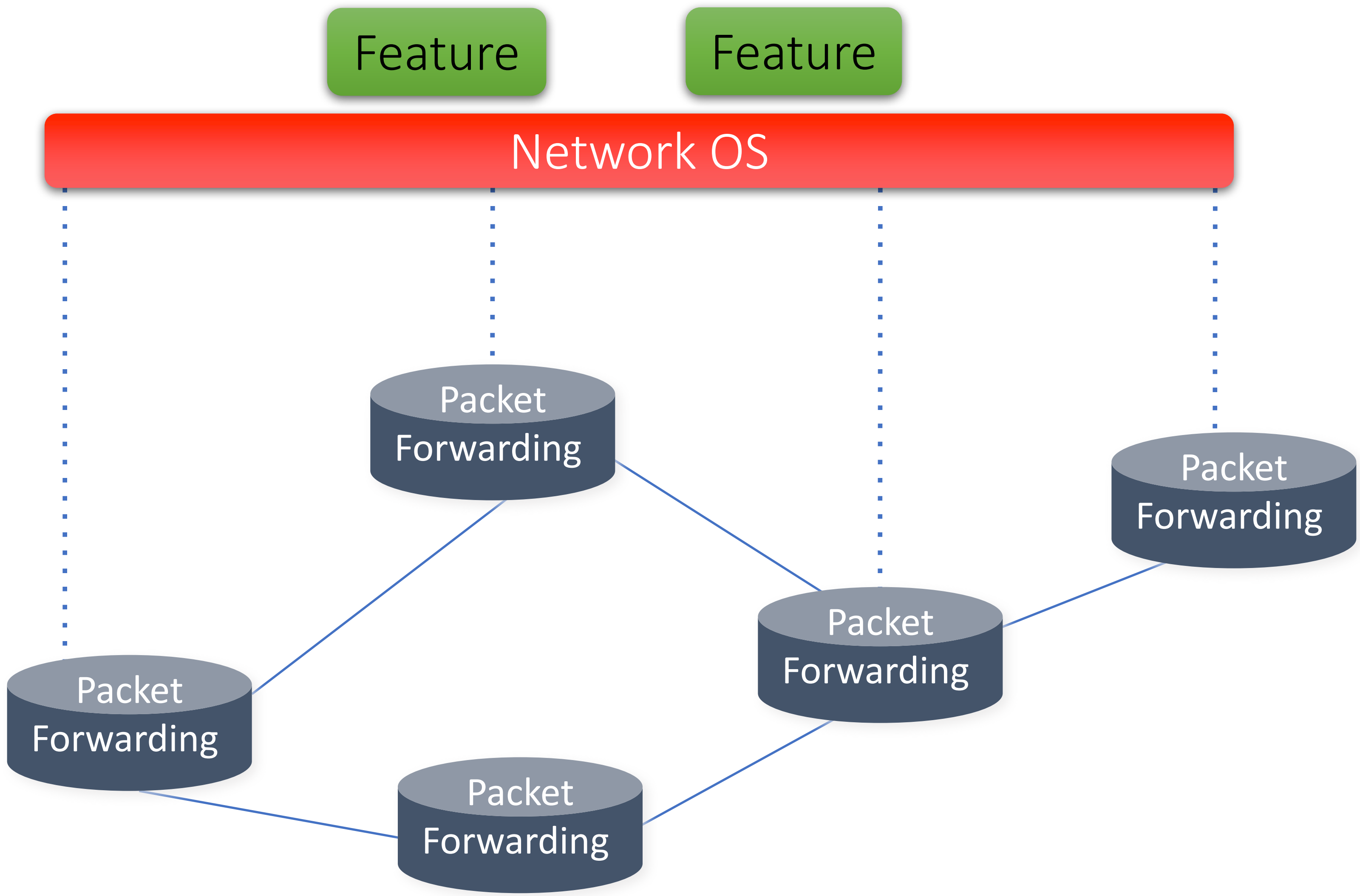
Software Defined Networking OpenFlow and NOX

ECE/CS598HPN

Radhika Mittal

Acknowledgements: Yashar Ganjali, Univ. of Toronto

Software Defined Network (SDN)

