: Software-Defined Networking

- Consolidate control plane.
- A single software.

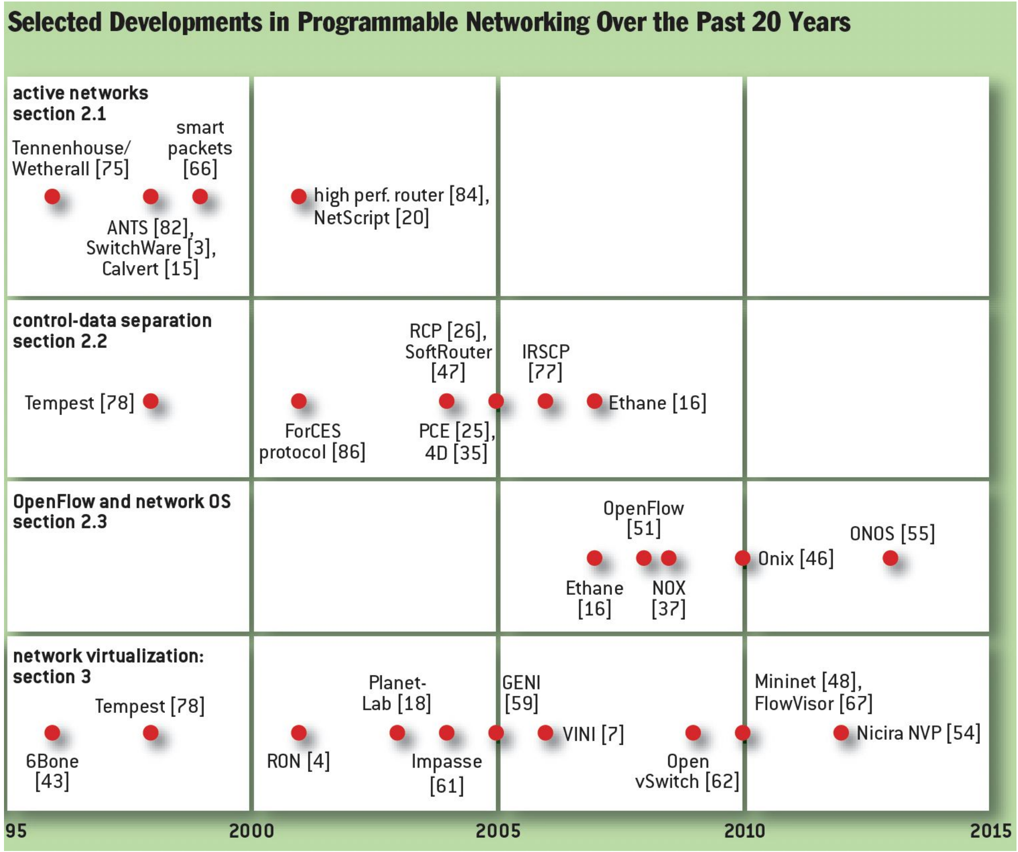
OpenFlow



(b) SDN network

The Roadmap to SDN

- Active Network (90-00)
  - Programmable Functions
- Control/Data Separation (01-07)
- OpenFlow & Net OS (07-10)
  - Scalable and Practical Interface
- Network Virtualization
  - Use case of SDN



The Need for Programmable Networks

- Not going to happen from the vendor’s side
- Open platforms exist but lack performance / port density

Active Networks - Motivation

- Use Pull
  - The huge cost to test and deploy new network services with hard-coded hardwares.
  - Interest to dynamically control the network for particular applications.
- Technology Push
  - Cost reduction in computation.
  - Advancement in Programming Language.
- High-level Idea
  - Open-up network control to users by a programming interface that exposes resources (e.g., processing, storage) and supports custom functionality to apply to packets.

Active Networks - Methods

- Capsule Model: executable codes are added in the packet.
- D. J. Wetherall, J. V. Guttag and D. L. Tennenhouse, "ANTS: a toolkit for building and dynamically deploying network protocols," *1998 IEEE Open Architectures and Network Programming*, San Francisco, CA, USA, 1998,
    - Packets are processed on each active nodes. It can use the environemnt access, manipulate the capsule control the capsule (add or delete), or store on the nodes.
    - Also supports code distribution, which caches the code on nodes.

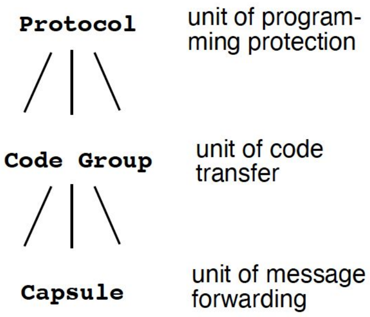


Figure 1: Capsule Composition Hierarchy

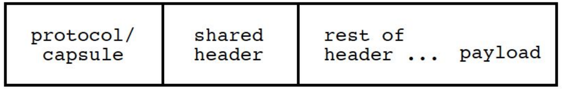


Figure 2: Capsule Format

Active Networks - Methods

- Programmable routers: code to execute are established ahead.
- Nodes contain an input queue, an output queue, state information, and a set of parameterized functions to call.

Remove a packet M from the input queue;
while (more functions need to be applied to M):
    Let f, p be the function ID and parameter from the next header of M;
    Let g be the state component relevant to f and p;
    Invoke function f on M, with p as parameter:
        (optionally) Modify M;
        (optionally) Update g;
        (optionally) Queue messages for output;

Bhattacharjee, Samrat, Kenneth L. Calvert, and Ellen W. Zegura. "An architecture for active networking." *High Performance Networking VII: IFIP TC6 Seventh International Conference on High Performance Networks (HPN '97), 28th April–2nd May 1997, White Plains, New York, USA*. Springer US, 1997.

