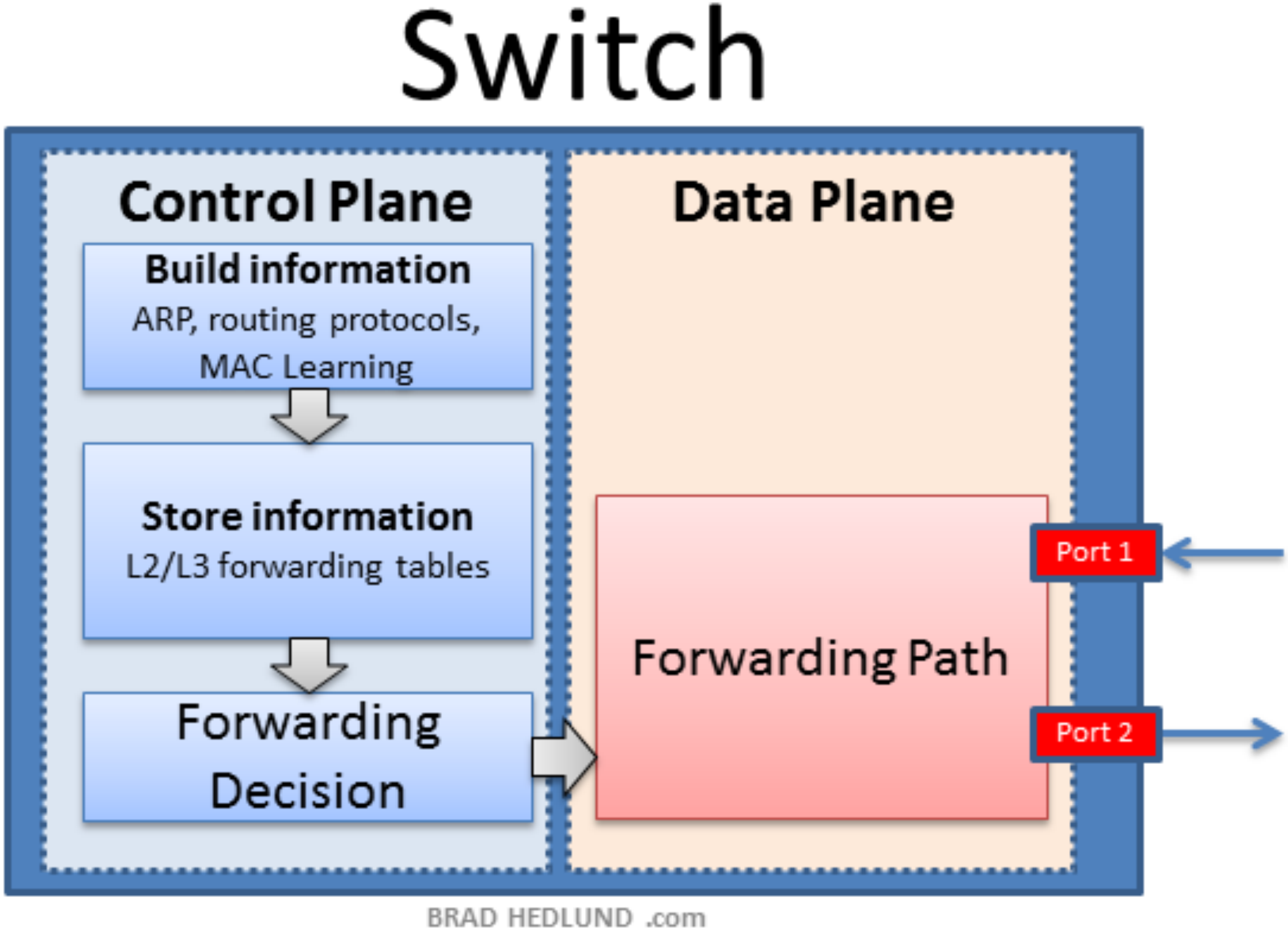
Tutorial: OpenFlow and GENI

GENI Engineering Conference 17

July 2013

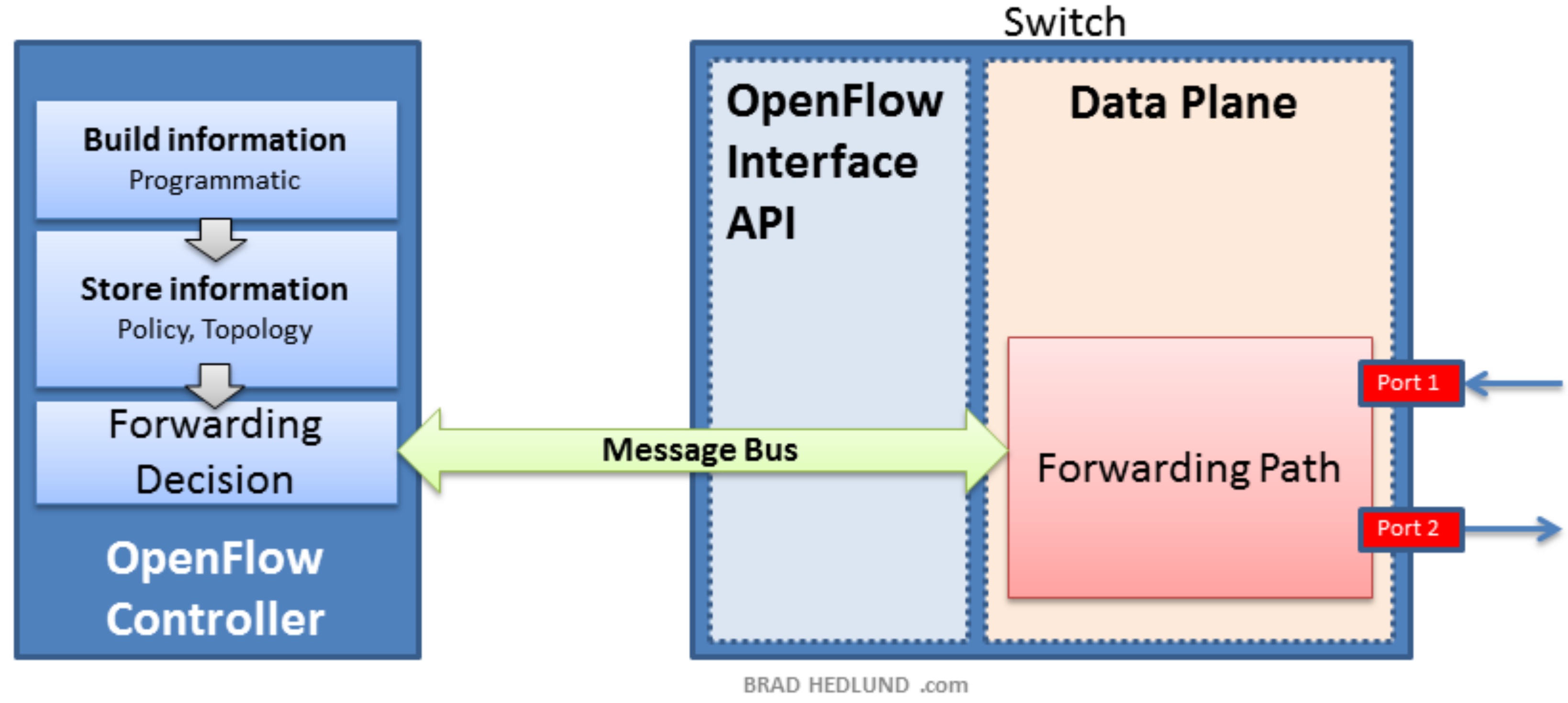
Sponsored by the National Science Foundation

Switch Architecture



Moving Control out of the Switch