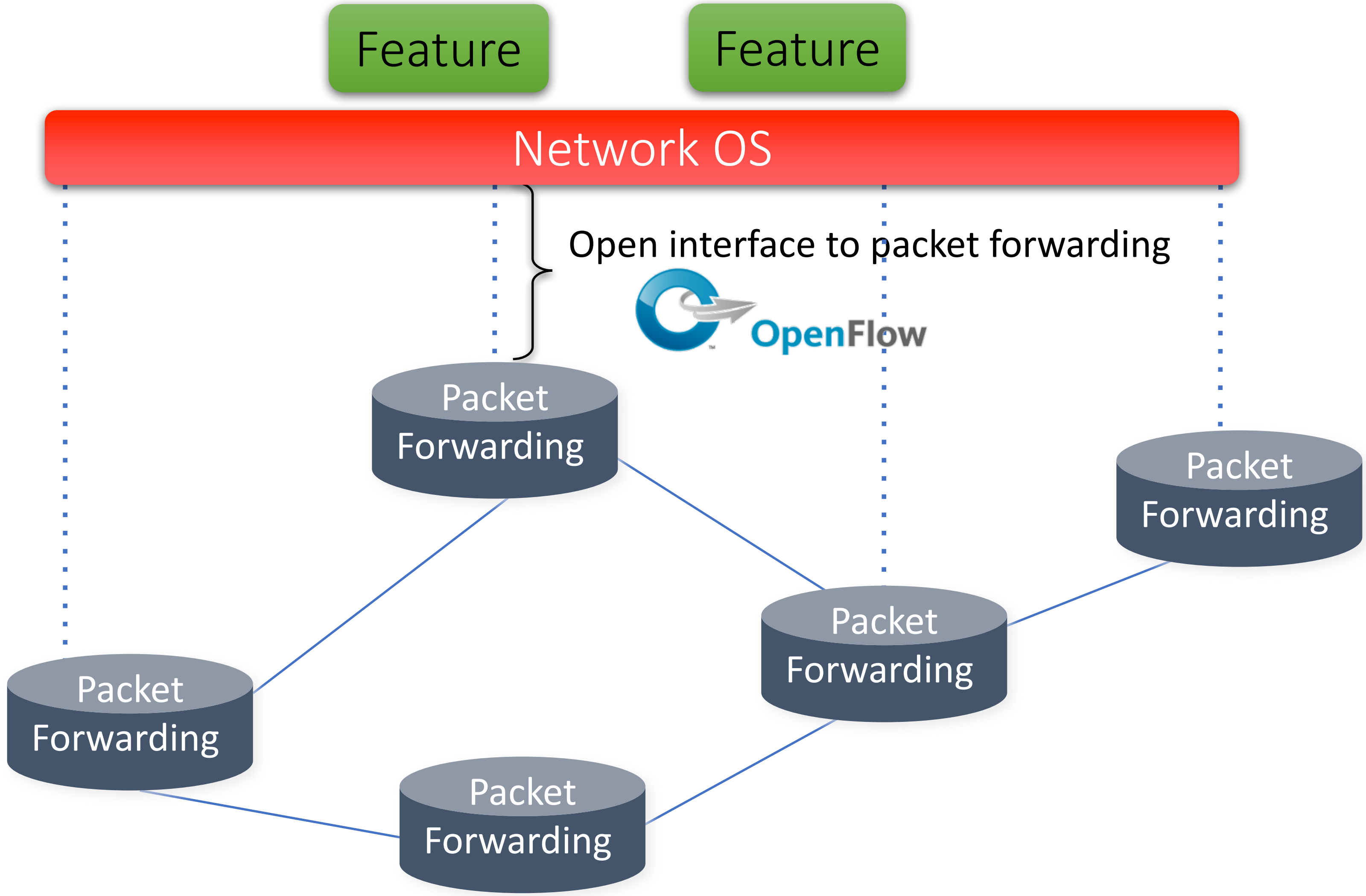


Abs#1: Forwarding Abstraction

- Express intent independent of implementation
  - Don't want to deal with proprietary HW and SW

- OpenFlow is a standardized interface to switch.

Software Defined Network (SDN)



OpenFlow

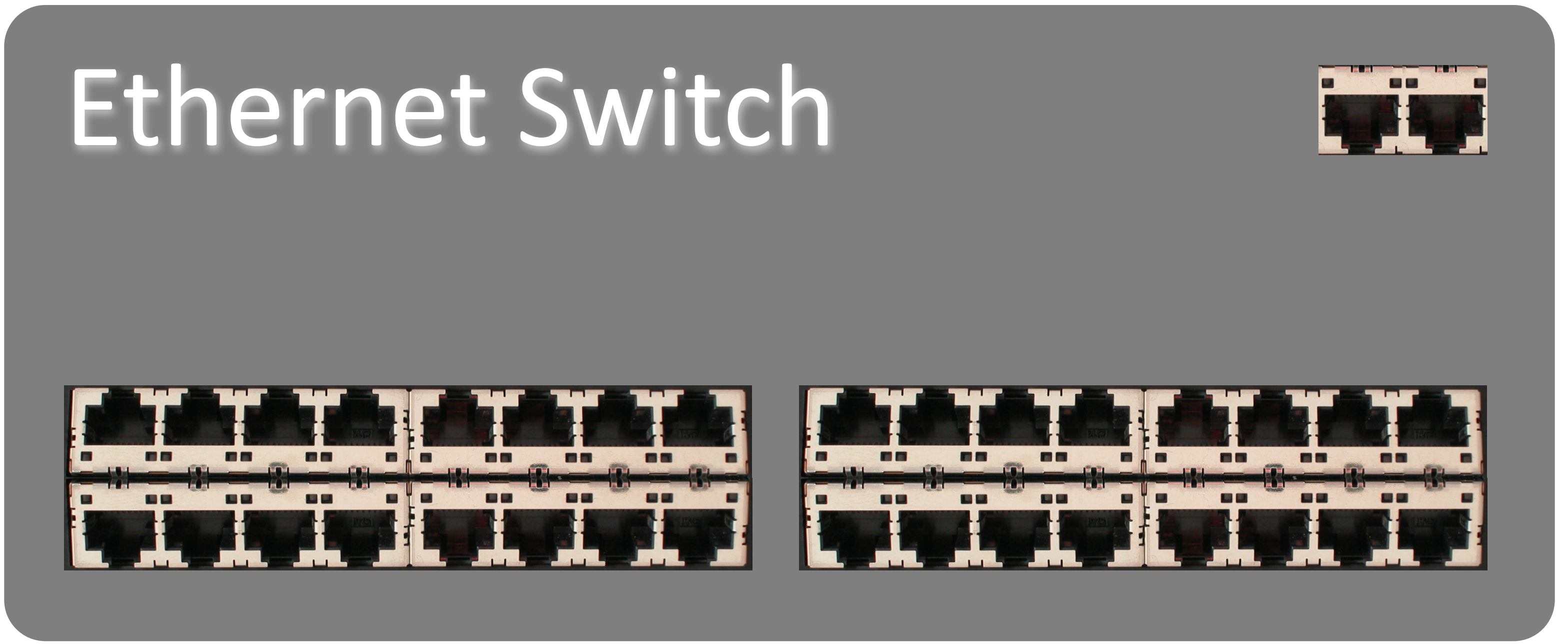
- **Initial objective:** Enable experimentation and innovation within universities.

- Developed at Stanford.

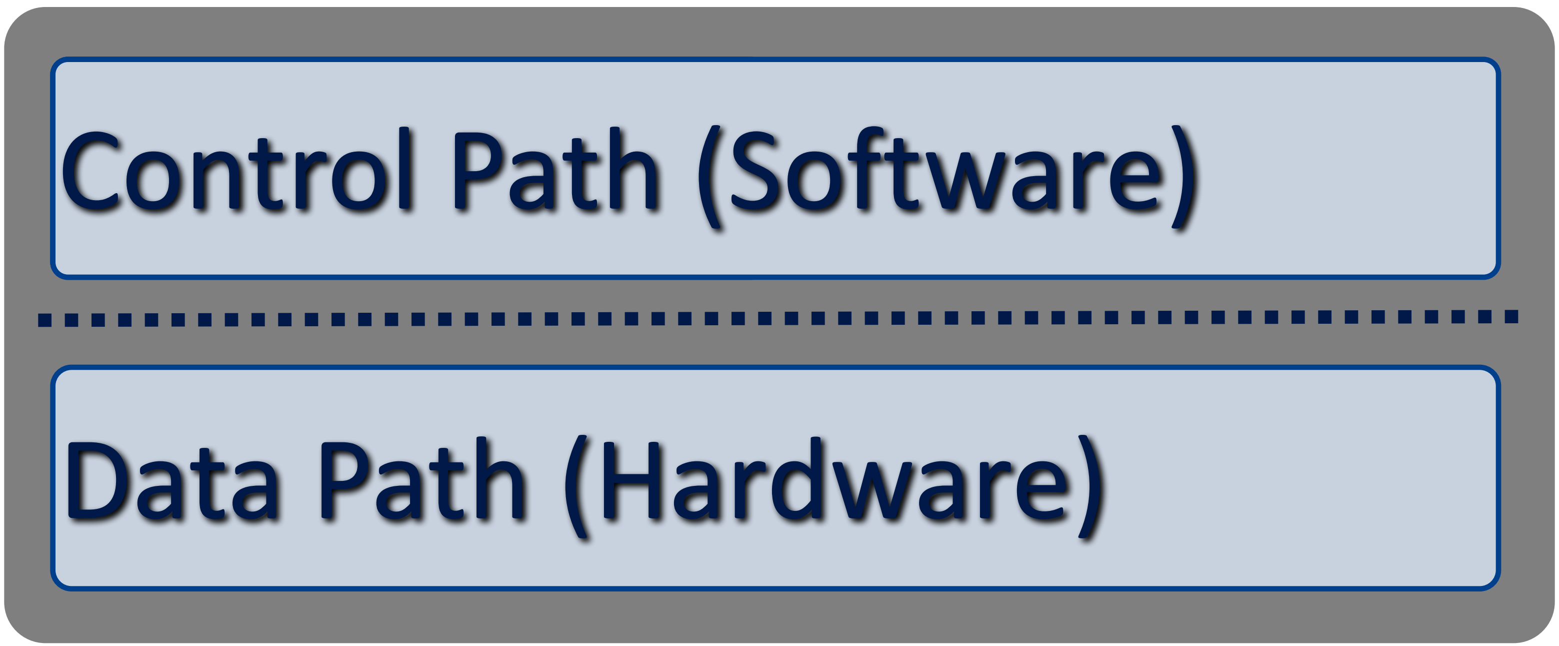
- Supported by various companies (Cisco, Juniper, HP, NEC, …)

