# Software Defined Networking

Dr. Nick Feamster
Professor

*In this course, you will learn about software defined networking and how it is changing the way communications networks are managed, maintained, and secured.*

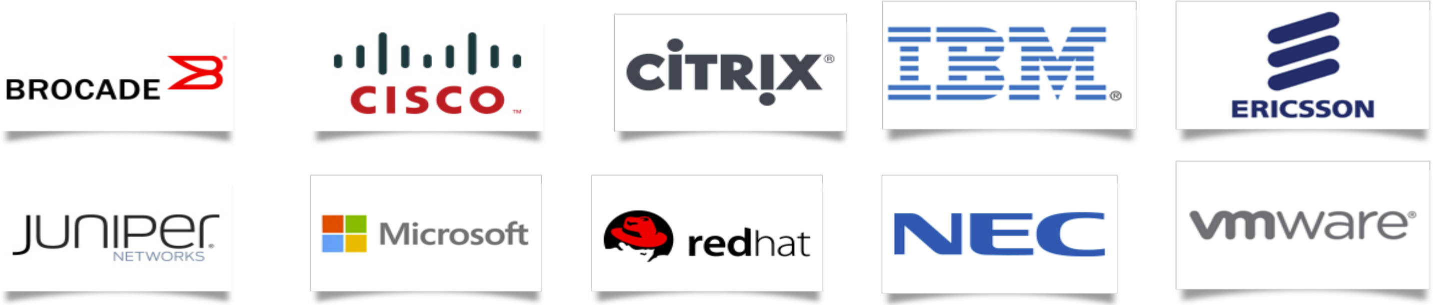
# This Lesson: OpenDaylight

- Overview of OpenDaylight
  - Consortium
  - Architecture
- Demonstration
  - Life of a packet, Web interface
  - Essential ODL functions
- More information: http://sdnhub.org/

Slides adapted from: http://goo.gl/k5Ba5J
Used with permission from SDNHub.

# OpenDaylight Consortium

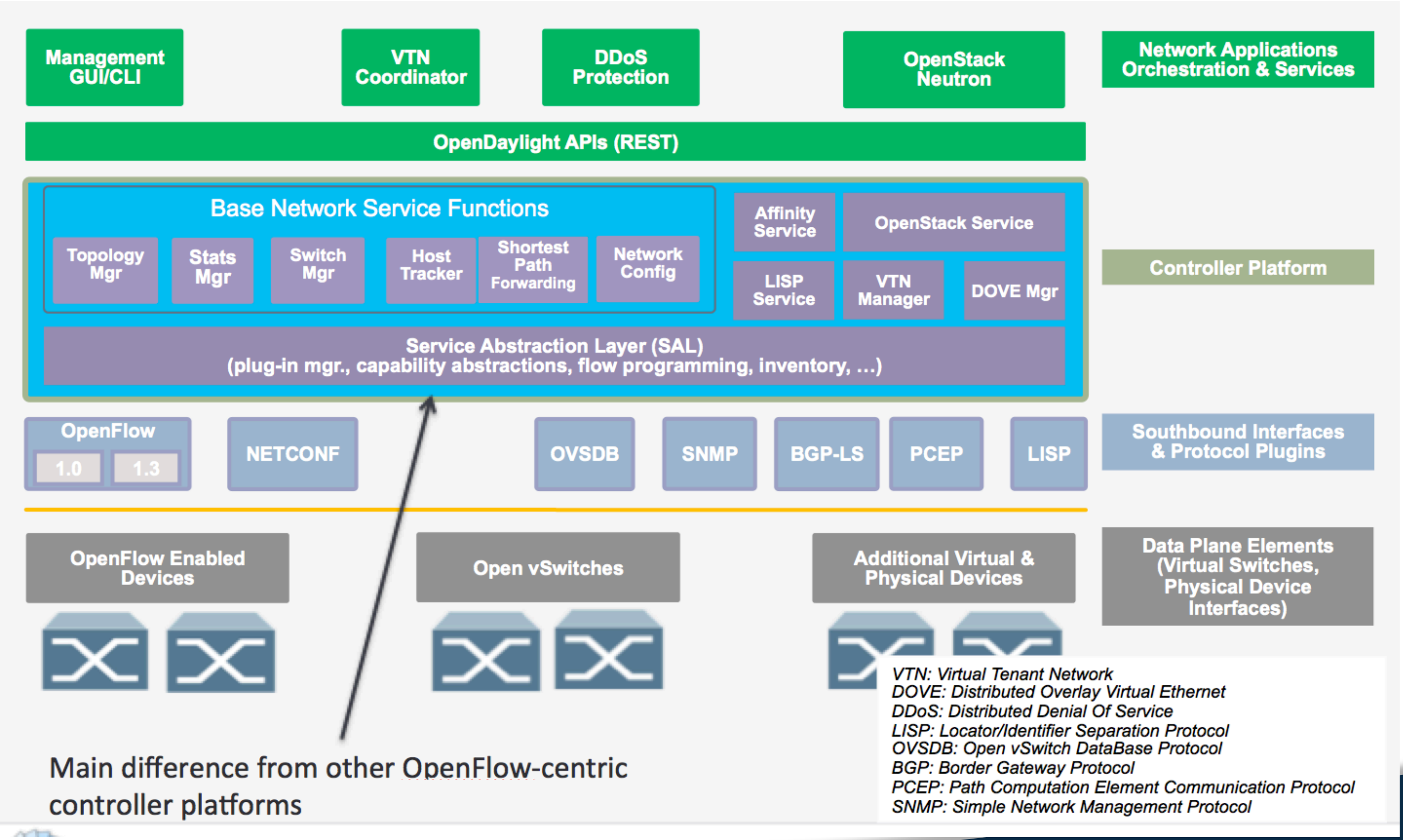
- Heavy industry involvement and backing



