An Introduction to Software Defined Networking and OpenFlow

Violet R. Syrotiuk, Arizona State University

Presentation by Vic Thomas, GENI Project Office

Sponsored by the National Science Foundation

- Software Defined Networking Basics
- OpenFlow

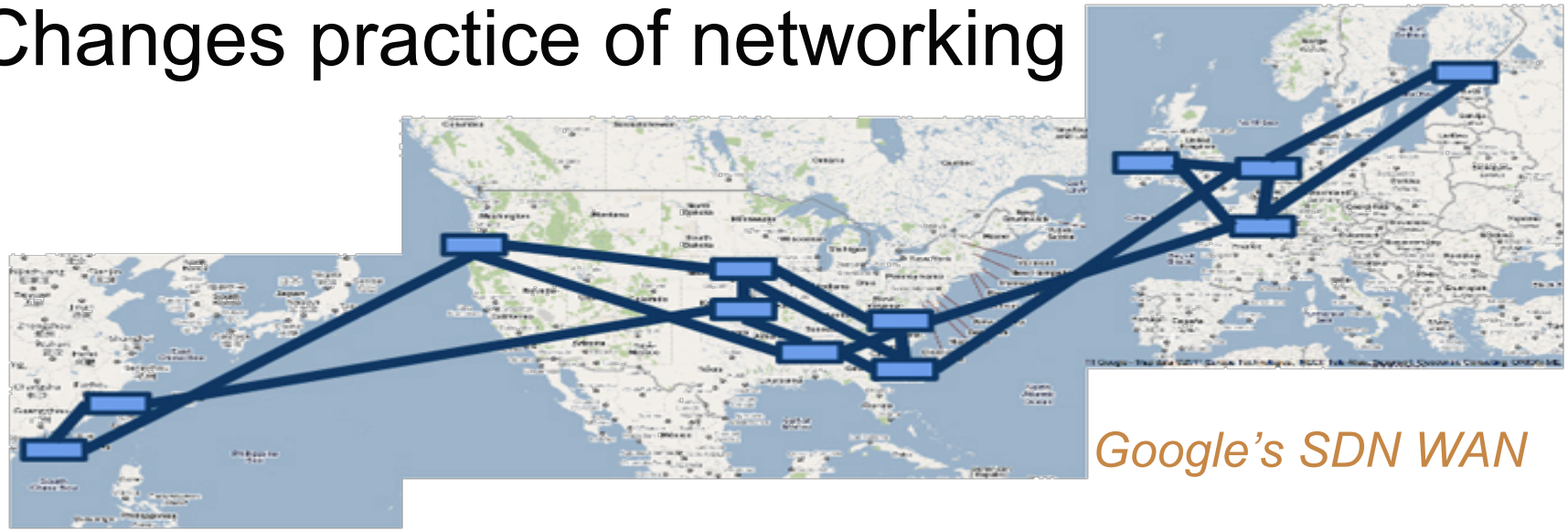
- **Software Defined Networking Basics**

"The current Internet is at an impasse because new architecture cannot be deployed or even adequately evaluated"

Overcoming the Internet Impasse through Virtualization, Larry Peterson, Scott Shenker, Jonothan Turner. Hotnets 2004

Software Defined Networking…

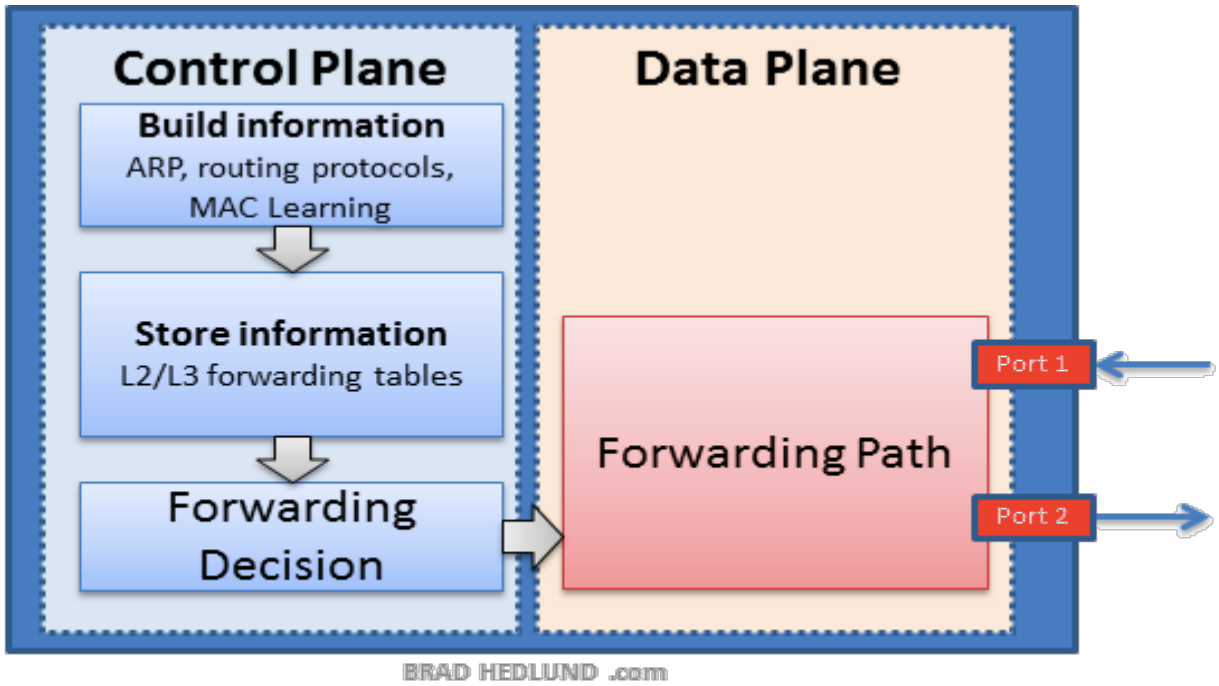
- Enables innovation in networking
- Changes practice of networking