Active Networks - Pragmatism

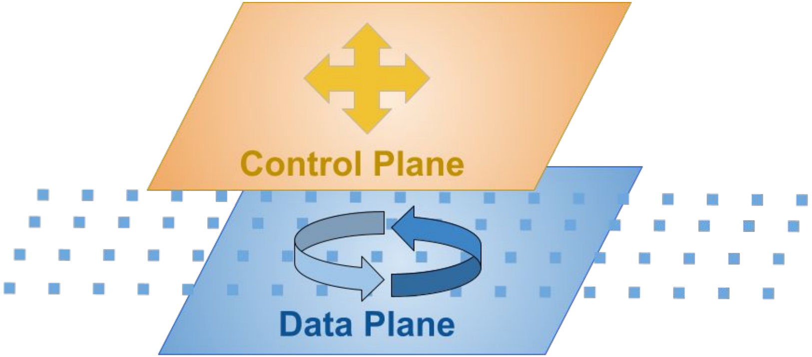
- Inherent unsafe to carry user-defined code.
- Low throughput with user-defined code.
- The design can offer various applications but none of them offer a compelling solution.

Active Networks - Contribution to SDN

- Programmability in the network. Many early work in SDN focus on control-plane programmability.
- Network virtualization and packet demultiplexing.
- Lessons for middlebox orchestration.

Separating the Control and Data Planes

- **Control Plane:** Logic for controlling the forwarding behavior.
  - Routing protocols, middlebox configuration, etc.
- **Data Plane:** Forward traffic according to the control plane.
  - IP forwarding, Layer 2 switching, etc.

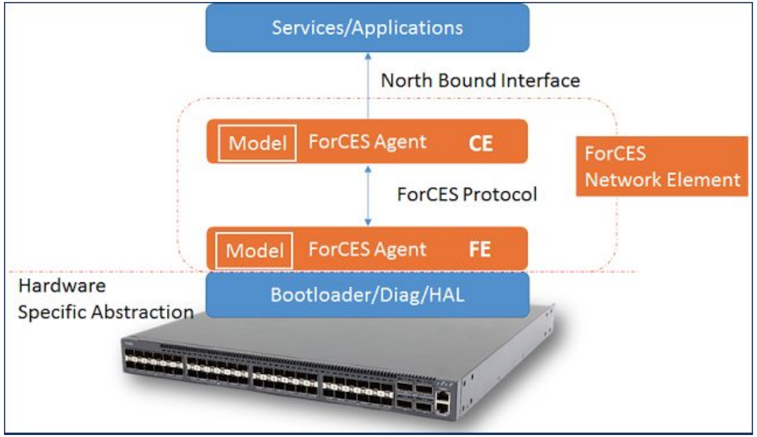


Separating the Control and Data Planes - Motivation

- Need for:
  - Easier debugging of configuration problems
  - Easier prediction and control of routing behavior
- Independent development and evolution of software and hardware.
- Better network management.
  - Innovation by and for network admins.
  - Programmable control planes.
  - Network-wide visibility and control.
  - Required external control planes.

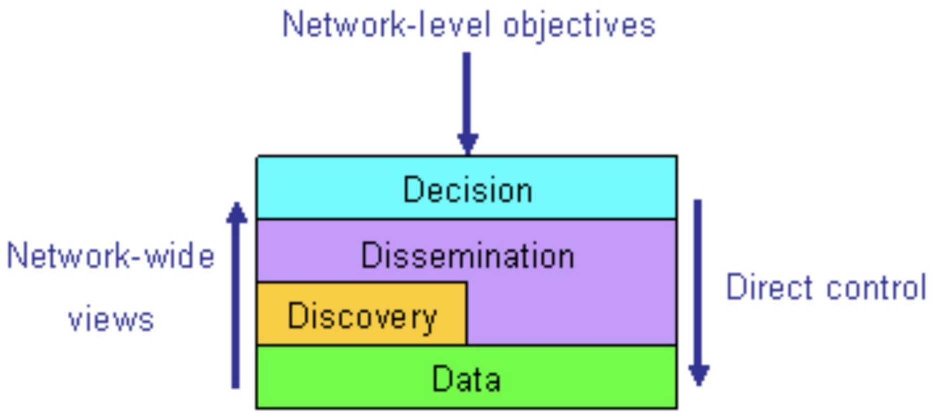
Separating the Control and Data Planes - ForCES ('04)

- Stands for Forwarding and Control Element Separation.
- Defined an open interface between logically separate control and data planes.
- Allows both:
  - Keeping control/data planes in the same device
  - Using an external centralized control plane (SoftRouter)



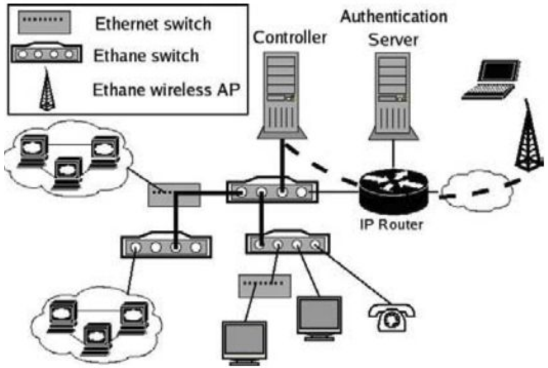
Separating the Control and Data Planes - 4D Project

- Proposed four planes:
  - Data plane (processing packets)
  - Discovery plane (measurement info)
  - Dissemination plane (installing packet processing rules)
  - Decision plane (centralized controllers)



Separating the Control and Data Planes - Ethane ('07)

- Centralized access control in enterprise networks.
- Manages admittance and routing of flows.
- Enforces network-wide security policies by populating Ethernet switch flow tables.
- Deployed in the Stanford CS department network.
- Gave rise to OpenFlow.