- Focused on having an open framework for building upon SDN/NFV innovations
  - Not limited to OpenFlow innovations

# Software Defined Networking

## Hydrogen Release

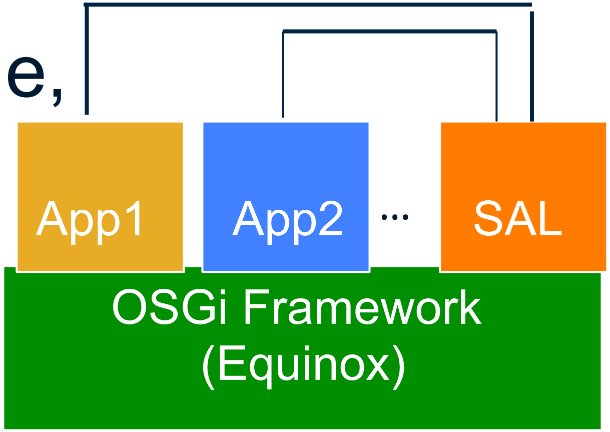


Main difference from other OpenFlow-centric controller platforms

*VTN: Virtual Tenant Network*
*DOVE: Distributed Overlay Virtual Ethernet*
*DDoS: Distributed Denial Of Service*
*LISP: Locator/Identifier Separation Protocol*
*OVSDB: Open vSwitch DataBase Protocol*
*BGP: Border Gateway Protocol*
*PCEP: Path Computation Element Communication Protocol*
*SNMP: Simple Network Management Protocol*

