MAC Learning Switch (L2) on OpenDaylight

Sai Kishore
Abhiram Krishna

COEN233 – Computer Systems
FALL 2014

1 TABLE OF CONTENTS

1.1 TABLE OF FIGURES

2 ACKNOWLEDGEMENTS

We are thankful to Prof. Wang for giving us this opportunity to get hands on experience with Software Defined Networks. His projects and class discussions immensly helped us explore Computer Networks from a bottom up approach. Lastly, Thanks are due to the amazing people of SCU for the environemtn and learning culture.

3 INTRODUCTION

3.1 OBJECTIVE

The aim of this project is to develop and implement in software – an L2 switch optimized with learning. The implementation will be done on the OpenDaylight platform using a model-driven Service Abstraction Layer (MD-SAL), which can be applied to Software Defined Networks.

3.2 PROBLEM BACKGROUND

Layer 2 switching has some significant disadvantages which include lack of router hardware which increase their susceptibility to broadcast storms. Also, as they currently work, they forward all traffic especially ARP and DHCP broadcasts. Anything transmitted by one device is forwarded to all devices. In case of large networks, this causes significant traffic congestion and decreases network efficiency.

3.3 APPLICATION OF THE PROBLEM TO COMPUTER NETWORKS

Layer 2 switching is a fairly important topic in computer networks. We attempt to implement the Switch with learning capabilities for Software Defined Networks, which is related to Networks domain.

3.4 DISADVANTAGES OF OTHER APPROACHES

Many approaches to solving this problem include adding administration of IP allocations across multiple sites in the subnet. This is a significant overhead. Other approaches include solving this by increasing the amount of router hardware. There are very few software-based approaches, which is a concern especially as the networking industry is heading towards Software Defined Networks.

3.5 ADVANTAGES OF OUR APPROACH

Our approach to solving this problem involves a complete, re-usable and extensible software solution to layer 2 switching. Our proposed implementation would be leveraging the OpenDaylight platform – one of the latest and highly advocated open platforms in software-defined networking. In our implementation, we aim to induce a high level of learning into the L2 switch. This allows the switch to learn the network topology and be aware of changes. Thereby, when a packet is received it is directly 'teleported' to the destination instead of flooding to

every other host. This results in very high efficiency as compared to current switching methodology in layer 2.

3.6 PROBLEM STATEMENT

To implement a reusable, extensible and scalable L2 switch optimized with learning for software defined networks using the OpenDaylight controller.

3.7 PROJECT SCOPE

The project is divided into two phases:

1. Background Research and Selecting an SDN Controller.
2. Implementation of the L2 Switch with a learning algorithm that helps in efficient routing.

4 THEORETICAL BASIS AND RELATED LITERATURE

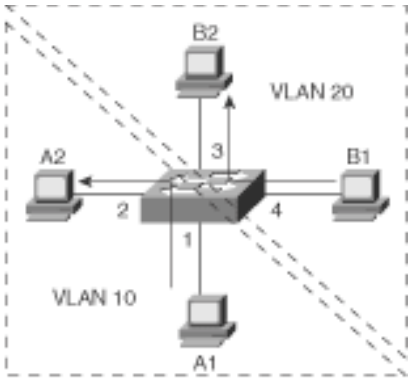
4.1 PROBLEM DEFINITION

Traditional Layer 2 switch lacks the intelligence and capability to learn network topology and forward packets from host to destination. The same has continued into implementations with regard to Software Defined Networks. Also, the software defined implementations of the L2 switch lack the modularity, scalability and ability for reuse.

4.2 THEORETICAL BACKGROUND

L2 Switching: Switching is a layer 2 operation in the OSI stack and its decisions are based in destination MAC addresses. The main objective of layer 2 switching was to split networks with too much end terminal devices in different “collision domains”. In order to improve the behavior of Ethernet, switches operate like hubs, that is they flood all the other hosts connected to it. This is a considerable inefficiency especially in large networks. The following basic concepts are involved in L2 switching:

- MAC Learning – Allows the Ethernet switch to learn the MAC addresses of the stations in the network to identify which port to send the traffic to. LAN switches normally keep a MAC learning table (or a bridge table) and a VLAN table. The MAC learning table associates MACs/VLANs with a given port, and the VLAN table associates the port with a VLAN.



MAC	Port
(MAC-A1, VLAN 10)	1
(MAC-A2, VLAN 10)	2
(MAC-B1, VLAN 20)	4
(MAC-B2, VLAN 20)	3

VLAN	Port
10	1,2
20	3,4

Figure 1: MAC Learning

- Flooding: When the switch gets a packet with an unknown destination MAC address, it sends this packet to all the connected ports which is called flooding.

Software Defined Implementation of L2 Switch: There are existing software-designed implementations, which replicate this functioning but do not solve the inefficiencies. Also, they are tightly coupled to specific controller API thereby not providing the extensibility and ability to reuse in large implementations of an SDN.

OpenDaylight and MD-SAL: OpenDaylight Controller currently contains Service Abstraction Layer (AD-SAL/MD-SAL), Several Services, UI, Samples etc. Depending on the plugins loaded, the controller also has a certain packet handling/forwarding behavior with the ARPHandlers, Host Tracker, Simple Forwarder etc. There is an existing AD-SAL friendly implementation of the L2 switch which also hinders the scalability of a larger and open SDN implementation. Also, there is no separation of Layer 2 specific functionality and handling - therefore no extensibility or ability to reuse.

OpenDaylight Controller:

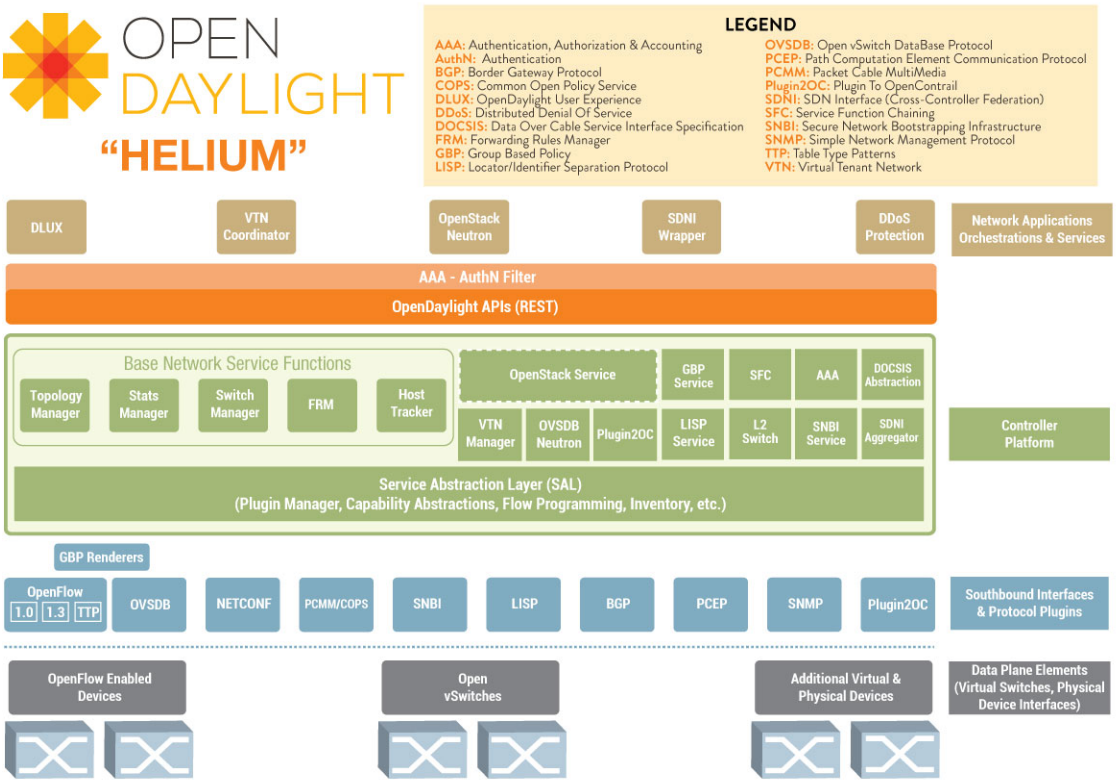


Figure 2: Opendaylight Architecture

OpenDaylight is a collaborative open-source SDN framework which has many modules which reuse common services and interfaces. These modules have been developed as sub-projects by multiple collaborators such as Cisco, Juniper, Brocade etc. It was inspired and was built based on Beacon which introduced the use of Open Service Gateway Interface (OSGi) which is key for modularity and run-time loading of plugins into the controller.