- Now being used world-wide in industries.

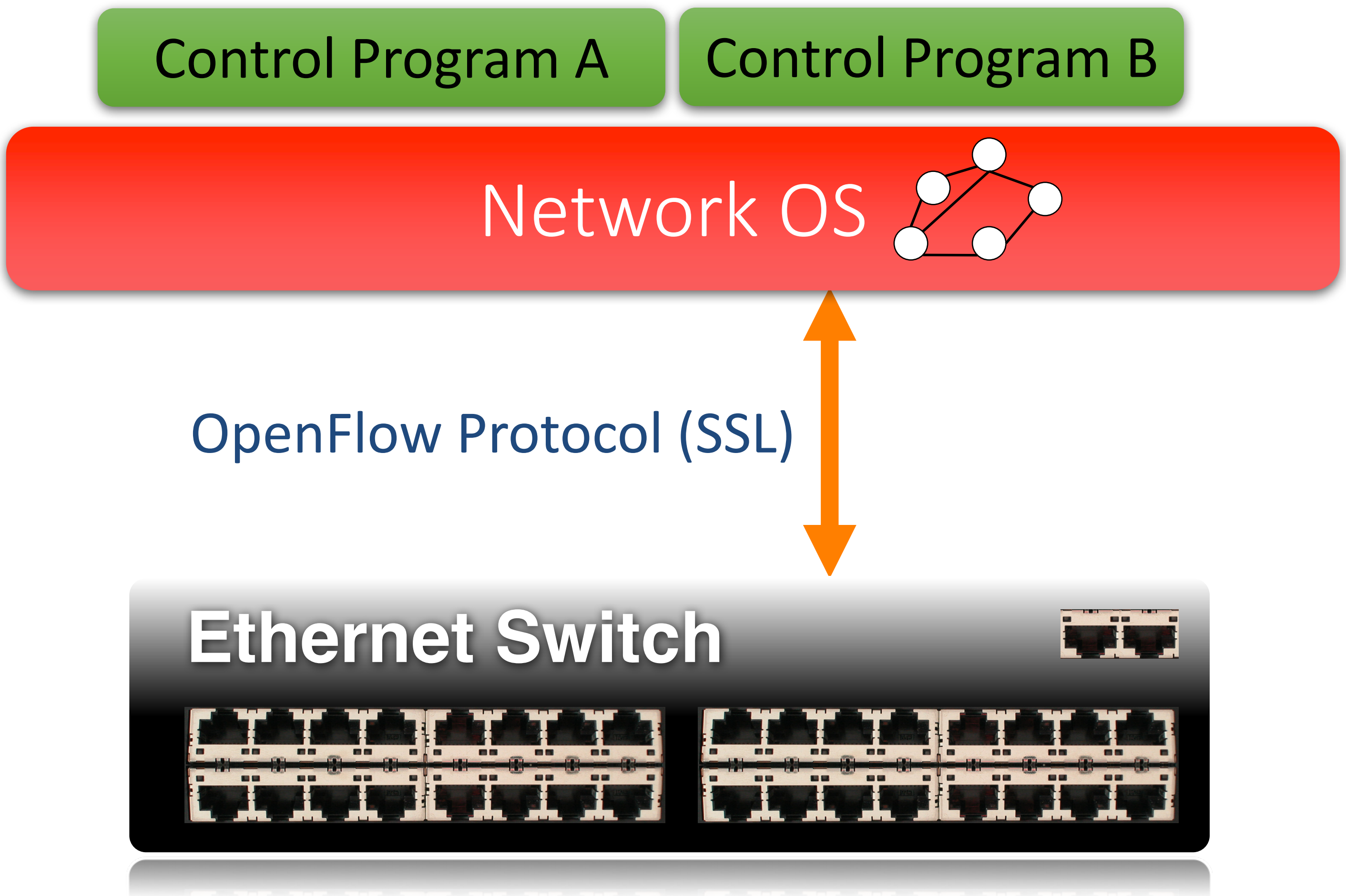
Traditional Switch



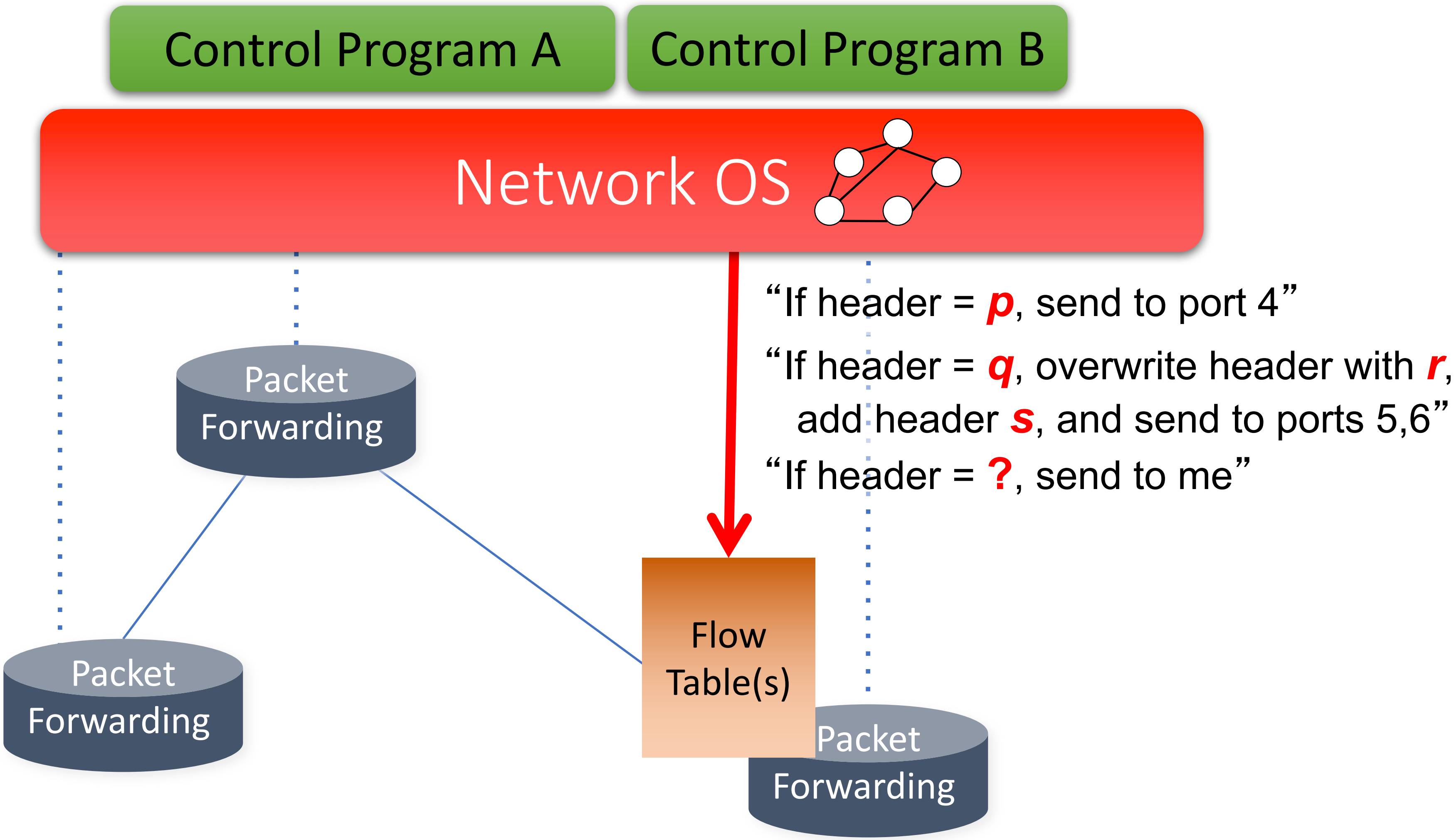
Traditional Switch



OpenFlow Switch



OpenFlow Rules



Match-Action Primitive

Match arbitrary bits in headers: Match: 1000x01xx0101001x

- Match on any of the supported header fields
- Allows any flow granularity

Action

- **Forward to port(s)**
- **Encapsulate and send to controller**
- **Drop**
- Rewrite packet headers, map to a particular priority level

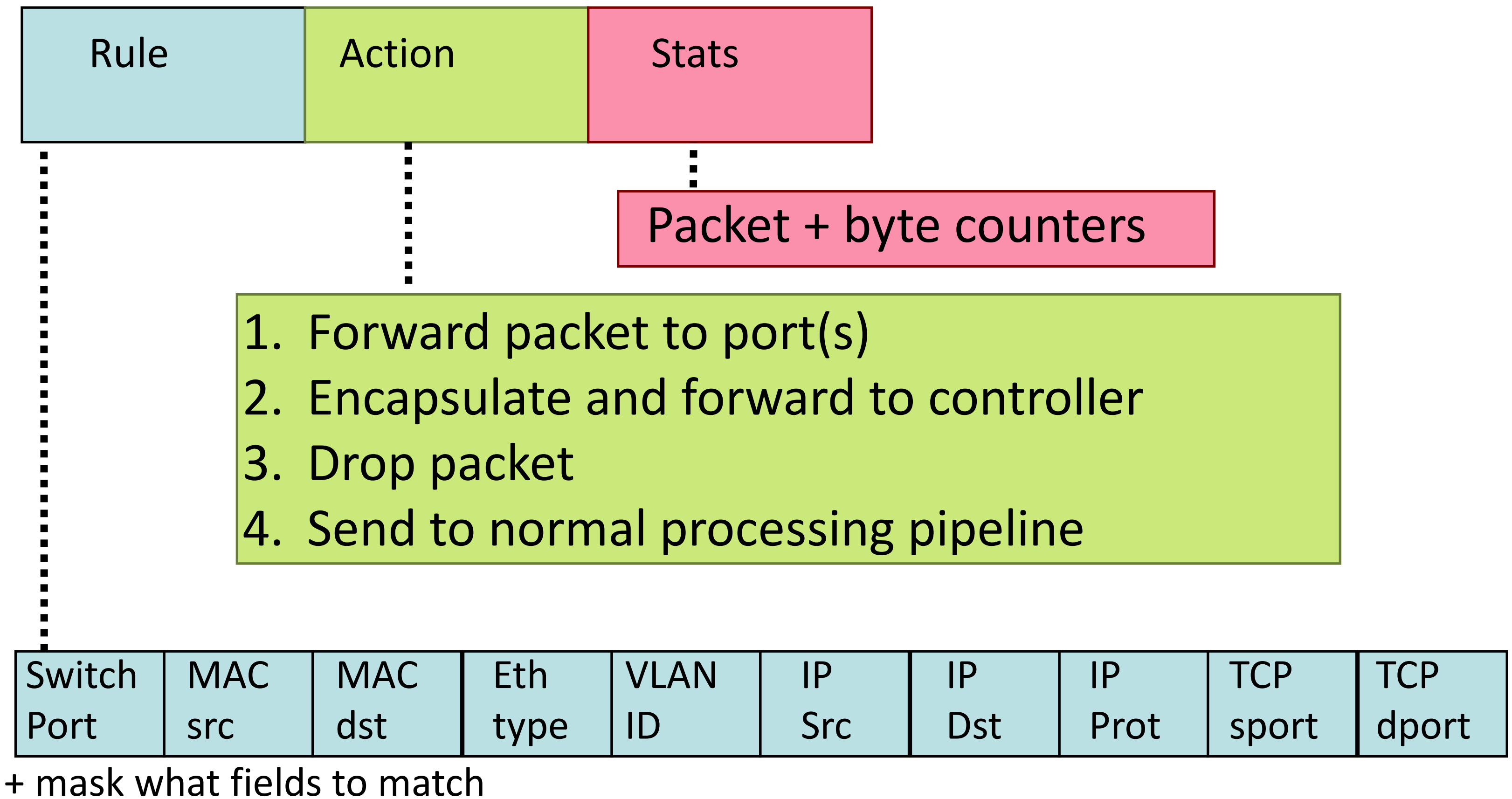
OpenFlow Rules – Cont'd

- Exploit the flow table in switches, routers, and chipsets

Flow 1.	Rule (exact & wildcard)	Action	Statistics
Flow 2.	Rule (exact & wildcard)	Action	Statistics
Flow 3.	Rule (exact & wildcard)	Action	Statistics
	……………………		
Flow N.	Rule (exact & wildcard)	Default Action	Statistics

Flow Table Entry

- OpenFlow Protocol Version 1.0



Flow Table Entry

- OpenFlow Protocol Version 1.0

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

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

+ mask what fields to match

Examples

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:2e..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	*	*	*	*	22	drop