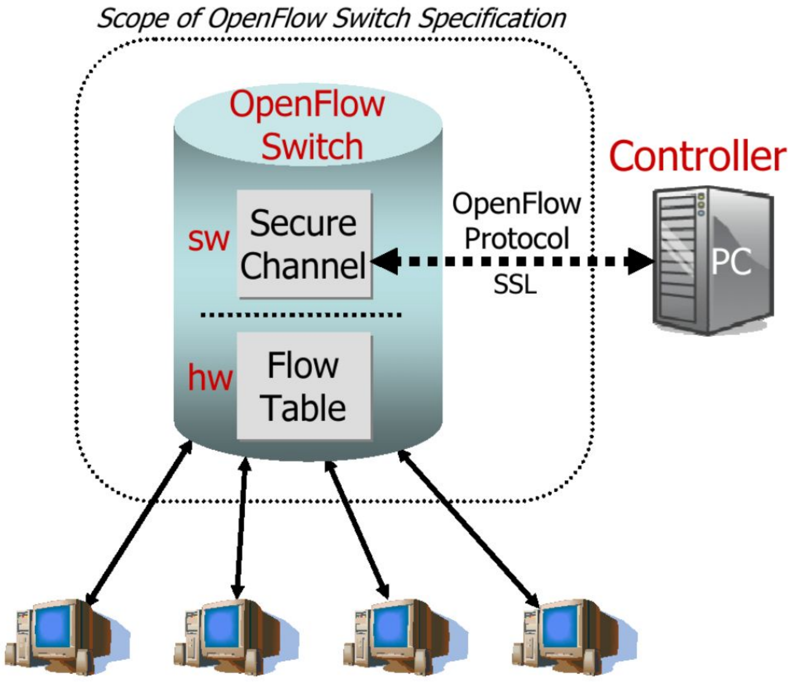


OpenFlow: Enabling Innovation in Campus Networks

Basic idea: exploit capabilities of *Flow Tables*

Switches already "contain *Flow Tables* built from TCAMs that run at line-rate"

OpenFlow exploits the capabilities common across switches / vendors, and provides an open protocol to program them.



Flow Tables in Practice

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Count …
*	*	10.1.1.1	10.2.1.5	80	*	Drop	
*	*	192.*	10.2.4.*	*	*	Port 2	
*	*	*	*	*	*	Drop	

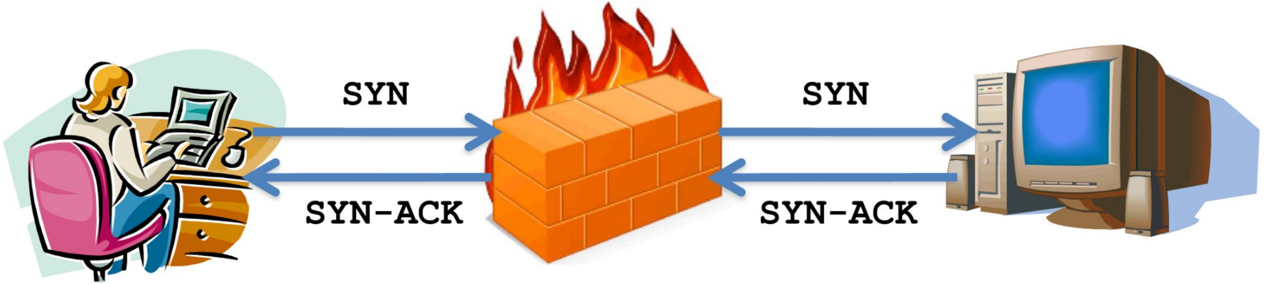


Image from COS461, Flow Table example from "What OpenFlow Tables Can Do" - Ivan Pepelnjak

Flow Tables in Practice

In Port	VLAN ID	Ethernet			IP			TCP		Action	Count …
		SA	DA	Type	SA	DA	Proto	Src	Dst		
*	*		10.1.1.1	10.2.1.*		80		*		Port 3	300
*	*		192.*	10.2.4.*		*		*		Port 2	800
*	*		*	*		*		*		Controller	

Image from COS461, Flow Table example from “What OpenFlow Tables Can Do” - Ivan Pepelnjak

OpenFlow

- Commercial switches are unlikely to open up their internal flexibility
- Internals are nonstandard, but researchers need a standard platform to develop
- Research friendly solutions (NetFPGA) don’t have enough fanout for a real college network.

We want a degree of switch flexibility that is:

- Practical, with high-performance low cost implementations
- Capable of supporting a broad range of research
- Assured to isolate experimental vs production traffic
- Consistent with vendor’s closed platforms