Google's SDN WAN

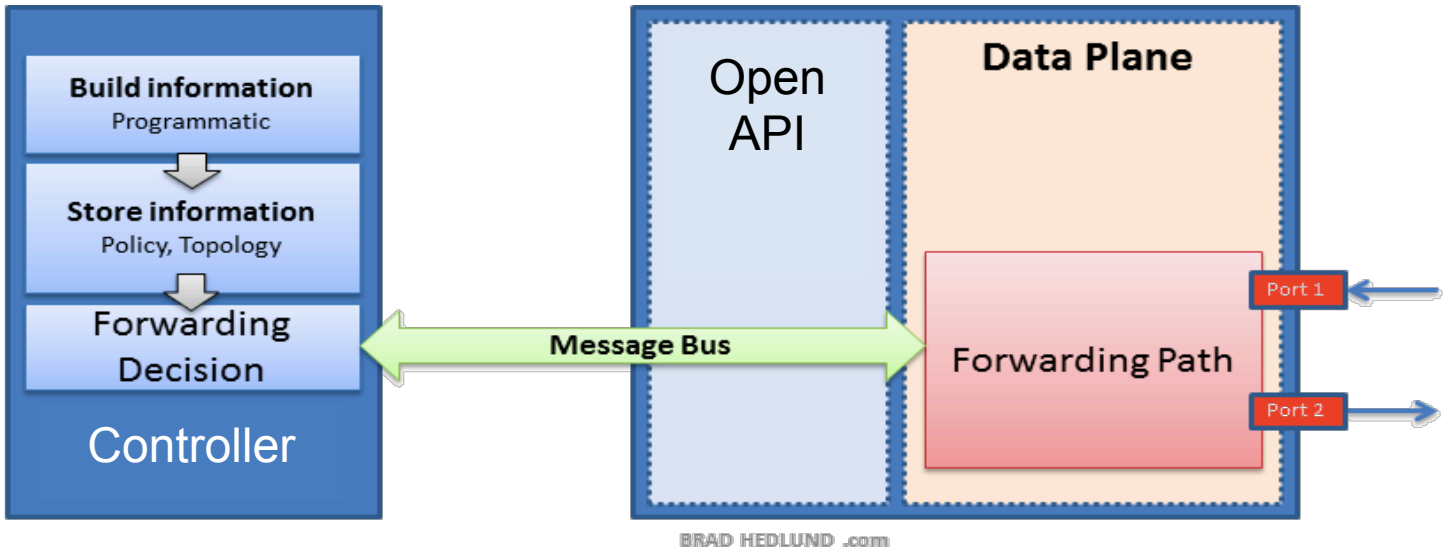
Traditional Switches and Routers

Smarts baked into switch



Network Switch/Router

SDN Basics



Smarts moved out of switch

Network Switch

SDN Basics

Forwarding table entries added by vendor provided logic internal to switch

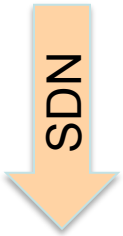
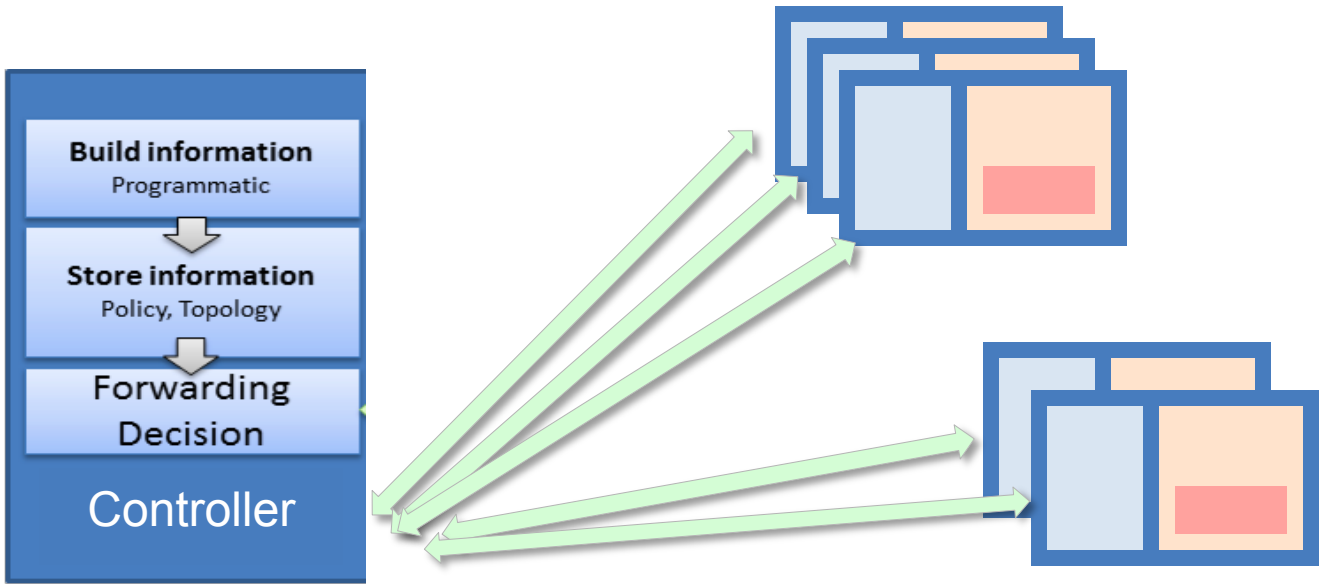


Table entries added by external controller written by anyone

MATCH	ACTION
dst subnet X	output port 48
dst subnet Y	output port 47
dst MAC: 00:00:00:00:00:01	output port 2
dst MAC: 00:00:00:00:00:01	output port 5
src subnet Z	drop
TCP port 80	output port 10

