Wireless Software Defined Networks
Ayaka Koshibe, Akash Baid and Ivan Seskar

WINLAB
Rutgers University

2014 Fall IAB
December 12th, 2014
Need for Inter-network Cooperation

- Interaction between managed wireless networks over the back-end wired link for making more efficient use of the spectrum

**Aggregate radio map**
- Range of operation: \((x_B, y_B, r_B)\)
- Technology type: Wi-Fi
- Device list: A1 params, A2 params, ...

**Policy/capabilities**
- Controller type: C2
- Information sharing enabled
- Merge RRM enabled

**Algorithm & policy negotiation**

**Per device parameters**
- Radio device A1
  - Type: Transmit/Receive (client)
  - Location: \((x_{A1}, y_{A1})\)
  - Power, BW, frequency, duty cycle

**Per device parameters**
- Radio device B1
  - Type: Transmit/Receive (client)
  - Location: \((x_{B1}, y_{B1})\)
  - Power, BW, frequency, duty cycle

**Aggregate radio map**
- Range of operation: \((x_B, y_B, r_B)\)
- Technology type: Wi-Fi
- Device list: B1 params, B2 params, ...

**Policy/capabilities**
- Controller type: C2
- Information sharing enabled
- Merge RRM enabled

**Network A**

**Network B**

**Spectrum info exchange**

Interaction between managed wireless networks over the back-end wired link for making more efficient use of the spectrum
SDN Approach to Wireless Control Plane

Introducing flexibility in the wireless control plane by leveraging software defined networking techniques

Inter-network cooperation translates to inter-controller interactions and setting of flow-rules

- Through extension of OpenFlow Match/Action Fields
- Through the ControlSwitch framework
A: Distributed Control Plane

Extension of traditional Enterprise Controller:

- Multiple copies of wireless controllers (WC) with mechanisms to cooperate, scattered throughout SDN based control plane
- Reduced distance between device and a controller – reduced flow setup times (reduced control latency)
B: Heterogeneous Distributed Control Plane

- Builds the control plane as a network of different controllers
  - Each controller is a part of a control stack
  - Controllers communicate by message passing
  - Multiple controllers process each event
    - SDN control plane can pick and choose services for each event, avoiding conflict

- Wireless Control Stack (WCS) realized as a complex interaction between controllers rather than (single) monolithic application
Groups of functional WCS controllers arranged in tiers
- Higher tier events = more global and less frequent
- Supports natural hierarchical inter-controller relations

Controllers connected with SDN inter-controller links
- Joins pieces of wireless control stacks together through SDN
- Additional benefit: dynamic provisioning and routing for events to be moved between controllers and/or tiers (if they can’t be handled at a tier)
Example: Pair of enterprises with heterogeneous decomposed controllers
Distributed SD(W)Ns: What’s Missing?

Scalable and resilient, but not readily available

- proprietary, experimental, or early-stage

Focuses on uniformity, expects controllers to be homogeneous

No standard mechanism to handle interaction of network stacks (SDX?)

- Relies on specialized mechanisms
  - network hypervisors - virtualize network
  - network compilers - resolve flow conflicts
  - semi-intelligent switches - delegate some work to switches
  - SDN-to-IP network peering applications
A complex control plane structure requires:

- Orchestration of event handling across multiple controllers
  → An inter-controller process chain

- Path selection across control plane
  → A control plane routing mechanism

- Allow controllers to learn of available services/topology
  → An information propagation scheme
**Enabling Coordination**

**Servers** configure **client process chains** via subscription messages containing:
- SID and events handled by server
- Packet process chain behavior directives for the client
  - Services may want clients to behave in specific ways
    → A client must wait for an ACL’s result for an event before further handling
  - Four types: DENY, ALLOW, DIVERT, SPLIT
- Service reachability information
  - Route to service by hop count
  - Hops increased as messages propagate towards data plane
  - Allow picking of smallest hop routes to each service
How is a controller implemented?

A controller is composed of three layers:

- Control channel handler: Sending/receiving control messages from the data plane
- Event dispatcher: Conversion between control messages and events, distribution of events to services
- Applications: Implementation of services, event handlers and interfaces for usability
Implementation: A Hierarchical Controller With Floodlight

- Base controller extended with modules, channel handlers, and dispatchers
Processing times, multiple tiers

<table>
<thead>
<tr>
<th>Hops</th>
<th>Mean (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.09</td>
</tr>
<tr>
<td>2</td>
<td>0.72</td>
</tr>
<tr>
<td>3</td>
<td>1.09</td>
</tr>
<tr>
<td>4</td>
<td>1.47</td>
</tr>
<tr>
<td>5</td>
<td>1.98</td>
</tr>
</tbody>
</table>
Processing times, multiple DIVERT

<table>
<thead>
<tr>
<th>hops</th>
<th>mean (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.56</td>
</tr>
<tr>
<td>2</td>
<td>1.13</td>
</tr>
<tr>
<td>3</td>
<td>1.89</td>
</tr>
<tr>
<td>4</td>
<td>2.52</td>
</tr>
<tr>
<td>5</td>
<td>3.27</td>
</tr>
</tbody>
</table>
Processing times, multiple SPLIT

<table>
<thead>
<tr>
<th>hops</th>
<th>mean (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.56</td>
</tr>
<tr>
<td>2</td>
<td>0.71</td>
</tr>
<tr>
<td>3</td>
<td>0.89</td>
</tr>
<tr>
<td>4</td>
<td>1.01</td>
</tr>
<tr>
<td>5</td>
<td>1.25</td>
</tr>
</tbody>
</table>
Scalability Testing - Hierarchical

Two-tiered control plane

- One, two, and four tier 1 controllers
- Fixed at four virtual datapaths/controller
- Tier 2 controller subscribed to new-device event triggers

We expect:

- Tier 2 subscriptions cut event escalations to manageable levels
- Event handling capacity of Tier 1 scales with the number of controllers
Event reception/handling

- Left chart: Graph showing the number of packets received per unit time with respect to the number of unique addresses (per switch, $10^3$).
- Right chart: Graph showing processing time in nanoseconds per packet with respect to the number of unique MAC addresses ($10^3$).
OPEN BTS

GENI Project – Open WiMAX BTS
- Exposed all controllable parameters through API
- Removed all default IP routing, simplified ASN controller*
- All switching purely based on MAC addresses
- Implemented the datapath virtualization and VNTS shaping mechanism in click/OpenVSwitch for slice isolation

Ongoing work – Open LTE BTS
- Exposed all controllable parameters through the same REST based API
- Implemented the datapath with openvswitch
- Current development: ePC replacement with open source aggregate manager (i.e. simplification/elimination of LTE control protocols)
OPEN BTS: WiMAX & LTE

WiMAX

Traffic Scheduler/Shaper

RF Aggregate Manager

OpenVSwitch

control

data

LTE

RF/ePC Aggregate Manager

OpenVSwitch

X2S1-U/ST-MME...

LTE-Uu

OpenVSwitch
4G in GENI: Larger Picture

- Adaptation/Handoff Controller
- Generic Resource Controller
- SDN Datapath Complex
- OPEN BS2
- OPEN BS3
More Info @

www.orbit-lab.org
www.geni.net
www.winlab.rutgers.edu
wimax.orbit-lab.org