Customizing SDN Control Plane Actions

- still need the “brains” - a controller
  - set flow table entries in switches being controlled
  - respond to event from switches
  - controller communicates with controlled switches using OpenFlow messages

- many controller environments exist, operating over OpenFlow:
  - NoX, PoX, Beacon, Floodlight, Maestro, Ryu, Frenetic

- event-based programming paradigm: controller programmed to respond to events:
  - switch joins network (initialize forwarding tables in controller)
  - link changes reported by network (recompute routes, change forwarding tables)
Simple PoX Controller example

- one Openflow switch
- 3 hosts
- one controller running PoX over Openflow
- **goal:** controller configures switch to broadcast each incoming frame over all outgoing links (function like a hub)
Simple PoX Controller example

def launch():
core.openflow.addListenerByName("ConnectionUp", _handle_ConnectionUp)
log.info("Controller launched and running.")

- launch() method adds listener to listen for OpenFlow switch to connect to controller

```
def _handle_ConnectionUp(event):
    msg = of.ofp_flow_mod()
    msg.actions.append(of.ofp_action_output(port = of.OFPP_FLOOD))
    event.connection.send(msg)
    log.info("Hubifying %s", dpidToStr(event.dpid))
```

- when switch connects, send back OpenFlow message switch to set its flowtable so that each incoming frame is flooded onto all links

(check out class readings for more examples)
Course outline: design principles

- **Network virtualization**: VLANs, VPN, virtualized sliced networks, NFV
- **Randomization**: Multiple access, routing, power of two choices
- **Indirection**: I3, Mobile IP
Network Virtualization

**virtualization**: build services based on abstraction of lower layer functions

- computing examples: virtual memory, virtual devices
  - virtual machines: e.g., java
  - IBM VM os from 1960’s/70’s
- layering of abstractions: don’t sweat the details of the lower layer, only deal with lower layers abstractly
Network Virtualization

- VLANS
- VPNS
- virtual nets
- GENI
- network function
- virtualization
- NFV

Timeline:
- 1960
- 1970
- 1980
- 1990
- 2000
- 2010
VLANs: motivation

consider:

- CS user moves office to EE, but wants to connect to CS switch?
- single broadcast domain:
  - all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN
  - security/privacy, efficiency issues
VLANs

Virtual Local Area Network

Switch(es) supporting VLAN capabilities can be configured to define multiple virtual LANS over single physical LAN infrastructure.

Port-based VLAN: switch ports grouped (by switch management software) so that single physical switch ……

… operates as multiple virtual switches

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-16)
Port-based VLAN

- **traffic isolation:** frames to/from ports 1-8 can only reach ports 1-8
  - can also define VLAN based on MAC addresses of endpoints, rather than switch port

- **dynamic membership:** ports can be dynamically assigned among VLANs

- **forwarding between VLANs:** done via routing (just as with separate switches)
  - in practice vendors sell combined switches plus routers
VLANS spanning multiple switches

- **trunk port**: carries frames between VLANS defined over multiple physical switches
  - frames forwarded within VLAN between switches can’t be vanilla 802.1 frames (must carry VLAN ID info)
  - 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports
802.1Q VLAN frame format

802.1Q frame

Tag Control Information (12 bit VLAN ID field, 3 bit priority field like IP TOS)

Recomputed CRC

2-byte Tag Protocol Identifier (value: 81-00)
Virtual Private Networks (VPN)

VPN networks perceived as being private networks by customers using them, but built over shared infrastructure owned by service provider (SP)

- **SP infrastructure:**
  - backbone
  - provider edge devices

- **Customer:**
  - customer edge networks and devices (communicating over shared backbone)
VPN reference architecture
VPN: logical view

virtual private network
Leased-line VPN

customer sites interconnected via static virtual channels (e.g., ATM VCs), leased lines
Customer premise VPN

- all VPN functions implemented by customer

- customer sites interconnected via tunnels
- tunnels encrypted typically
- SP treats VPN packets like all other packets
Drawbacks

- **leased-line VPN**: configuration costs, maintainance by SP: long time, much manpower
- **CPE-based VPN**: expertise by customer to acquire, configure, manage VPN

## Network-based VPN

- customer’s routers connect to SP routers
- SP routers maintain separate (independent) IP contexts for each VPN
  - sites can use private addressing
  - traffic from one vpn cannot be injected into another
Network-based Layer 3 VPNs

Tunnel encapsulation/de-capsulation performed in provider edge equipment

multiple virtual routers in single provider edge device
Tunneling (recall 4G cellular core)
VPNs: why?

- privacy
- security
- works well with mobile users (looks like you are always at home)
- cost: many forms of newer vpns are cheaper than leased line vpns
  - ability to share at lower layers even though logically separate means lower cost
  - exploit multiple paths, redundancy, fault-recovery in lower layers
  - need isolation mechanisms to ensure resources shared appropriately
- abstraction and manageability: all machines with addresses that are “in” are trusted no matter where they are
Network Virtualization via SDN

- logically divide single physical network into separate non-interacting virtual networks
  - each with its own routing, other protocols

Controller #1
- controls all IP addresses in range 0..*.*.* - 2.*.*.*

Controller #2
- controls all IP addresses in range 3.*.*.* - 9.*.*.*
Network function virtualization (NFV)

Classical Network Appliance Approach

- Message Router
- CDN
- Session Border Controller
- WAN Acceleration
- Tester/QoE monitor
- Carrier Grade NAT
- BRAS
- PE Router
- SGSN/GGSN
- DPI

- Fragmented non-commodity hardware.
- Physical install per appliance per site.
- Hardware development large barrier to entry for new vendors, constraining innovation & competition.

Independent Software Vendors

Orchestrated, automatic & remote install.

- Standard High Volume Servers
- Standard High Volume Storage
- Standard High Volume Ethernet Switches

Network Virtualisation Approach
Network function virtualization (NFV)

- vision: replace various network devices (routers,witches, CDN nodes, middleboxes) with programmable base set of hardware:
  - high-speed, commoditized switch (programmable)
  - storage
  - multiple processors, VM processing capabilities (programmable)