Putting
IPv6
to work

North American IPv6 Summit

Grand Hyatt, Denver, Colorado
September 23-25, 2014

Rocky Mountain IPv6 Task Force
RMv6TF

OpenFlow and IPv6

Two great tastes that taste great together!

Scott Hogg, CTO GTRI

Chair Emeritus RMv6TF

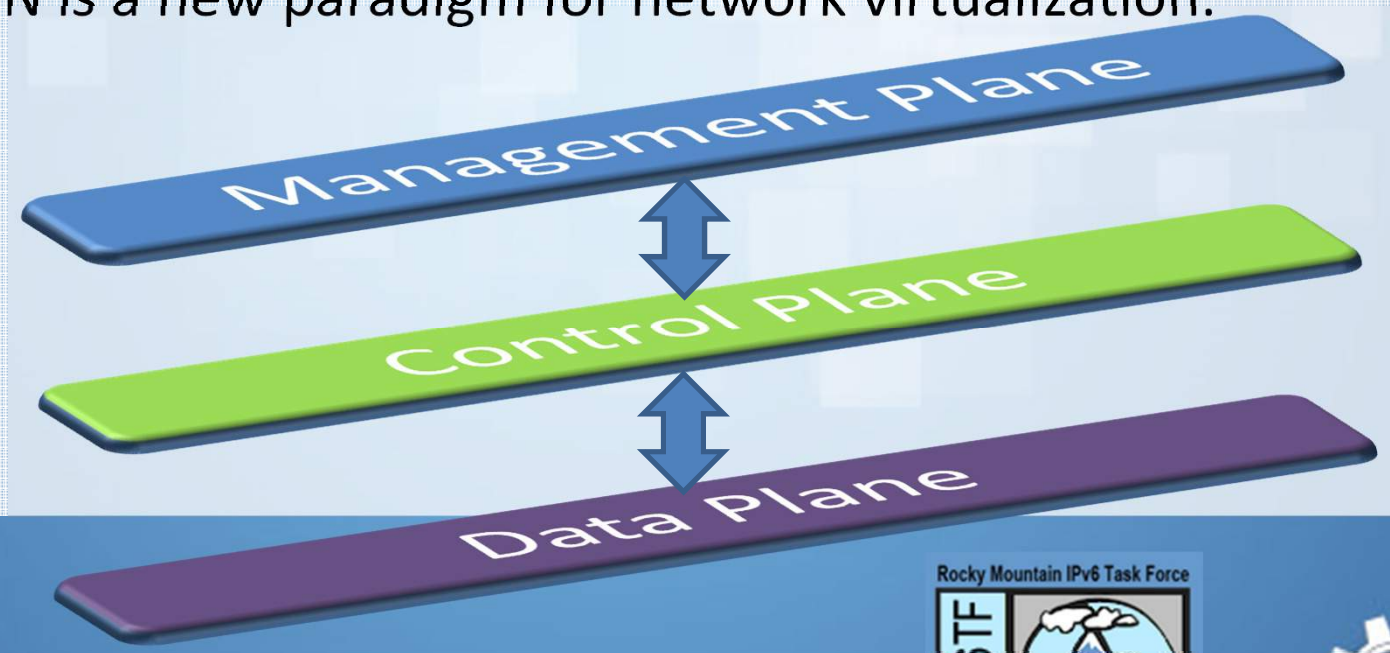
Infoblox IPv6 COE

Today's Outline

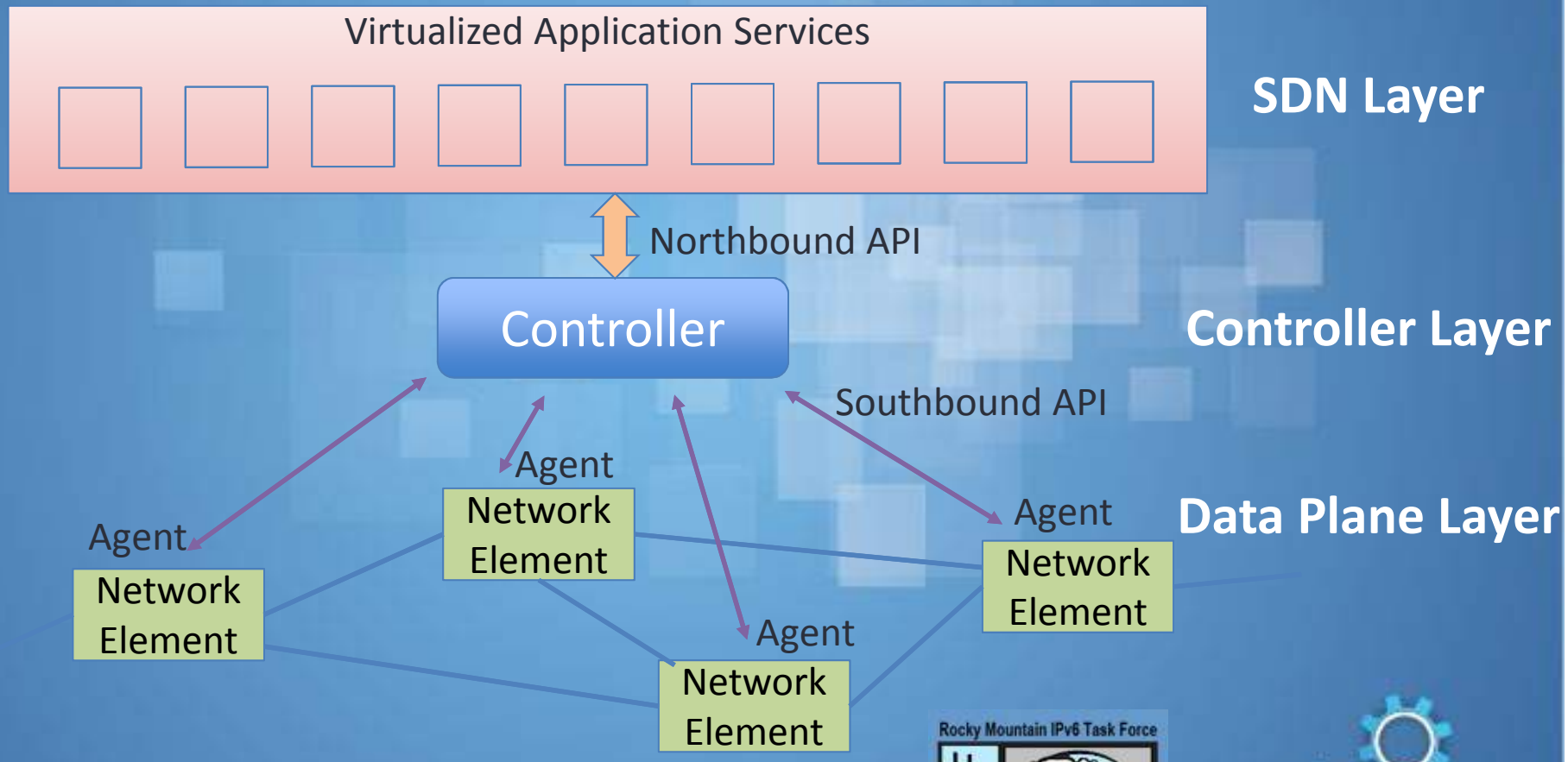
- Software-Defined Networking Background
- Introduction to OpenFlow
- OpenFlow History and Lineage
- IPv6 Capabilities within OpenFlow
- OpenFlow v1.3 demonstration using OpenDaylight & Mininet/OVS
- Q&A

Software-Defined Networking Background

- Software-Defined Networking is an approach to networking that separates the control plane from the forwarding plane to support virtualization.
- SDN is a new paradigm for network virtualization.

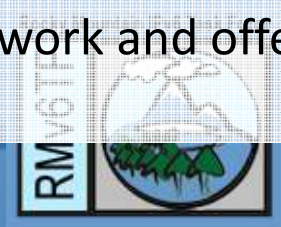


SDN High-Level Architecture



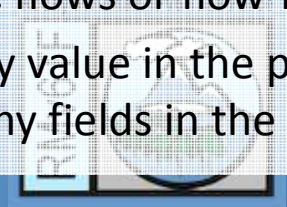
SDN Benefits

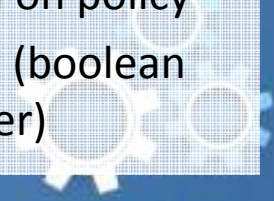
- SDN offers a new way to innovate use of a virtualized network (greater customization of networks).
- SDN provides for greater span of control and network analytics and response. A single controller can have global or network-wide view of the network.
- SDN provides better intelligence with a global view of the network rather than each network element looking at the network from its own viewpoint.
- SDN provides for improved application experience and empower the network owner/operator
- SDN offer simplified IT administration.