The architecture mainly comprises of three layers:

1. Southbound protocol plugins
2. Service Adaptation Layer (SAL)
3. Northbound Application/Service functions.

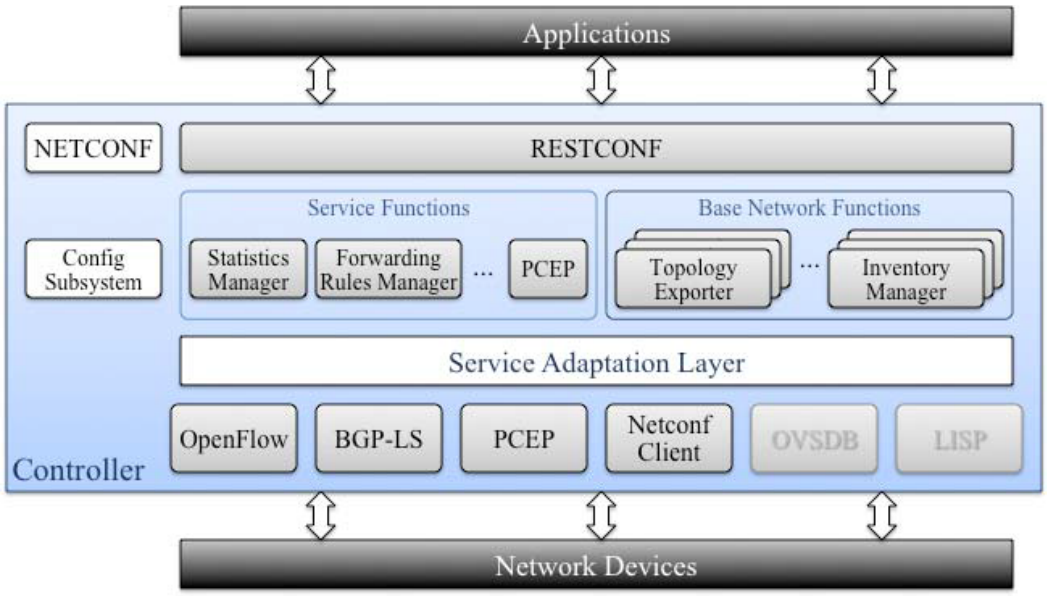


Figure 3: High-level network view of the OpenDaylight controller

The Southbound protocol plugins interface with the network devices – the SAL adapts these SB plugin functions to the Northbound application functions. The layered architecture allows the controller to support multiple Southbound protocols.

Coming to some of the technical paradigms used by the OpenDaylight platform, the following are important to understand.

- Interfaces: The platform provides Java interfaces which are used for event listening, specifications and forming patterns. This is the main way in which specific bundles implement call-back functions or events. They also notify if there are changes in the specific state.
- Maven: OpenDaylight uses Maven for build automation and dependency management.
- OSGi: The Open Service Gateway Interface (OSGi) forms the backend of the OpenDaylight platform. It allows the dynamic loading and binding of bundles and packaged Jar files.
- Karaf: It is an OSGi based runtime which provides a lightweight container for loading of different modules.

OpenDaylight also offers core bundles with many important services exposed through the Java interfaces. Some of them important to our project are:

Bundle	Exported interface	Description
arphandler	IHostFinder	Component responsible for learning about host location by handling ARP.
hosttracker	IfIptoHost	Track the location of the host relatively to the SDN network.

switchmanager	ISwitchManager	Component holding the inventory information for all the known nodes (i.e., switches) in the controller.
topologymanager	ITopologyManager	Component holding the whole network graph.
usermanager	IUserManager	Component taking care of user management.
statisticsmanager	IStatisticsManager	Component in charge of using the SAL ReadService to collect several statistics from the SDN network.
sal	IReadService	Interface for retrieving the network node's flow/port/queue hardware view
sal	ITopologyService	Topology methods provided by SAL toward the applications
sal	IFlowProgrammerService	Interface for installing/modifying/removing flows on a network node
sal	IDataPacketService	Data Packet Services SAL provides to the applications
web	IDaylightWeb	Component tracking the several pieces of the UI depending on bundles installed on the system.

Table 5: Important core bundles in OpenDaylight

Mininet :

Mininet is an OpenFlow based testing platform to simulate virtual networks and facilitate testing of SDN platforms. Formally, it is defined as a realistic virtual network, running real kernel, switch and application code on a single machine in seconds, with a single command. Mininet essentially emulates an OpenFlow network and end-hosts within a single machine. It includes built-in support to create several commonly used topologies, and it allows for construction of custom topologies using python scripts.

4.3 RELATED RESEARCH

We based our research related to selecting an appropriate SDN platform and exploring OpenDayLight platform.

4.3.1 Featured based comparison and selection of SDN controllers.

Authors of this paper talk about mathematical method to do a feature comparison on various available controllers. They solve this Multi Creteria Decision problem using Analytic Hierarchy Process model. This helped us compare various controllers. We chose OpenDaylight platform as our framework for implementing by following the process mentioned by the authors.

4.3.2 OpenDayLight: Towards a model driven SDN controller architecture.

Authors of this paper talk about applying the principles of model-driven software engineering to the OpenDaylight platform. They propose a model-driven service abstraction layer as opposed to the API driven SAL developed for the first release of the OpenDaylight platform. This is a novel approach and has been a major upgrade from the first version of the platform. The latest release i.e., Helium incorporates these changes and the platform now supports the many privileges offered by model-driven service abstraction layer (MD-SAL).

4.4 OUR SOLUTION

We plan to use OpenDayLight platform after evaluating some well-known SDN controllers. Our solution is to develop and implement an L2 switch with basic learning algorithm. So with added intelligence, our switch performs better than traditional L2 switches.

5 HYPOTHESIS

Using an L2 Switch developed on OpenDaylight platform, with learning capability, more efficient L2 forwarding will be achieved when compared to a traditional L2 Switch.

6 METHODOLOGY

OpenDayLight platform is chosen for this project implementation. It is free and recent SDN platform to develop solutions for networks based on software.

6.1 DATA COLLECTION

This project will implement a L2 switch with learning algorithm. We plan to analyze single packet across multiple network topologies of varying complexity and multiple packets on a single topology to measure our learning L2 switch performance.

6.2 PROPOSED SOLUTION

We will develop a software defined L2 switch and implement a simple learning algorithm. The idea is not emphasize on machine learning but to show that learning improves switch performance.

6.2.1 Algorithm Design

The pseudo-code for the MAC learning switch is as follows:

1. Create and maintain a table to manage the mac address to port mapping.
2. In case of multiple switches, maintain a table for each switch.
3. For every input packet
    i. Parse the packet
    ii. Determine the source and destination MAC addresses.
    iii. Keep in table according to mac address to port

iv. Check if destination mac address is mapped to a port.
v. If mapping is found, create new flow according to algorithm and send
vi. If mapping is not found flood it like hub.
vii. When flooding happens, the same procedure will enable learning of other source mac – port mappings.

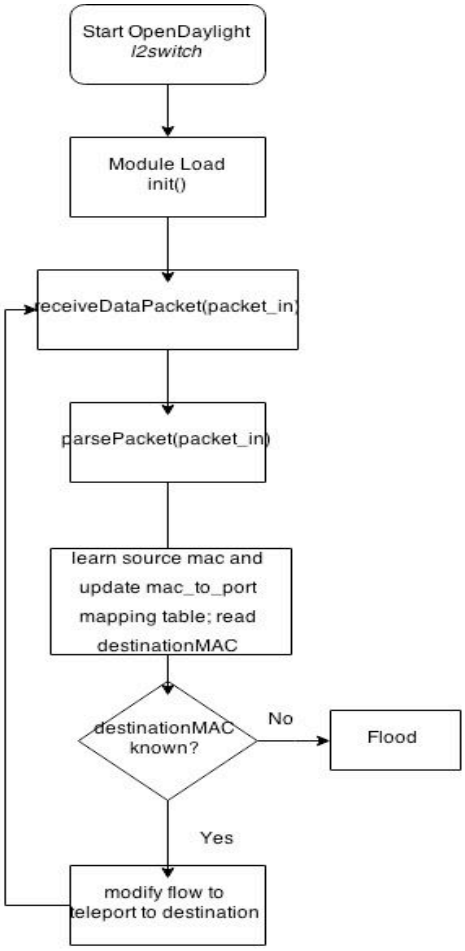


Figure 4: MAC Learning Switch Program Flow

6.2.2 Programming Environment

Java, OpenDayLight Controller.