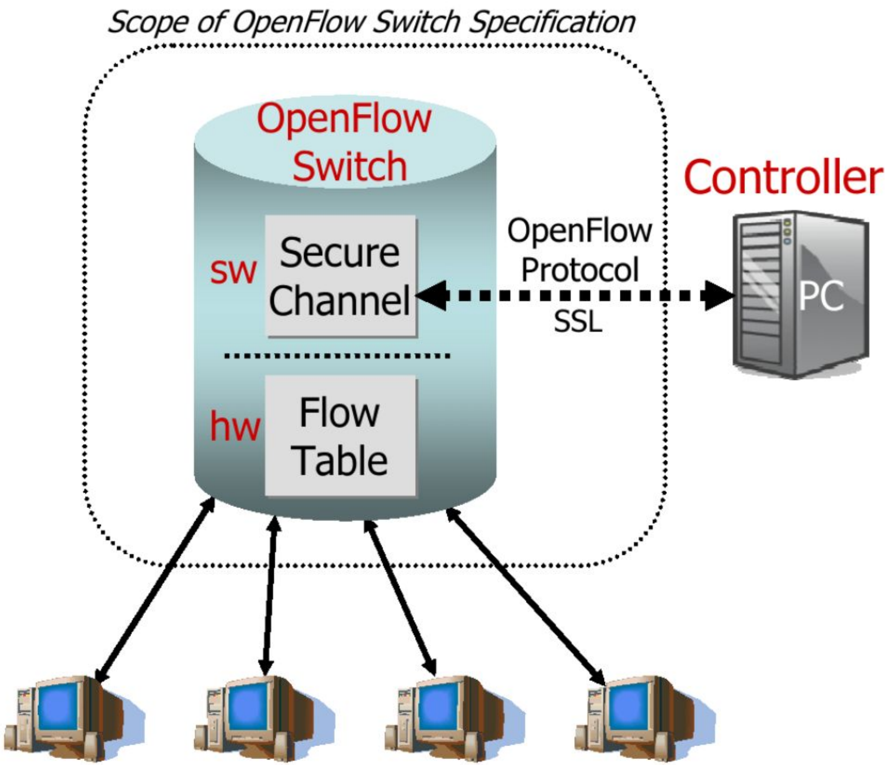
Dedicated OpenFlow Switch

In Port	VLAN ID	Ethernet			IP			TCP	
		SA	DA	Type	SA	DA	Proto	Src	Dst

Basic actions required:

1. Forward to Port X
2. Forward to Controller
3. Drop

Stats: Count, Bytes, Time of last match



OpenFlow-enabled switch

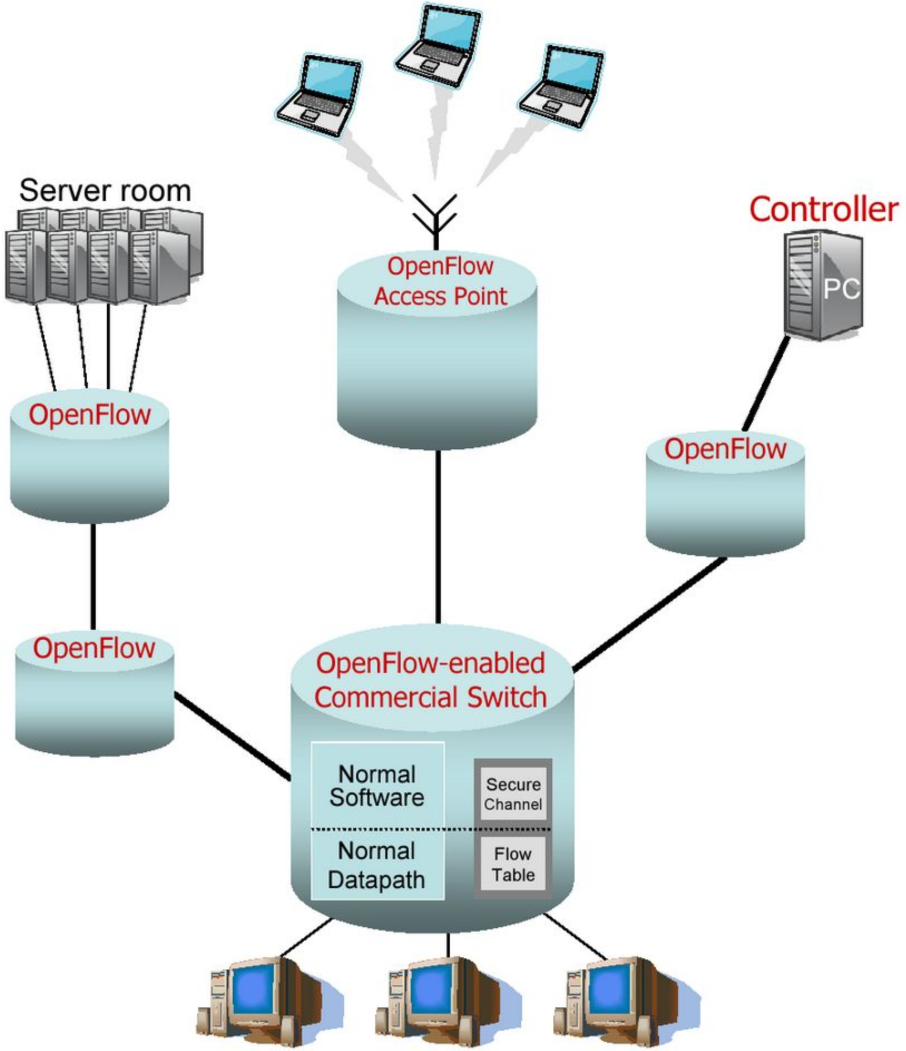
In Port	VLAN ID	Ethernet			IP			TCP	
		SA	DA	Type	SA	DA	Proto	Src	Dst

Basic actions required:

1. Forward to Port X
2. Forward to Controller
3. Drop
4. Forward via normal packet processing

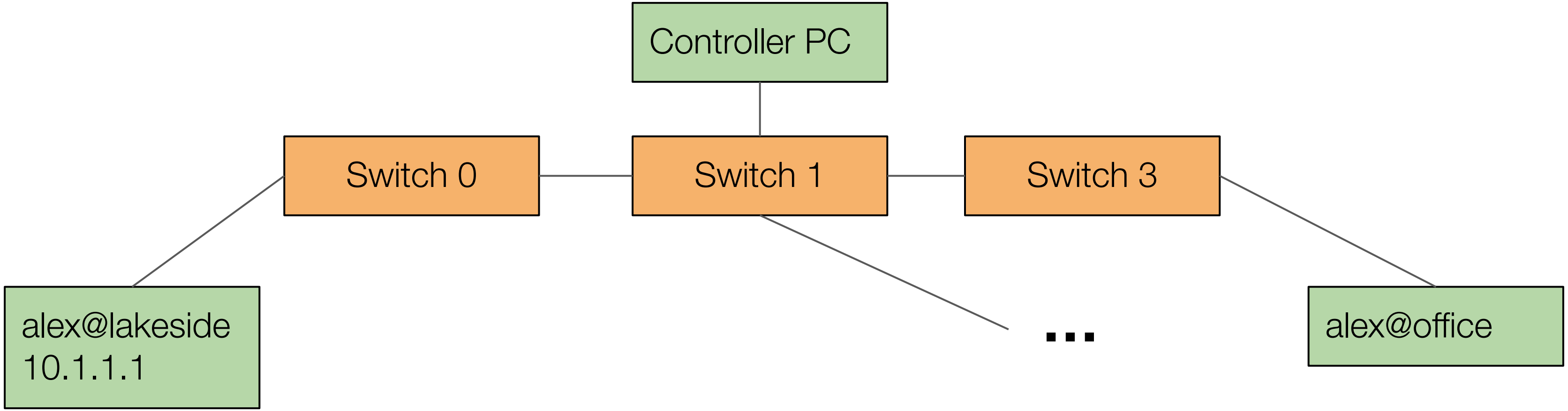
Called a *Type 0* switch.

Type 1 was TBD



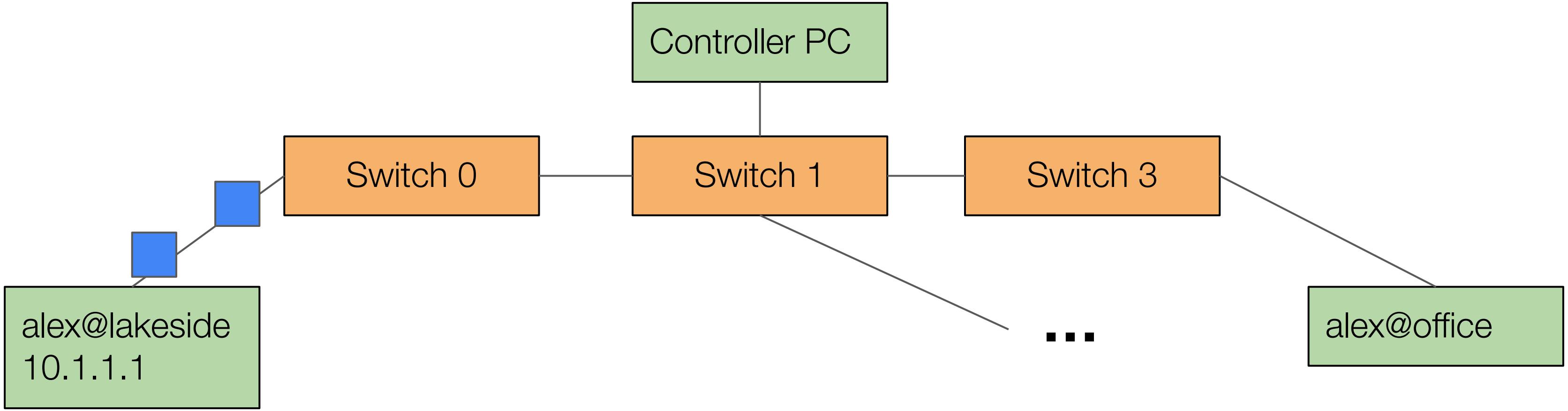
Practical use case of OpenFlow

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Stats
*	*	10.1.1.1	*	*	*	Controller	0
*	*	*	*	*	*	Forward Normally	50000



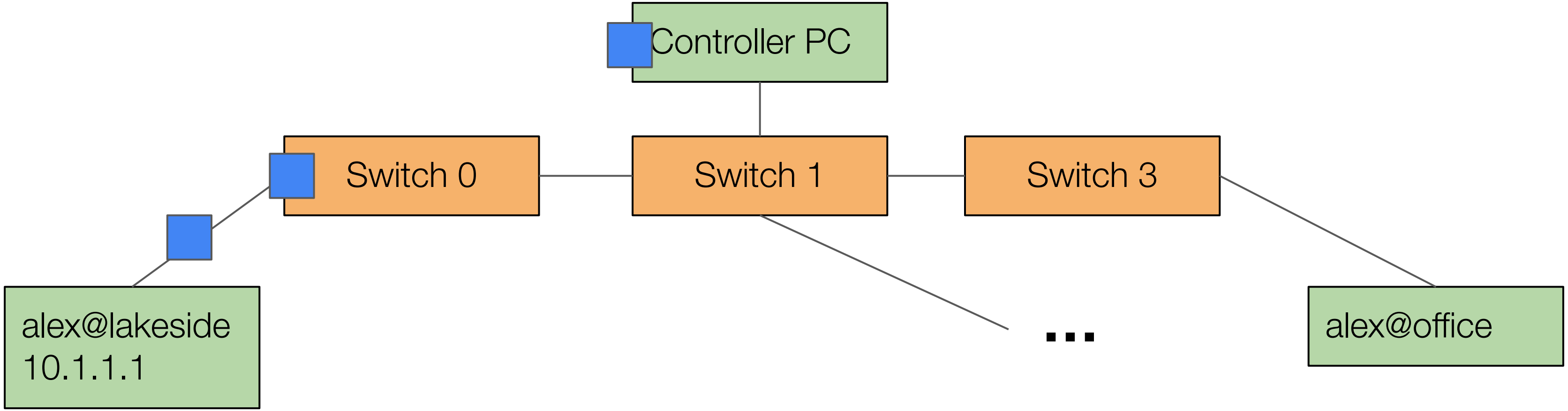
Practical use case of OpenFlow

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Stats
*	*	10.1.1.1	*	*	*	Controller	0
*	*	*	*	*	*	Forward Normally	50000