Switch Forwarding Table

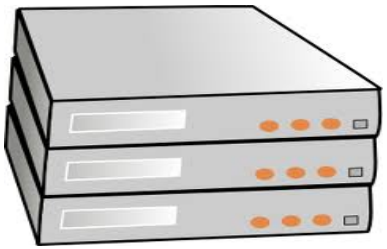
SDN Basics



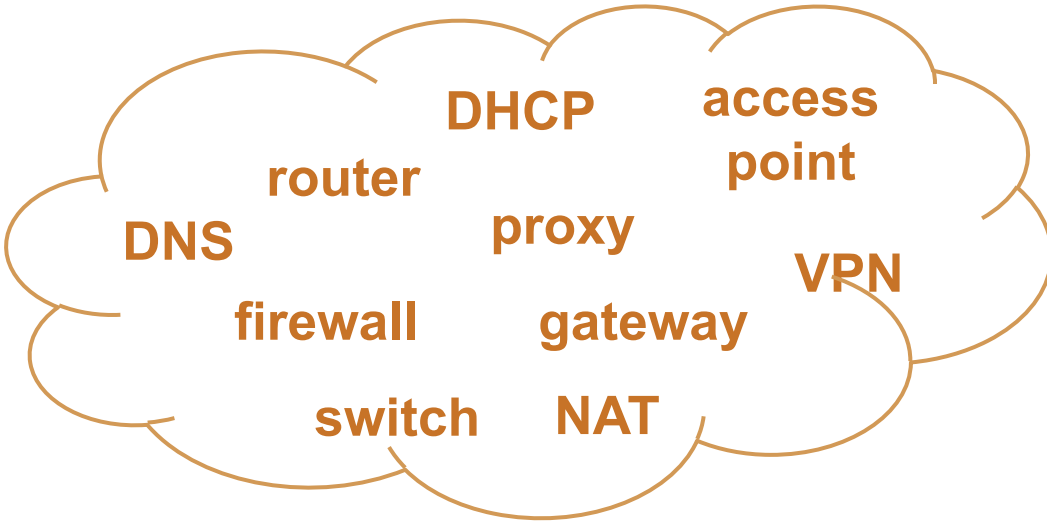
One controller can manage many switches

SDN Enables Network Function Virtualization

+

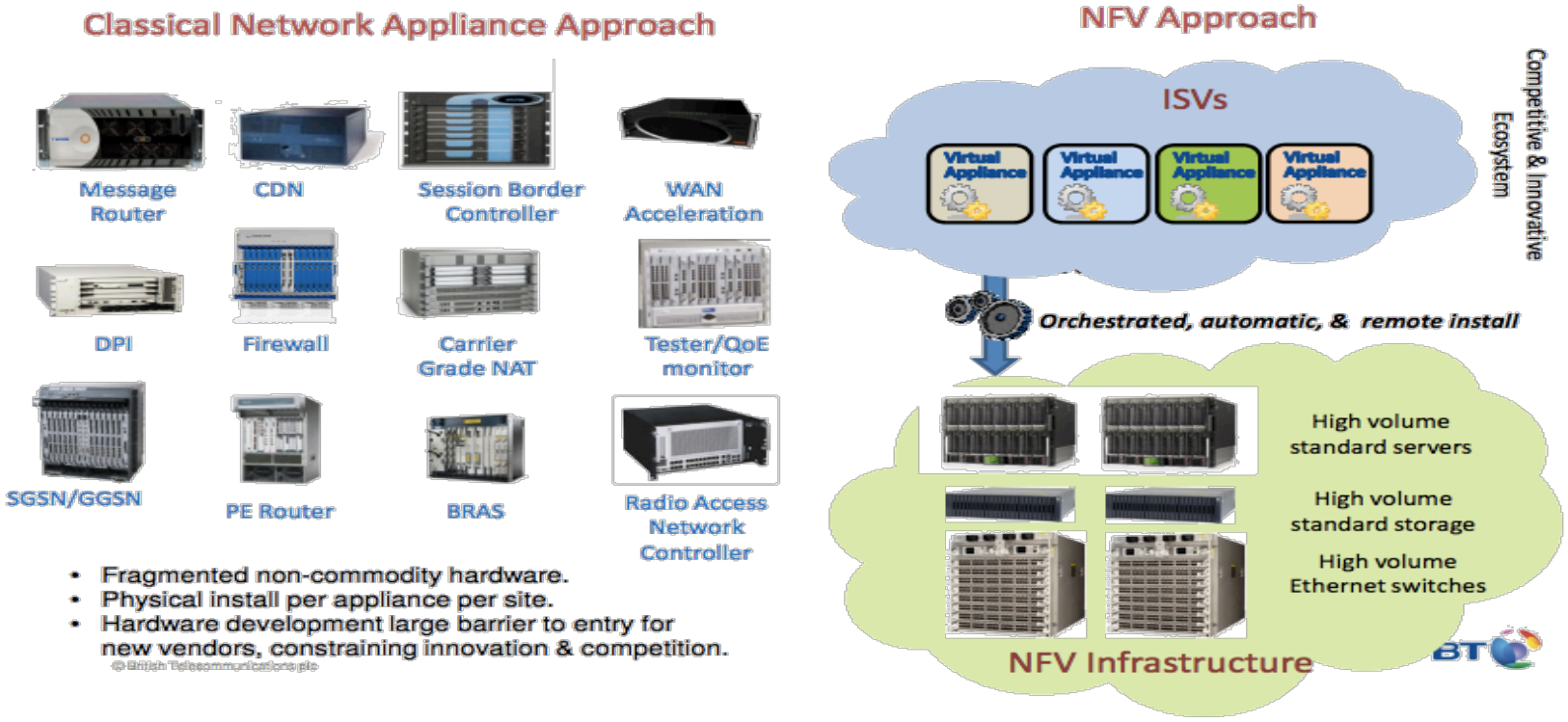


Network Device



Many network functions can be implemented using a generic network device

NFV: Network Function Virtualization



Slide from: http://docbox.etsi.org/Workshop/2013/201304_FNTWORKSHOP/S07_NFV/BT_REID.pdf

Software Defined Infrastructure

User defined virtual networks with compute, storage, networking

Orchestration Layer (e.g. ONOS)

Physical infrastructure

Everything is virtualized

Highly optimized networks

Dynamic reconfigurations

Network snapshotting

Network engineering ~
~ Software engineering

Figure adapted from http://www.slideshare.net/LarryCover/virtualizing-the-network-to-enable-a-software-defined-infrastructure-sdi?related=1

SDN Benefits*

- **External control**
  - Enables network apps
  - Fosters innovation: Not limited to vendor provided switch logic
  - Leverages general-purpose computers (Moore's Law)
  - Drives down costs: Network hardware becomes a commodity
- **Centralized control**
  - Enterprise-wide optimization and planning
  - Dynamic network reconfiguration
  - One place for apps to interact (auth & auth, etc.)

* OpenFlow: A Radical New idea in Networking, Thomas A. Limoncelli CACM 08/12 (Vol 55 No. 8)