Externally controlled Switch



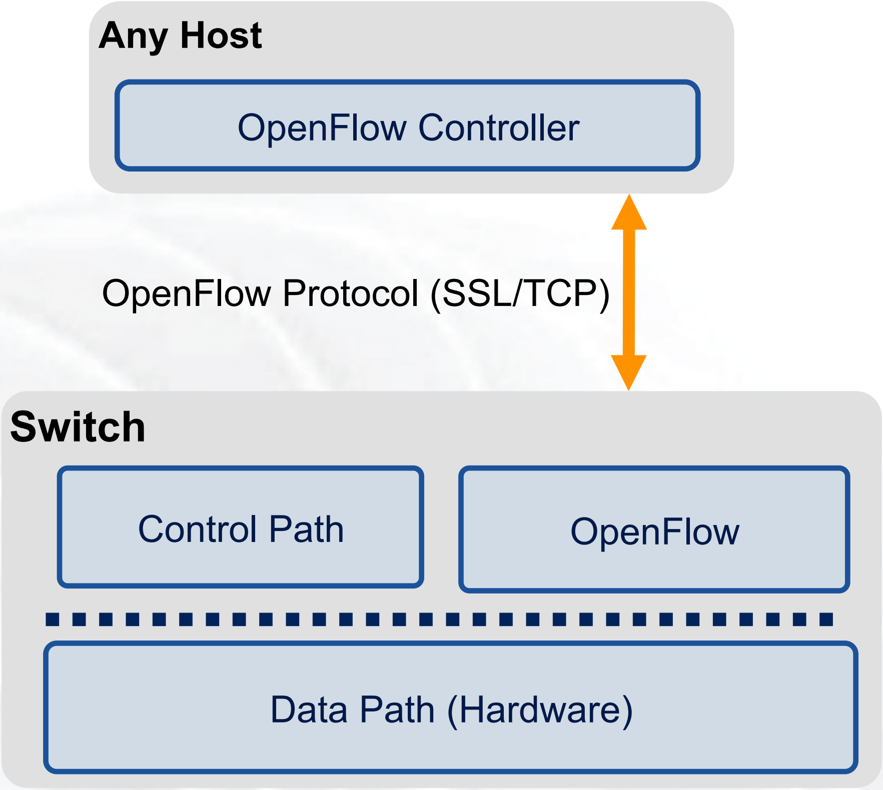
BRAD HEDLUND .com

OpenFlow is an API

- Control how packets are forwarded
- Implementable on COTS hardware
- Make deployed networks programmable
  - not just configurable
- Makes innovation easier

