OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)

Stefano Salsano\textsuperscript{(1)}, Pier Luigi Ventre\textsuperscript{(2)}, Luca Prete\textsuperscript{(2)}, Giuseppe Siracusano\textsuperscript{(1)}, Matteo Gerola\textsuperscript{(3)}, Elio Salvadori\textsuperscript{(3)}

\textsuperscript{(1)} Univ. of Rome Tor Vergata, \textsuperscript{(2)} Consortium GARR, \textsuperscript{(3)} CREATE-NET
DREAMER Project
http://netgroup.uniroma2.it/DREAMER

Distributed RESilient sdn Architecture MEeting carrier grade Requirements

- Partners:
  - CNIT
  - Consortium GARR
  - CREATE-NET

- Main goal:
  - Design a “carrier grade” IP backbone based on OpenFlow/SDN and experiment with its prototype on GÉANT SDN testbeds

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
Objectives of this work

- Introduce the Software Defined Networking paradigm in IP backbones
  - replicating the services of IP/MPLS networks
  - ... and their non-functional properties (“carrier grade”)

- Do it in an open way !!
  - Open source components
  - Simple tools for setting up and performing experiments
Outline

1. Open Source Hybrid IP/SDN (OSHI) data plane
2. An example service: Ethernet VLL
3. OSHI emulation tools (and short video demo)
4. Performance evaluation
Open Source Hybrid IP/SDN (OSHI)
http://netgroup.uniroma2.it/OSHI

Hybrid IP/SDN resilient data plane
IP routing & forwarding
SDN/OpenFlow switch

Open Source Hybrid IP/SDN (OSHI) nodes
OSHI Node architecture

1. Open Source Hybrid IP/SDN

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
Hybrid IP/SDN data plane

1. Coexistence mechanisms for IP traffic and SDN traffic
2. Ingress classification functions / tunneling mechanisms
Hybrid IP/SDN data plane

1. Coexistence mechanisms for IP traffic and SDN traffic
2. Ingress classification functions / tunneling mechanisms

Current prototype:
VLAN tags for coexistence, classification & tunneling mechanisms
Ethernet Virtual Leased Line

- VLL is provided through a SDN Based Path (SBP)
  - we use VLAN tags switching (in current prototype)
Virtual Leased Line Pusher

2. An example service (VLL)

OSH! - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
3. Emulation tools

Design & deployment workflow

Graphical Topology and Service Designer

Topography representation file (JSON)

networkx (automatic topology generator)

Topology to testbed mapping

Topology Deployer

Topology Parser

Mininet Deployer

OFELIA Deployer

OFELIA Setup scripts

OFELIA config scripts

Management Console

Measurement Tools

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
Emulation on OCF testbeds

(OCF : OFELIA Control Framework)

3. Emulation tools

Overlay Experimental Topology

VMs and Tunneling

How to map an arbitrary topology on a set of VM servers and links, with minimal configuration effort?
3. Emulation tools

Our toolset:
- VXLAN (or OpenVPN) for making tunnels
- Bash and Python scripts to automate VMs setup
- DSH for distributed setup and maintenance
Short demo

1. Topology and Service Designer

2. Topology Deployer (on OFELIA)

3. Virtual Leased Lines operation
Performance evaluation
Measurements tools

- iperf tool for traffic sources/sinks
- A client-server measurement tool to gather CPU load info of VMs

3. Emulation tools

iperf

xentop on the XEN server

getVMInfo(VM3)

getVMInfo(VM3)

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
OSHI vs. Plain Router
(no tunnels in both cases)

No tunnels, comparison between routing with OSHI and a plain router

4. Performance evaluation

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
4. Performance evaluation

Tunneling comparisons

OpenVPN tunnels vs. VXLAN tunnels vs. No tunnels

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
References

- DREAMER project home page: http://netgroup.uniroma2.it/DREAMER

- Home page of OSHI http://netgroup.uniroma2.it/OSHI
  – Code from GitHub see Software download section
Conclusions

• We designed and implemented an Open Source Hybrid IP/SDN solution (OSHI) ... it works well!

• The VXLAN tunneling solution is very effective to map overlay topology over distributed testbeds

• The proposed deployment workflow is a key element to ease innovation in IP/SDN networks
Next steps (work in progress)

• Designing a full Ethernet “pseudo-wire” service... using MPLS for tunneling rather than VLAN

• Using ONOS controller clusters to replace Floodlight

• Designing a solution with multiple controller clusters, each one controlling a portion of a wide area SDN based IP/SDN backbone
Thank you! (questions)
Backup slides
Mininet deployer

- Deploys arbitrary experimental topologies in Mininet
- Nodes ready to start the experiments
- One click experiments
- Easy to extend with new features
4. OSHI emulation tools

TestbedDeployer Framework

1 //Deploy of Full Mesh Topology
2 // Initialization section of temporary variable
3 testbed = TestbedOFELIA(“ofelia.map”) // We create Testbed object passing as
4 parameter the mapping file
5 for i in range (0, size): // the size parameter is the dimension of the mesh topology
6 oshi = testbed.addOshi(name) // Add a new OSHI to the experiment. Return an
7 Oshi object
8 for lhs in oshis:
9 testbed.addPPLink(lhs_name, oshi_name) // Add an overlay link among lhs
10 and oshi. Return a composite object, that represents in our case an overlay link.
11 oshis.append(oshi)
12 ctrl = testbed.addController(name, OF_tcp_port) // Add a controller to the
13 experiment. Return a Controller object
14 testbed.addPPLink(oshi, ctrl) // Connect the Controller to the overlay network
15 testbed.configure() // Generate the configuration file for the overlay topology to deploy

- Able to start in a few minutes through config. script whatever experimental topology;
- Nodes ready to start the experiments;
- Easy to extend with new features;

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
1. Deploy of Full Mesh Topology

```python
// Initialization section of temporary variable
2 testbed = TestbedOFELIA(“ofelia.map”)  // We create Testbed object passing as parameter the mapping file
3 for i in range (0, size):  // the size parameter is the dimension of the mesh topology
4   oshi = testbed.addOshi(name)  // Add a new OSHI to the experiment. Return an Oshi object
5   for lhs in oshis:
6     testbed.addPPLink(lhs_name, oshi_name)  // Add an overlay link among lhs and oshi. Return a composite object, that represents in our case an overlay link.
7   oshis.append(oshi)
8 ctrl = testbed.addController(name, OF_tcp_port)  // Add a controller to the experiment. Return a Controller object
9 testbed.addPPLink(oshi, ctrl)  // Connect the Controller to the overlay network
10 testbed.configure()  // Generate the configuration file for the overlay topology to deploy
```

- Able to start in a few minutes through config. script whatever experimental topology;
- Nodes ready to start the experiments;
- Easy to extend with new features;

OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds)
VLL switching vs. IP routing

- Mininet Emulator;
- 7 EUHs, 4 Switches, 3 Access OSHIs and 5 Core OSHIs in a laptop;
- Evaluation of TCP throughput;
- Comparing IP vs. VLL solution;

![VLL Diagram]

### Performance Evaluation

<table>
<thead>
<tr>
<th>#</th>
<th>VLL (Mb/s)</th>
<th>IP (Mb/s)</th>
<th>% GAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG</td>
<td>1555</td>
<td>1150</td>
<td>26.04%</td>
</tr>
<tr>
<td>STD DEV</td>
<td>21.8</td>
<td>20</td>
<td>#</td>
</tr>
</tbody>
</table>

- TCP throughput is limited by sum of CPU load;
- Label switching is less expensive than IP forwarding