6.2.3 Other Tools Used

Linux, Maven, Eclipse, Supporting OpenDayLight components.

6.3 Output Generation

Simulate network using mininet with an OpenDaylight controller.

Environment 1: A regular switch without learning.

Environment 2: MAC Learning Switch.

Both setups have traces to print required information as described above in section 6.1. This data is the output to verify the hypothesis.

6.4 Testing Methodology

Environment 1: A Software Defined Network topology with regular L2 switch.

Environment 2: A Software Defined Network topology with our L2 switch with learning capability.

Network is simulated with heavy traffic on above environments under identical conditions. Data is tabulated and compared against each run. This comparative analysis will test our hypothesis.

Using Mininet:

There are many default topologies provided in Mininet such as minimal, single, reversed, linear and tree.

Single Switch:

It is a simple topology with one switch and N hosts.

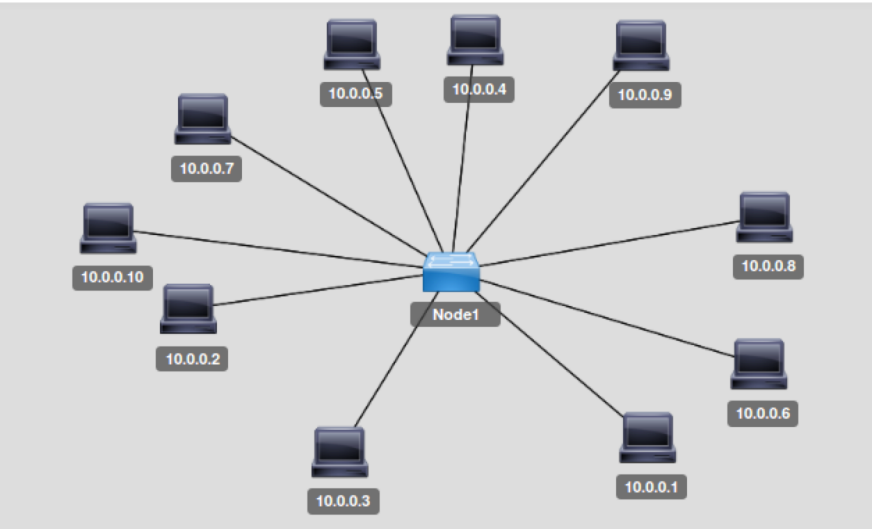


Figure 5: Mininet Single Switch topology

To generate this topology the following command can be used:

$ sudo mn --arp --topo single,10 --mac --switch ovsk --controller remote

Linear Topology:

This topology consists of N switches and N hosts connected serially.

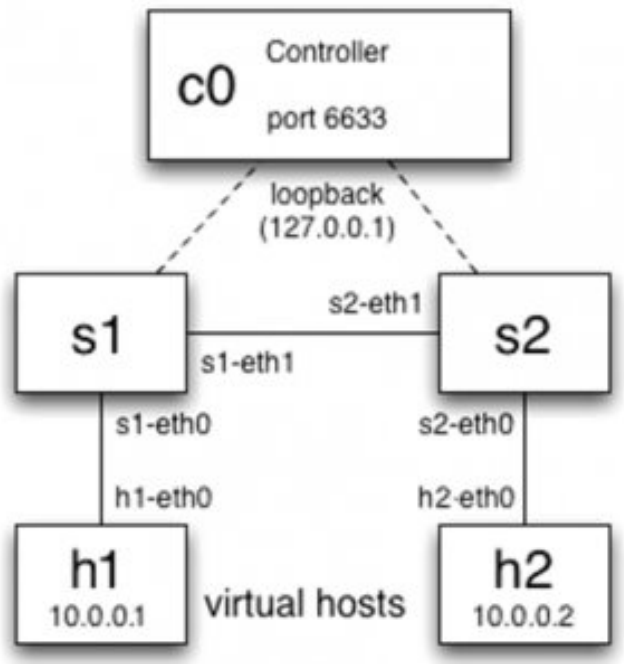


Figure 7: Mininet Linear Topology

The command to generate this topology is:

$ sudo mn --topo linear --switch ovsk --controller remote

6.5 CHANGES TO THE PROPOSED SOLUTION

Our initial idea of implementing the switch using both AD-SAL and MD-SAL has been partly successful. We were able to implement the switch in AD-SAL but due to time constraints we were not able to complete the implementation using MD-SAL. However, this is one of our scheduled follow-up activities with respect to this project.

As part of our ideas for extending and improving our implementation, we plan to include machine learning to improve performance especially in case of multiple switches with N hosts.

7 IMPLEMENTATION

We have developed a L2 Switch in

7.1.1 Code Flow

1. The packet arrives at Switch (S1) and will be forwarded to appropriate plugin corresponding to the switch.
2. The plugin will parse the packet and generate an event.
3. The SAL will transfer the packet to the event listener – IListenDataPacket
4. IListenDataPacket will receive the packet and dispatch it through its service – IdataPacketService.
5. SAL dispatches the packet to the module listening for the packet.
6. OpenFlow message will be sent to appropriate switch.

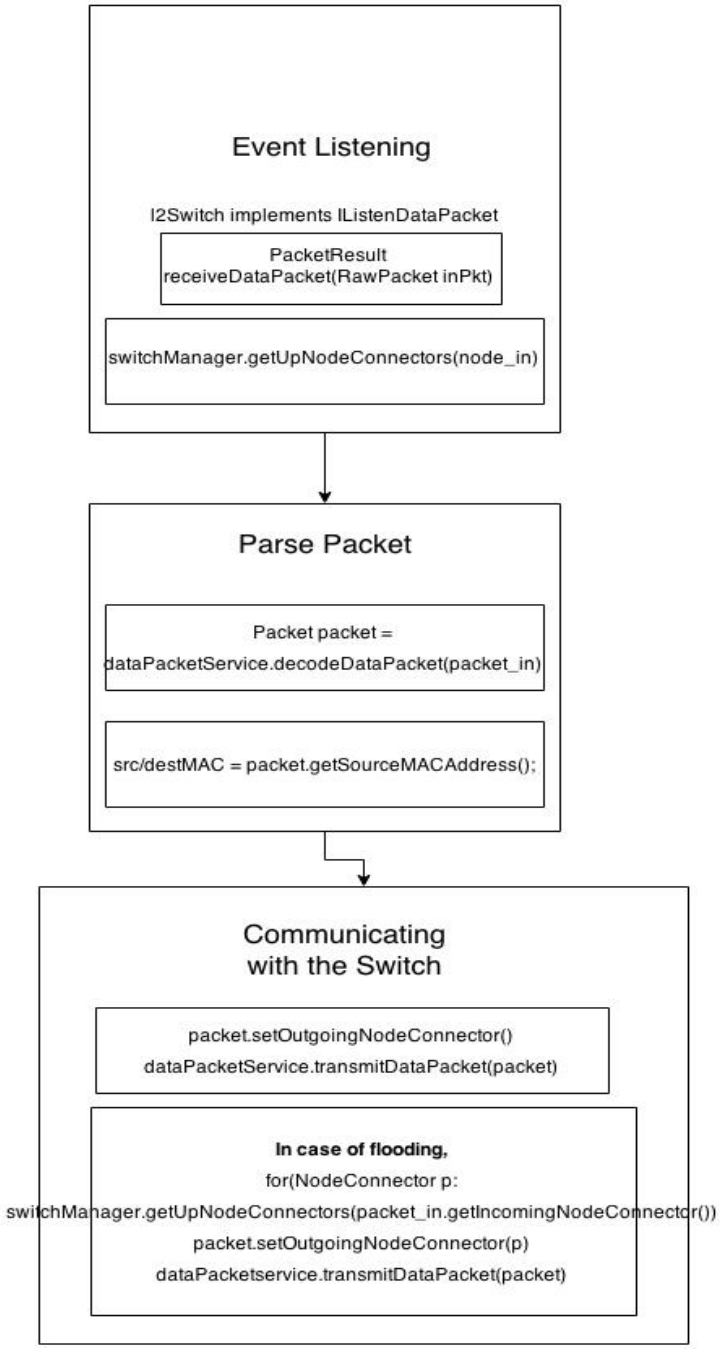


Figure 5: MAC learning switch code flow

8 DATA ANALYSIS & DISCUSSION

We used mininet to emulate the network and flow packets between hosts. Following is the data collected as a result of packet flow from two hosts in a topology.

