

OpenFlow on top of NetFPGA

Part I: Introduction to OpenFlow NetFPGA Spring School 2010

Nadi Sarrar
TU-Berlin / T-Labs
Research group Prof. Anja Feldmann, Ph.D.

Some slides with permission from Prof. Nick McKeown.
OpenFlow was originally developed by Stanford University.



What is OpenFlow?

Support experimental protocols in production networks

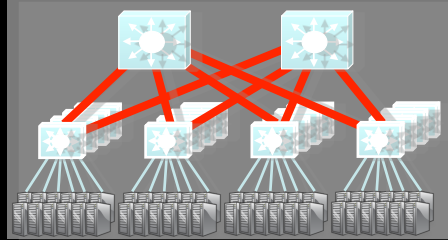
→ **Foundation for innovation**

Specify an API to control Ethernet switch forwarding decisions

→ **Towards software-defined, open-source networks**

Why OpenFlow?

Example:
New data center



Cost

500,000 servers
Fanout of 50 \Rightarrow 10,000 switches
\$10k commercial switch \Rightarrow \$100M
\$1k custom-built switch \Rightarrow \$10M

Savings in 10 data centers = **\$900M**

Control

1. Optimize for features needed
2. Customize for services & apps
3. Quickly improve and innovate

Why OpenFlow?

Network innovation today is difficult:

- Switches and routers are “closed”
- Software-only approach lacks performance and scalability

\rightarrow Need high-performance, programmable networking equipment (substrate)

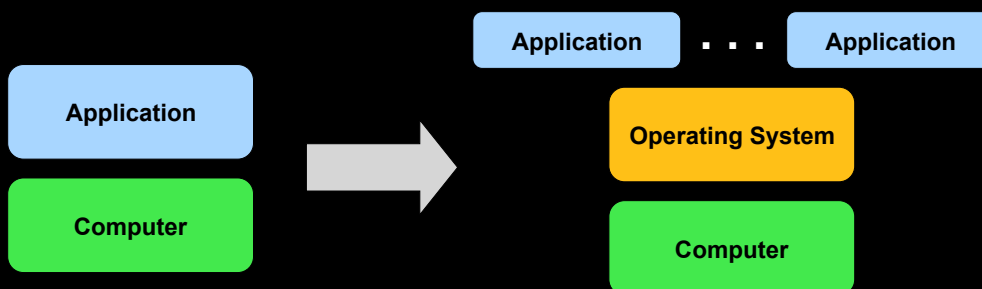
Networking hardware substrate

What do we mean by programmable equipment?

Need to define a **substrate** to build upon.

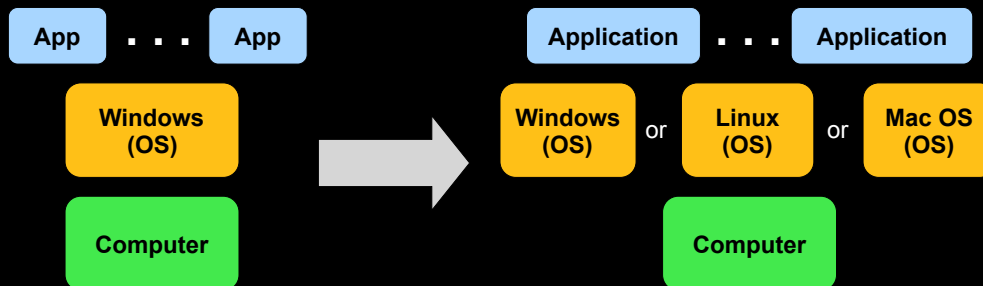
Let's learn from evolution of the PC!