- SDN support rapid enhanced multi-tenant cloud services deployment.
- SDN offers an opportunity to open up the network and offer a diverse set of vendors.



SDN Use Cases

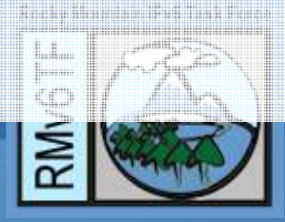
- SDN automates and speeds up deployment of creating a network to support a specific application.
- SDN is well suited to network services that are complex or time consuming to implement manually.
- SDN can ease deployment of functions that are difficult to provision or require manual configuration that are not dynamic.
  - Network Segmentation/Separation
  - Multi-tenant cloud data centers
  - Network monitoring - TAP aggregation
  - WAN traffic engineering (L2/L3 MPLS automation), optimal traffic engineering, bandwidth on demand
  - QoS, Policy-based-Routing, RSVP/TE, firewalls
  - Network programmability with dynamic traffic flows or flow-based on policy
  - Special forwarding of packets, matching on any value in the packet (boolean logic, match on l2/l3 source/dest addresses, any fields in the header)





Introduction to OpenFlow

- OpenFlow (OF) is a Software-Defined Networking (SDN) protocol used for southbound communications from an SDN controller to and from a network device.
- OpenFlow is the protocol used to inform the topology of network switches on which flows should be added to their flow tables and advise switches how they should handle traffic flows that are not in the current flow tables.
- OpenFlow uses TCP port 6633 (legacy) or 6653 (IANA assigned, OF 1.4+) & TLS



