eecs589

ADVANCED COMPUTER NETWORKS

Sherwood, R., et al., " Can the Production Network Be the Testbed?" *Proc. of the 9th USENIX Symposium on OSDI*, 2010

Reference:
[C+07] Cascado et al., "Ethane: Taking Control of the Enterprise," ACM SIGCOMM '07, 37(4):69-74, Oct. 2007

Slicing a Network

Want to create virtual networks from slices of physical network

Each virtual network forwards traffic at line speed: no extra overhead in packet forwarding (data plane), no extra overhead in the forwarding rule specifications (control plane)

Slicing isolates bandwidth, switch CPU, and flow table entries between virtual networks

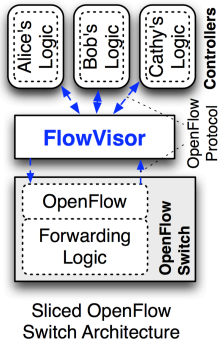
FlowVisor

Assumes software-defined network with separate control and data planes

Built on OpenFlow switches: NC and switches communicate using OpenFlow protocol

Provides network slicing by adding a layer between the control and data planes

Extra overhead in the communication between an OpenFlow switch and the centralized NC

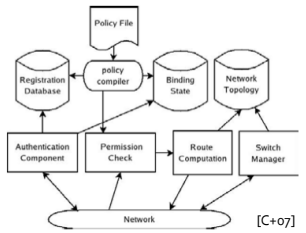


Sliced OpenFlow Switch Architecture

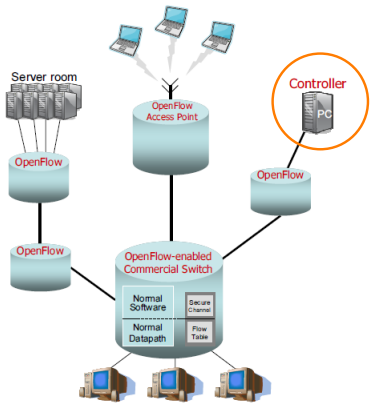
Software-Defined Network (SDN)

Centralized Network Control (NC)

- monitors and approves all traffic
  - allows for complete policy-based control of the network
- creates and populates switches with forwarding rules
- access controls built in
  - network understands users, hardware, topology, and policies

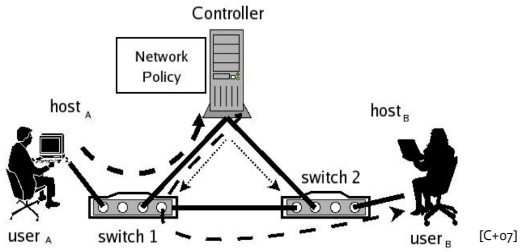


[C+07]



Flow Setup Process

1. User$_A$ tries to connect to User$_B$
2. User$_A$-to-User$_B$ "flow" isn't in Switch 1's flow table, so the packet is queued and the NC "notified"
3. NC either approves or denies route
4. If approved, NC adds a new rule into Switch 1's and Switch 2's flow tables to establish a flow from User$_A$ to User$_B$

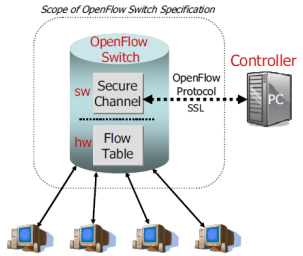


[C+07]

SDN Switches and OpenFlow

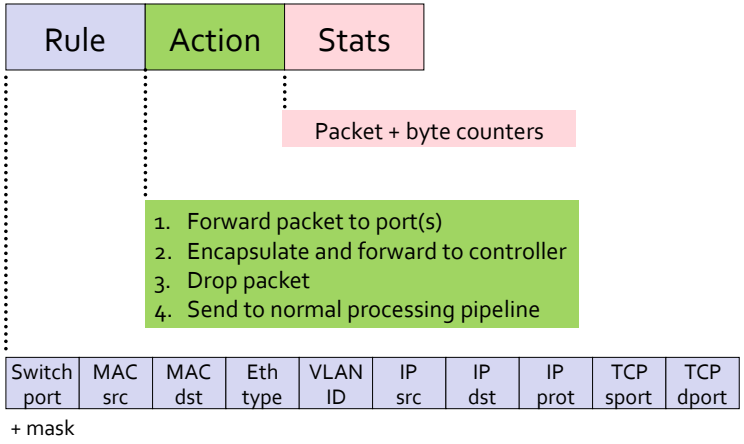
Switch forwarding controlled by NC
- communicates with controller over a secure channel
- OpenFlow is an open standard NC-switch communication protocol

Assume simple, off-the-shelf switches
- minimal on-board logic
  - "flow" table lookup only
  - only stores active flows
  - no understanding of network topology
  - no NAT knowledge
- OpenFlow standard specifies lowest common denominator hardware features exposed to NC's control



Flow Table Entry

Type 0 OpenFlow Switch



Network Slice Definition

A network slice is specified in terms of topology, bandwidth, switch CPU rate, forwarding table quota, and the set of flows that the slice controls

Traffic handled by a slice is defined by bit patterns in packet headers (flowspace)

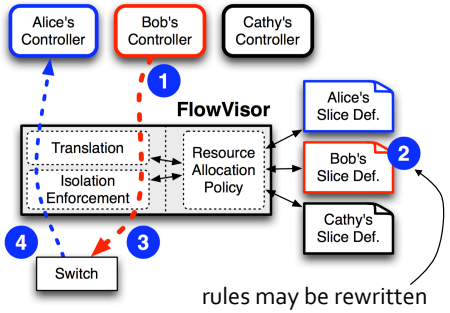
Each slice has its own control plane that defines how packets are forwarded and rewritten in the slice, e.g., Bob's HTTP load-balancer slice specifies:
- topology: encompassing the web servers
- flowspace: comprising flows with port 80