Examples

Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

VLAN

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	vlan1	*	*	*	*	*	port6, port7, port9

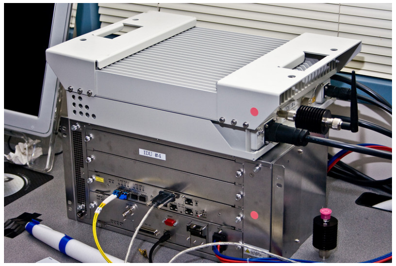
Supported Header Fields

Version	Date	# Headers
OF 1.0	Dec 2009	12
OF 1.1	Feb 2011	15
OF 1.2	Dec 2011	36
OF 1.3	Jun 2012	40
OF 1.4	Oct 2013	41

OpenFlow Switches

Juniper MX-series

NEC IP8800



WiMax (NEC)

HP Procurve 5400

Cisco Catalyst 6k

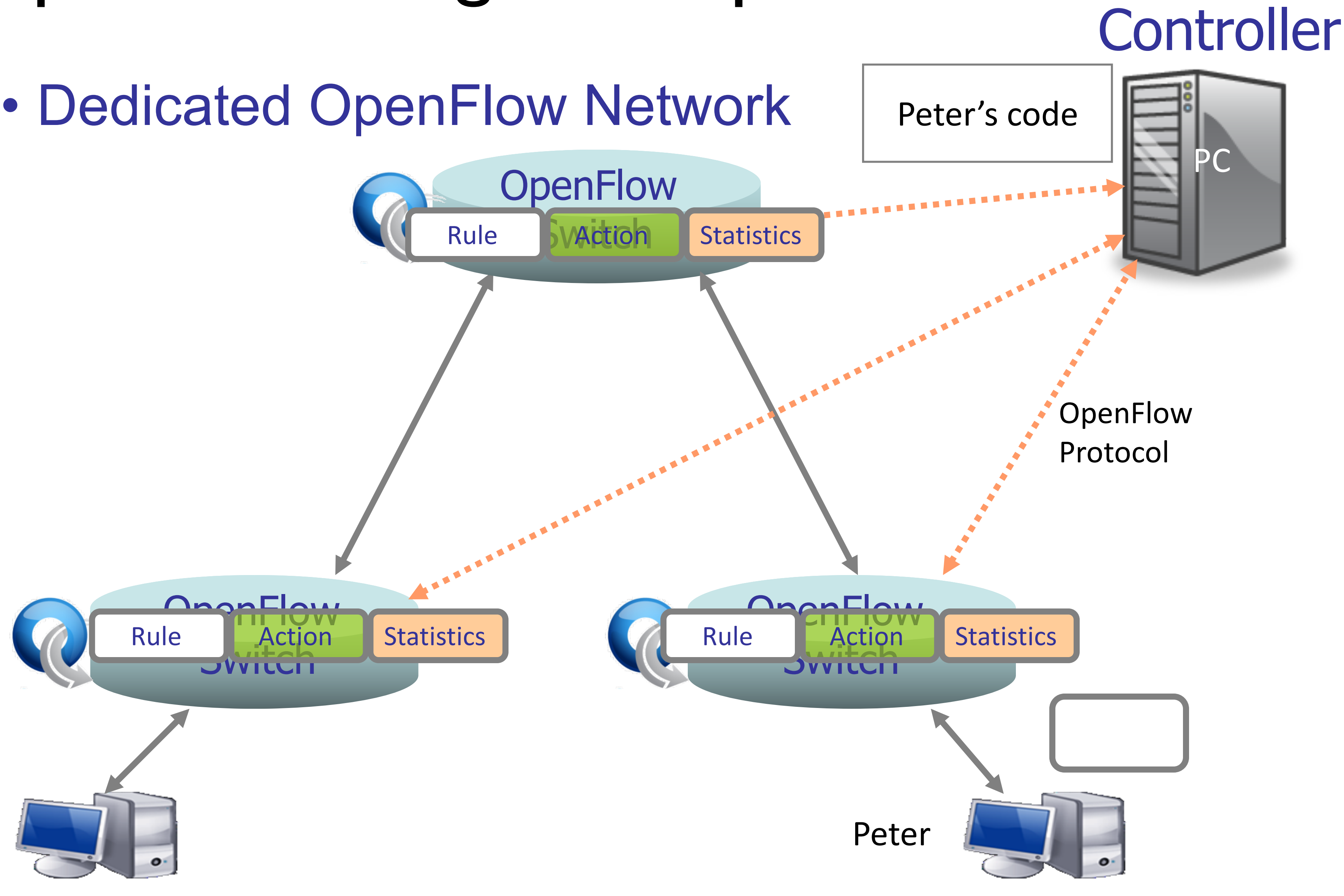
PC Engines

Quanta LB4G

More coming soon...