8.1.1 Without Learning L2 Switch acts as a simple hub

			Time in ms
Ping from 10.0.0.2	icmp_seq=1	ttl=64	**151**
Ping from 10.0.0.2	icmp_seq=2	ttl=64	**125**
Ping from 10.0.0.2	icmp_seq=3	ttl=64	**160**
Ping from 10.0.0.2	icmp_seq=4	ttl=64	**163**
Ping from 10.0.0.2	icmp_seq=5	ttl=64	**168**
Ping from 10.0.0.2	icmp_seq=6	ttl=64	**172**
Ping from 10.0.0.2	icmp_seq=7	ttl=64	**117**
Ping from 10.0.0.2	icmp_seq=8	ttl=64	**174**
Ping from 10.0.0.2	icmp_seq=9	ttl=64	**173**
Ping from 10.0.0.2	icmp_seq=10	ttl=64	**174**
Ping from 10.0.0.2	icmp_seq=11	ttl=64	**176**
Ping from 10.0.0.2	icmp_seq=12	ttl=64	**115**
Ping from 10.0.0.2	icmp_seq=13	ttl=64	**178**
Ping from 10.0.0.2	icmp_seq=14	ttl=64	**179**
Ping from 10.0.0.2	icmp_seq=15	ttl=64	**104**
Ping from 10.0.0.2	icmp_seq=16	ttl=64	**107**
Ping from 10.0.0.2	icmp_seq=17	ttl=64	**106**
Ping from 10.0.0.2	icmp_seq=18	ttl=64	**114**
Ping from 10.0.0.2	icmp_seq=19	ttl=64	**117**
Ping from 10.0.0.2	icmp_seq=20	ttl=64	**119**
Ping from 10.0.0.2	icmp_seq=21	ttl=64	**123**
Ping from 10.0.0.2	icmp_seq=22	ttl=64	**100**
Ping from 10.0.0.2	icmp_seq=23	ttl=64	**131**
Ping from 10.0.0.2	icmp_seq=24	ttl=64	**134**
Ping from 10.0.0.2	icmp_seq=25	ttl=64	**138**
Ping from 10.0.0.2	icmp_seq=26	ttl=64	**140**
Ping from 10.0.0.2	icmp_seq=27	ttl=64	**94.9**
Ping from 10.0.0.2	icmp_seq=28	ttl=64	**146**
Ping from 10.0.0.2	icmp_seq=29	ttl=64	**151**
Ping from 10.0.0.2	icmp_seq=30	ttl=64	**156**
Ping from 10.0.0.2	icmp_seq=31	ttl=64	**158**

Ping from 10.0.0.2	icmp_seq=32	ttl=64	**87.3**
Ping from 10.0.0.2	icmp_seq=33	ttl=64	**168**
Ping from 10.0.0.2	icmp_seq=34	ttl=64	**171**
Ping from 10.0.0.2	icmp_seq=35	ttl=64	**80**
Ping from 10.0.0.2	icmp_seq=36	ttl=64	**180**
Ping from 10.0.0.2	icmp_seq=37	ttl=64	**183**
Ping from 10.0.0.2	icmp_seq=38	ttl=64	**185**
Ping from 10.0.0.2	icmp_seq=39	ttl=64	**189**
		Average	**143.774359**

8.1.2 Without Learning L2 Switch acts as a simple hub

			Time in ms
Ping from 10.0.0.2	icmp_seq=1	ttl=64	**156**
Ping from 10.0.0.2	icmp_seq=2	ttl=64	**46**
Ping from 10.0.0.2	icmp_seq=3	ttl=64	**0.237**
Ping from 10.0.0.2	icmp_seq=4	ttl=64	**0.053**
Ping from 10.0.0.2	icmp_seq=5	ttl=64	**0.044**
Ping from 10.0.0.2	icmp_seq=6	ttl=64	**0.072**
Ping from 10.0.0.2	icmp_seq=7	ttl=64	**0.064**
Ping from 10.0.0.2	icmp_seq=8	ttl=64	**0.054**
Ping from 10.0.0.2	icmp_seq=9	ttl=64	**0.057**
Ping from 10.0.0.2	icmp_seq=10	ttl=64	**0.081**
Ping from 10.0.0.2	icmp_seq=11	ttl=64	**0.062**
Ping from 10.0.0.2	icmp_seq=12	ttl=64	**0.052**
Ping from 10.0.0.2	icmp_seq=13	ttl=64	**0.051**
Ping from 10.0.0.2	icmp_seq=14	ttl=64	**0.065**
Ping from 10.0.0.2	icmp_seq=15	ttl=64	**0.105**
Ping from 10.0.0.2	icmp_seq=16	ttl=64	**0.061**
Ping from 10.0.0.2	icmp_seq=17	ttl=64	**0.048**
Ping from 10.0.0.2	icmp_seq=18	ttl=64	**0.051**
Ping from 10.0.0.2	icmp_seq=19	ttl=64	**0.055**
Ping from 10.0.0.2	icmp_seq=20	ttl=64	**0.048**
Ping from 10.0.0.2	icmp_seq=21	ttl=64	**0.068**
Ping from 10.0.0.2	icmp_seq=22	ttl=64	**0.067**
Ping from 10.0.0.2	icmp_seq=23	ttl=64	**0.066**
Ping from 10.0.0.2	icmp_seq=24	ttl=64	**0.076**
Ping from 10.0.0.2	icmp_seq=25	ttl=64	**0.047**
Ping from 10.0.0.2	icmp_seq=26	ttl=64	**0.139**
Ping from 10.0.0.2	icmp_seq=27	ttl=64	**0.055**
Ping from 10.0.0.2	icmp_seq=28	ttl=64	**0.059**
Ping from 10.0.0.2	icmp_seq=29	ttl=64	**0.06**

Ping from 10.0.0.2	icmp_seq=30	ttl=64	**0.048**
Ping from 10.0.0.2	icmp_seq=31	ttl=64	**0.06**
Ping from 10.0.0.2	icmp_seq=32	ttl=64	**0.047**
Ping from 10.0.0.2	icmp_seq=33	ttl=64	**0.049**
Ping from 10.0.0.2	icmp_seq=34	ttl=64	**0.048**
Ping from 10.0.0.2	icmp_seq=35	ttl=64	**0.049**
Ping from 10.0.0.2	icmp_seq=36	ttl=64	**0.059**
Ping from 10.0.0.2	icmp_seq=37	ttl=64	**0.046**
Ping from 10.0.0.2	icmp_seq=38	ttl=64	**0.047**
Ping from 10.0.0.2	icmp_seq=39	ttl=64	**0.049**
		Average	**5.241**

As we can there is significant decrease in average time (RTT) a packet transfers between two hosts.

9 CONCLUSIONS & RECOMMENDATIONS

We have successfully implemented a MAC learning switch on the OpenDaylight platform. We have observed that inducing the capability of learning into a switch increases its performance multifold, as evident in the RTT times observed. We plan to replicate this implementation to use the MD-SAL or Model-Driven SAL in the latest release of OpenDaylight. Also, an interesting extension to the MAC learning switch would be to use sophisticated machine learning algorithms.

10 BIBLIOGRAPHY

[1] Rahamatullah Khondoker, Adel Zaalouk, Ronald Marx, Kpatcha Bayarou, "Feature-based Comparison and Selection of Software Defined Networking (SDN) Controllers", IEEE-: WCCAIS 2014, World Congress on Computer Applications and Information Systems : Hammamet, Tunisia, 17-19 January 2014 Piscataway, NJ: IEEE, 2014.

[2] Achim Autenrieth, Thomas Szyrkowiec, Klaus Grobe, Jörg-Peter Elbers, Paweł Kaczmarek, Paweł Kostecki, Wolfgang Kellerer, "Evaluation of Virtualization Models for Optical Connectivity Service Providers", 18th International Conference on Optical Network Design and Modeling (ONDM), Stockholm, Sweden, May 2014.

[3] Medved, J. etal., "OpenDaylight: Towards a Model-Driven SDN Controller Architecture", IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014.