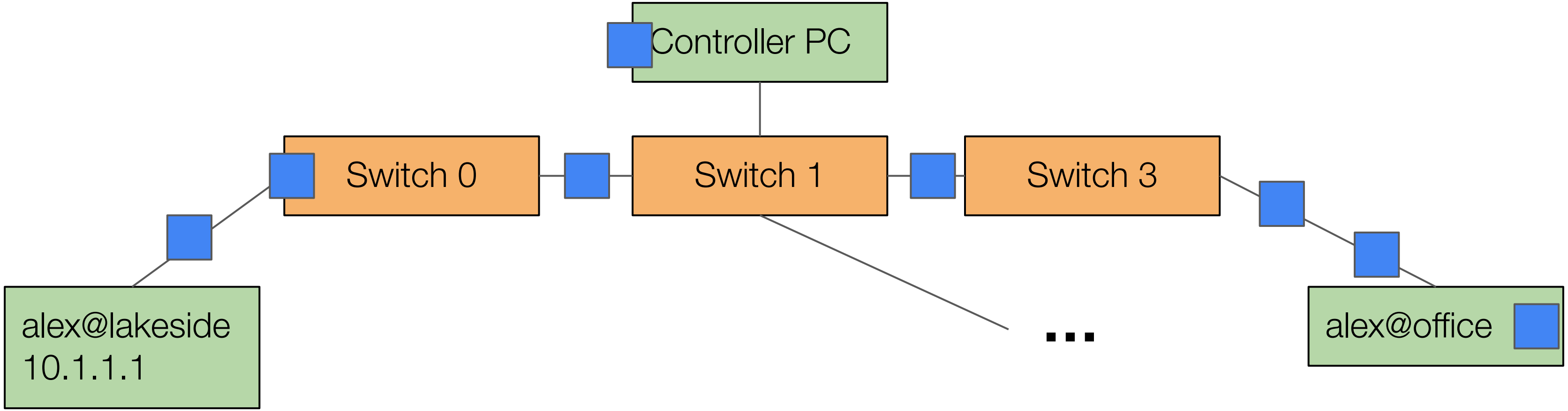
Practical use case of OpenFlow

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Stats
*	*	10.1.1.1	*	*	*	Port X	0
*	*	*	*	*	*	Forward Normally	50000



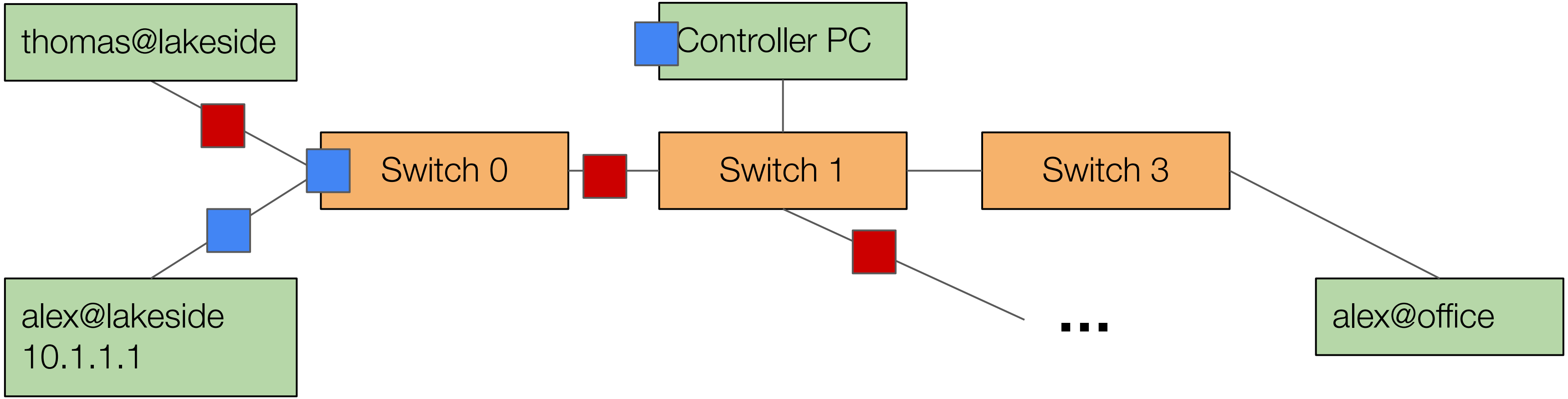
Practical use case of OpenFlow

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Stats
*	*	10.1.1.1	*	*	*	Port X	0
*	*	*	*	*	*	Forward Normally	50000