OpenFlow Usage Example

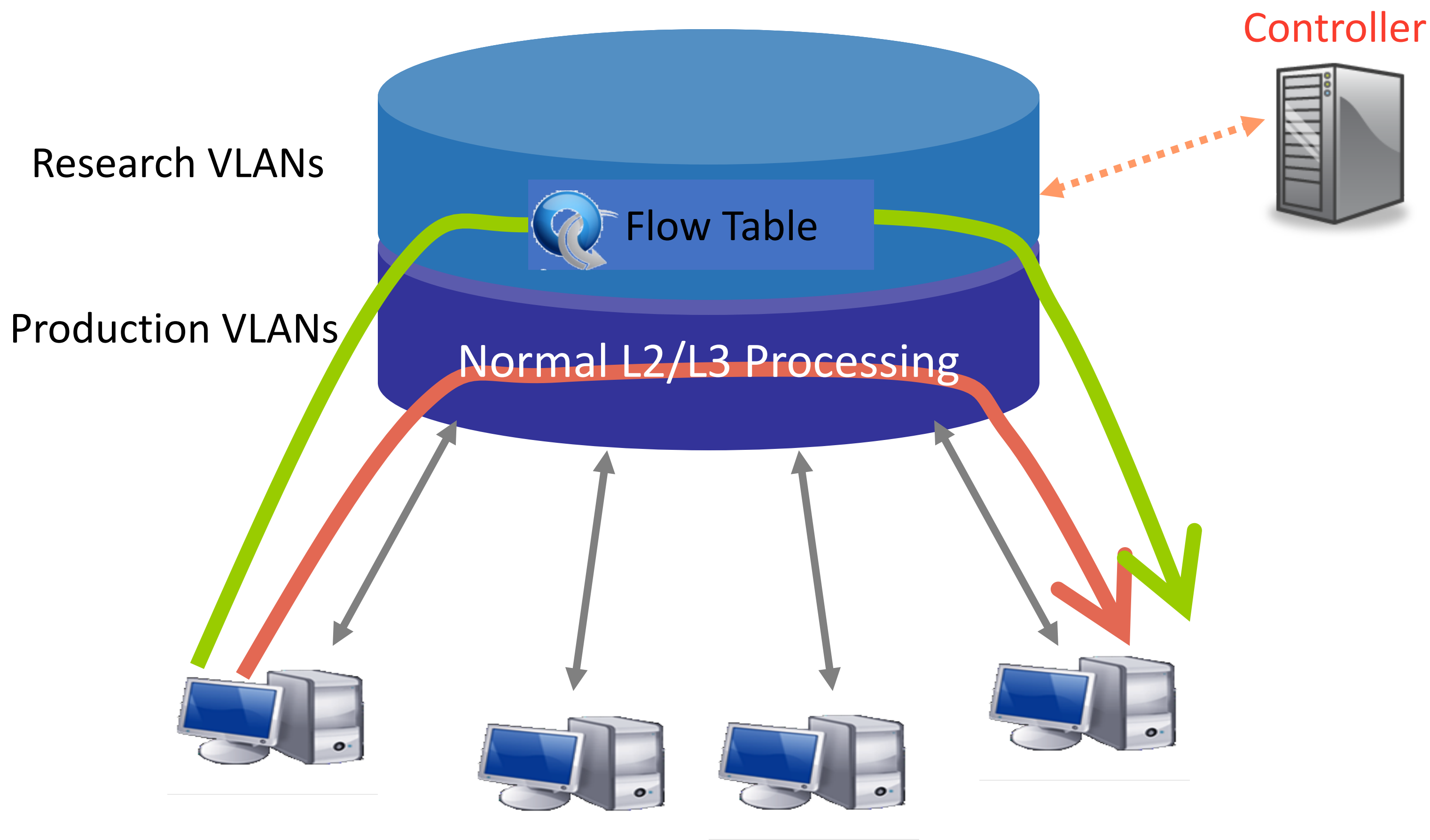
- Dedicated OpenFlow Network



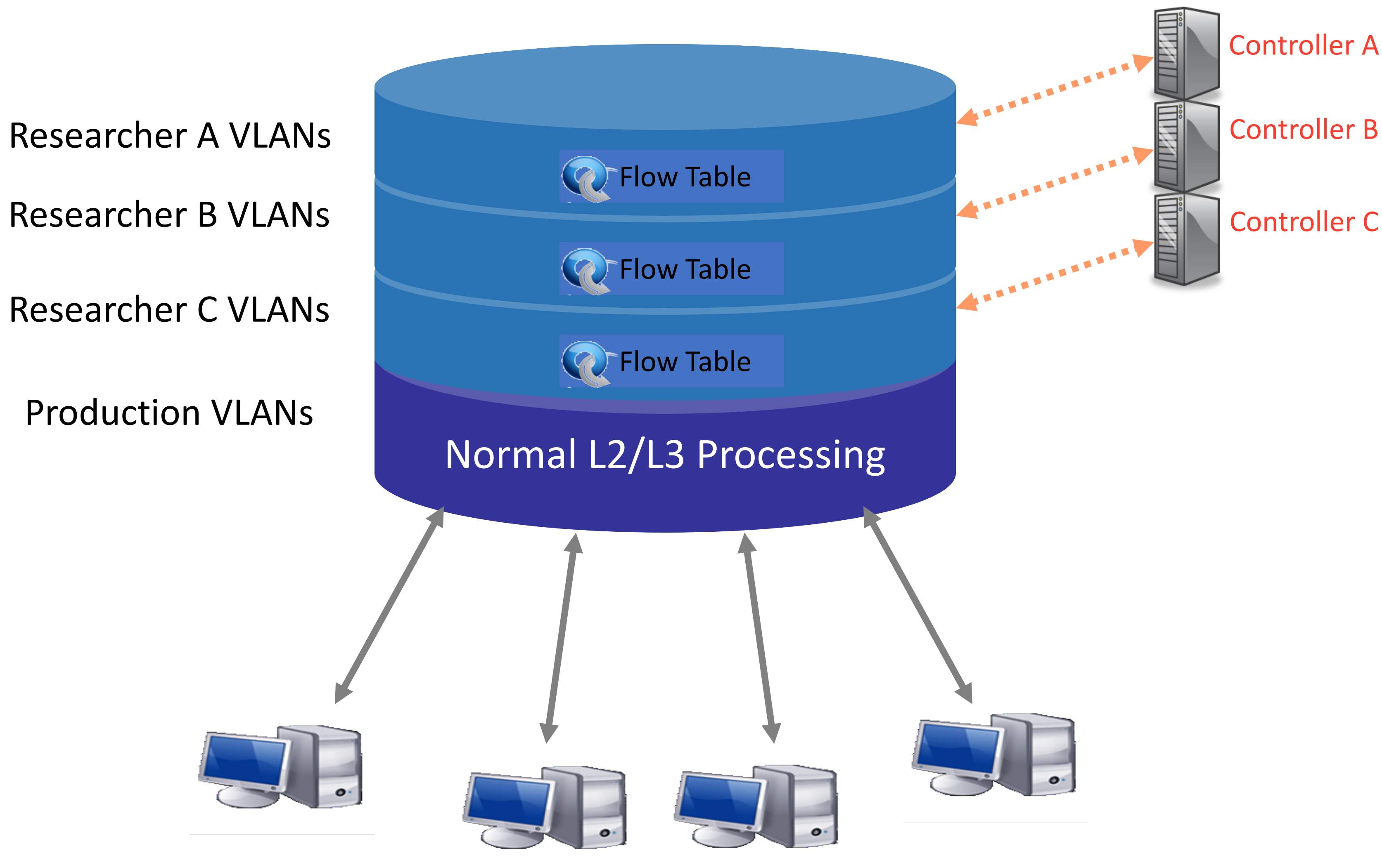
Usage examples

- Peter’s code:
  - Static “VLANs”
  - His own new routing protocol: unicast, multicast, multipath, load-balancing
  - Network access control
  - Home network manager
  - Mobility manager
  - Energy manager
  - Packet processor (in controller)
  - IPvPeter
  - Network measurement and visualization
  - …