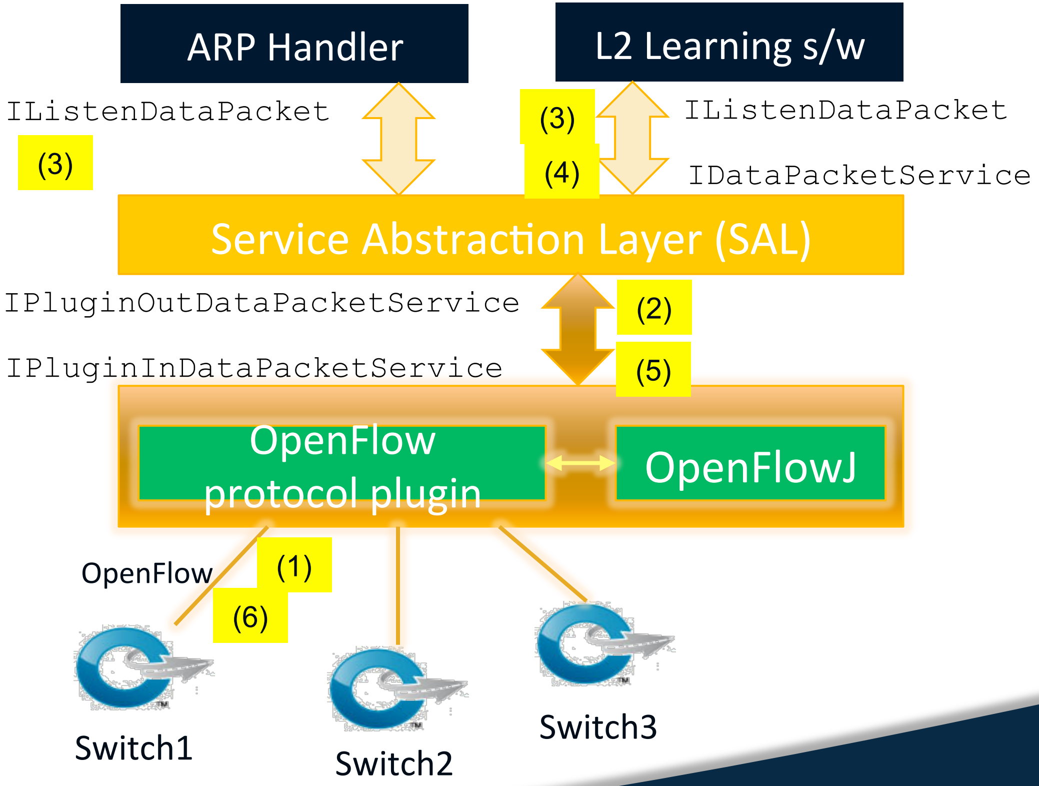
# Java, Maven, OSGi, Interface

- **Java** chosen as an enterprise-grade, cross-platform compatible language
- **Maven** – build system for Java
- **OSGi:**
  - Allows dynamically loading bundles
  - Allows registering dependencies and services exported
  - For exchanging information across bundles
- Java Interfaces are used for event listening, specifications, and forming patterns