PC hardware abstraction



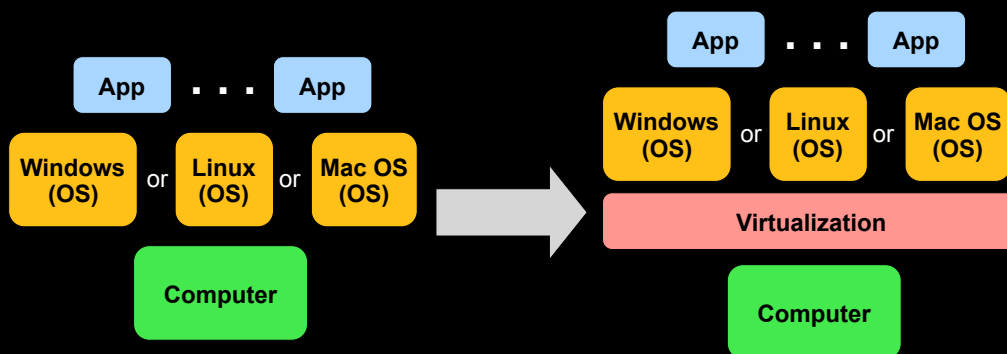
OS abstracts hardware substrate
→ **Innovation in applications**

PC hardware abstraction



Simple, common, stable, hardware substrate below.
Programmability and competition:
→ **Innovation in OS and applications**

PC hardware virtualization

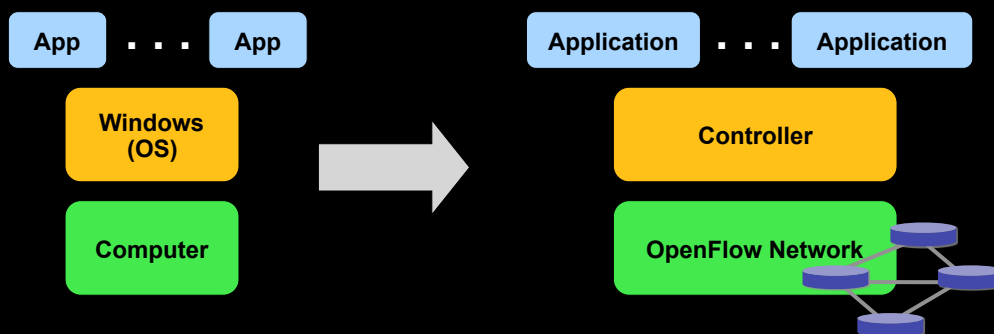


Strong isolation model:
→ **Innovation in infrastructure**

Networking substrate


Let's apply a similar abstraction model to networking.

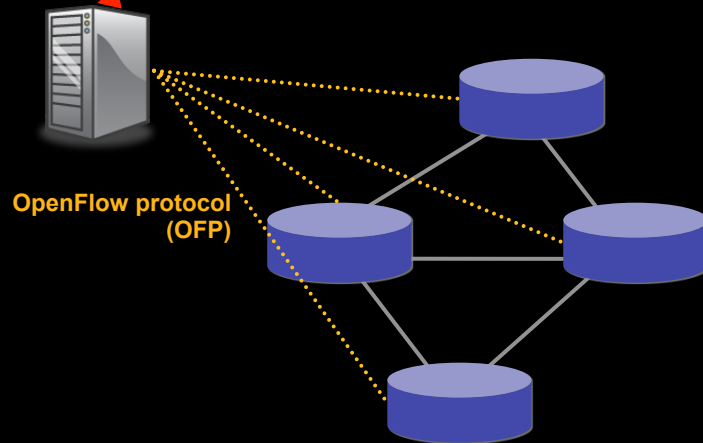
Network hardware abstraction



Simple, common, stable, hardware substrate below.
Programmability and competition:
→ **Innovation in network controllers**

Step 1: Separate intelligence from datapath

New function!  Operators, users,
3rd party developers, researchers, ...