Slice set up is done manually in the prototype

Network Slice Implementation

FlowVisor intercepts and rewrites OpenFlow messages between NC and switches to enforce that:

- NC → switch:
  - forwarding rules only apply to the traffic and topology of the slice and observe resource quota
  - rules may be rewritten, e.g.,
    `all traffic → port 80`
    `all ports → ports in slice`
- switch → NC:
  - only messages from switches in the slice's topology are forwarded to its NC
  - port-related messages are pruned or rewritten such that NC only sees relevant ports



FlowSpace Definition

Flowspace specified (manually) as an ordered list of tuples similar to firewall rules, example:

Bob's HTTP load-balancer network:

```
Allow: tcp-port: 80 and ip=Doug'sIP
Allow: tcp-port: 80 and ip=Eric'sIP
```

Implications:

- new HTTP flow notifications with Doug's or Eric's IPs (non-contiguous flowspace) are all sent to Bob's NC
- any flow table entries Bob's NC tries to add are modified to match only HTTP traffic with Doug's or Eric's IPs

FlowSpace Definition

Alice's production network:

```
Deny: tcp-port:80 and ip=Doug'sIP
Deny: tcp-port:80 and ip=Eric'sIP
Allow: all ; lowest priority rule
```

Implications:

- only OpenFlow messages not intended for Bob's NC are forwarded to the production network's NC
- the production network's NC is not allowed to add any forwarding entries for HTTP traffic with Doug's or Eric's IPs

Resource Isolation

Bandwidth isolation: relies on hardware capability exposed to OpenFlow to assign fractional link bandwidth to user-created queue

Flow table entry isolation: limit the number of entries per slice, must take into account any automatic rule expansion, e.g., when the rule applies to multiple input ports

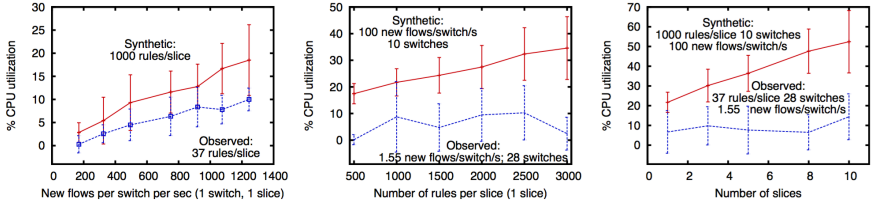
Resource Isolation

switch CPU isolation: hardware capabilities to rate limit CPU usage are usually not exposed to OpenFlow, instead relies on work around:

- if new flow arrivals exceeds some threshold, insert a lowest priority, time-limited forwarding rule to drop all packets matching the rule (e.g., drop all HTTPs packets not belonging to existing flows)
- manually rate limit NC's OpenFlow requests to switch
- rewrites "slow-path" forwarding rules to one-time forwarding rule
- manually tune the above rate limits to ensure sufficient CPU for internal bookkeeping

Scaling

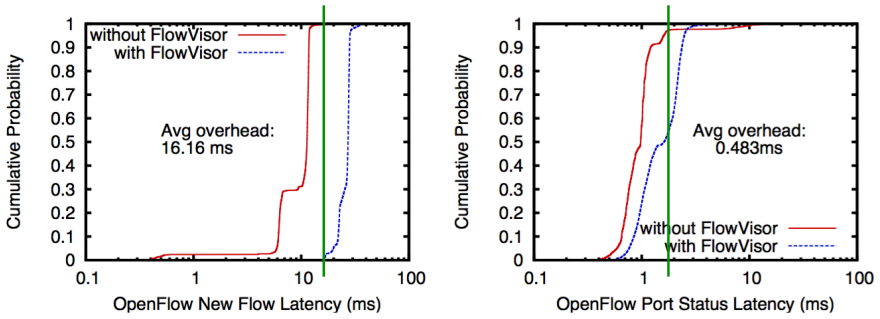
FlowVisor scales linearly with new flow rate, number of rules/slice, and number of slices



Performance Overhead

FlowVisor adds extra overhead only to OpenFlow messages:

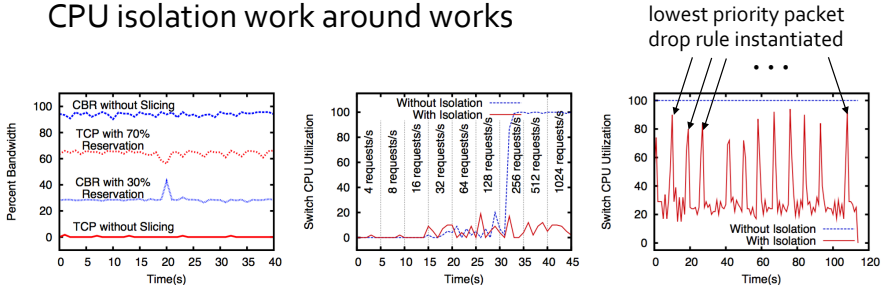
- switch ⟶ NC: new flow messages, affects connection setup latency
- NC ⟶ switch: port status requests, must be rewritten to match topology



Isolation

Hardware bandwidth isolation works

CPU isolation work around works



rate limiting NC requests

capping new flow setups

Deployment Issues

Incompatibilities with hardware features, e.g., multiple physical interfaces mapped into one logical interface

OpenFlow spanning tree does not match underlying spanning tree for loop detection

Different OpenFlow messages have different costs and other practical realities

Limitations

Prototype requires a lot of manual set up

OpenFlow doesn't expose many hardware capabilities

FlowVisor doesn't allow for deep packet inspection and other arbitrary packet modification, e.g., payload processing