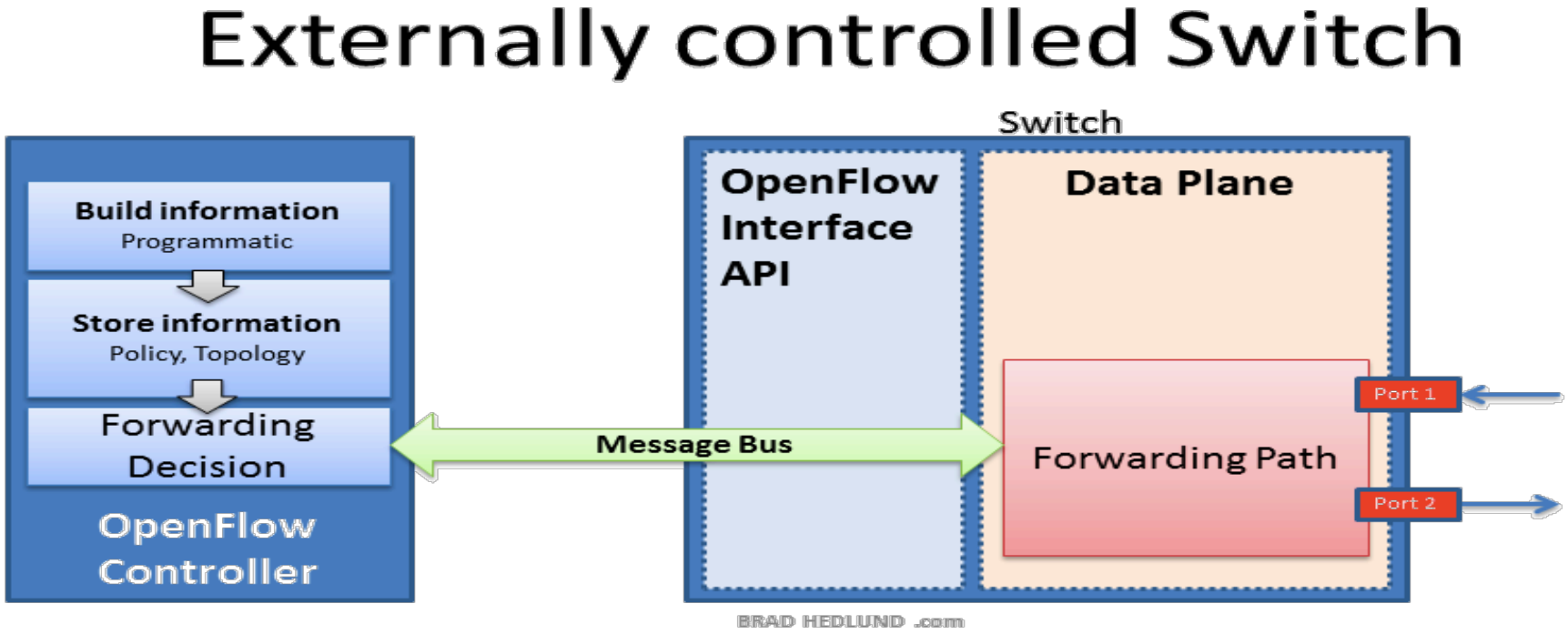
SDN Drawbacks

- Unexpected interactions between features
- Controller reliability and stability
- Controller security (runs on a general purpose computer and OS)