[4] Izquierdo-Zaragoza, J.-L. etal. "Leveraging Net2Plan planning tool for network orchestration in OpenDaylight", International Conference on Smart Communications in Network Technologies (SaCoNeT), 2014

[5] https://wiki.opendaylight.org/view/OpenDaylight_Controller:Eclipse_CLI_Setup

11 APPENDICES

1. **Program Source code**

```
package edu.scu.coen233.l2switch;

import java.util.List;
import java.util.ArrayList;
import java.util.Set;
import java.lang.String;
import java.util.Map;
import java.util.HashMap;


import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import org.osgi.framework.Bundle;
import org.osgi.framework.BundleContext;
import org.osgi.framework.BundleException;
import org.osgi.framework.FrameworkUtil;
import org.opendaylight.controller.sal.core.ConstructionException;
import org.opendaylight.controller.sal.core.Node;
import org.opendaylight.controller.sal.core.NodeConnector;
import org.opendaylight.controller.sal.flowprogrammer.IFlowProgrammerService;
import org.opendaylight.controller.sal.flowprogrammer.Flow;
import org.opendaylight.controller.sal.packet.BitBufferHelper;
import org.opendaylight.controller.sal.packet.Ethernet;
import org.opendaylight.controller.sal.packet.IDataPacketService;
import org.opendaylight.controller.sal.packet.IListenDataPacket;
import org.opendaylight.controller.sal.packet.Packet;
import org.opendaylight.controller.sal.packet.PacketResult;
import org.opendaylight.controller.sal.packet.RawPacket;
import org.opendaylight.controller.sal.action.Action;
import org.opendaylight.controller.sal.action.Output;
import org.opendaylight.controller.sal.match.Match;
import org.opendaylight.controller.sal.match.MatchType;
import org.opendaylight.controller.sal.match.MatchField;
import org.opendaylight.controller.sal.utils.Status;
import org.opendaylight.controller.switchmanager.ISwitchManager;

public class L2Switch implements IListenDataPacket {
```

```
    private static final Logger logger = LoggerFactory
            .getLogger(L2Switch.class);
    private ISwitchManager sw_manager = null;
    private IFlowProgrammerService flowService = null;
    private IDataPacketService dataPacketService = null;
    private Map<Long, NodeConnector> mac_to_port = new HashMap<Long,
NodeConnector>();
    private String function = "switch";

            /*
            * Standard methods while handling OSGi - Data Packet Service
            *
            */
    void setDataPacketService(IDataPacketService s) {
      this.dataPacketService = s;
    }

    void unsetDataPacketService(IDataPacketService s) {
      if (this.dataPacketService == s) {
        this.dataPacketService = null;
      }
    }

    public void setFlowProgrammerService(IFlowProgrammerService s)
    {
      this.flowService = s;
    }

    public void unsetFlowProgrammerService(IFlowProgrammerService s) {
      if (this.flowService == s) {
        this.flowService = null;
      }
    }

    void setSwitchManager(ISwitchManager s) {
      logger.debug("sw_manager set");
      this.sw_manager = s;
    }

    void unsetSwitchManager(ISwitchManager s) {
      if (this.sw_manager == s) {
        logger.debug("sw_manager removed!");
        this.sw_manager = null;
      }
```

```
  }

  /**
   * Function called by the dependency manager when all the required
   * dependencies are satisfied
   *
   */
  void init() {
    logger.info("Initialized");
                /*
                * Removing conflicting bundles for ARP handling
                */
    BundleContext bundleContext =
FrameworkUtil.getBundle(this.getClass()).getBundleContext();
    for(Bundle bundle : bundleContext.getBundles()) {
      if (bundle.getSymbolicName().contains("simpleforwarding")) {
        try {
          bundle.uninstall();
        } catch (BundleException e) {
          logger.error("Exception in Bundle uninstall "+bundle.getSymbolicName(),
e);
        }
      }
    }

  }

  /**
   * Function called by the dependency manager when at least one
   * dependency become unsatisfied or when the component is shutting
   * down because for example bundle is being stopped.
   *
   */
  void destroy() {
        logger.info("destroyed");
  }

  /**
   * Function called by dependency manager after "init ()" is called
   * and after the services provided by the class are registered in
   * the service registry
   *
   */
  void start() {
```

```
    logger.info("Started");
}

/**
 * Function called by the dependency manager before the services
 * exported by the component are unregistered, this will be
 * followed by a "destroy ()" calls
 *
 */
void stop() {
    logger.info("Stopped");
}

private void floodPacket(RawPacket packet_in) {
    NodeConnector inc_connector = packet_in.getIncomingNodeConnector();
    Node in_node = inc_connector.getNode();

    Set<NodeConnector> nodeConnectors =
            this.sw_manager.getUpNodeConnectors(in_node);

    for (NodeConnector p : nodeConnectors) {
        if (!p.equals(inc_connector)) {
            try {
                RawPacket destPkt = new RawPacket(packet_in);
                destPkt.setOutgoingNodeConnector(p);
                this.dataPacketService.transmitDataPacket(destPkt);
            } catch (ConstructionException e2) {
                continue;
            }
        }
    }
}

@Override
public PacketResult receiveDataPacket(RawPacket packet_in) {
    if (packet_in == null) {
        return PacketResult.IGNORED;
    }

    NodeConnector in_connector = packet_in.getIncomingNodeConnector();
    Node in_node = in_connector.getNode();

    if(!function.equals("hub")){
        NodeConnector dst_connector = null;
```

```
        Packet packet = this.dataPacketService.decodeDataPacket(packet_in);

        if (!(packet instanceof Ethernet)) {
          return PacketResult.IGNORED;
                        }

                        NodeConnector inc_connector =
packet_in.getIncomingNodeConnector();

                        byte[] srcMAC = ((Ethernet) packet).getSourceMACAddress();
                        long srcMAC_val = BitBufferHelper.toNumber(srcMAC);
                        this.mac_to_port.put(srcMAC_val, in_connector);

      byte[] dstMAC = ((Ethernet)packet).getDestinationMACAddress();
      long dstMAC_val = BitBufferHelper.toNumber(dstMAC);
      dst_connector = this.mac_to_port.get(dstMAC_val);

      if (dst_connector != null) {
        List<Action> actions = new ArrayList<Action>();
        actions.add(new Output(dst_connector));

        Match match = new Match();
        match.setField( new MatchField(MatchType.IN_PORT, in_connector) );
        match.setField( new MatchField(MatchType.DL_DST, dstMAC.clone()) );


        // Create new the flow on the network node
        Flow f = new Flow(match, actions);

        Status status = flowService.addFlow(in_node, f);
        if (!status.isSuccess()) {
           logger.warn("Plugin failed at {}. The failure is: {}",
                    f, status.getDescription());
           return PacketResult.IGNORED;
         }
         logger.info("Installed flow {} in node {}", f, in_node);
      } else{
        floodPacket(packet_in);
      }
    }
    return PacketResult.CONSUME;
  }
```

}

2. Maven POM files

POM1.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd"
xmlns="http://maven.apache.org/POM/4.0.0"

      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <modelVersion>4.0.0</modelVersion>


 <parent>

  <groupId>org.sdnhub.odl.common</groupId>

  <artifactId>commons</artifactId>

  <version>1.1.0-SNAPSHOT</version>

  <relativePath>commons/parent</relativePath>

 </parent>


 <artifactId>l2switch</artifactId>

 <version>0.6.0-SNAPSHOT</version>

 <packaging>pom</packaging>

 <modules>

  <module>commons/parent</module>

  <module>coen233_l2switch</module>

  <module>distribution/opendaylight-osgi-adsal</module>

 </modules>


</project>
```

POM2.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd"
xmlns="http://maven.apache.org/POM/4.0.0"

      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <modelVersion>4.0.0</modelVersion>

 <prerequisites>

  <maven>3.0</maven>

 </prerequisites>

 <parent>

  <groupId>org.opendaylight.odlparent</groupId>

  <artifactId>odlparent</artifactId>

  <version>1.4.2-SNAPSHOT</version>

  <relativePath></relativePath>

 </parent>


 <groupId>org.sdnhub.odl.common</groupId>

 <artifactId>commons</artifactId>

 <name>Open Day Light Common POM</name>

 <version>1.1.0-SNAPSHOT</version>

 <packaging>pom</packaging>


 <properties>

  <sdnhubnexus>http://repo.sdnhub.org:8080/nexus/content</sdnhubnexus>

  <nexusproxy>http://nexus.opendaylight.org/content</nexusproxy>

  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

  <java.version.source>1.7</java.version.source>

  <java.version.target>1.7</java.version.target>
```

```
    <checkstyle.skip>true</checkstyle.skip>

    <skip.distribution>false</skip.distribution>

    <spring.current.version>4.0.2.RELEASE</spring.current.version>

    <sdnhub.of_plugins.version>0.1.0-SNAPSHOT</sdnhub.of_plugins.version>

    <!-- ODL Controller Dependency Versions -->

    <appauth.version>0.4.2-SNAPSHOT</appauth.version>

    <arphandler.version>0.5.2-SNAPSHOT</arphandler.version>

    <bundlescanner.implementation.version>0.4.2-
SNAPSHOT</bundlescanner.implementation.version>

    <bundlescanner.version>0.4.2-SNAPSHOT</bundlescanner.version>

    <clustering.services.version>0.5.1-SNAPSHOT</clustering.services.version>

    <clustering.services_implementation.version>0.4.3-
SNAPSHOT</clustering.services_implementation.version>

    <commons.httpclient.version>0.1.2-SNAPSHOT</commons.httpclient.version>

    <configuration.implementation.version>0.4.3-
SNAPSHOT</configuration.implementation.version>

    <configuration.version>0.4.3-SNAPSHOT</configuration.version>

    <connectionmanager.version>0.1.2-SNAPSHOT</connectionmanager.version>

    <connectionmanager.northbound.version>0.1.2-
SNAPSHOT</connectionmanager.northbound.version>

    <containermanager.version>0.5.2-SNAPSHOT</containermanager.version>

    <containermanager.implementation.version>0.5.2-
SNAPSHOT</containermanager.implementation.version>

    <containermanager.northbound.version>0.4.2-
SNAPSHOT</containermanager.northbound.version>

    <controllermanager.northbound.version>0.0.2-
SNAPSHOT</controllermanager.northbound.version>

    <devices.web.version>0.4.2-SNAPSHOT</devices.web.version>

    <flows.web.version>0.4.2-SNAPSHOT</flows.web.version>

    <flowprogrammer.northbound.version>0.4.2-
SNAPSHOT</flowprogrammer.northbound.version>

    <forwarding.staticrouting>0.5.2-SNAPSHOT</forwarding.staticrouting>
```