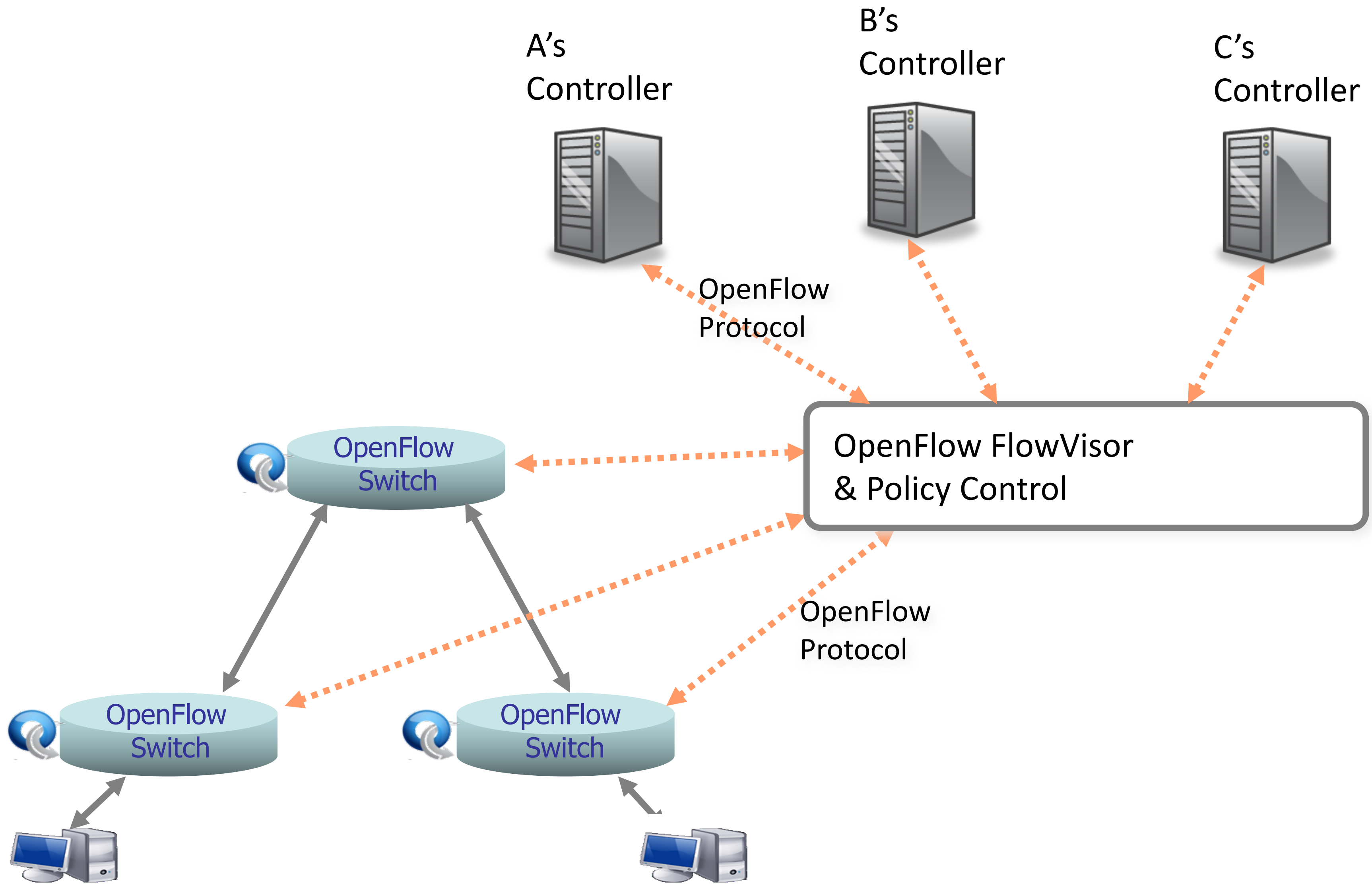
Research/Production VLANS



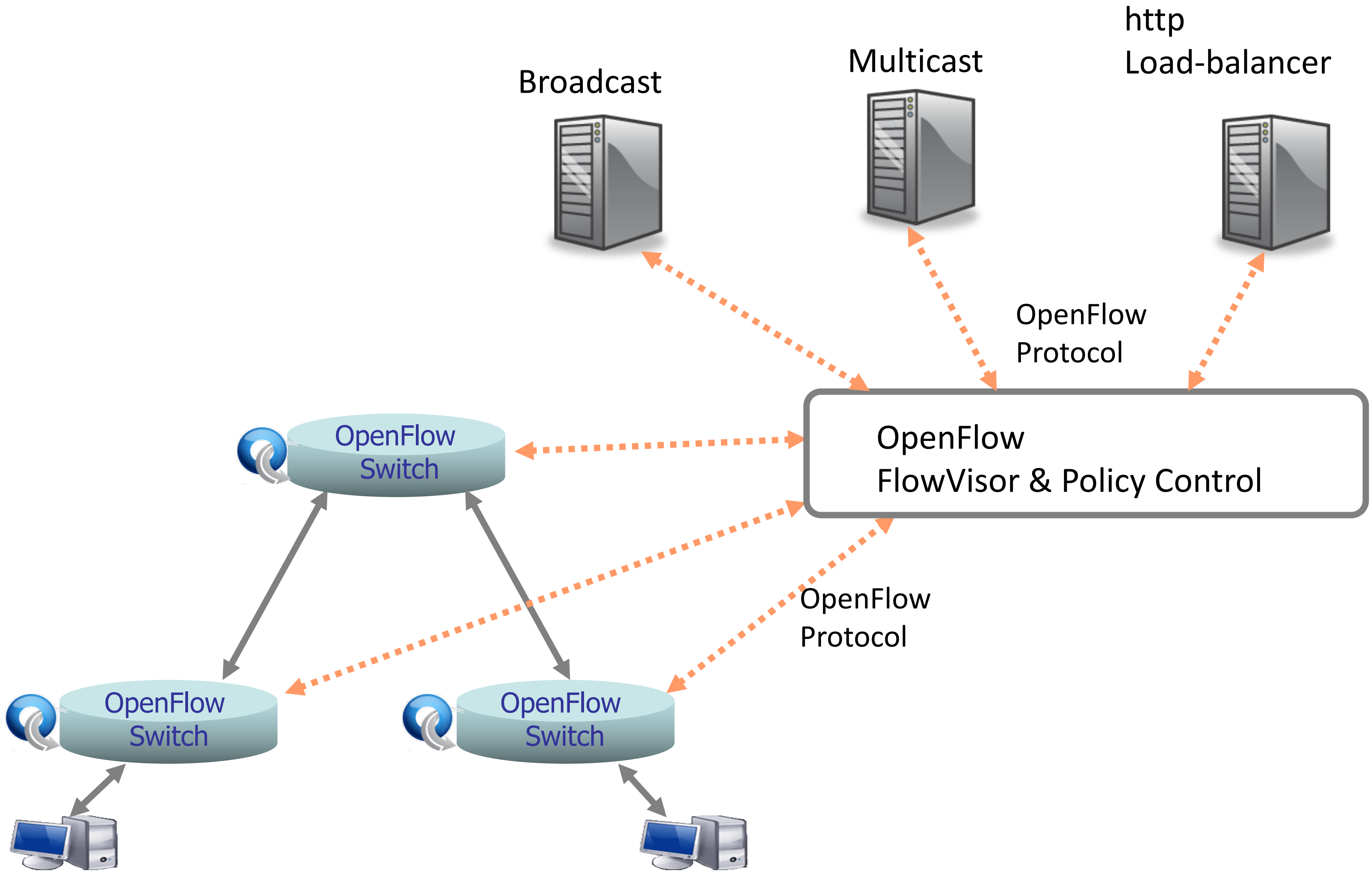
Virtualize OpenFlow Switch



Virtualizing OpenFlow



Virtualizing OpenFlow



Discuss!