OpenFlow on top of NetFPGA -
Introduction

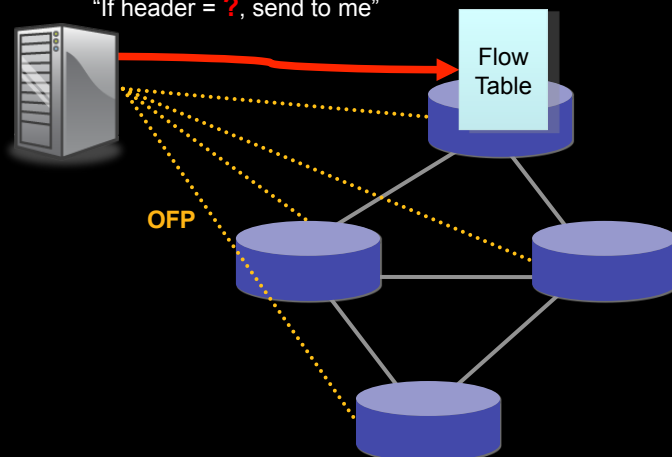
11

Step 2: Cache decisions in minimal flow-based datapath

“If header = **x**, send to port 4”

“If header = **y**, overwrite header with **z**, send to ports 5,6”

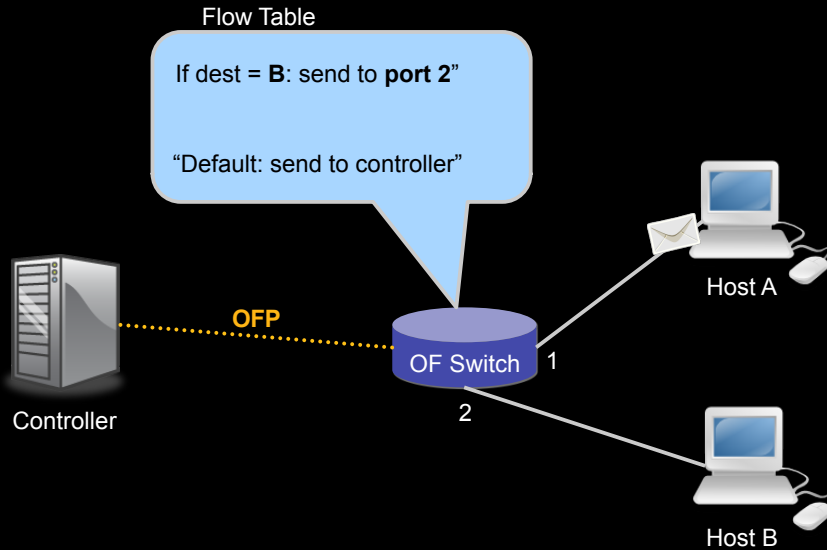
“If header = **?**, send to me”



OpenFlow on top of NetFPGA -
Introduction

12

Example



Flow

Flow's can be defined via 10-tuples:

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

Allows for flexible, multidimensional flow definitions.

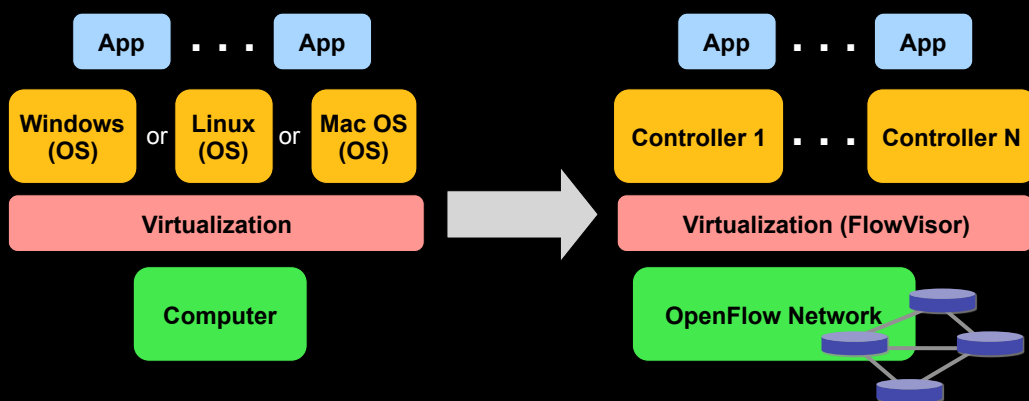
Topology slicing

How to **slice** the network to allow production and experimental use at the same time?