<forwarding.staticrouting.northbound.version>0.4.2-SNAPSHOT</forwarding.staticrouting.northbound.version>

<forwardingrulesmanager.version>0.6.0-SNAPSHOT</forwardingrulesmanager.version>

<forwardingrulesmanager.implementation.version>0.4.2-SNAPSHOT</forwardingrulesmanager.implementation.version>

<hosttracker.api.version>0.5.2-SNAPSHOT</hosttracker.api.version>

<hosttracker.implementation.version>0.5.2-SNAPSHOT</hosttracker.implementation.version>

<hosttracker.northbound.version>0.4.2-SNAPSHOT</hosttracker.northbound.version>

<logging.bridge.version>0.4.2-SNAPSHOT</logging.bridge.version>

<protocol_plugins.stub.version>0.4.2-SNAPSHOT</protocol_plugins.stub.version>

<routing.dijkstra_implementation.version>0.4.2-SNAPSHOT</routing.dijkstra_implementation.version>

<sal.version>0.8.1-SNAPSHOT</sal.version>

<sal.implementation.version>0.4.2-SNAPSHOT</sal.implementation.version>

<sal.connection.version>0.1.2-SNAPSHOT</sal.connection.version>

<sal.networkconfiguration.version>0.0.3-SNAPSHOT</sal.networkconfiguration.version>

<security.version>0.4.2-SNAPSHOT</security.version>

<statistics.northbound.version>0.4.2-SNAPSHOT</statistics.northbound.version>

<statisticsmanager.implementation.version>0.4.2-SNAPSHOT</statisticsmanager.implementation.version>

<statisticsmanager.version>0.5.1-SNAPSHOT</statisticsmanager.version>

<subnets.northbound.version>0.4.2-SNAPSHOT</subnets.northbound.version>

<switchmanager.api.version>0.7.1-SNAPSHOT</switchmanager.api.version>

<switchmanager.implementation.version>0.4.2-SNAPSHOT</switchmanager.implementation.version>

<switchmanager.northbound.version>0.4.2-SNAPSHOT</switchmanager.northbound.version>

<topology.northbound.version>0.4.2-SNAPSHOT</topology.northbound.version>

<topologymanager.version>0.4.2-SNAPSHOT</topologymanager.version>

<topology.web.version>0.4.2-SNAPSHOT</topology.web.version>

<troubleshoot.web.version>0.4.2-SNAPSHOT</troubleshoot.web.version>

<usermanager.implementation.version>0.4.2-SNAPSHOT</usermanager.implementation.version>

<usermanager.northbound.version>0.0.2-SNAPSHOT</usermanager.northbound.version>

<usermanager.version>0.4.2-SNAPSHOT</usermanager.version>

<web.version>0.4.2-SNAPSHOT</web.version>

<mdsal.version>1.1-SNAPSHOT</mdsal.version>

<openflowj.version>1.0.2</openflowj.version>

<protocol_plugins.openflow.version>0.4.2-SNAPSHOT</protocol_plugins.openflow.version>

<openflowplugin.version>0.0.3-SNAPSHOT</openflowplugin.version>

<openflowplugin-nicira.version>0.0.3-SNAPSHOT</openflowplugin-nicira.version>

<openflowplugin-extension.version>0.0.3-SNAPSHOT</openflowplugin-extension.version>

<openflowjava-nicira.version>0.0.3-SNAPSHOT</openflowjava-nicira.version>

<openflowjava-extension.version>0.0.3-SNAPSHOT</openflowjava-extension.version>

<openflowjava.version>0.5-SNAPSHOT</openflowjava.version>

<ovsdb.library.version>1.0.0-SNAPSHOT</ovsdb.library.version>

<ovsdb.plugin.version>1.0.0-SNAPSHOT</ovsdb.plugin.version>

<ovsdb.schema.openvswitch.version>1.0.0-SNAPSHOT</ovsdb.schema.openvswitch.version>

<ovsdb.schema.hardwarevtep.version>1.0.0-SNAPSHOT</ovsdb.schema.hardwarevtep.version>