OpenFlow History and Lineage

- OpenFlow evolved from the CleanSlate Program at Stanford University around 2007/2008, re-imagining networking, starting over
  - http://cleanslate.stanford.edu/
- Open Networking Foundation (ONF) formed in 2011 to standardize OpenFlow and speed innovation and collaboration between member companies
  - https://www.opennetworking.org/
- OpenFlow Switch Specification 1.1.0 – February 2011
- Then research began on adding IPv6
- OpenFlow Switch Specification 1.2 – December 2011
- OpenFlow v1.2 Included definitions/match-fields for IPv6 flows

OpenFlow Version 1.0.0 Flow Entry

Match Fields	Priority	Counters	Timeouts	Cookie	Actions

- Ingress Port
- Header Fields
  - Ether Src
  - Ether dst
  - Ether type
  - VLAN Id
  - VLAN prio
  - IP Src
  - IP Dst
  - IP Proto
  - IP TOS (DSCP)
  - TP Src Port
  - TP Dst Port

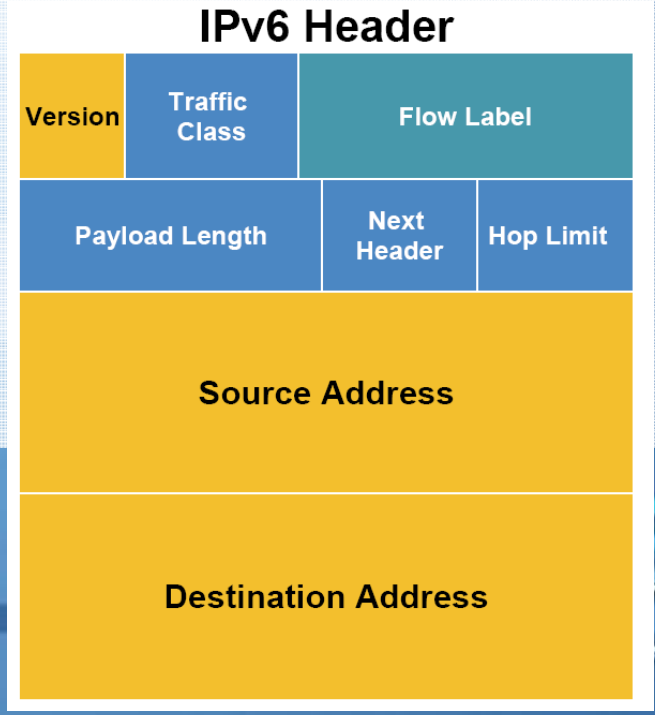
- Per Table
  - Active Entries
  - Packet Lookups
  - Packet Matches
- Per Flow
  - Received Packets
  - Received Bytes
  - Duration (Seconds)
  - Duration (Nano Secs)
- Per Port
  - Received Packets
  - Transmitted Packets
  - Received Bytes
  - Transmitted Bytes
  - Receive Drops
  - Transmit Drops
  - Receive Errors
  - Transmit Errors
  - Receive Frame Errors
  - Receive Overrun Errors
  - Receive CRC Errors
  - Collisions
- Per Queue
  - Transmit Packets
  - Transmit Bytes
  - Transmit Overrun Errors

- ALL
- CONTROLLER
- LOCAL
- TABLE
- IN_PORT
- NORMAL
- FLOOD
- Enqueue
- Drop
- Modify-Field
  - VLAN Id
  - VLAN Prio
  - Pop VLAN
  - Ether Src
  - Ether Dst
  - IP Src
  - IP Dst
  - IP Tos
  - TP Src Port
  - TP Dst Port

Source: https://wiki.opendaylight.org/images/d/dc/Openflow1.3_Support_for_Opendaylight.pdf

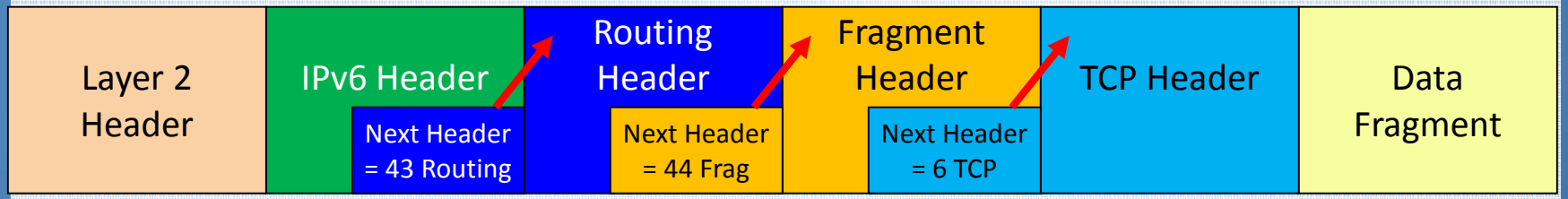
IPv6 Capabilities in OpenFlow v1.2

- Starting with OpenFlow version 1.2, IPv6 flow capabilities added
- IP protocol number (Ethernet type 0x86DD = IPv6)
- IPv6 source/destination address
- IPv6 traffic class
- IPv6 flow label
- ICMPv6 types/codes
- IPv6 unicast and multicast

