S5: Software Defined Networking

(SDN)

Instructor:
Behnam Momeni

Fall 2018
Software-Defined Networking

• Motivation
• Enterprise network management
• Scalable SDN

• Readings:
  • A Clean Slate 4D Approach to Network Control and Management
  • Onix: A Distributed Control Platform for Large-scale Production Networks

• Optional reading
  • Ethane: Taking Control of the Enterprise
Software-Defined Networking

• Motivation
• Enterprise network management
• Scalable SDN

• Readings:
  • A Clean Slate 4D Approach to Network Control and Management
  • Onix: A Distributed Control Platform for Large-scale Production Networks

• Optional reading
  • Ethane: Taking Control of the Enterprise
4D: Motivation

• Network management is difficult!

• Observation: current Internet architecture bundles control logic and packet handling (e.g., OSPF)

• Challenge: how to systematically enforce various, increasingly complex high-level goals?
Design choices

• Incremental deployment
  • Advantage: easier to implement
  • Disadvantage: point solution?

• 4D advocates a clean-slate approach
  • Build control plane/network management from the ground up
  • Constraint: no change of packet formats

• Insight: Decouple the control and data planes
Example 1:
Front-Office Data Center ACL
Example 2: Spurious Routing
Management today

• Data plane
  • Packet forwarding mechanisms

• Control plane
  • Routing protocols
  • Distributed

• Management plane
  • Has to reverse engineer what the control plane does
  • Work around rather than work with!
Driving principles

• Network-level objectives
  • High-level, not after-the-fact

• Network-wide views
  • Measurement/monitoring/diagnosis

• Direct control
  • No more “reverse engineering” or “inversion”
  • Direct configuration
Three Principles for Network Control & Management

Network-level Objectives:

• Express goals explicitly
  • Security policies, QoS, egress point selection
• Do not bury goals in box-specific configuration

Reachability matrix
Traffic engineering rules
Management Logic
Three Principles for Network Control & Management

Network-wide Views:

- Design network to provide timely, accurate info
  - Topology, traffic, resource limitations
- Give logic the inputs it needs

Read state info

Reachability matrix
Traffic engineering rules

Management Logic
Three Principles for Network Control & Management

Direct Control:

- Allow logic to directly set forwarding state
  - FIB entries, packet filters, queuing parameters
- Logic computes desired network state, let it implement it

Reachability matrix
Traffic engineering rules

Write state

Management Logic

Read state info
4D Architecture

- Decision plane
  - routing, access control, load balancing, …

- Dissemination plane
  - control information through an independent channel from data

- Discovery plane
  - discover net. elements and create a logical net. map

- Data plane
  - handle individual packets given state by decision plane (e.g., forwarding tables, load balancing schemes, …)
Overview of the 4D Architecture

**Decision Plane:**
- *All* management logic implemented on centralized servers making *all* decisions
- *Decision Elements* use *views* to compute data plane state that meets *objectives*, then *directly writes* this state to routers

**Network-level objectives**

**Direct control**

**Network-wide views**
Overview of the 4D Architecture

Dissemination Plane:

- Provides a robust communication channel to each router – and robustness is the *only* goal!
- May run over same links as user data, but logically separate and independently controlled
Overview of the 4D Architecture

Discovery Plane:
- Each router discovers its own resources and its local environment
- E.g., the identity of its immediate neighbors
Overview of the 4D Architecture

Data Plane:
- Spatially distributed routers/switches
- Can deploy with today’s technology
- Looking at ways to unify forwarding paradigms across technologies

Network-wide views

Network-level objectives

Direct control
Good Abstractions Reduce Complexity

All decision making logic lifted out of control plane

- Eliminates duplicate logic in management plane
- Dissemination plane provides robust communication to/from data plane switches
Putting the pieces together

Controller

4D (vision)

Config

Config
Putting the pieces together

Controller

OpenFlow

Config

Config
Putting the pieces together

Controller

Ethane (concrete example)

Config

Config
Putting the pieces together

Controller

E.g., ONIX

Config

Config
Software-Defined Networking

- Motivation
- Enterprise network management
- Scalable SDN

Readings:
- A Clean Slate 4D Approach to Network Control and Management
- Onix: A Distributed Control Platform for Large-scale Production Networks

Optional reading
- Ethane: Taking Control of the Enterprise
ONIX: How to build a controller platform?
What are the key challenges?

- Usability
- Performance
- Flexibility
- Scalability
- Reliability/availability
- ...
ONIX

Server 1
- Network Control Logic
- NIB
  - Switch Import / Export
  - Distribution I / E

Management Connectivity Network Infrastructure
Managed Physical Network Infrastructure

Server N
- Network Control Logic
- NIB
  - Distribution I / E
  - Switch Import / Export
ONIX Design Decisions

- “Data-centric” API

- Treat all networking actions as data actions
  - Read
  - Alter
  - Register for changes in network state
Core component == NIB

- Network information base
  - Analogous to forwarding information base

- Graph of all network entities
  - Switches, ports, interfaces, links etc

- Applications read/register/manipulate NIB
Core component == NIB

- NIB is a collection network entities

- Each entity contains a set of key-value pairs

Default network entity classes
## ONIX NIB APIs

<table>
<thead>
<tr>
<th>Category</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>Find entities.</td>
</tr>
<tr>
<td>Create, destroy</td>
<td>Create and remove entities.</td>
</tr>
<tr>
<td>Access attributes</td>
<td>Inspect and modify entities.</td>
</tr>
<tr>
<td>Notifications</td>
<td>Receive updates about changes.</td>
</tr>
<tr>
<td>Synchronize</td>
<td>Wait for updates being exported to network elements and controllers.</td>
</tr>
<tr>
<td>Configuration</td>
<td>Configure how state is imported to and exported from the NIB.</td>
</tr>
<tr>
<td>Pull</td>
<td>Ask for entities to be imported on-demand.</td>
</tr>
</tbody>
</table>

### Functions provided by the ONIX NIB API
Three scalability strategies

- **Partition**
  - Can we split the state into independent sub-sets?
  - E.g., different subnet forwarding rules on a switch

- **Aggregate**
  - zoom-in/zoom-out at different aggregation levels

- **Tradeoff with weaker consistency/durability**
  - E.g., replicated transactional DB for network topology
  - E.g., one-hop DHT for link utilization info
Killer apps for ONIX

• Why did VMware buy Nicira?

• Distributed Virtual Switch

• Multi-tenant virtualization
Summary

• 4D: An general vision for design

• Ethane: End-to-end enterprise network management

• ONIX: A distributed control platform
Next Lecture

• Router Design

• Readings:
  • A Fast Switched Backplane for a Gigabit Switched Router
  • Scaling Internet Routers Using Optics (read intro)