How to do virtualization?

Again, let's learn from PC's :)

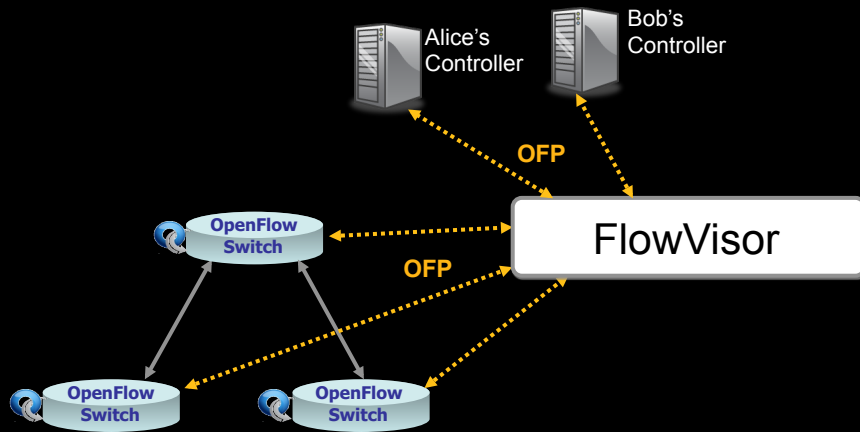
Virtualization



Strong isolation model:
→ **Innovation in infrastructure**

FlowVisor

Rob Sherwood (rob.sherwood@stanford.edu)



OpenFlow on top of NetFPGA -
Introduction

17

FlowVisor

Rob Sherwood (rob.sherwood@stanford.edu)

- A **proxy** between switch and guest controller
 - Parses and rewrites OpenFlow messages as they pass
- Ensures that **one experiment doesn't affect another**
- Allows **rich virtual network boundaries**
 - By port, by IP, by flow, by time, etc.
- Define **virtualization rules in software**

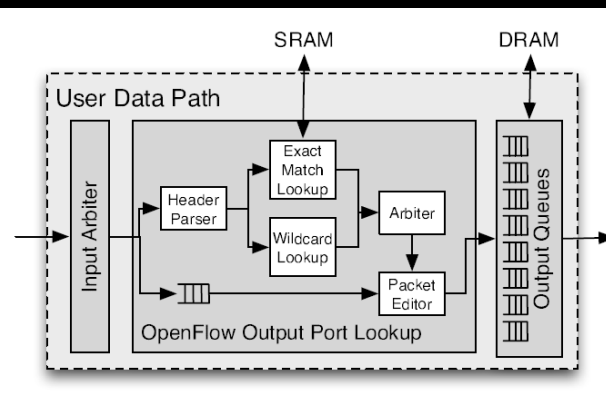
OpenFlow on top of NetFPGA -
Introduction

18

Available controllers and switches

- NOX (<http://noxrepo.org/>, GNU GPLv3)
 - Provides network-wide view of the topology
 - C++ and Python modules make decisions
- OpenVSwitch (<http://openvswitch.org/>, Apache 2)
 - Soft-switch, replaces Linux bridge
 - Designed to be used with VM's
- Simple controller (reference implementation)
- Hardware switches:
 - Quanta LB4G (Broadcom), HP ProCurve 5400 series, NEC IP8800/S3640, Juniper MX, Cisco Catalyst, **NetFPGA**

NetFPGA as an OpenFlow switch



J. Naous et. al., "Implementing an OpenFlow Switch on the NetFPGA platform", ANCS 2008

OpenFlow activity in Europe

Successful EU proposals:

OFELIA (5 OpenFlow islands across Europe,
partners from academia and industry)

CHANGE (Flow-processing research, uses
OFELIA testbed)

SPARC (Split architecture, MPLS)

Thank you!