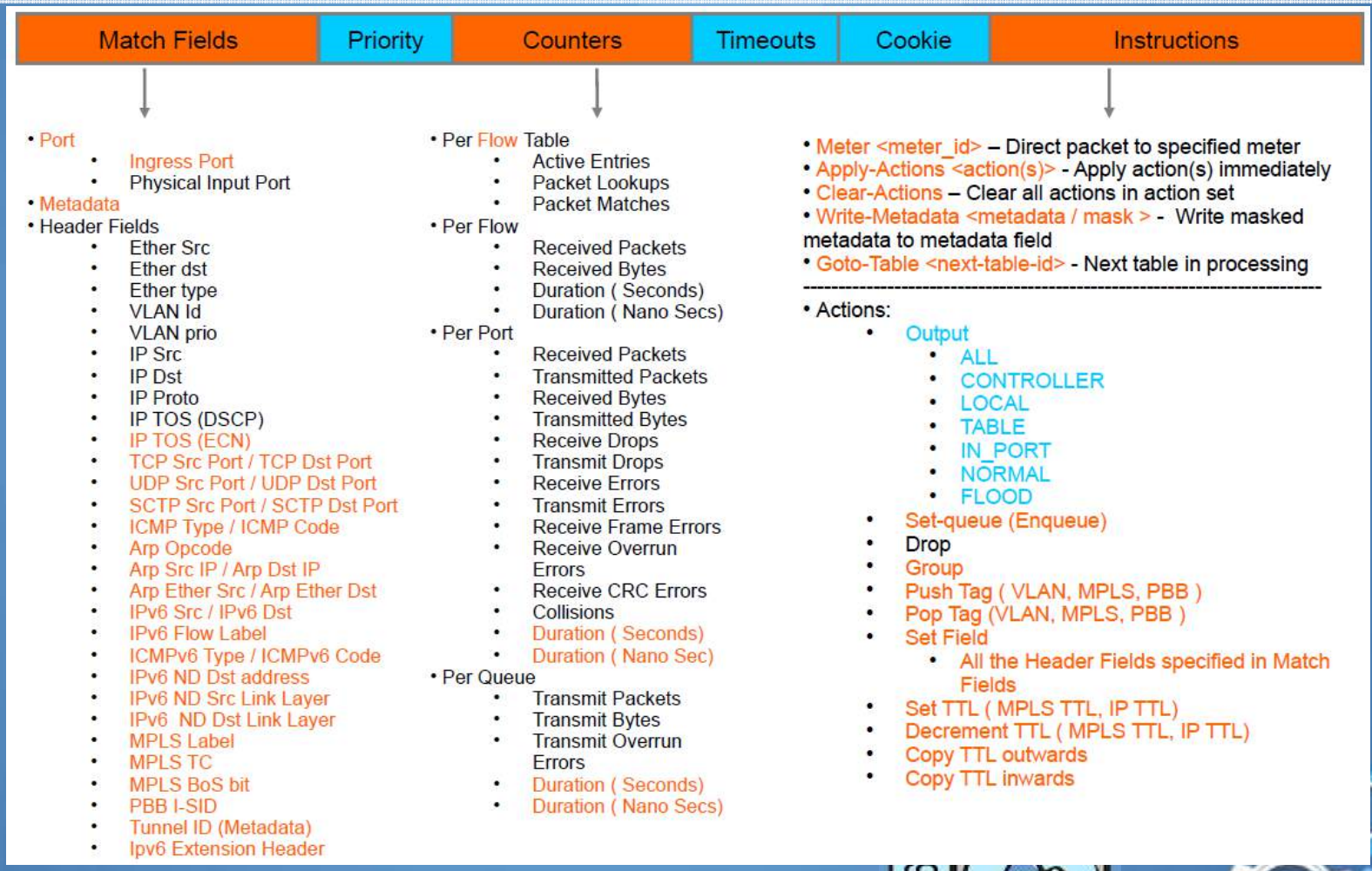


IPv6 Capabilities in OpenFlow v1.3

- OpenFlow version 1.3 added some new IPv6 features
- Ability to rewrite packet headers via Flexible Match Support
- Match on IPv6 header fields the same as OpenFlow v1.2
- Match on Next Header/Extension Header



OpenFlow Version 1.3.1 Flow Entry



Source: https://wiki.opendaylight.org/images/d/dc/Openflow1.3_Support_for_Opendaylight.pdf