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

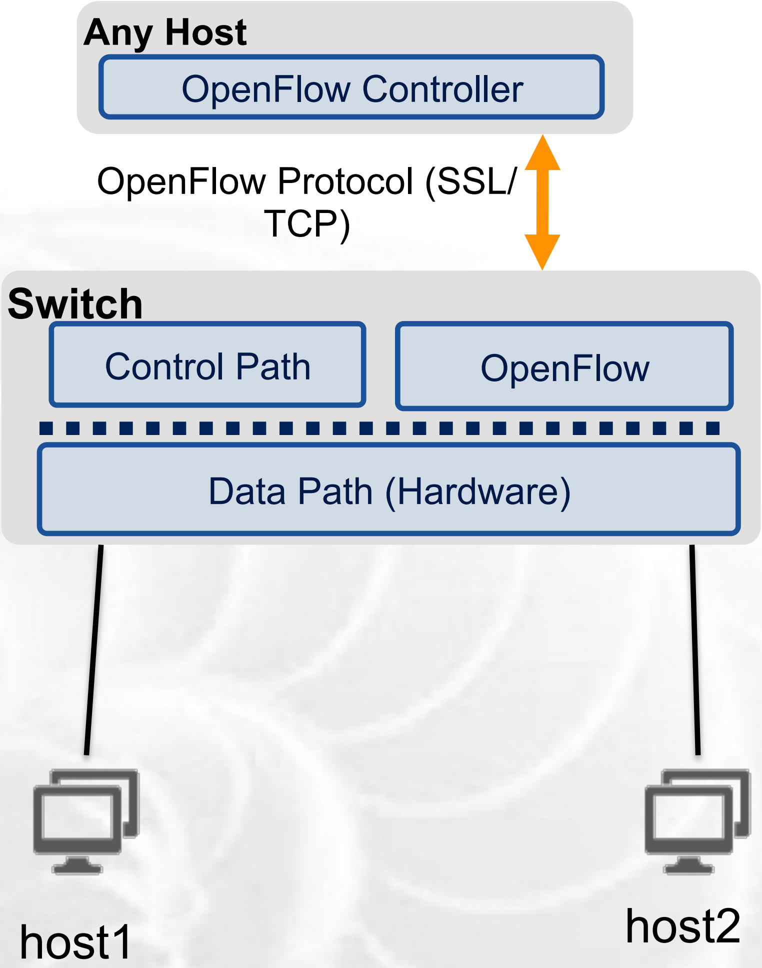
OpenFlow



- 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

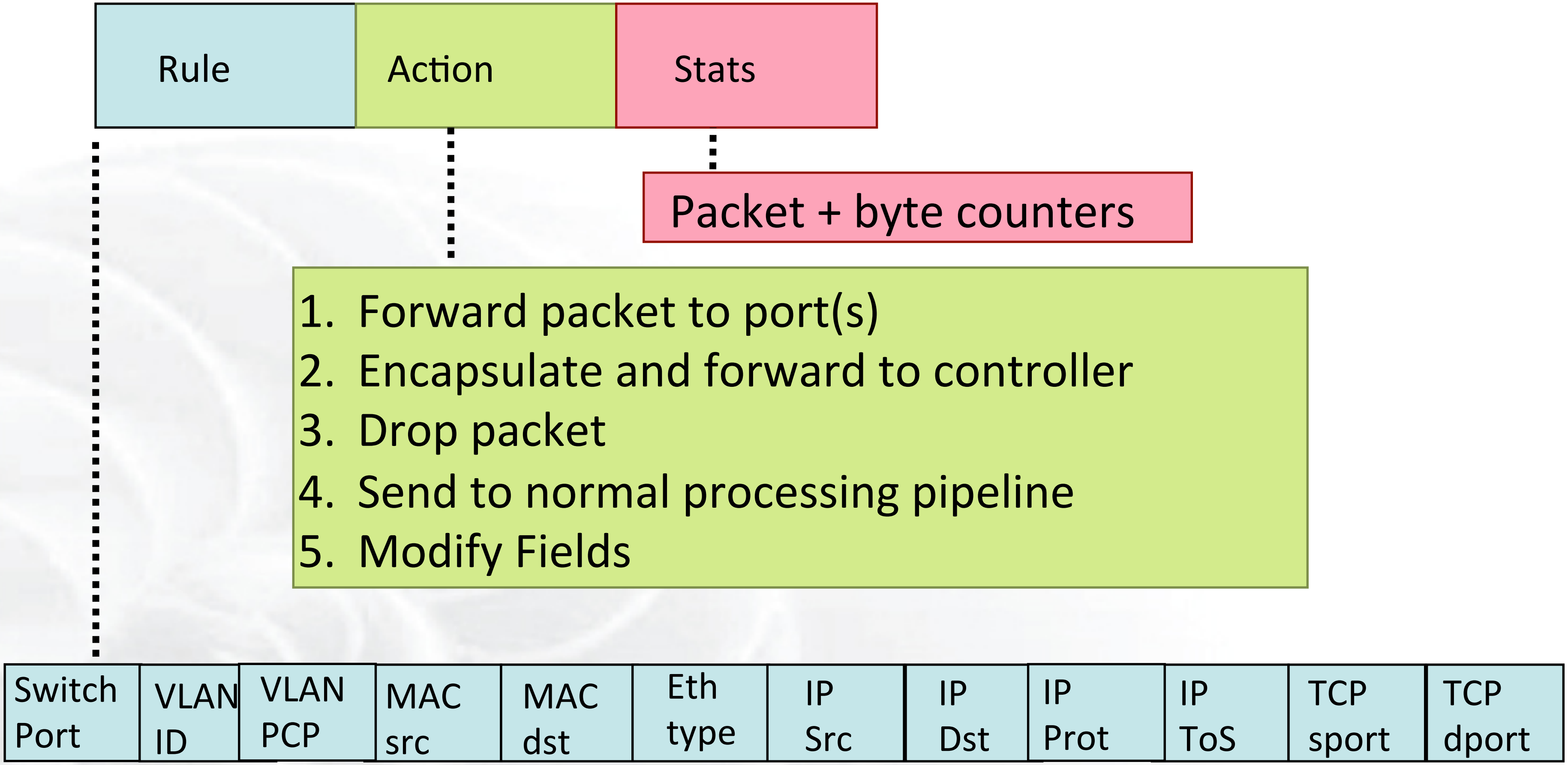
OpenFlow in action



- Host1 sends a packet
- If there are no rules about handling this packet
  - Forward packet to the controller
  - Controller installs a flow
- Subsequent packets do not go through the controller

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

OpenFlow Basics
Flow Table Entries



Rule | Action | Stats

Packet + byte counters

1. Forward packet to port(s)
2. Encapsulate and forward to controller
3. Drop packet
4. Send to normal processing pipeline