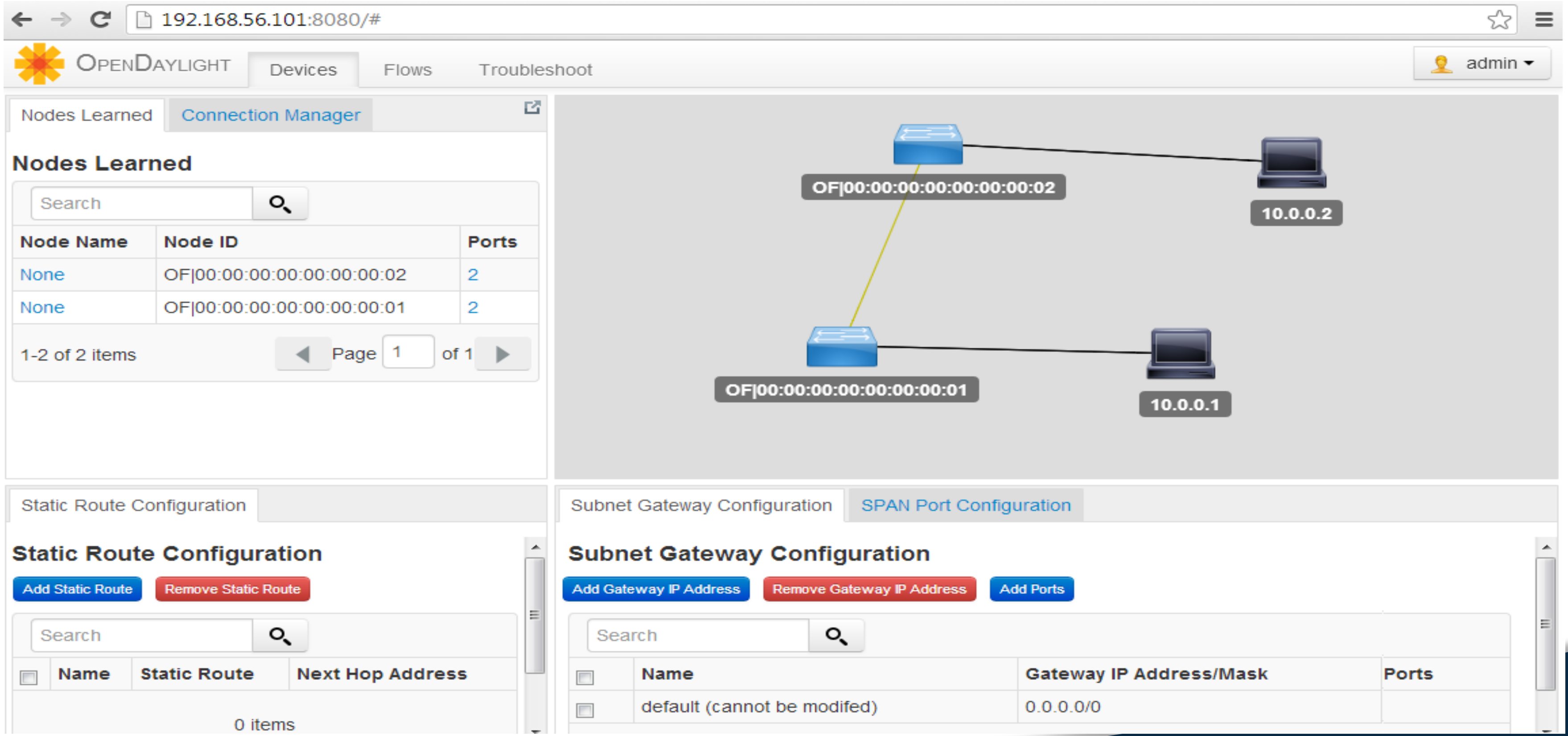


# Life of a Packet

1. A packet arriving at Switch1 will be sent to the appropriate plugin managing the switch
2. The plugin will parse the packet, generate an event for SAL
3. SAL will dispatch the packet to the modules listening for DataPacket
4. Module handles packet and sends packet_out through IDataPacketService
5. SAL dispatches the packet to the modules listening for DataPacket
6. OpenFlow message sent to appropriate switch



# OpenDaylight Web Interface



## Nodes Learned

| Node Name | Node ID | Ports |
|---|---|---|
| None | OF\|00:00:00:00:00:00:00:02 | 2 |
| None | OF\|00:00:00:00:00:00:00:01 | 2 |

1-2 of 2 items Page 1 of 1

## Static Route Configuration

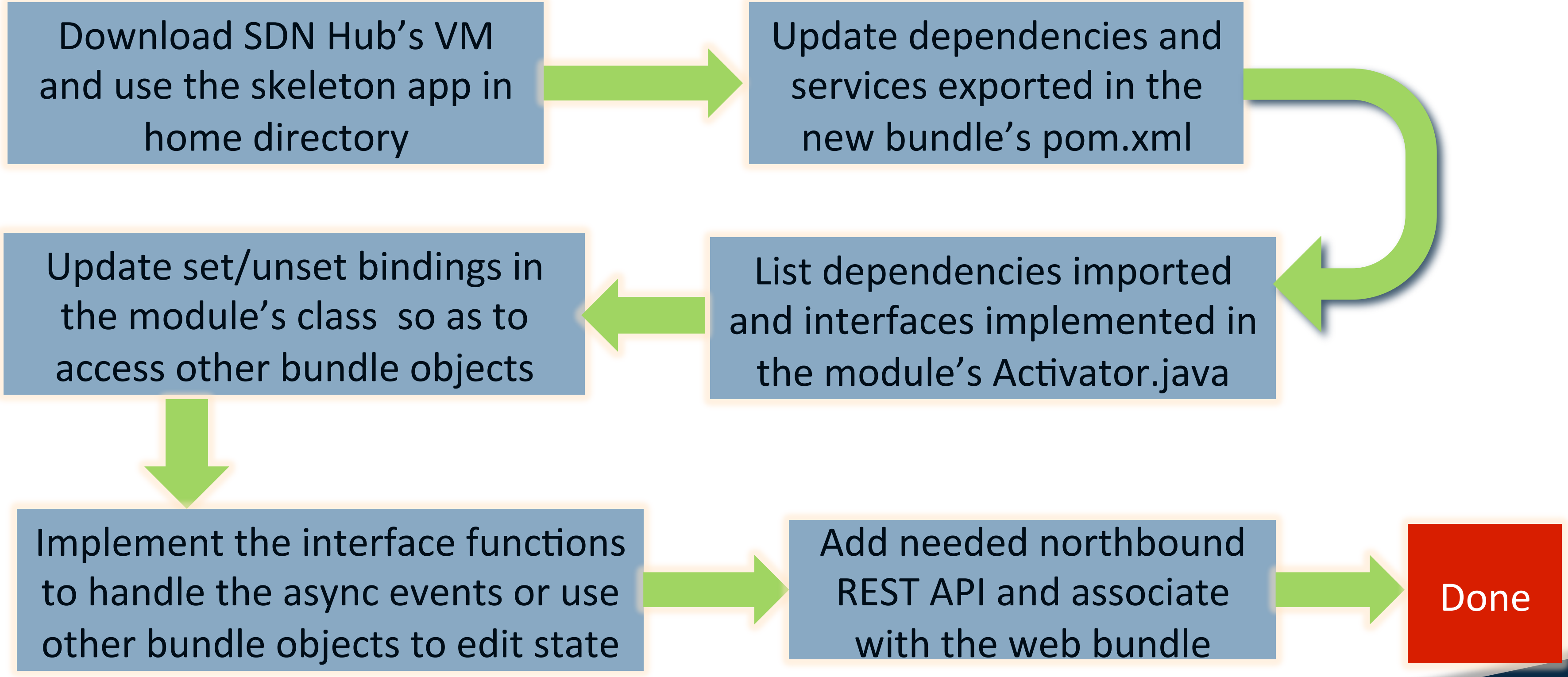
| □ | Name | Static Route | Next Hop Address |
|---|---|---|---|