<controller.model.version>1.1-SNAPSHOT</controller.model.version>

<yang.binding.version>0.6.2-SNAPSHOT</yang.binding.version>

<yangtools.version>0.6.2-SNAPSHOT</yangtools.version>

```
<controller.config.version>0.2.5-SNAPSHOT</controller.config.version>
<northbound.commons.version>0.4.2-SNAPSHOT</northbound.commons.version>
<!-- 3rd Pary Dependency Versions -->
<sf.net.jung2.version>2.0.1</sf.net.jung2.version>
<commons.collection.version>1.0</commons.collection.version>
<portlet.version>2.0</portlet.version>
<httpcomponents.version>4.2.1</httpcomponents.version>
<karaf.shell.version>3.0.0</karaf.shell.version>
<nsf.version>0.4.2-SNAPSHOT</nsf.version>
<sfc-model.version>0.0.1-SNAPSHOT</sfc-model.version>
<netty.version>4.0.2.Final</netty.version>
<adsal_L2_forwarding.version>0.5.0-SNAPSHOT</adsal_L2_forwarding.version>
<mdsal_L2_forwarding.version>0.1.0-SNAPSHOT</mdsal_L2_forwarding.version>
<plugin_exercise.version>0.1.0-SNAPSHOT</plugin_exercise.version>
<!-- Karaf Dependency Versions -->
<karaf.empty.version>1.4.2-SNAPSHOT</karaf.empty.version>
<feature.adsal-tutorial.version>1.0.0-SNAPSHOT</feature.adsal-tutorial.version>
<feature.mdsal-tutorial.version>1.0.0-SNAPSHOT</feature.mdsal-tutorial.version>
<feature.plugin-exercise.version>1.0.0-SNAPSHOT</feature.plugin-
exercise.version>
<karaf.shell.version>3.0.0</karaf.shell.version>
<karaf.base.version>1.4.2-SNAPSHOT</karaf.base.version>
<karaf.nsf.version>0.4.2-SNAPSHOT</karaf.nsf.version>
<karaf.mdsal.version>1.1-SNAPSHOT</karaf.mdsal.version>
<karaf.adsal.version>0.8.1-SNAPSHOT</karaf.adsal.version>
<karaf.adsal-compat.version>1.4.2-SNAPSHOT</karaf.adsal-compat.version>
<karaf.ovsdb.version>1.0.0-SNAPSHOT</karaf.ovsdb.version>
<karaf.empty.version>1.4.2-SNAPSHOT</karaf.empty.version>
<karaf.restconf.version>1.1-SNAPSHOT</karaf.restconf.version>
```

```
 </properties>

 <build>
  <pluginManagement>
   <plugins>
    <plugin>
     <groupId>org.apache.maven.plugins</groupId>
     <artifactId>maven-compiler-plugin</artifactId>
     <configuration>
      <source>${java.version.source}</source>
      <target>${java.version.target}</target>
      <testSource>${java.version.source}</testSource>
      <testTarget>${java.version.target}</testTarget>
     </configuration>
    </plugin>
   </plugins>
  </pluginManagement>
 </build>

 <repositories>
  <!-- OpenDayLight Released artifact -->
  <repository>
   <id>opendaylight-release</id>
   <name>opendaylight-release</name>
<url>http://nexus.opendaylight.org/content/repositories/opendaylight.release/</url
>
   <releases>
    <enabled>true</enabled>
```

```
    </releases>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
  </repository>
  <!-- OpenDayLight Snapshot artifact -->
  <repository>
    <id>opendaylight-snapshot</id>
    <name>opendaylight-snapshot</name>
<url>http://nexus.opendaylight.org/content/repositories/opendaylight.snapshot/</u
rl>
    <releases>
      <enabled>false</enabled>
    </releases>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
  </repository>
  <!-- SDNHub Public Repo group -->
  <repository>
    <id>sdnhub-nexus</id>
    <name>sdnhub</name>
    <url>http://repo.sdnhub.org:8080/nexus/content/groups/public</url>
    <releases>
      <enabled>false</enabled>
    </releases>
    <snapshots>
      <enabled>true</enabled>
```

```
    </snapshots>
   </repository>
  </repositories>

  <pluginRepositories>
   <pluginRepository>
    <id>opendaylight-release</id>
    <name>opendaylight-release</name>
<url>http://nexus.opendaylight.org/content/repositories/opendaylight.release/</url
>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
    <releases>
      <enabled>true</enabled>
    </releases>
   </pluginRepository>
   <pluginRepository>
    <id>opendaylight-snapshot</id>
    <name>opendaylight-snapshot</name>
<url>http://nexus.opendaylight.org/content/repositories/opendaylight.snapshot/</u
rl>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
    <releases>
      <enabled>false</enabled>
    </releases>
```

```
</pluginRepository>
</pluginRepositories>

<dependencies>
 <!-- SDNHub common artifacts -->
 <dependency>
   <groupId>org.sdnhub.odl</groupId>
   <artifactId>web.brandfragment</artifactId>
   <version>0.1.0-SNAPSHOT</version>
 </dependency>
 <!-- Other common bundles -->
 <dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-classic</artifactId>
 </dependency>
 <dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-core</artifactId>
 </dependency>
  <dependency>
  <groupId>org.apache.felix</groupId>
  <artifactId>org.apache.felix.dependencymanager</artifactId>
  </dependency>
 <dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-annotations</artifactId>
 </dependency>
 <dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
```

```
  <artifactId>jackson-core</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-databind</artifactId>
</dependency>
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>com.google.guava</groupId>
  <artifactId>guava</artifactId>
</dependency>
<dependency>
  <groupId>commons-codec</groupId>
  <artifactId>commons-codec</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>commons-lang</groupId>
  <artifactId>commons-lang</artifactId>
</dependency>
<dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>org.eclipse.osgi</artifactId>
</dependency>
<dependency>
```

```
    <groupId>org.eclipse.equinox</groupId>
    <artifactId>region</artifactId>
    <version>1.0.0.v20110506</version>
   </dependency>
   <dependency>
    <groupId>io.netty</groupId>
    <artifactId>netty-all</artifactId>
    <version>${netty.version}</version>
   </dependency>
   <dependency>
    <groupId>commons-collections</groupId>
    <artifactId>commons-collections</artifactId>
    <version>${commons.collection.version}</version>
   </dependency>
   <dependency>
    <groupId>javax.portlet</groupId>
    <artifactId>portlet-api</artifactId>
    <version>${portlet.version}</version>
   </dependency>
   <dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpcore-nio</artifactId>
    <version>${httpcomponents.version}</version>
   </dependency>
  </dependencies>
</project>
```

POM3.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">
 <modelVersion>4.0.0</modelVersion>
 <parent>
  <groupId>org.sdnhub.odl.common</groupId>
  <artifactId>commons</artifactId>
  <version>1.1.0-SNAPSHOT</version>
  <relativePath>../../commons/parent/</relativePath>
 </parent>

 <artifactId>distribution-osgi-adsal</artifactId>
 <packaging>pom</packaging>
 <name>OpenDaylight OSGi AD-SAL Distribution Pack</name>

 <dependencies>
  <!-- L2 Switch -->
  <dependency>
   <groupId>edu.scu.coen233.l2switch</groupId>
   <artifactId>coen233_l2switch</artifactId>
   <version>0.5.0-SNAPSHOT</version>
  </dependency>

  <!-- SDNHub Artifacts -->
  <dependency>
   <groupId>org.sdnhub.odl</groupId>
   <artifactId>ofbroker</artifactId>
    <version>0.1.0-SNAPSHOT</version>
```

```
</dependency>
<dependency>
 <groupId>org.sdnhub.odl</groupId>
 <artifactId>protocol_plugins.openflow13</artifactId>
  <version>0.1.0-SNAPSHOT</version>
</dependency>
<dependency>
 <groupId>org.sdnhub.odl</groupId>
 <artifactId>protocol_plugins.openflow10</artifactId>
  <version>0.1.0-SNAPSHOT</version>
</dependency>
<!-- ODL Web -->
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>web</artifactId>
 <version>${web.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>flows.web</artifactId>
 <version>${flows.web.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>devices.web</artifactId>
 <version>${devices.web.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
```

```
  <artifactId>troubleshoot.web</artifactId>
  <version>${troubleshoot.web.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>topology.web</artifactId>
  <version>${topology.web.version}</version>
</dependency>
<!-- ODL Mirror -->
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>arphandler</artifactId>
  <version>${arphandler.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>bundlescanner</artifactId>
  <version>${bundlescanner.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>bundlescanner.implementation</artifactId>
  <version>${bundlescanner.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>clustering.services</artifactId>
  <version>${clustering.services.version}</version>
</dependency>
```

```
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>clustering.services-implementation</artifactId>
  <version>${clustering.services_implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>configuration</artifactId>
  <version>${configuration.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>configuration.implementation</artifactId>
  <version>${configuration.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>containermanager</artifactId>
  <version>${containermanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>appauth</artifactId>
  <version>${appauth.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>containermanager.implementation</artifactId>
  <version>${containermanager.implementation.version}</version>
```

```
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>forwarding.staticrouting</artifactId>
 <version>${forwarding.staticrouting}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>routing.dijkstra_implementation</artifactId>
 <version>${routing.dijkstra_implementation.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>forwardingrulesmanager</artifactId>
 <version>${forwardingrulesmanager.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>forwardingrulesmanager.implementation</artifactId>
 <version>${forwardingrulesmanager.implementation.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>hosttracker</artifactId>
 <version>${hosttracker.api.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>hosttracker.implementation</artifactId>
```

```
  <version>${hosttracker.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>switchmanager</artifactId>
  <version>${switchmanager.api.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>switchmanager.implementation</artifactId>
  <version>${switchmanager.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>statisticsmanager</artifactId>
  <version>${statisticsmanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>statisticsmanager.implementation</artifactId>
  <version>${statisticsmanager.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>topologymanager</artifactId>
  <version>${topologymanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
```

```
  <artifactId>usermanager</artifactId>
  <version>${usermanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>usermanager.implementation</artifactId>
  <version>${usermanager.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>sal</artifactId>
  <version>${sal.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>sal.implementation</artifactId>
  <version>${sal.implementation.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>logging.bridge</artifactId>
  <version>${logging.bridge.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>security</artifactId>
  <version>${security.version}</version>
</dependency>
<dependency>
```

```
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>sal.connection</artifactId>
  <version>${sal.connection.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>sal.connection.implementation</artifactId>
  <version>${sal.connection.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>connectionmanager</artifactId>
  <version>${connectionmanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>connectionmanager.implementation</artifactId>
  <version>${connectionmanager.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller.thirdparty</groupId>
  <artifactId>net.sf.jung2</artifactId>
  <version>${sf.net.jung2.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller.thirdparty</groupId>
  <artifactId>com.sun.jersey.jersey-servlet</artifactId>
  <version>${jersey-servlet.version}</version>
</dependency>
```

```
<dependency>
 <groupId>org.opendaylight.controller.thirdparty</groupId>
 <artifactId>org.apache.catalina.filters.CorsFilter</artifactId>
 <version>${corsfilter.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>sal.networkconfiguration</artifactId>
 <version>${sal.networkconfiguration.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>sal.networkconfiguration.implementation</artifactId>
 <version>${sal.networkconfiguration.version}</version>
</dependency>

<!-- ODL Northbound -->
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>commons.northbound</artifactId>
 <version>${northbound.commons.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
 <artifactId>connectionmanager.northbound</artifactId>
 <version>${connectionmanager.northbound.version}</version>
</dependency>
<dependency>
 <groupId>org.opendaylight.controller</groupId>
```