There an now many more ways of messing up a nework

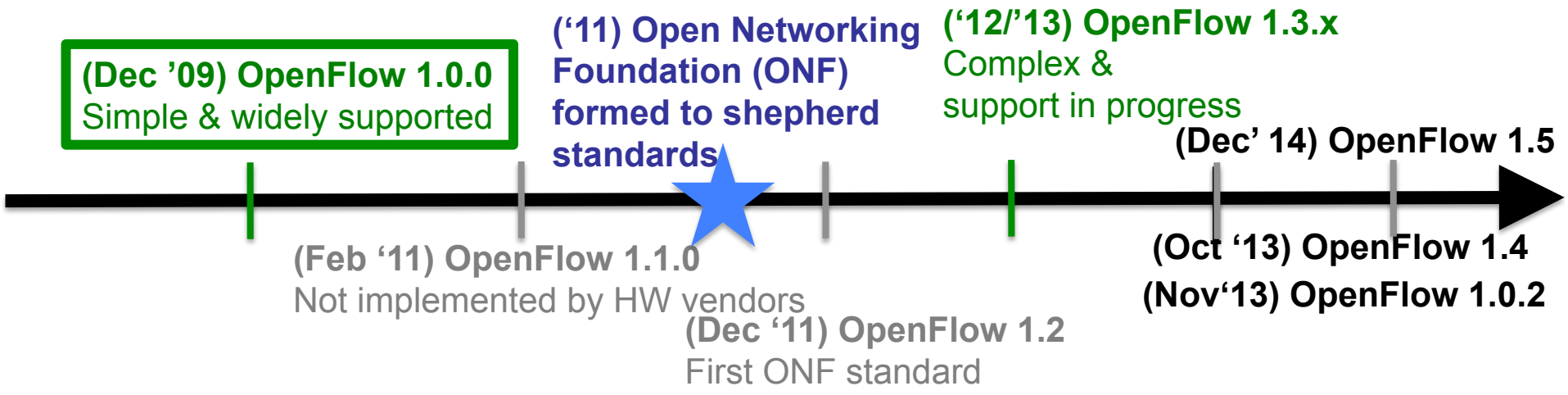
- **OpenFlow**

OpenFlow is an SDN API



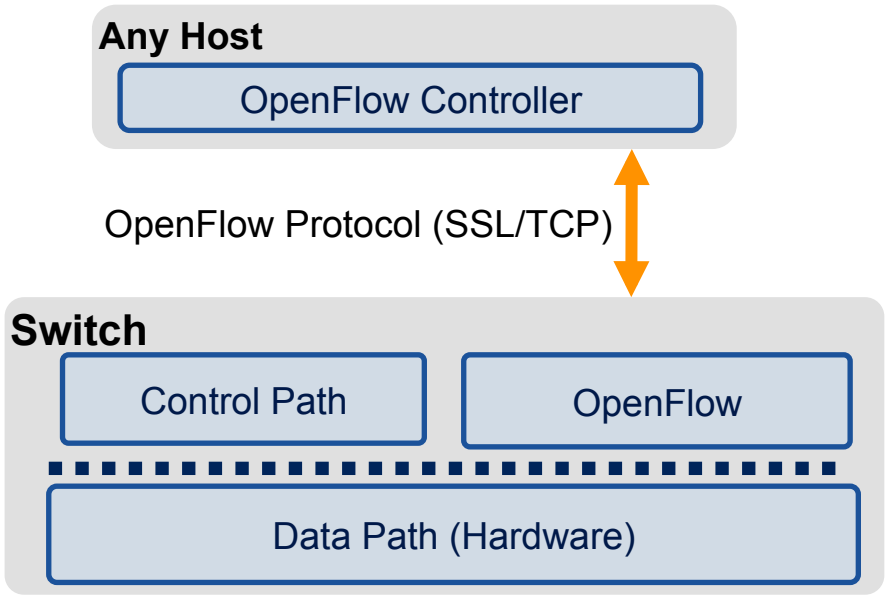
OpenFlow is the most widely implemented controller-switch API

OpenFlow Versions



https://www.opennetworking.org/sdn-resources/technical-library

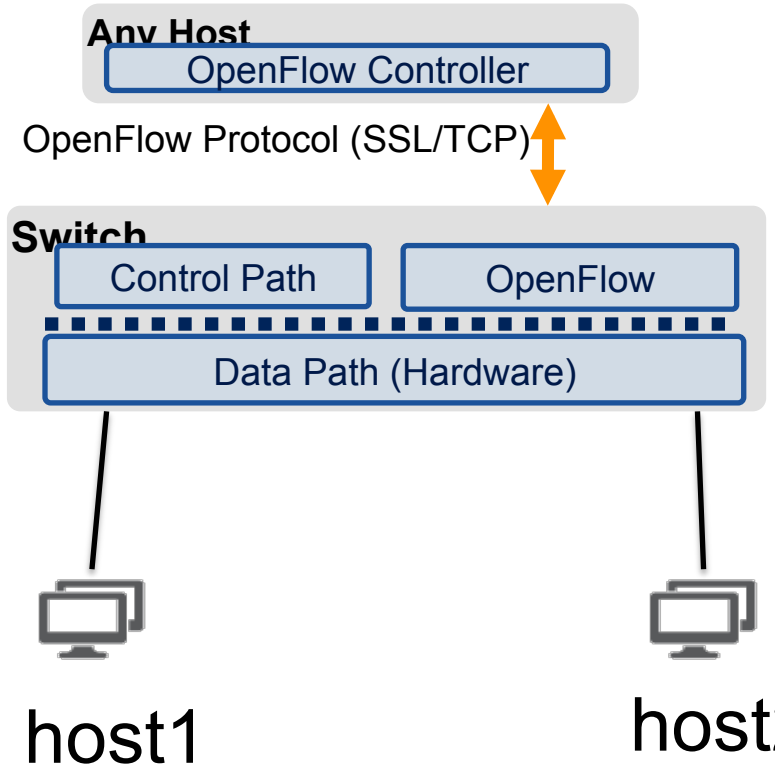
OpenFlow



- The controller is responsible for populating forwarding table of the switch
- In a table miss the switch asks the controller

Modified slide from : http://www.deutsche-telekom-laboratories.de/~robert/GENI-Experimenters-Workshop.ppt