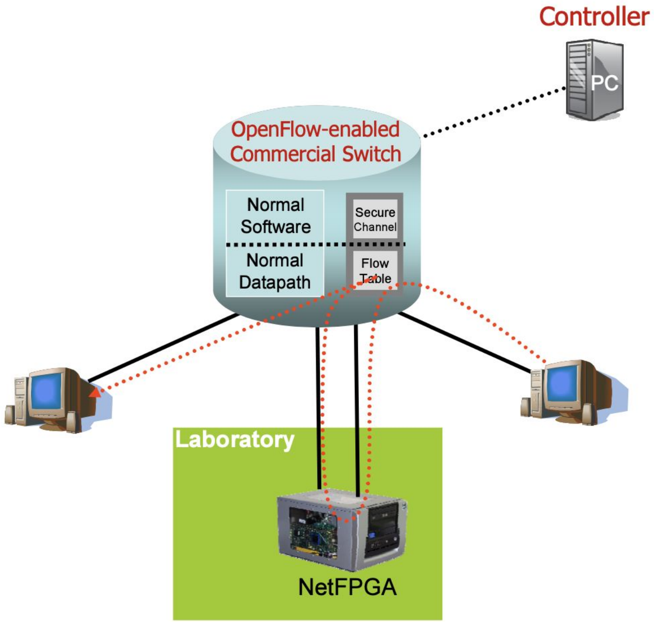
Practical use case of OpenFlow

MAC SRC	MAC DST	IP SRC	IP DST	TCP Dest	TCP Source	Action	Stats
*	*	10.1.1.1	*	*	*	Port X	0
*	*	*	*	*	*	Forward Normally	50000



Using OpenFlow

Example of processing packets through an external line-rate packet-processing device, such as a programmable NetFPGA router.



Using OpenFlow | Production Network

Network Management and Access Control

VLANs

Mobile wireless VOIP clients

A non-IP network

Processing packets rather than flows

Using OpenFlow | Production Network

Network Management and Access Control

VLANs

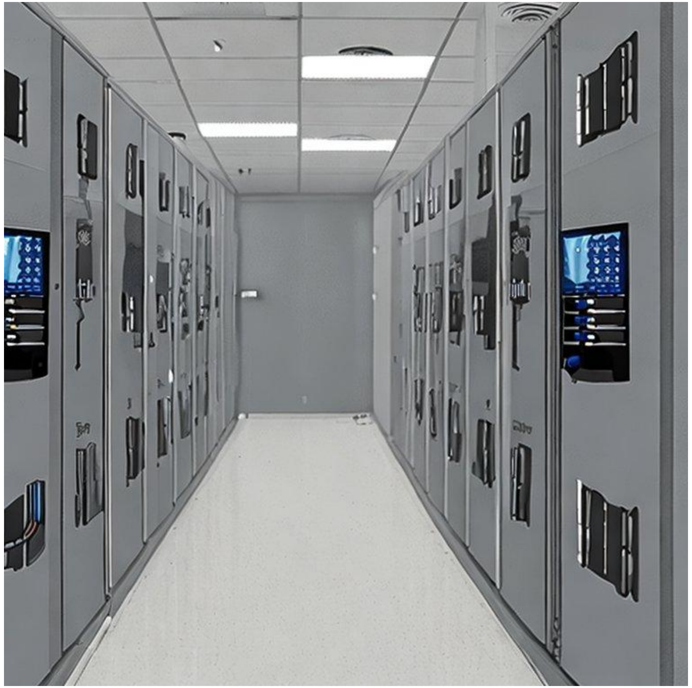
Mobile wireless VOIP clients

A non-IP network

Processing packets rather than flows

Using OpenFlow | Network Management and Access Control

Ethane uses a specific controller implementation to manage flow admittance and routing by allowing network managers to define a network-wide policy that the controller enforces through admission control decisions for each new flow.



Martin Casado, Michael J. Freedman, Justin Pettit, Jianying Luo, Nick McKeown, Scott Shenker. “Ethane: Taking Control of the Enterprise,” ACM SIGCOMM ’07, August 2007, Kyoto, Japan.

Using OpenFlow | VLANS

- OpenFlow provides isolated networks for users similar to VLANs.
- Static flow declaration can specify accessible ports on a VLAN ID and tag traffic from a single user.
- A dynamic approach involves a controller managing user authentication and location information to tag traffic at runtime.

Martin Casado, Michael J. Freedman, Justin Pettit, Jianying Luo, Nick McKeown, Scott Shenker. “Ethane: Taking Control of the Enterprise,” ACM SIGCOMM ’07, August 2007, Kyoto, Japan.

Using OpenFlow | Process packets in an OpenFlow-enabled network

- Force all of a flow’s packets to pass through a controller
- Route the packets to a programmable switch that does packet processing

Using OpenFlow | Mobile wireless VOIP clients

- Experiments were conducted to test a new call-handoff mechanism for WiFi-enabled phones using OpenFlow-enabled network.
- A controller tracks the location of clients, reprogramming Flow Tables to allow seamless handoff from one access point to another as users move through the network.

Deploying OpenFlow Switches

- There is an opportunity for companies to sell OpenFlow-enabled switches to universities.
- Some companies are already adding OpenFlow to their switches without needing to change the hardware.
- Companies are willing to support the research community without revealing their product's internal workings.