<http://www.openflowswitch.org>

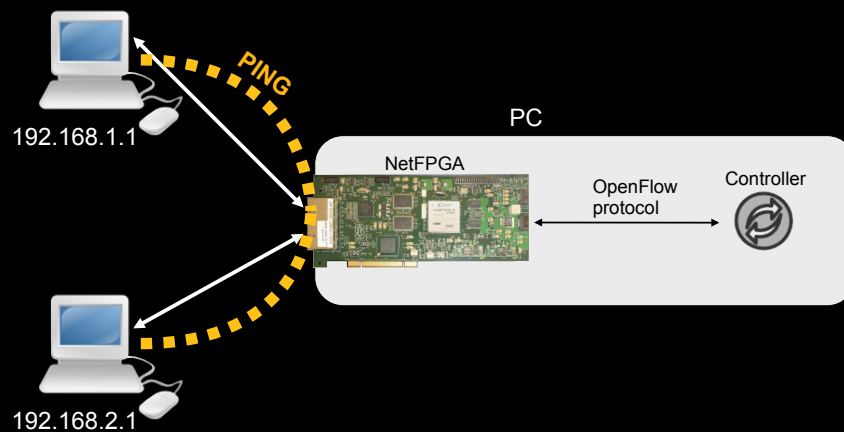
OpenFlow on top of NetFPGA

Part II: Hands-on workshop
NetFPGA Spring School 2010

Nadi Sarrar
TU-Berlin / T-Labs
Research group Prof. Anja Feldmann, Ph.D.



Exercise overview



192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit

1. Bitfile

Bitfile to program the NetFPGA:
/root/openflow/openflow_switch.bit

Download bitfile to NetFPGA:
nf2_download \$bitfile

OpenFlow on top of NetFPGA - Hands-on workshop

25

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

2. Kernel modules

Load general OpenFlow datapath module:
insmod /root/openflow/ofdatapath.ko

Load NetFPGA datapath module:
insmod /root/openflow/ofdatapath_netfpga.ko

OpenFlow on top of NetFPGA - Hands-on workshop

26

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

3a. Datapath

Create a datapath:
`dpctl adddp nl:0`

Display datapath info:
`dpctl show nl:0`

OpenFlow on top of NetFPGA - Hands-on workshop 27

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

3b. Datapath

Add interfaces to datapath:
`dpctl addif nl:0 nf2c0 nf2c1 nf2c2 nf2c3`

Display datapath info:
`dpctl show nl:0`

OpenFlow on top of NetFPGA - Hands-on workshop 28

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

4. Statistics

Display flow table entries:
`dpctl dump-flows nl:0`

Display hardware information:
`cat /proc/net/openflow/netfpga`

OpenFlow on top of NetFPGA - Hands-on workshop 29

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

ofprotocol

controller

5. Controller

Run controller (passive mode):
`controller tcp:localhost:6633`

Run OpenFlow protocol daemon (active mode):
`ofprotocol nl:0 tcp:localhost:6633`

OpenFlow on top of NetFPGA - Hands-on workshop 30

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

ofprotocol

controller

Kernel / Hardware

Userspace

5. Controller

Run controller (passive mode):
`controller ptcp:localhost:6633`

Run OpenFlow protocol daemon (active mode):
`ofprotocol nl:0 tcp:localhost:6633`

OpenFlow on top of NetFPGA - Hands-on workshop

31

192.168.1.1

192.168.2.1

PC

NetFPGA

OpenFlow protocol

Controller

openflow_switch.bit
ofdatapath.ko
ofdatapath_netfpga.ko

ofprotocol

controller

Kernel / Hardware

Userspace

6. Testing

Attach two nodes to any two ports.
Assign IP addresses.
IP should work (try to ping).
Monitor flow table: `dpctl dump-flows nl:0`

OpenFlow on top of NetFPGA - Hands-on workshop

32