OpenFlow Switch Specification v1.4.0 (Oct 14, 2013)

```
/* OXM Flow match field types for OpenFlow basic class. */
enum oxm_ofb_match_fields {
OFPXMT_OFB_IN_PORT = 0, /* Switch input port. */
OFPXMT
OFPXMT  OFPXMT_OFB_ETH_TYPE        - 5, /* Ethernet frame type. */
OFPXMT
OFPXMT  …
OFPXMT
OFPXMT  OFPXMT_OFB_IP_PROTO        - 10, /* IP protocol. */
OFPXMT
OFPXMT  …
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_SRC        - 26, /* IPv6 source address. */
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_DST        - 27, /* IPv6 destination address. */
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_FLABEL     - 28, /* IPv6 Flow Label */
OFPXMT
OFPXMT  OFPXMT_OFB_ICMPV6_TYPE     - 29, /* ICMPv6 type. */
OFPXMT
OFPXMT  OFPXMT_OFB_ICMPV6_CODE     - 30, /* ICMPv6 code. */
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_ND_TARGET  - 31, /* Target address for ND. */
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_ND_SLL     - 32, /* Source link-layer for ND. */
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_ND_TLL     - 33, /* Target link-layer for ND. */
OFPXMT
OFPXMT  …
OFPXMT
OFPXMT  OFPXMT_OFB_IPV6_EXTHDR     - 39, /* IPv6 Extension Header pseudo-field */
OFPXMT
OFPXMT_OFB_ICMPV6_CODE = 30, /* ICMPv6 code. */
OFPXMT_OFB_IPV6_ND_TARGET = 31, /* Target address for ND. */
OFPXMT_OFB_IPV6_ND_SLL = 32, /* Source link-layer for ND. */
OFPXMT_OFB_IPV6_ND_TLL = 33, /* Target link-layer for ND. */
OFPXMT_OFB_MPLS_LABEL = 34, /* MPLS label. */
OFPXMT_OFB_MPLS_TC = 35, /* MPLS TC. */
OFPXMT_OFP_MPLS_BOS = 36, /* MPLS BoS bit. */
OFPXMT_OFB_PBB_ISID = 37, /* PBB I-SID. */
OFPXMT_OFB_TUNNEL_ID = 38, /* Logical Port Metadata. */
OFPXMT_OFB_IPV6_EXTHDR = 39, /* IPv6 Extension Header pseudo-field */
OFPXMT_OFB_PBB_UCA = 41, /* PBB UCA header field. */
```

Source: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf

OpenFlow Switch Specification v1.4.0 (Oct 14, 2013)

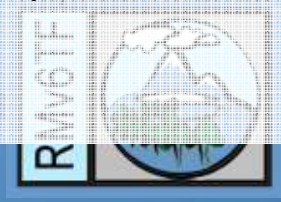
```
OXM_OF_IPV6_EXTHDR

/* Bit definitions for IPv6 Extension Header pseudo-field. */
enum ofp_ipv6exthdr_flags {
OFPIEH_NONEXT = 1 << 0,   /* "No next header" encountered. */
OFPIEH_ESP = 1 << 1,      /* Encrypted Sec Payload header present. */
OFPIEH_AUTH = 1 << 2,     /* Authentication header present. */
OFPIEH_DEST = 1 << 3,     /* 1 or 2 dest headers present. */
OFPIEH_FRAG = 1 << 4,     /* Fragment header present. */
OFPIEH_ROUTER = 1 << 5,   /* Router header present. */
OFPIEH_HOP = 1 << 6,      /* Hop-by-hop header present. */
OFPIEH_UNREP = 1 << 7,    /* Unexpected repeats encountered. */
OFPIEH_UNSEQ = 1 << 8,    /* Unexpected sequencing encountered. */
};
```