5. Modify Fields

Switch Port	VLAN ID	VLAN PCP	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP Prot	IP ToS	TCP sport	TCP dport

+ mask what fields to match

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
- Installing Flows either proactively or reactively is the right thing to do:
- A Flow Mod consists off :
  - 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

OpenFlow common PitFalls

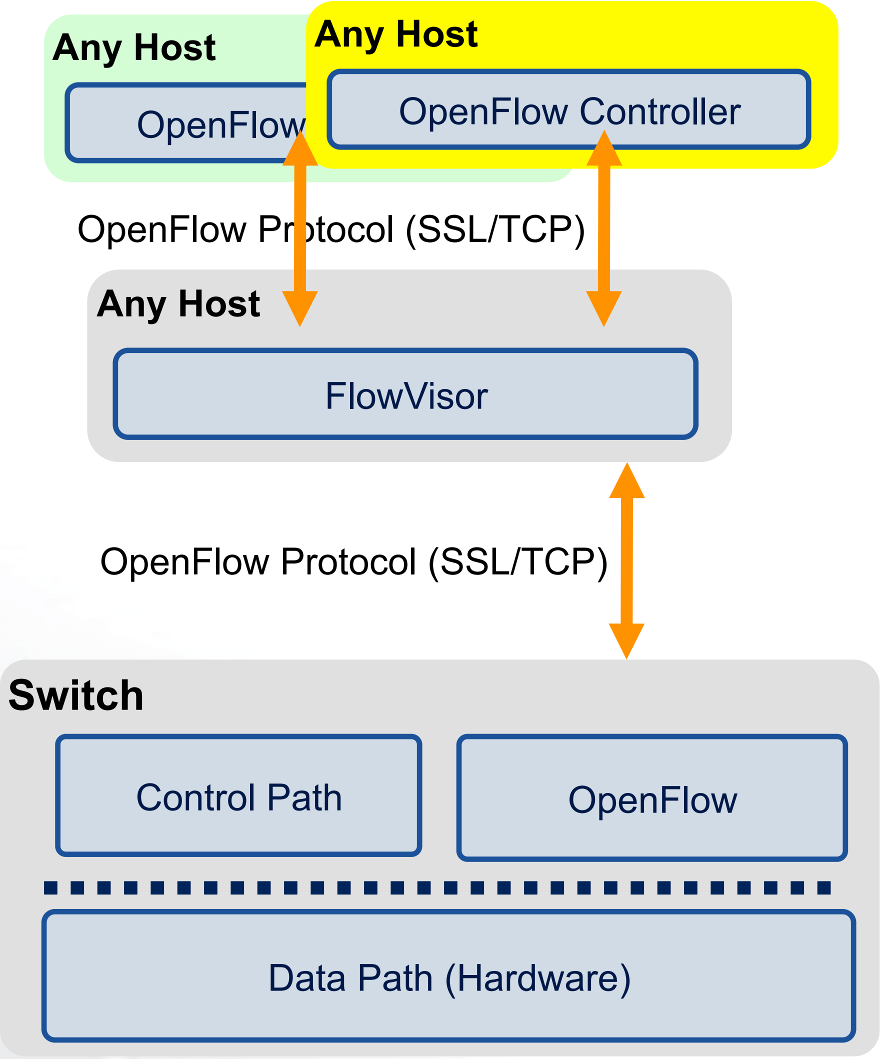
- Controller is responsible for all traffic, not just your application!
  - ARPs
  - DHCP
  - LLDP
- Reactive controllers
  - UDP
- Performance in hardware switches
  - Not all actions are supported in hardware
- No STP
  - Broadcast storms

