Facilitating Network Management with Software Defined Networking

Nick Feamster
Georgia Tech
feamster@cc.gatech.edu

Steve Woodrow, Srikanth Sundaresan,
Hyojoon Kim, Russ Clark
Georgia Tech
Andreas Voellmy
Yale University

http://projectbismark.net/
Software Defined Network Management

• Software defined networking (SDN) makes it easier for network operators to evolve network capabilities

• Can SDN also help network operators manage their networks, once they are deployed?
  – Home networks
  – Campus/Enterprise networks
Why is network management so hard?
Changes are Frequent, Unwieldy

- Changes to the network configuration occur daily
  - Errors are frequent

- Operators must determine
  - What will happen in response to a configuration change
  - Whether the configuration is correct
Minimal Visibility Into Performance

• **Access ISPs**
  – What performance are customers seeing?
  – Can they gain better visibility into downtimes?
  – Can visibility into problems help reduce service calls?

• **Content Providers**
  – How do content routing or traffic engineering decisions affect end user performance

• Also, consumers and regulators
Configuration is Complex, Low-Level

- A campus network may have
  - More than one million lines of configuration
  - Thousands of devices
  - Hundreds of thousands of changes every year

- Home networks can be complex, too
## Addressing the Challenges

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Approach</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal Visibility and Control</td>
<td>Programmable Measurement Platform</td>
<td>BISmark</td>
</tr>
<tr>
<td>Frequent Changes</td>
<td>Event-Based Network Control</td>
<td>Lithium</td>
</tr>
<tr>
<td>Low-Level Configuration</td>
<td>High-Level Policy Language</td>
<td>Procera</td>
</tr>
</tbody>
</table>
SDN Forwarding Abstraction

- Control Program A
- Control Program B

Network OS

```
“If header = p, send to port 4”
“If header = q, overwrite header with r, add header s, and send to ports 5,6”
“If header = ?, send to me”
```
OpenFlow 1.0 Flow Table Entry

<table>
<thead>
<tr>
<th>Rule</th>
<th>Action</th>
<th>Stats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch Port</td>
<td>MAC src</td>
<td>MAC dst</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eth type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VLAN ID</td>
</tr>
<tr>
<td></td>
<td>IP Src</td>
<td>IP Dst</td>
</tr>
<tr>
<td></td>
<td>IP Prot</td>
<td>TCP sport</td>
</tr>
<tr>
<td></td>
<td>TCP sport</td>
<td>TCP dport</td>
</tr>
</tbody>
</table>

- Rule: Forward packet to port(s)
- Action: Encapsulate and forward to controller
- Stats: Drop packet
- Stats: Send to normal processing pipeline

+ mask

Packet + byte counters
Big Problem: Configuration Changes Frequently

- Changes to the network configuration occur daily
  - Errors are also frequent

- Operators must determine
  - What will happen in response to a configuration change
  - Whether the configuration is correct
But, Network Configuration is Really Just Event Processing!

• Rate limit all Bittorrent traffic between the hours of 9 a.m. and 5 p.m.
• Do not use more than 100 GB of my monthly allocation for Netflix traffic
• If a host becomes infected, re-direct it to a captive portal with software patches
• ...
BISmark: Bringing SDN Home

- Better monitoring and management of home and access networks
- **Deployment:** 225 Routers in ~30 countries

[Map showing routers in various countries](http://projectbismark.net/)
BISmark: Better Home Networks

• **Monitoring and Measurement**
  – ISP performance
  – Wireless characteristics and interference
  – Traffic use inside the home
  – Security
  – Human activity patterns

• **Control (with Software Defined Networking)**
  – Usage cap management (ongoing w/HCI researchers)
  – Traffic prioritization (e.g., ensure file sharing does not clobber critical traffic)
  – Parental controls
Better Visibility & Control

- **Better visibility**: Continuous performance monitoring
  - Network and application-level monitoring

- **Better control**: SDN
  - Control applications with simple programs and interfaces

**What is the network performance?**

**How do users use apps and devices?**

**Can we manage resource allocation?**
Usage Control in Home Networks

• Network management in homes is challenging
• One aspect of management: usage control
  – Usage cap management
  – Parental control
  – Bandwidth management

• Idea: Outsource network management/control
  – Home router runs OpenFlow switch
  – Usage reported to off-site controller
  – Controller adjusts behavior of traffic flows
Control: SDN + Intuitive Interfaces

Real-Time Monitoring and Control

Joint work with Boris de Souza, Bethany Sumner, Marshini Chetty.
Lithium:
Event-Based Network Control

Main Idea: Express network policies as event-based programs.

