### Software Defined Networking

Jennifer Rexford  
COS 461: Computer Networks  
Lectures: MW 10:10:50am in Architecture N101  
http://www.cs.princeton.edu/courses/archive/spr12/cos461/

#### The Internet: A Remarkable Story

- **Tremendous success**  
  - From research experiment to global infrastructure
- **Brilliance of under-specifying**  
  - Network: best-effort packet delivery  
  - Hosts: arbitrary applications
- **Enables innovation in applications**  
  - Web, P2P, VoIP, social networks, virtual worlds
- But, change is easy only at the edge... 😊

#### Inside the ‘Net: A Different Story...

- **Closed equipment**  
  - Software bundled with hardware  
  - Vendor-specific interfaces
- **Over specified**  
  - Slow protocol standardization
- **Few people can innovate**  
  - Equipment vendors write the code  
  - Long delays to introduce new features

Impacts performance, security, reliability, cost...

#### Networks are Hard to Manage

- **Operating a network is expensive**  
  - More than half the cost of a network  
  - Yet, operator error causes most outages
- **Buggy software in the equipment**  
  - Routers with 20+ million lines of code  
  - Cascading failures, vulnerabilities, etc.
- **The network is “in the way”**  
  - Especially a problem in data centers  
  - ... and home networks

#### Creating Foundation for Networking

- **A domain, not (yet?) a discipline**  
  - Alphabet soup of protocols  
  - Header formats, bit twiddling  
  - Preoccupation with artifacts
- **From practice, to principles**  
  - Intellectual foundation for networking  
  - Identify the key abstractions  
  - ... and support them efficiently
- **To build networks worthy of society’s trust**

#### Rethinking the “Division of Labor”
Traditional Computer Networks

Data plane:
Packet streaming
Forward, filter, buffer, mark, rate-limit, and measure packets

Track topology changes, compute routes, install forwarding rules

Traditional Computer Networks

Management plane:
Human time scale
Collect measurements and configure the equipment

Death to the Control Plane!

• Simpler management
  – No need to “invert” control-plane operations
• Faster pace of innovation
  – Less dependence on vendors and standards
• Easier interoperability
  – Compatibility only in “wire” protocols
• Simpler, cheaper equipment
  – Minimal software

Software Defined Networking (SDN)

Logically-centralized control
API to the data plane (e.g., OpenFlow)

OpenFlow Networks
Data-Plane: Simple Packet Handling

- Simple packet-handling rules
  - Pattern: match packet header bits
  - Actions: drop, forward, modify, send to controller
  - Priority: disambiguate overlapping patterns
  - Counters: #bytes and #packets

1. src=1.2.*.*, dest=3.4.5.* \(\rightarrow\) drop
2. src = *,.*,.*, dest=3.4.*.* \(\rightarrow\) forward(2)
3. src=10.1.2.3, dest=.*,.*,.* \(\rightarrow\) send to controller

Unifies Different Kinds of Boxes

- Router
  - Match: longest destination IP prefix
  - Action: forward out a link

- Switch
  - Match: destination MAC address
  - Action: forward or flood

- Firewall
  - Match: IP addresses and TCP/UDP port numbers
  - Action: permit or deny

- NAT
  - Match: IP address and port
  - Action: rewrite address and port

Controller: Programmability

Eventos from switches
- Topology changes,
- Traffic statistics,
- Arriving packets

Commands to switches
- (Un)install rules,
- Query statistics,
- Send packets

Example OpenFlow Applications

- Dynamic access control
- Seamless mobility/migration
- Server load balancing
- Network virtualization
- Using multiple wireless access points
- Energy-efficient networking
- Adaptive traffic monitoring
- Denial-of-Service attack detection

See http://www.openflow.org/videos/

E.g.: Dynamic Access Control

- Inspect first packet of a connection
- Consult the access control policy
- Install rules to block or route traffic

E.g.: Seamless Mobility/Migration

- See host send traffic at new location
- Modify rules to reroute the traffic
OpenFlow in the Wild

- Open Networking Foundation
  - Google, Facebook, Microsoft, Yahoo, Verizon, Deutsche Telekom, and many other companies
- Commercial OpenFlow switches
  - HP, NEC, Quanta, Dell, IBM, Juniper, ...
- Network operating systems
  - NOX, Beacon, Floodlight, Nettle, ONIX, POX, Frenetic
- Network deployments
  - Eight campuses, and two research backbone networks
  - Commercial deployments (e.g., Google backbone)

A Helpful Analogy

From Nick McKeown’s talk “Making SDN Work” at the Open Networking Summit, April 2012
Challenges

Heterogeneous Switches
- Number of packet-handling rules
- Range of matches and actions
- Multi-stage pipeline of packet processing
- Offload some control-plane functionality (?)

Controller Delay and Overhead
- Controller is much slower than the switch
- Processing packets leads to delay and overhead
- Need to keep most packets in the “fast path”

Distributed Controller

Testing and Debugging
- OpenFlow makes programming possible
  - Network-wide view at controller
  - Direct control over data plane
- Plenty of room for bugs
  - Still a complex, distributed system
- Need for testing techniques
  - Controller applications
  - Controller and switches
  - Rules installed in the switches

Programming Abstractions
- Controller APIs are low-level
  - Thin veneer on the underlying hardware
- Need better languages
  - Composition of modules
  - Managing concurrency
  - Querying network state
  - Network-wide abstractions
- Ongoing at Princeton
  - http://www.frenetic-lang.org/
Conclusion

- Rethinking networking
  - Open interfaces to the data plane
  - Separation of control and data
  - Leveraging techniques from distributed systems
- Significant momentum
  - In both research and industry
- Next time
  - Closing lecture
  - No precept this week