FlowVisor

- Only one controller per switch
- FlowVisor is a proxy controller that can support multiple controllers

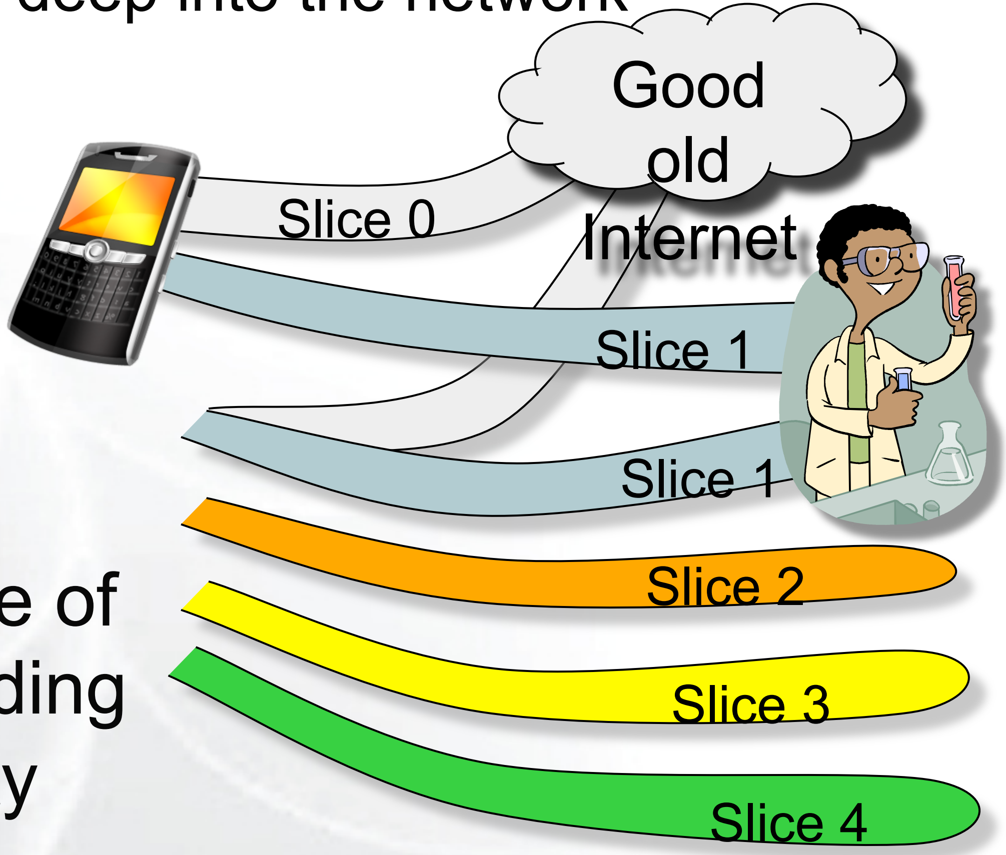
FlowSpace describes packet flows :

- Layer 1: Incoming port on switch
- Layer 2: Ethernet src/dst addr, type, vlanid, vlanpcp
- Layer 3: IP src/dst addr, protocol, ToS
- Layer 4: TCP/UDP src/dst port



GENI Programmable Network

- Key GENI concept: slices & deep programmability
  - Internet: open innovation in application programs
  - GENI: open innovation deep into the network



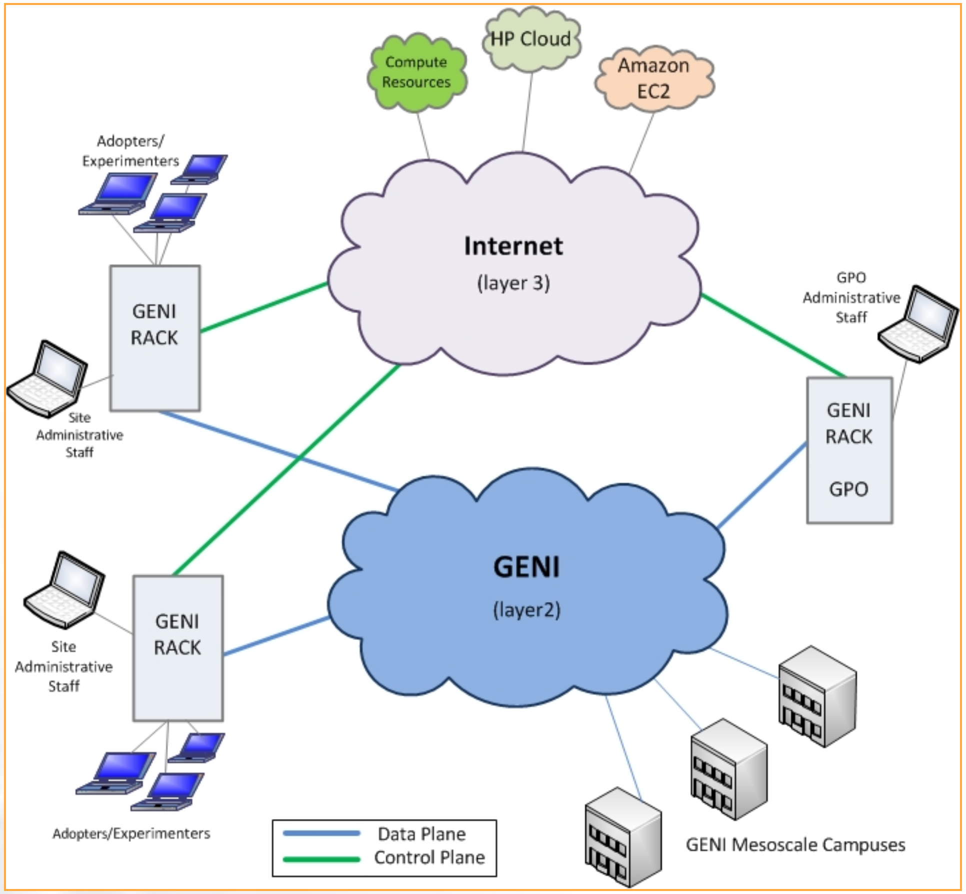
OpenFlow switches one of the ways GENI is providing deep programmability