```
  <artifactId>containermanager.northbound</artifactId>
  <version>${containermanager.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>controllermanager.northbound</artifactId>
  <version>${controllermanager.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>flowprogrammer.northbound</artifactId>
  <version>${flowprogrammer.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>forwarding.staticrouting.northbound</artifactId>
  <version>${forwarding.staticrouting.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>hosttracker.northbound</artifactId>
  <version>${hosttracker.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>statistics.northbound</artifactId>
  <version>${statistics.northbound.version}</version>
</dependency>
<dependency>
```

```
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>subnets.northbound</artifactId>
  <version>${statistics.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>switchmanager.northbound</artifactId>
  <version>${switchmanager.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>topology.northbound</artifactId>
  <version>${topology.northbound.version}</version>
</dependency>
<dependency>
  <groupId>org.opendaylight.controller</groupId>
  <artifactId>usermanager.northbound</artifactId>
  <version>${usermanager.northbound.version}</version>
</dependency>

<!-- enunciate -->
<dependency>
  <groupId>org.codehaus.enunciate</groupId>
  <artifactId>enunciate-core-annotations</artifactId>
</dependency>

<!-- javax -->
<dependency>
  <groupId>equinoxSDK381</groupId>
```

```
  <artifactId>javax.servlet</artifactId>
</dependency>
<dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>javax.servlet.jsp</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.el</artifactId>
</dependency>
<dependency>
  <groupId>eclipselink</groupId>
  <artifactId>javax.resource</artifactId>
</dependency>
<dependency>
  <groupId>org.jboss.spec.javax.transaction</groupId>
  <artifactId>jboss-transaction-api_1.1_spec</artifactId>
</dependency>

<!-- objectweb -->
<dependency>
  <groupId>org.ow2.asm</groupId>
  <artifactId>asm-all</artifactId>
</dependency>

<!-- equinox + felix -->
<dependency>
  <groupId>org.eclipse.equinox.http</groupId>
  <artifactId>servlet</artifactId>
```

```
 </dependency>
 <dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>org.eclipse.osgi</artifactId>
 </dependency>
 <dependency>
  <groupId>org.apache.felix</groupId>
  <artifactId>org.apache.felix.gogo.command</artifactId>
  <version>0.8.0</version>
</dependency>
<dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>org.apache.felix.gogo.runtime</artifactId>
</dependency>
<dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>org.apache.felix.gogo.shell</artifactId>
</dependency>
<dependency>
  <groupId>org.apache.felix</groupId>
  <artifactId>org.apache.felix.dependencymanager</artifactId>
</dependency>
<dependency>
  <groupId>org.apache.felix</groupId>
  <artifactId>org.apache.felix.dependencymanager.shell</artifactId>
</dependency>
 <dependency>
  <groupId>equinoxSDK381</groupId>
  <artifactId>org.eclipse.equinox.console</artifactId>
```

```
</dependency>
<dependency>
 <groupId>equinoxSDK381</groupId>
 <artifactId>org.eclipse.equinox.launcher</artifactId>
</dependency>
<dependency>
 <groupId>equinoxSDK381</groupId>
 <artifactId>org.eclipse.equinox.ds</artifactId>
</dependency>
<dependency>
 <groupId>equinoxSDK381</groupId>
 <artifactId>org.eclipse.equinox.util</artifactId>
</dependency>
<dependency>
 <groupId>equinoxSDK381</groupId>
 <artifactId>org.eclipse.osgi.services</artifactId>
</dependency>

<!-- virgomirror -->
<dependency>
 <groupId>virgomirror</groupId>
 <artifactId>org.eclipse.jdt.core.compiler.batch</artifactId>
</dependency>

<!-- felix -->
<dependency>
 <groupId>org.apache.felix</groupId>
 <artifactId>org.apache.felix.fileinstall</artifactId>
</dependency>
```

```
<!-- Spring Framework -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${spring.current.version}</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-web</artifactId>
  <version>${spring.current.version}</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-websocket</artifactId>
  <version>${spring.current.version}</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-mock</artifactId>
  <version>2.0.8</version>
</dependency>

<!-- Spring Framework ODL -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.asm</artifactId>
</dependency>
<dependency>
```

```
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.aop</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.context</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.context.support</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.core</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.beans</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.expression</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.web</artifactId>
</dependency>
<dependency>
  <groupId>org.aopalliance</groupId>
```

```
  <artifactId>com.springsource.org.aopalliance</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.web.servlet</artifactId>
</dependency>

<!-- Spring security -->
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-config</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-core</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-web</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-taglibs</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.transaction</artifactId>
</dependency>
```

```
<!-- slf4j -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>log4j-over-slf4j</artifactId>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-slf4j</artifactId>
</dependency>

<!-- apache -->
<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-lang3</artifactId>
</dependency>

<!-- logback -->
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-core</artifactId>
</dependency>
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-classic</artifactId>
```

```
</dependency>

<!-- Jersey for JAXRS -->
<dependency>
  <groupId>com.sun.jersey</groupId>
  <artifactId>jersey-core</artifactId>
</dependency>
<dependency>
  <groupId>com.sun.jersey</groupId>
  <artifactId>jersey-server</artifactId>
</dependency>

<!-- JUnit -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <scope>test</scope>
</dependency>

<!-- Web -->
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.servlet.jsp.jstl</artifactId>
</dependency>
<dependency>
```

```
  <groupId>orbit</groupId>
  <artifactId>javax.servlet.jsp.jstl.impl</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-annotations</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-core</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.core</groupId>
  <artifactId>jackson-databind</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.jaxrs</groupId>
  <artifactId>jackson-jaxrs-base</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.jaxrs</groupId>
  <artifactId>jackson-jaxrs-json-provider</artifactId>
</dependency>
<dependency>
  <groupId>com.fasterxml.jackson.module</groupId>
  <artifactId>jackson-module-jaxb-annotations</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
```

```
  <artifactId>org.apache.catalina</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.annotation</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.ejb</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.xml.rpc</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.mail.glassfish</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>javax.activation</artifactId>
</dependency>
<dependency>
  <groupId>eclipselink</groupId>
  <artifactId>javax.persistence</artifactId>
</dependency>
<dependency>
  <groupId>orbit</groupId>
  <artifactId>org.apache.coyote</artifactId>
```

```
</dependency>
<dependency>
 <groupId>orbit</groupId>
 <artifactId>org.apache.juli.extras</artifactId>
</dependency>
<dependency>
 <groupId>orbit</groupId>
 <artifactId>org.apache.tomcat.api</artifactId>
</dependency>
<dependency>
 <groupId>orbit</groupId>
 <artifactId>org.apache.tomcat.util</artifactId>
</dependency>
<dependency>
 <groupId>orbit</groupId>
 <artifactId>org.apache.jasper</artifactId>
</dependency>
<!-- Gemini Web -->
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.gemini.web.core</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.gemini.web.extender</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.gemini.web.tomcat</artifactId>
```

```
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.kernel.equinox.extensions</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.common</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.io</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.math</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.osgi</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.osgi.manifest</artifactId>
</dependency>
<dependency>
 <groupId>geminiweb</groupId>
 <artifactId>org.eclipse.virgo.util.parser.manifest</artifactId>
</dependency>
```

```
<dependency>
 <groupId>orbit</groupId>
 <artifactId>org.apache.el</artifactId>
</dependency>

<dependency>
 <groupId>com.google.guava</groupId>
 <artifactId>guava</artifactId>
</dependency>

<!--Netty-->
<dependency>
 <groupId>io.netty</groupId>
 <artifactId>netty-handler</artifactId>
 </dependency>
<dependency>
 <groupId>io.netty</groupId>
 <artifactId>netty-codec</artifactId>
</dependency>
<dependency>
 <groupId>io.netty</groupId>
 <artifactId>netty-buffer</artifactId>
</dependency>
<dependency>
 <groupId>io.netty</groupId>
 <artifactId>netty-transport</artifactId>
</dependency>
 <dependency>
 <groupId>io.netty</groupId>
```

```
      <artifactId>netty-common</artifactId>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-assembly-plugin</artifactId>
        <executions>
          <execution>
            <id>distro-assembly</id>
            <phase>package</phase>
            <goals>
              <goal>single</goal>
            </goals>
            <configuration>
              <descriptors>
                <descriptor>src/assemble/bin.xml</descriptor>
              </descriptors>
              <skip>${skip.distribution}</skip>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>
</project>
```