0 items

## Subnet Gateway Configuration

| □ | Name | Gateway IP Address/Mask | Ports |
|---|---|---|---|
| □ | default (cannot be modifed) | 0.0.0.0/0 | |

# Steps for Writing a new application



Download SDN Hub's VM and use the skeleton app in home directory

Update dependencies and services exported in the new bundle's pom.xml

List dependencies imported and interfaces implemented in the module's Activator.java

Update set/unset bindings in the module's class so as to access other bundle objects

Implement the interface functions to handle the async events or use other bundle objects to edit state

Add needed northbound REST API and associate with the web bundle

Done

# Essential Code Constructs

| | Beacon | OpenDaylight |
|---|---|---|
| Packet_in handling | public class XX implements IOFMessageListener {<br>  public Command receive(IOFSwitch sw, OFMessage msg) throws IOException { … }<br>} | public class XX implements IListenDataPacket {<br>  public PacketResult receiveDataPacket(RawPacket inPkt) { … }<br>} |
| Packet parsing | Ethernet ethHdr = new Ethernet(pi.getPacketData());<br>IPv4 ipv4Hdr = (IPv4) ethHdr.getPayload(); | Ethernet ethHdr  = (Ethernet) this.dataPacketService.decodeDataPacket(inPkt);<br>IPv4 ipv4Hdr = (IPv4) ethHdr.getPayload(); |
| Send msg to switch | OFPacketOut pktOut = new OFPacketOut(packetData, actions,<br>OFPacketOut.BUFFER_ID_NONE);<br>actions.add(new OFActionOutput(port));<br>router.getOutputStream().write(pktOut); | RawPacket destPkt = new RawPacket(inPkt);<br>destPkt.setOutgoingNodeConnector(p);<br>this.dataPacketService.transmitDataPacket(destPkt); |

- Several similarities between Beacon and OpenDaylight
  - This goes beyond just these two controller platforms
  - The above three functions are basic to all controller platforms

# Main Constructs

A. Packet in event handling:

- public class TutorialL2Forwarding implements IListenDataPacket
  - Indicates that the class will handle any packet_in events
- public PacketResult receiveDataPacket(RawPacket inPkt) { ... }
  - Call-back function to implement in the class for receiving packets

B. Packet parsing

- Packet formattedPak = this.dataPacketService.decodeDataPacket(inPkt);
- byte[] srcMAC = ((Ethernet)formattedPak).getSourceMACAddress();
- long srcMAC_val = BitBufferHelper.toNumber(srcMAC);

C. Send message (packet_out or flow_mod) to switch

- RawPacket destPkt = new RawPacket(inPkt);
- destPkt.setOutgoingNodeConnector(p);
- this.dataPacketService.transmitDataPacket(destPkt);

# Useful Interfaces and Bundles

| 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. |

# Useful Interfaces and Bundles

| Bundle | Exported interface | Description |
|---|---|---|
| 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. |

## Summary

- OpenDaylight is an industry-backed effort to develop broader set of SDN solutions
- SDN is no longer just OpenFlow!
  - Possible to integrate a broad set of cloud-based applications
  - Set of OpenFlow functions is similar to other controllers
- Learning curve is significant.  
  SDN Hub has good starter kit!