Source: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf

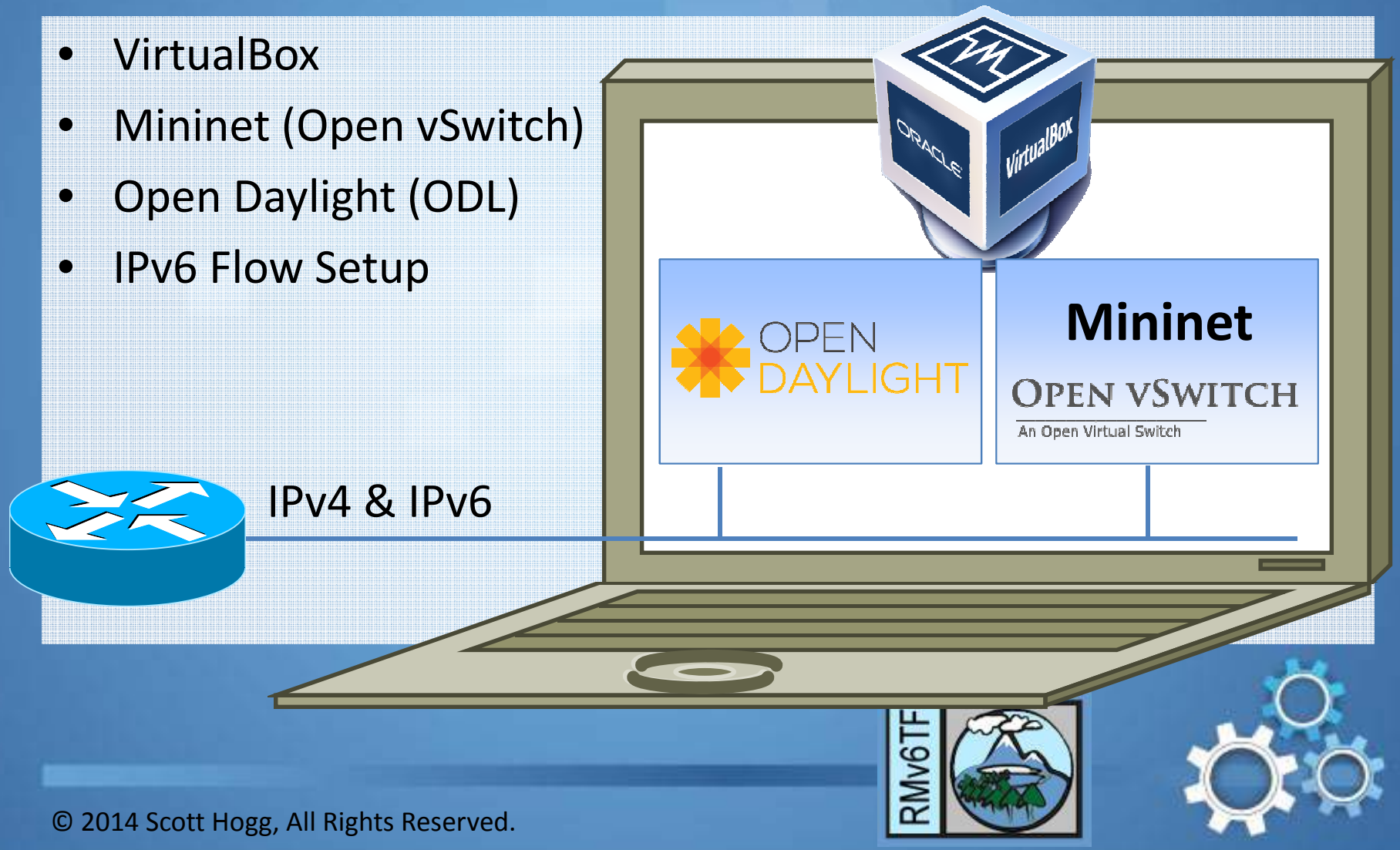
OpenFlow 1.3 Switches & Controllers

- Arista 7280SE, 7280SE-64
- Big Switch Switch Light, Floodlight-plus, Indigo Agent, Indigo Virtual Switch (IVS), LoxiGen
- Brocade MLXe, CER/CES, ICX, VDX, and VCS, NetIron XMR Series
- Centec V330 switch
- CPqD OpenFlow Switch
- HP Virtual Application Networks (VAN) SDN Controller, 50+ HP ProCurve switch models
- Infoblox FlowForwarding Erlang-Solutions LINC implementation
- NEC ProgrammableFlow PF6800 Controller, PF5240, PF5248, PF5820 switches
- NoviFlow switches (NoviSwitch)
- NOX & POX open source SDN controller
- OpenDaylight Hydrogen v1.0
- Pica8 SDN switches, P-3290, P-3295, P-3780, P3920 (OpenFlow v1.4 in PicOS 2.3+)
- Ryu (OpenFlow 1.0, 1.2, 1.3, & 1.4)
- Vello Systems VX1048 & VX3048