Deploying OpenFlow Switches

OpenFlow-enabled switches

Network equipment vendors

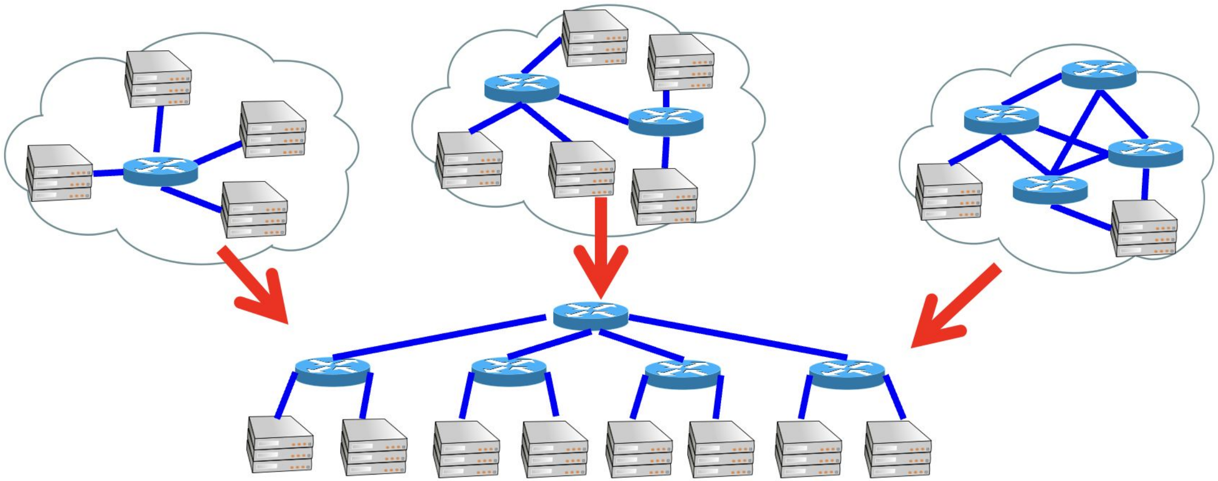
Research community

Reference designs

Open-source implementations

Network Virtualization

- Representation of multiple logical network topologies on the same infrastructure.
- Examples:
  - Virtual LANs (VLANs)
  - VPNs
  - Hosted Cloud Infrastructure

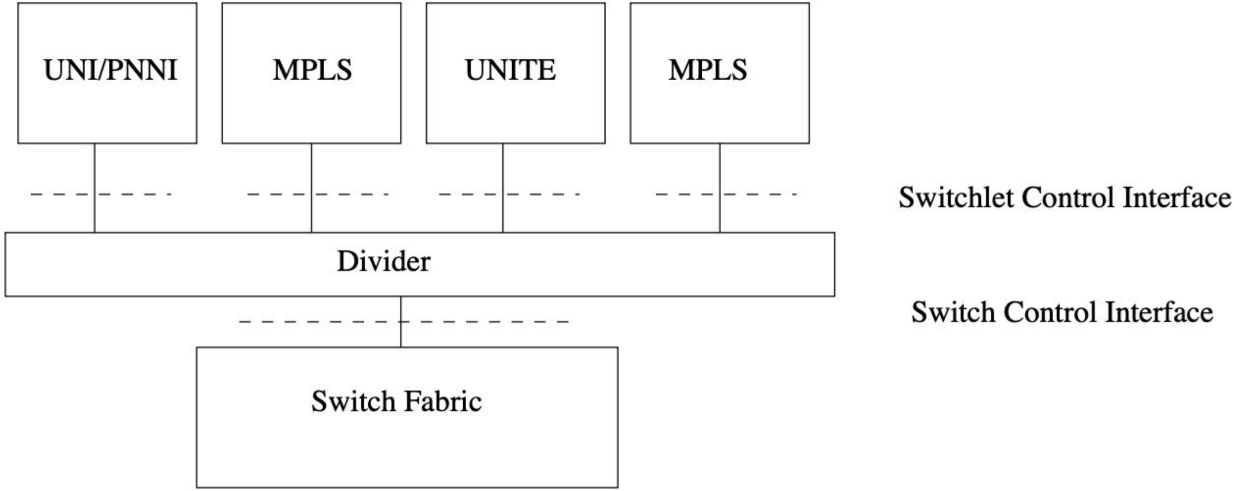


Network Virtualization - Benefits

- **Sharing**
  - Multiple logical networks on a platform
  - Resource isolation (CPU, memory, bandwidth, etc.)
- **Customizability**
  - Customizable routing/forwarding software for each logical network
  - General purpose CPUs for the control plane

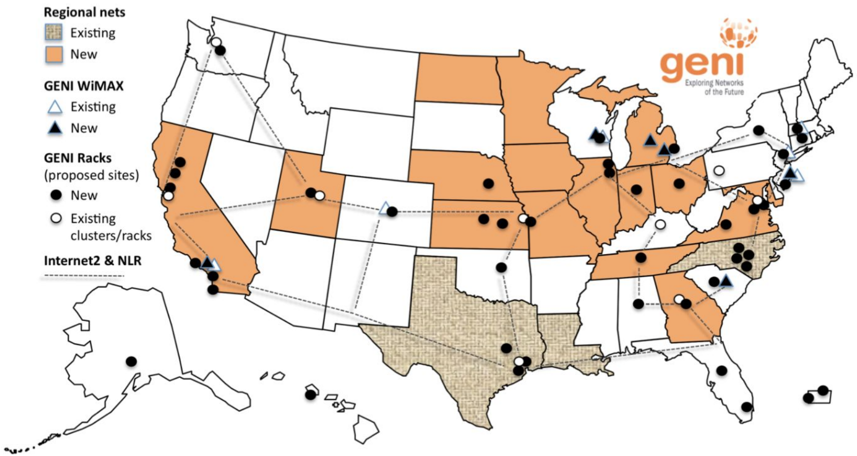
Network Virtualization - The Tempest ('98)

- Multiple controllers over single ATM switch network.
- Separation of controller and switch fabric via an open interface.
- Partitioning of resources across controllers.



Network Virtualization - GENI ('05)

- National experimental infrastructure for networking research.
- Obtain compute resources & run custom protocols.
- Uses slicing to manage resources.



Summary and Conclusion

- Data plane programmability through Active Networks.
- Control and data plane separation.
- OpenFlow project and how it works in practice.
- The use cases of OpenFlow.
- Network virtualization enabled SDN and vice versa.
- SDN is a highly active research field today.