Racks and Campuses

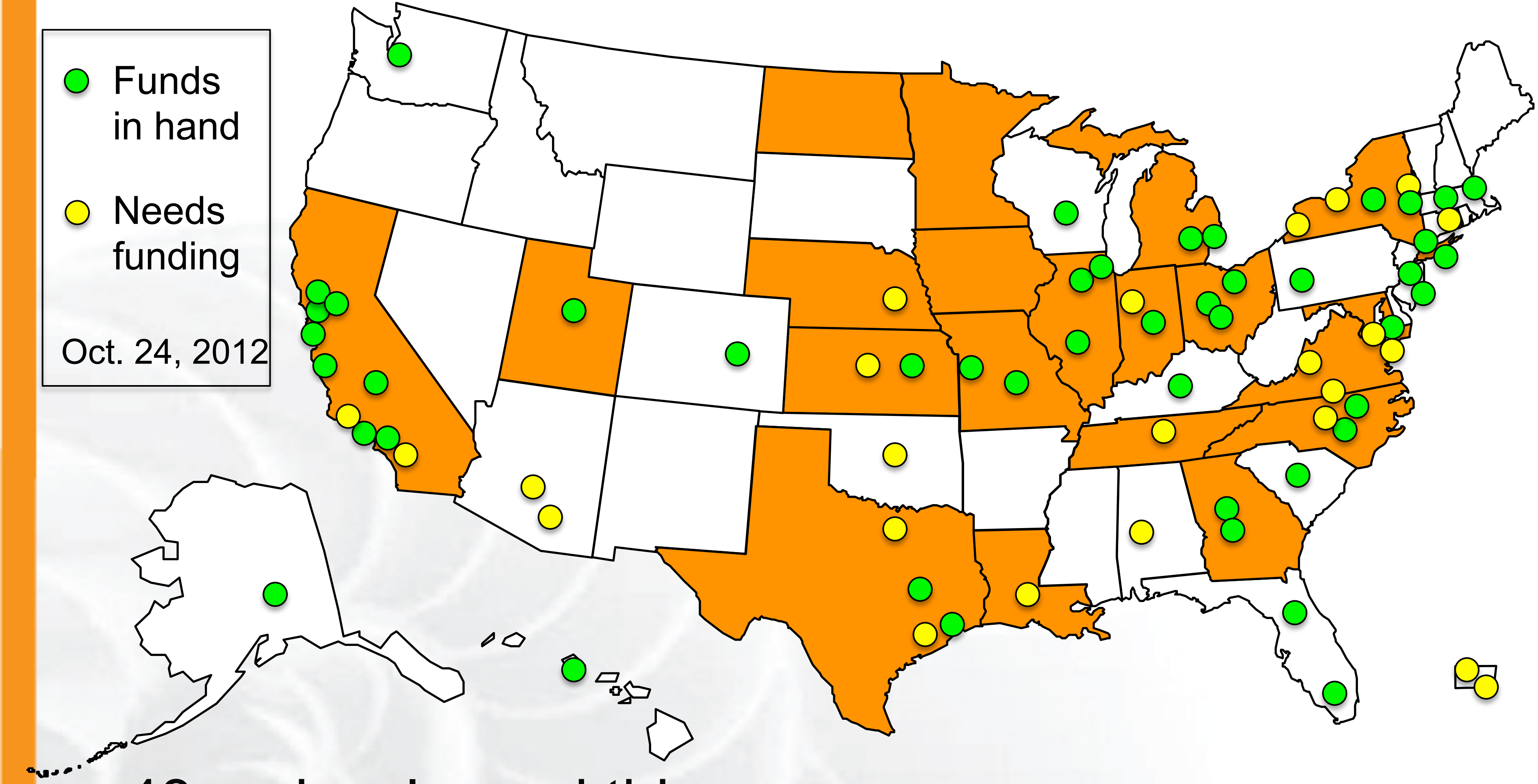
- GENI Rack projects are expanding available GENI infrastructure in the US.

- Racks provide reservable, sliceable compute and network resources using Aggregate Managers.