- What are the challenges in switching from traditional networks to OpenFlow networks?
  - Performance
  - Security or DoS
  - Dealing with very large network, scalability
- What are the opportunities?
  - Test network without disrupting production
  - Functionality within switches, middleboxes (caching…)

OpenFlow -- your opinions

- Pros:
  - concrete, clear workflow, comprehensive examples, achievable
  - flexible packet format (somewhat)
  - use existing switch mechanisms -- flow tables
  - Not overly ambitious – first focus on campus networks

OpenFlow -- your opinions

Cons:

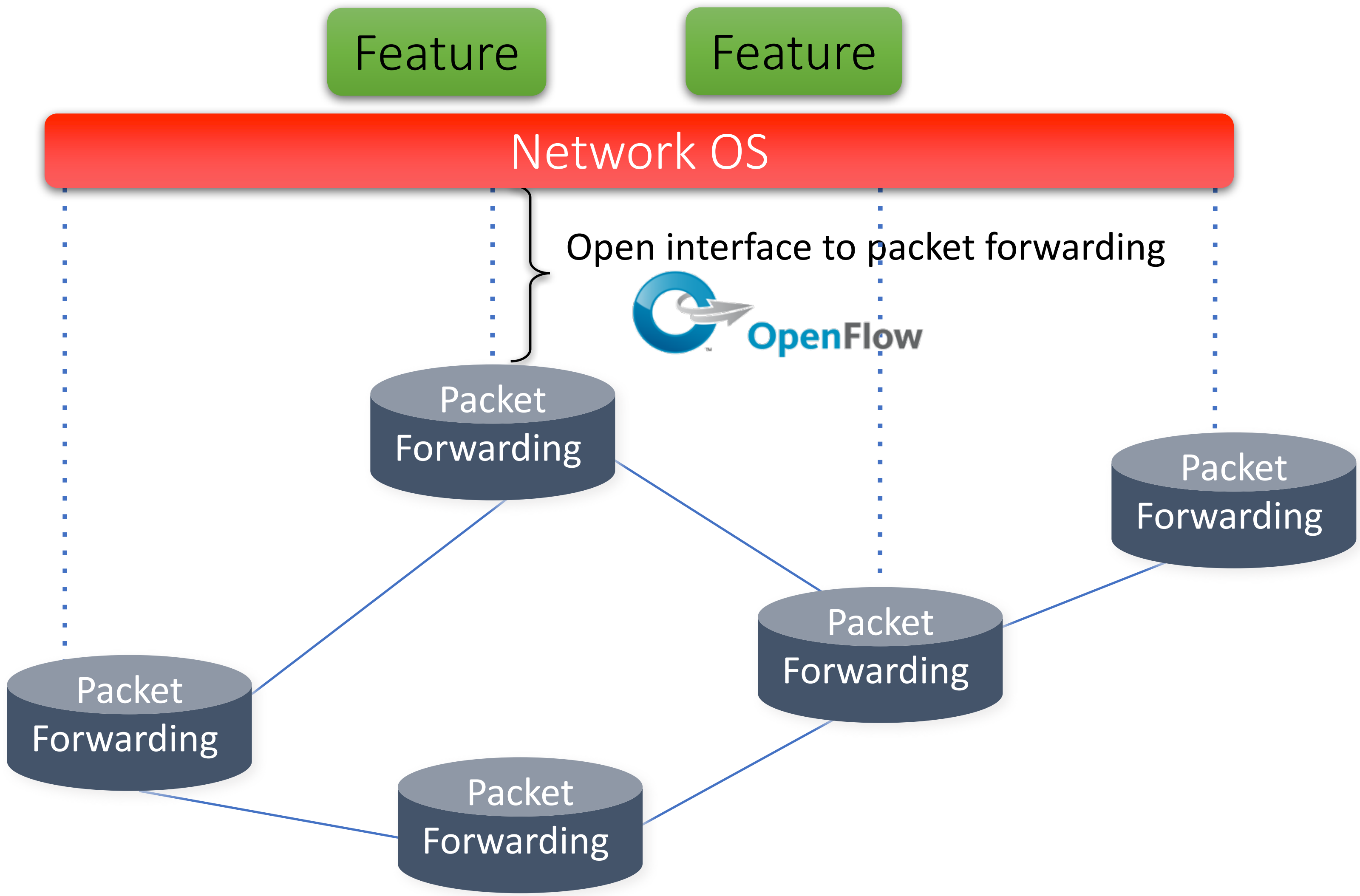
- Reliability of performance (?)
- Security (?)
- Performance (?)
    - Latency to the controller
- Size of flow table
- Incentive for vendors
- Impact on production traffic
- More details on controller
- Sharing resources across multiple OpenFlow users
- How to support multiple controller instances?

OpenFlow -- your opinions

Ideas:

- QoS for production and experimental traffic
- ML + controller for network resource regulation (?)
- Make OpenFlow more flexible and expressive
- Refactoring middlebox functionality using OpenFlow
- Evaluate scalability
- Use OpenFlow to handle link failures
- Can it really be deployed at large scale?

Software Defined Network (SDN)



Design choices for scalability

- Granularity of network view
  - Topology (switches, hosts, middleboxes)
  - Bindings between names and addresses
  - *Exclude network traffic state.*

- Granularity of control
  - Per-packet control will not scale.
  - Prefix-based control too coarse-grained.
  - Use *flow-based* control.

Scalability Argument

0 - 10/s	Modification of Control Program	Strong Consistency
$10^1 – 10^3$/s	Per Network Event	Eventual Consistency
$10^3 – 10^6$/s	Per Flow	No Consistency
$10^6 – 10^8$/s	Per Packet	No Consistency

Implication

- Can replicate controllers.
- Each replica can independently handle flow initiations.
- With network change events being less frequent, a consistent network view can be maintained across replicas.

Discuss!

- Do you buy the scalability argument?

- Are there any other concerns?

NOX was just the beginning…

- Support different languages
  - POX: Python
  - OpenDaylight, Floodlight, ONOS, Beacon, Maestro: Java
  - Onix: C++
  - ….
- Improved APIs/flexibility/scalability:
  - Maestro: exploit mutli-core parallelism.
  - Onix: richer state (network information base), that is replicated and distributed across instances.
  - Many many more…..

NOX -- your opinions

- Pros:
  - ''flow'' granularity – trade-off flexibility and scalability
  - OS-like abstraction -- multiple applications
  - Functional prototype
  - Good motivation, examples

NOX -- your opinions

Cons:

- Controller energy consumption
- No experimental results
- What are the pitfalls?
- How well can it scale?
  - Costly to maintain network view?
- Performance issues?
- Security issues?
- How to handle packet losses?

NOX -- your opinions

Ideas:

- What level of consistency is required for network state?
- More functionality
- Evaluation performance and scalability
- What if network topology changes very rapidly?
- More powerful distributed algorithm?