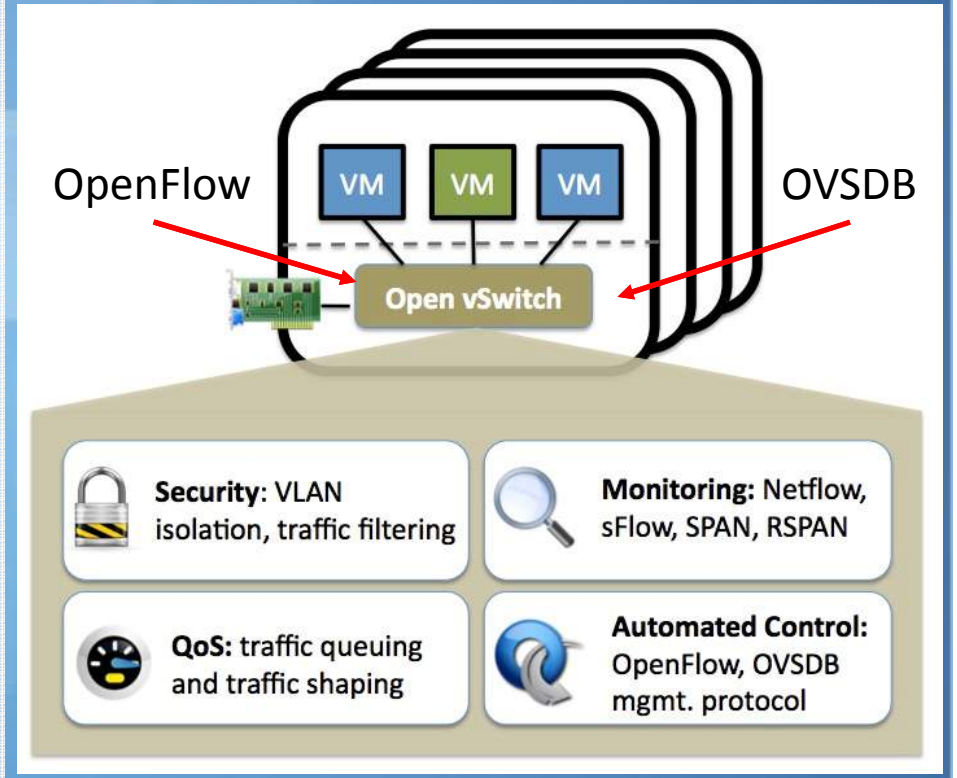
OpenFlow and IPv6 Demonstration

- VirtualBox
- Mininet (Open vSwitch)
- Open Daylight (ODL)
- IPv6 Flow Setup



OpenFlow and IPv6 Demonstration – Open vSwitch

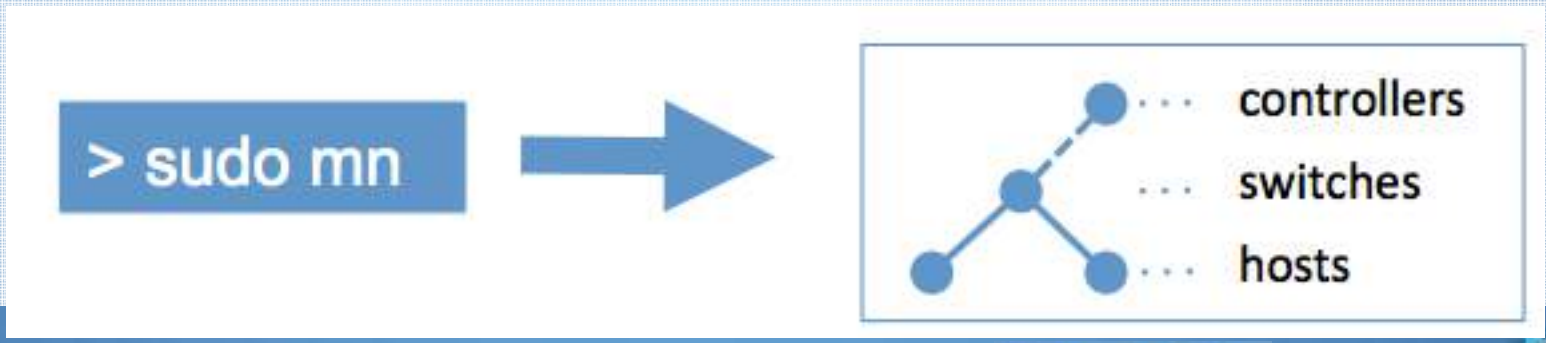
- Open vSwitch (OVS) is a multilayer virtual switch implemented at the hypervisor layer that allows for programmatic extensions (e.g. OpenFlow and Open vSwitch Database (OVSDB) with JSON) (RFC 7047)
- IPv6 support in Sept 2011
- http://openvswitch.org/



Rocky Mountain IPv6 Task Force
RMv6TF

OpenFlow and IPv6 Demonstration - Mininet

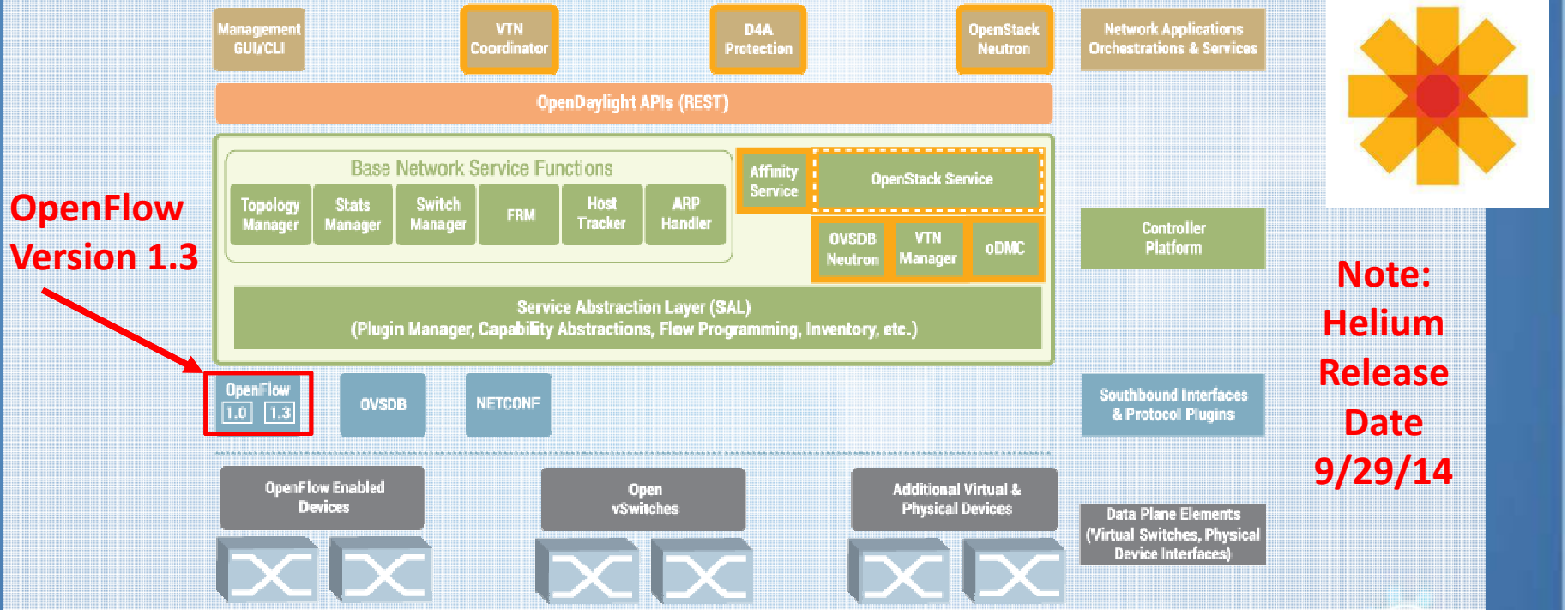
- Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine (VM, cloud or native), in seconds, with a single command
- Mininet version 2.1.0 uses OSV version 2.0.1 which supports OpenFlow 1.3 & OVSDB (RFC 7158/7159)
- http://mininet.org/



