

L2 Switch Proposal

- [Name](#)
- [Repo Name](#)
- [Description](#)
- [Scope](#)
- [Resources Committed \(developers committed to working\)](#)
- [Initial Committers](#)
- [Vendor Neutral](#)
- [Meets Board Policy \(including IPR\)](#)

Name

L2 Switch

Repo Name

l2switch

Description

ODL Controller project currently contains Service Abstraction Layer (AD-SAL/MD-SAL), Several Services, UI, Samples etc. Depending on the Plugins loaded, Controller also has a certain packet handling/forwarding behavior with the ARPHandlers, Host Tracker, Simple Forwarder etc. This proposal is to separate out the Layer 2 specific handling code into a separate project and also create several reusable services. Following are some of the generic services we intend to create.

PacketHandler:

PacketHandler would provide infrastructure in ODL controller, to process incoming packets and forwarding the packet as required. It would give a pluggable architecture for handling a lifecycle of a "packet" (term from openflow "OFPT_PACKET_IN" message) in ODL controller. This infrastructure can limit to handle only packets with reason as "No matching flow (table-miss flow entry).

- Packets would go through following stages in controller:

Decode : packet goes through "packet decoder(s)" registered/active) Modify : packet goes through "packet modifier" registered for a type of packet Transmit : packet gets dispatched by a dispatcher registered for that type of packet.

- Default behavior could be :

Decode -> do nothing (when no decoder is available /active) Modify -> do nothing (when no packet modifier is participating in packet modification) Transmit -> do nothing/drop (when no packet dispatcher is available to dispatch/transmit packet)

- We would provide a few protocol decoders out of the box. For example, A L2 Ethernet that decodes raw packet to Ethernet Packet. It would:

Have a Yang model for ethernet packet Generate EthernetPacket notification Key Benefits: Packet decoding would happen once and interested parties can subscribe to the type of packet notification. (Eg. EthernetPacket or specific protocol, LLDP packet, ARP packet etc.) Any application/module interested in learning from a packet of a "type" can subscribe to notification of that type and doesn't have to duplicate any decoding code. Yang models would be available for each type of packet that gets decoded in this process. If required some governance can be put so that a "type" of packet gets transmitted back to switch only through one dispatcher, to avoid duplicate packet transmits.

AddressTracker(Endpoint tracker):

HostTracker functionality can be generalized as an Address Tracker to track mappings and reverse mappings from IP Address to Mac Address, Mac Address to Switch/Port etc. It would augment the Node and NodeConnector models appropriately. The tracker would be extensible enough to accommodate other address tuples that can be maintained as well. For example vlan, vti, mpls tag, etc.

PathComputationService:

PathComputationService would utilize graph algorithms to provide optimal path between hosts such as shortest path, lowest cost etc. PathComputationService would leverage and augment Topology data. This service would accommodate and operate over multiple topologies if present.

STPService:

This would provide ability to participate in spanning tree protocol for ODL controlled switches. It would be essential service when ODL is part of hybrid network, i.e. switches controlled by ODL interact with switches that are not controlled by ODL.

FlowWriterService:

FlowWriterService takes care of programming all the intermediate switches with appropriate flows once an optimal path has been computed between two hosts. It would keep track of the mapping from the meta-flow to the individual flows such that appropriate flows can be reprogrammed when a port goes down or an entire switch goes down.

Additional functionality that can be implemented by the L2 Switch includes Mac address aging, Packet filtering, Static Mac Addresses etc.

Component Diagram:

[blocked URL](#)

Scope

Scope of L2 Switch includes:

- Providing basic L2 switch functionality
- Providing exemplary services in support of L2 switch functionality in a way that can be reused by others, including:

Modular event driven PacketHandlersAddress TrackingOptimum Path ComputationsBasic Spanning Tree Protocol (so as to prevent network meltdowns)

Resources Committed (developers committed to working)

Amit Mandke <ammandke@cisco.com>
Alex Fan <alefan@cisco.com>
Evan Zeller <evanzeller@gmail.com>
Raghurama Bhat <ragbhat@cisco.com>

Initial Committers

Amit Mandke <ammandke@cisco.com>
Alex Fan <alefan@cisco.com>
Evan Zeller <evanzeller@gmail.com>
Raghurama Bhat <ragbhat@cisco.com>

Vendor Neutral

Completely Vendor agnostic

Meets Board Policy (including IPR)

Yes. Most/all of the code is written from scratch & it meets Board Policy.