OpenFlow in Action



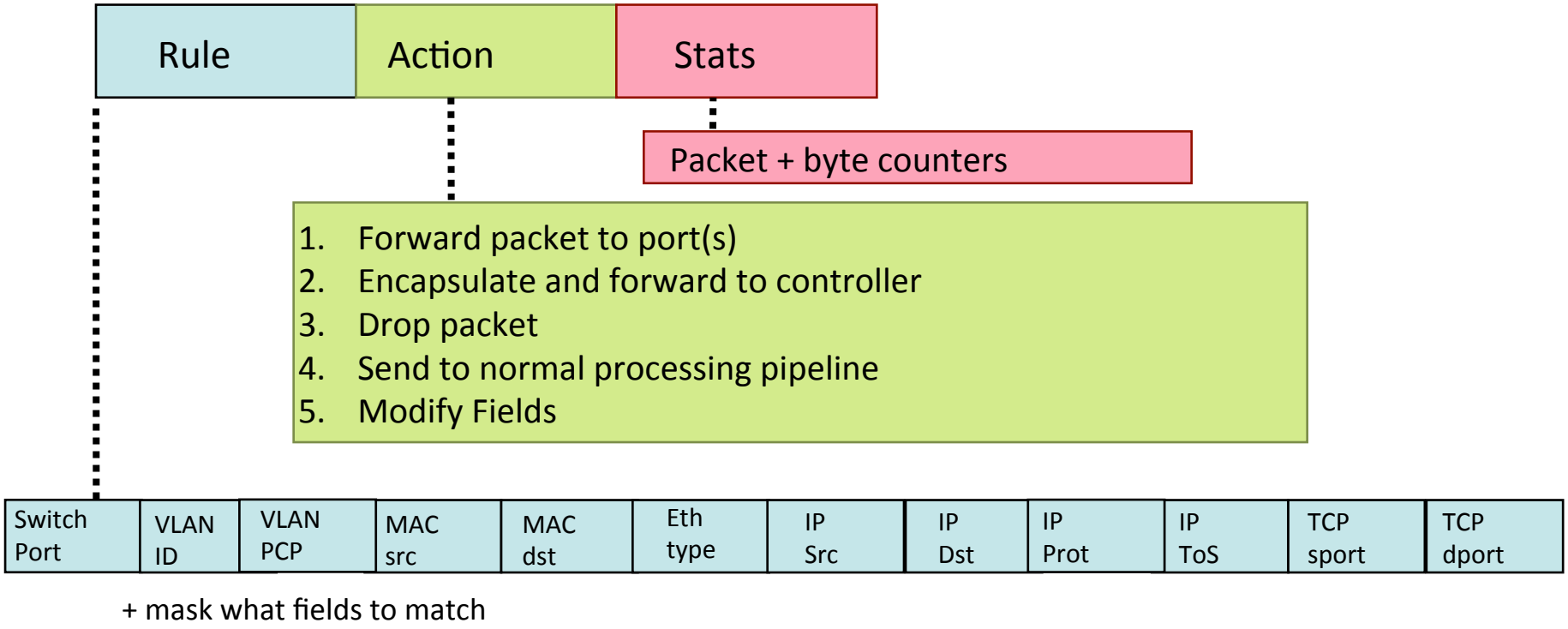
Host1 sends a packet

If there are no rules for handling this packet

- Forward packet to the controller
- Which installs a rule on the forwarding table (flow table)

Subsequent packets do not go through the controller

OpenFlow 1.0 Basics



slide from : http://www.deutsche-telekom-laboratories.de/~robert/GENI-Experimenters-Workshop.ppt

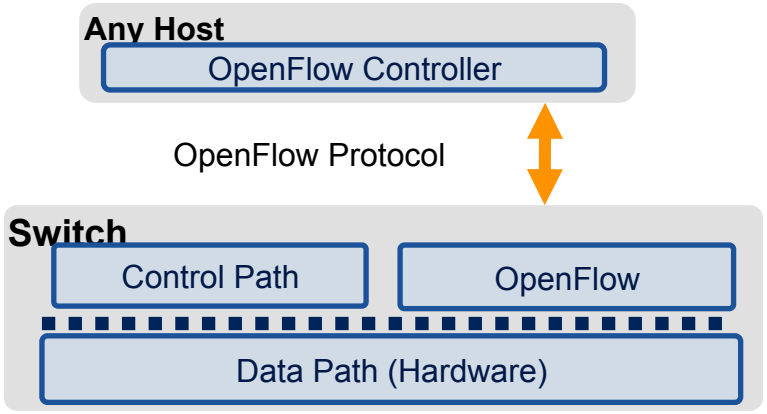
Use Flow Mods

- Going through the controller on every packet is inefficient
- Install flows proactively (preferred) or reactively
- A Flow Mod consists of :
  - A **match** on any of the 12 supported fields
  - A **rule** about what to do matched packets
  - Timeouts about the rules:
    - Hard timeouts
    - Idle timeouts
  - The packet id in reactive controllers
  - Priority of the rule

OpenFlow Datapaths

OpenFlow enabled devices are usually referred to as ***datapaths*** with a unique ***dpid***

It is not necessary that 1 physical device corresponds to 1 dpid



Different OpenFlow modes

- switches in **pure OF** mode are acting as one datapath
- **Hybrid VLAN switches** are one datapath per VLAN
- **Hybrid port switches** are two datapaths (one OF and one non-OF)

Each datapath can point to only one controller at a time!