Rocky Mountain IPv6 Task Force
RMv6TF

OpenFlow and IPv6 Demonstration - ODL

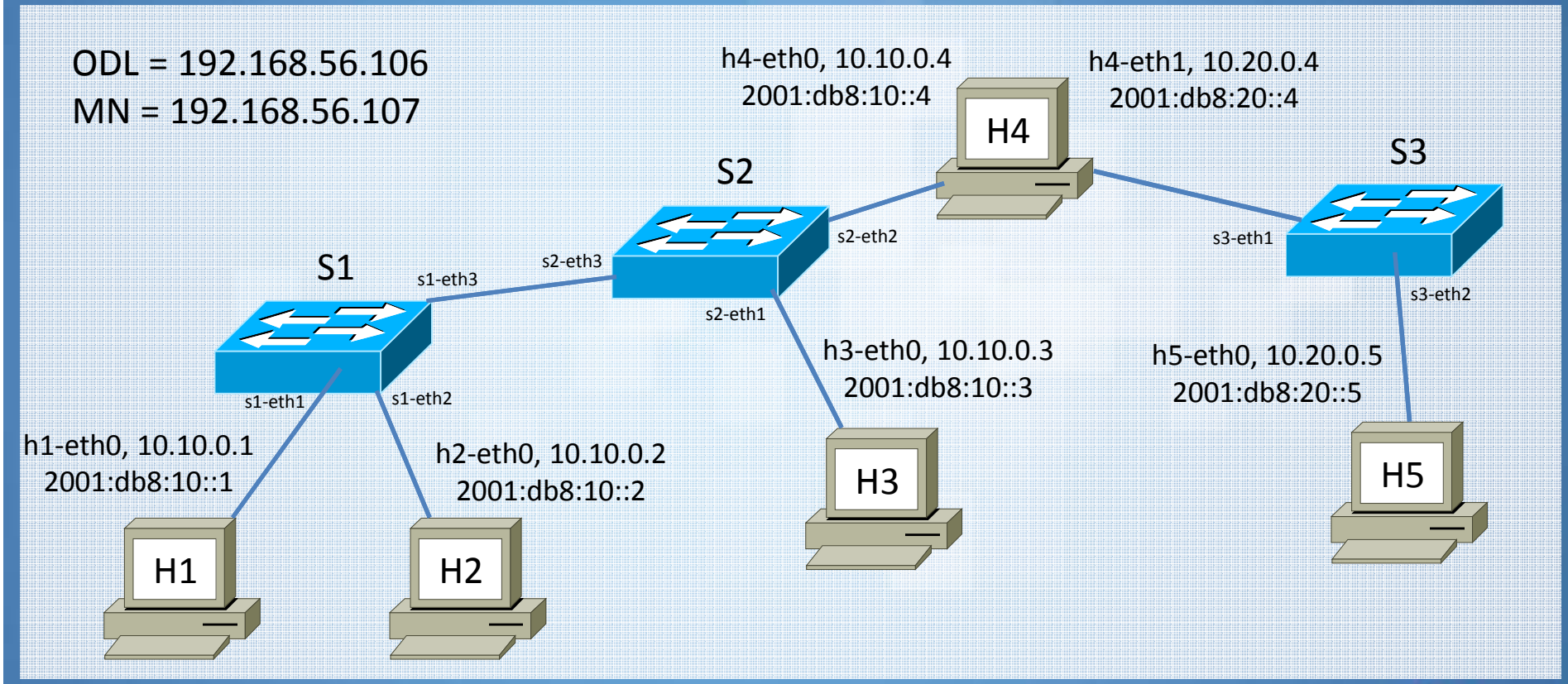
- Open Daylight Hydrogen Release, http://opendaylight.org



OpenFlow Version 1.3

Note: Helium Release Date 9/29/14

OpenFlow and IPv6 Demonstration – Mininet Topology



OpenFlow and IPv6 Demonstration

- Start up Mininet VM
- Start up OpenDaylight VM
- Start OpenDaylight Controller
- Start Wireshark
- Start Mininet topology
- IPv6 Flow Setup
- Test IPv6 connectivity
- Observe OpenFlow packets and flows in table

Thank you for your time!

Scott Hogg, @scotthogg

Scott { at } RMv6TF.org

SHogg { at } GTRI.com