Extending the Control Model

- OpenFlow only operates on flow properties
- Lithium **extends the control model** so that actions can be taken on **time**, **history**, and **user**
Two Real-World Deployments

• **Usage control in home networks**
  – Implementation of user controls (e.g., usage cap management, parental controls) in home networks
  – **Today**: Not possible
  – **With SDN**: Intuitive, simple

• **Access control in enterprise networks**
  – Re-implementation of access control on the Georgia Tech campus network
  – **Today**: Complicated, low-level
  – **With SDN**: Simpler, more flexible
Deployment Status

- Over 300 routers deployed in home networks “in the wild”
- Collaboration with Measurement Lab on monitoring network performance from various regions and ISPs.
- Ongoing trials with several ISPs as part of private deployments

- **Firmware**
  - OpenWrt, with luci web interface
  - IPv6-capable
- **Netgear 3800 router**
  - Atheros chipset
  - MIPS processor, 16 MB flash, 64 MB RAM
  - Gigabit ethernet
  - 2.4 GHz and 5 GHz radio
Ongoing Extensions

• **More measurements:** Denser deployments (e.g., apartments)

• **Broader scope:** More measurements (e.g., integration with Tor’s OONI project)

• **Sensor fusion:** Tighter integration with other in-home, *in situ* sensing capabilities (e.g., phones)

• **Open programming interface:** Enable other researchers to perform measurements
The Need for a Policy Language

- Network policies
  - Are dynamic
  - Depend on temporal conditions defined in terms of external events

- Need a way to configure these policies without resorting to general-purpose programming of a network controller

- Intuitive user interfaces can ultimately be built on top of this language
The Need for Reactive Control

• Simple policies are doable in FML: “Ban the device if usage exceeds 10 GB in the last 5 days”

  ```
  deny(Us, Hs, As, Ut, Ht, At, Prot, Req) <- over(Hs).
  over(Hs) <- usage(Hs, lastDays(5), amt), amt > 10.
  ```

• But, adding **temporal predicates** is difficult!
  – “Remove the ban if usage drops below 10 GB.”
  – “Remove the ban when an administrator resets.”

• Each condition requires a new predicate.

  ```
  over(Hs) <- usageOnceExceeded(Hs, lastDays(5), 10).
  ```
Language Design Goals

- **Declarative Reactivity**: Describing when events happen, what changes they trigger, and how permissions change over time.
- **Expressive and Compositional Operators**: Building reactive permissions out of smaller reactive components.
- **Well-defined Semantics**: Simple semantics, simplifying policy specification.
- **Error Checking & Conflict Resolution**: Leveraging well-defined, mathematical semantics.
Controller: signal functions and a flow constraint function

- Receives input signals from environment
- Periodically updates a flow constraint function that controls the forwarding elements

Define a signal function for a device going over (or under) the usage cap:

```
overUnderEvent = proc env -> do
    capMap ← capTracker → env
    usagedb ← usageTracker → env
    usageChanges ← usageChangesTracker → env
    let now = calendarTime env
    let over src =
        monthlyUsage usagedb now > capMap ! src
        condSplit over ← usageChanges
```

Define the set of devices over the cap:

```
overSetStream = proc env -> do
    (over, under) ← overUnderEvent → env
    toSetStream ← (over, under)
```
Next Steps: Faster, Programmable Data Plane

- Augment OpenFlow switches with custom packet processors
- **Device abstraction layer** to allow programmability of this substrate
  - Single device
  - Network wide
- **Applications**
  - Big data applications
  - On-the-fly encryption, transcoding, classification
  - Selective deep packet inspection
Summary

• Software Defined Networking can simplify network monitoring and management, but we still need new control models.

• **BISmark**: Better visibility and control of home networks
• **Lithium**: Event-based network control
• **Procera**: Policy language for SDNs
• Next
  – A fast, programmable data plane for SDN