OpenFlow Controllers

- **Open source controller frameworks**
  - NoX – *C++*
  - PoX **-** *Python*
  - OpenDaylight - *Java*
  - FloodLight - *Java*
  - Trema – *C / Ruby*
  - Maestro - *Java*
  - Ryu - *Python*
- **Production controllers**
  - Mostly customized solutions based on Open Source frameworks
  - ProgrammableFlow - NEC

OpenFlow: Common Pit Falls

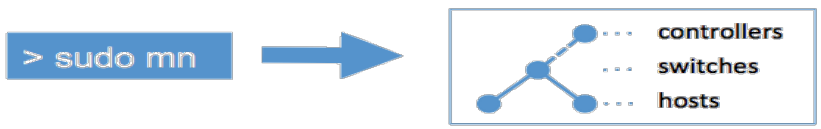
- Reactive controllers
  - Cause additional latency on some packets
  - UDP – many packets queued for your controller before flow is set up
- Hardware switch limitations
  - Not all actions are supported in hardware
- No STP to prevent broadcast storms
- Controller is **responsible for all traffic**, not just your application!
  - ARP, DHCP, LLDP

Running OpenFlow Experiments

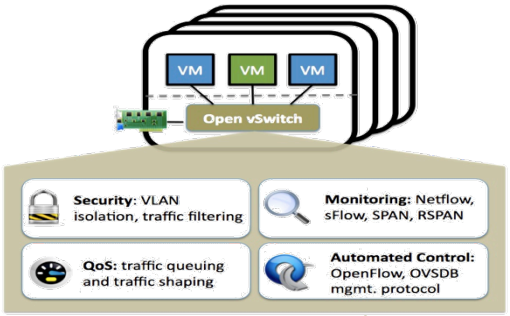
Debugging OpenFlow experiments is hard:

- Network configuration debugging requires coordination
- Many networking elements in play
- No console access to the switch

Before deploying your OpenFlow experiment test your controller.



http://mininet.github.com/



http://openvswitch.org/

Evolution of the OpenFlow Protocol

- *OpenFlow 1.0*
  + What you know and love!
- *OpenFlow 1.1*
  + Multiple tables and group tables
  + Some more matches and actions
- *OpenFlow 1.2*
  + The OpenFlow Extensible Match (OXM)
- *OpenFlow 1.3*
  + Meters
  + Table features

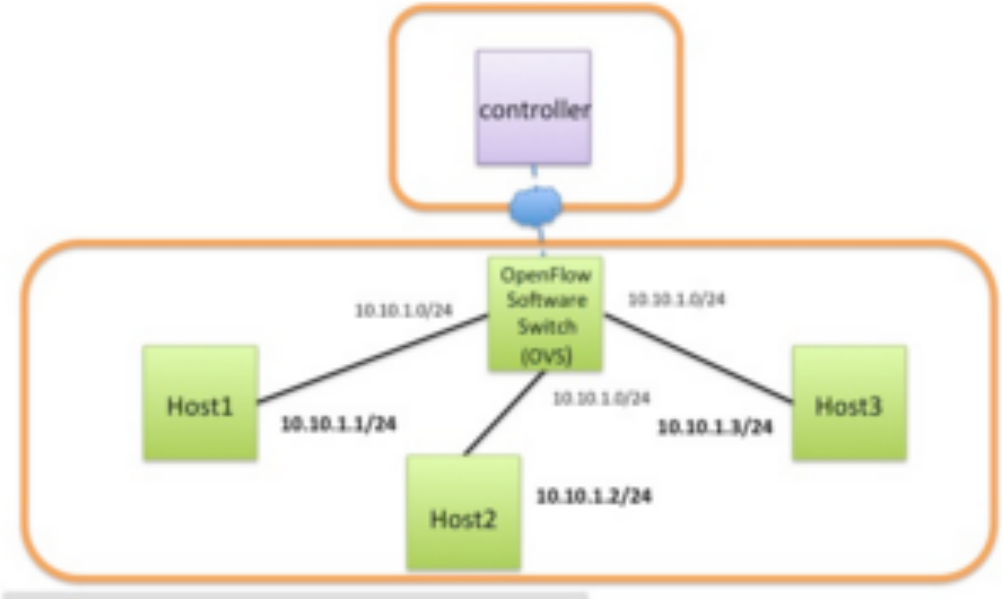
Evolution of the OpenFlow Protocol

- *OpenFlow 1.4*
  - Bundles
  - Flow table synchronization
  - Flow monitoring
- *OpenFlow 1.5*
  - More fine-grained matches and actions
  - Egress tables
  - Packet type aware pipeline & pipeline registers
  - Group/meter table improvements
- …But we struggle to keep up…

Why OpenFlow 1.3

- OF 1.0 primary complaint = too rigid
- OF 1.3 gains
  - ✓ Greater match and action support
  - ✓ Instructions add flexibility and capability
  - ✓ Groups facilitate advanced actions
  - ✓ Meters provide advanced counters
  - ✓ Per-table features
  - ✓ Custom table-miss behavior
  - ✓ …and more!

OpenFlow Tutorial Exercise



Experiment with simple controllers to control the traffic between three hosts.

1. *Traffic duplicator*
2. *TCP port forwarder*
3. *Proxy controller*

Project ideas

- Write a load balancer that uses packet statistics in its decision making
- Controller placement problem: Determine how many controllers are needed for a topology and where to place them
- Write a controller to establish an IPSEC tunnel to encrypt a flow