- GENI AM API compliance

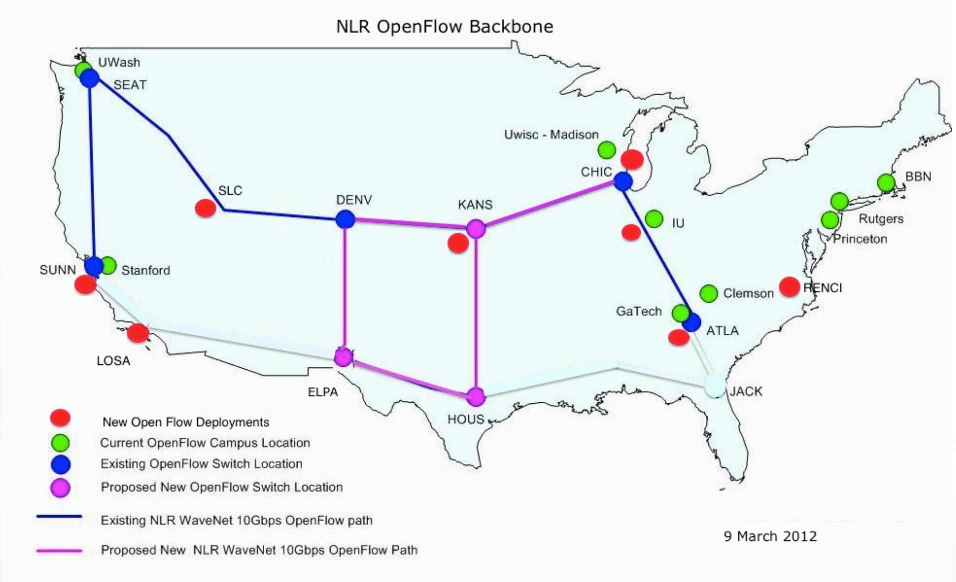


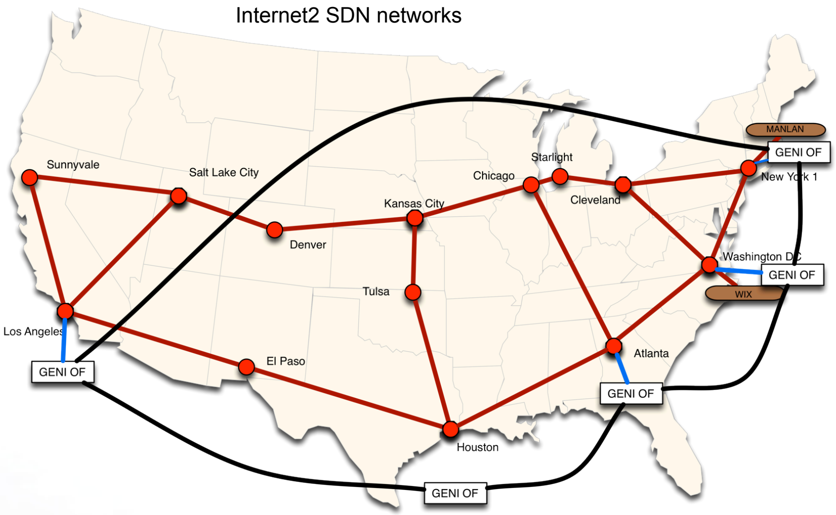
GENI Rack Campuses



- 43 racks planned this year
- Each rack has an **OpenFlow-enabled** switch

Core Networks


NLR OpenFlow Backbone
UWash
SEAT
SLC
DENV
KANS
Uwisc - Madison
CHIC
IU
BBN
Rutgers
Princeton
SUNN
Stanford
Clemson
RENCI
GaTech
ATLA
LOSA
ELPA
HOUS
JACK
New Open Flow Deployments
Current OpenFlow Campus Location
Existing OpenFlow Switch Location
Proposed New OpenFlow Switch Location
Existing NLR WaveNet 10Gbps OpenFlow path
Proposed New NLR WaveNet 10Gbps OpenFlow Path
9 March 2012



- NLR committed to 2013 meso-scale expansion following reorganization
- Internet2 adding 10GbE paths to Advanced Layer 2 Services (AL2S) at 4 of 5 OpenFlow meso-scale/ProtoGENI Pops
- GENI Aggregate Manager in Internet2 AL2S and dynamic stitching with GENI coming in Spiral 5

FOAM

- An OpenFlow Aggregate Manager
- It’s a GENI compliant reservation service
  - Helps experimenters reserve flowspace in the FlowVisor
- Speaks AM API v1
- Rspecs GENI v3, openflow v3 extension

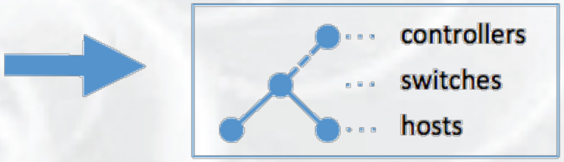
OpenFlow Experiments

Debugging OpenFlow experiments is hard:

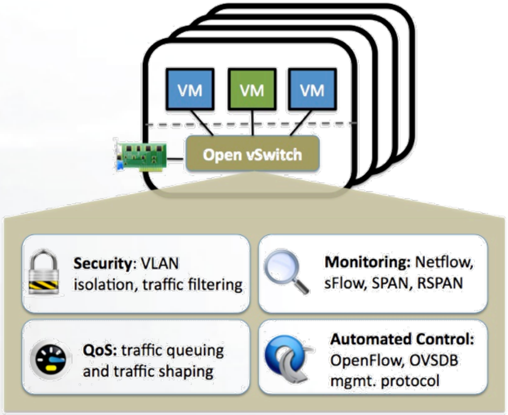
- Network configuration debugging requires coordination
- Many networking elements in play
- No console access to the switch

Before deploying your OpenFlow experiment test your controller.

```
> sudo mn
```



http://mininet.github.com/

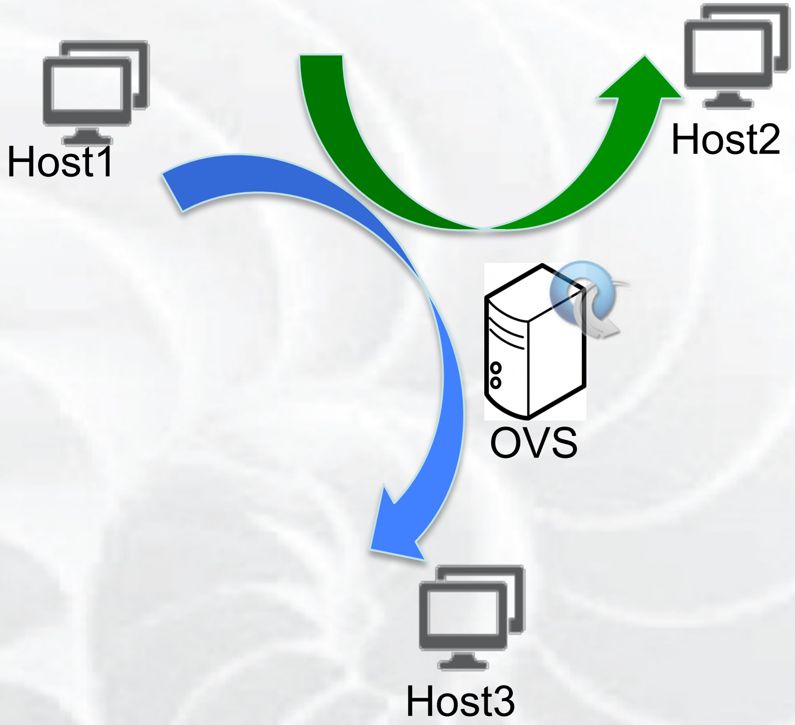

VM
VM
VM
Open vSwitch
Security: VLAN isolation, traffic filtering
Monitoring: Netflow, sFlow, SPAN, RSPAN
QoS: traffic queuing and traffic shaping
Automated Control: OpenFlow, OVSDB mgmt. protocol

http://openvswitch.org/

Run an OpenFlow experiment

1 host as OVS switch

3 VMs connected to OVS



- Setup OVS
- Write simple controllers
  - e.g. divert traffic to a different server
  - Use Python controller PoX

To Save Time…

- Your slices have been created for you
  - Slice name: oftutnn
- Resources have been added to your slice
  - 1 raw PC running OVS
  - 3 VMs that act as traffic sources & sinks
  - All resources from Emulab
- Shared public key installed on the resources
  - Private key: ~/.ssh/geni-shared-key
  - You will need to add this key to your ssh-agent
    - ssh-add ~/.ssh/geni-shared-key (password: gec17)
  - To login: nriga@pcxxx.emulab.net
- Luisa will add your account to the slice

Design/Setup
Execute
Finish

- Part I: Design/Setup
  - Obtain Resources
  - What is OpenFlow, what can I do with Openflow?
  - Demo: Using OpenFlow in GENI
- **Part II: Execute**
  - **Configure and Initialize Services**
  - Execute Experiment
- Part III: Finish
  - Teardown Experiment

Configure OVS

OVS is a virtual switch running on a bare metal node.

- The interfaces of the node are the ports of the switch
  - Configure an ethernet bridge
  - add all dataplane ports to the switch
- Can be an OpenFlow switch
  - Need to specify the controller (for convinience on the same host but it can be anywhere)
- Userspace OVS for this exercise

Design/Setup

Execute

Finish

- Part I: Design/Setup
  - Obtain Resources
  - What is OpenFlow, what can I do with Openflow?
  - Demo: Using OpenFlow in GENI
- **Part II: Execute**
  - Configure and Initialize Services
  - **Execute Experiment**
- Part III: Finish
  - Teardown Experiment

Experiments (1/4)

1. Verify connectivity with using a learning switch
    1. See the flow between host start and stop based on the controller
    2. Soft versus hard timeouts

Experiments (2/4)

2. Write a controller that will duplicate traffic to a different port on the switch
    1. Use tcpdump to see the duplication

Experiments (3/4)

3. Write a controller that will do port forwarding on your server
    1. Use netcat to run two servers on host2

Experiments (4/4)

3. Write a controller that will redirect packets to a proxy
   1. What fields do you need to overwrite?
   2. Which packets needs special handling?
   3. Use netcat to see the deflection

Part III: Finish Experiment

When your experiment is done, you should always release your resources.

- Normally this is when you would archive your data
- Delete your slivers at **each** aggregate