Introduction to OpenFlow

Raj Jain

Washington University in Saint Louis

Saint Louis, MO 63130

Jain@cse.wustl.edu

These slides and audio/video recordings of this class lecture are at:

http://www.cse.wustl.edu/~jain/cse570-15/

Overview

1. Planes of Networking
2. OpenFlow
3. OpenFlow Operation
4. OpenFlow Switches including Open vSwitch
5. OpenFlow Evolution
6. Current Limitations and Issues

Note: This is the first module of four modules on OpenFlow, OpenFlow Controllers, SDN and NFV in this course.

Planes of Networking

❑ **Data Plane**: All activities involving as well as resulting from data packets sent by the end user, e.g.,

  ➢ Forwarding

  ➢ Fragmentation and reassembly

  ➢ Replication for multicasting

❑ **Control Plane**: All activities that are <u>necessary</u> to perform data plane activities but do not involve end-user data packets

  ➢ Making routing tables

  ➢ Setting packet handling policies (e.g., security)

  ➢ Base station beacons announcing availability of services

Planes of Networking (Cont)

❑ **Management Plane**: All activities related to provisioning and monitoring of the networks

  ➢ Fault, Configuration, Accounting, Performance and Security (**FCAPS**).

  ➢ Instantiate new devices and protocols (Turn devices on/off)

  ➢ Optional ⟹ May be handled manually for small networks.

❑ **Services Plane**: Middlebox services to improve performance or security, e.g.,

  ➢ Load Balancers, Proxy Service, Intrusion Detection, Firewalls, SSL Off-loaders

  ➢ Optional ⟹ Not required for small networks

Data vs. Control Logic

❑ Data plane runs at line rate,
e.g., 100 Gbps for 100 Gbps Ethernet ⟹ Fast Path
⟹ Typically implemented using special hardware,
e.g., Ternary Content Addressable Memories (TCAMs)

❑ Some exceptional data plane activities are handled by the CPU
in the switch ⟹ Slow path
e.g., Broadcast, Unknown, and Multicast (BUM) traffic

❑ All control activities are generally handled by CPU
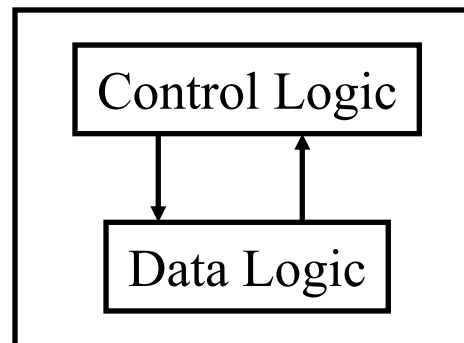
OpenFlow: Key Ideas

1. Separation of control and data planes
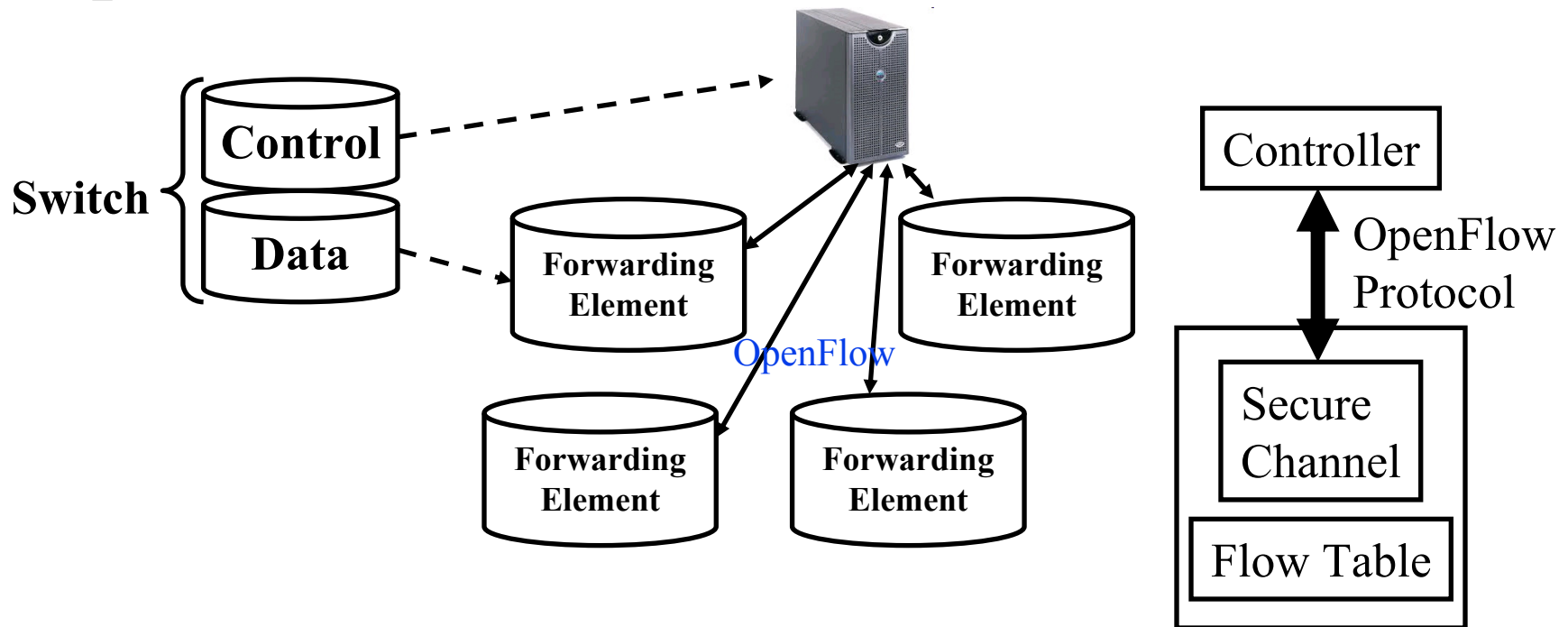2. Centralization of control
3. Flow based control

Ref: N. McKeown, et al., ``OpenFlow: Enabling Innovation in Campus Networks,'' ACM SIGCOMM CCR, Vol. 38, No. 2, April 2008, pp. 69-74.

http://www.cse.wustl.edu/~jain/cse570-15/

Separation of Control and Data Plane

- ❑ Control logic is moved to a controller
- ❑ Switches only have forwarding elements
- ❑ One expensive controller with a lot of cheap switches
- ❑ OpenFlow is the protocol to send/receive forwarding rules from controller to switches

OpenFlow V1.0

❑ On packet arrival, match the header fields with flow entries in a table, if any entry matches, update the counters indicated in that entry and perform indicated actions

Flow Table:

Header Fields	Counters	Actions
Header Fields	Counters	Actions
…	…	…
Header Fields	Counters	Actions

Ingress Port	Ether Source	Ether Dest	VLAN ID	VLAN Priority	IP Src	IP Dst	IP Proto	IP ToS	Src L4 Port	Dst L4 Port

Washington University in St. Louis   http://www.cse.wustl.edu/~jain/cse570-15/   ©2015 Raj Jain

Flow Table Example

Port	Src MAC	Dst MAC	VLAN ID	Priority	EtherType	Src IP	Dst IP	IP Proto	IP ToS	Src L4 Port ICMP Type	Dst L4 Port ICMP Code	Action	Counter
*	*	0A:C8:*	*	*	*	*	*	*	*	*	*	Port 1	102
*	*	*	*	*	*	*	192.168.*.*	*	*	*	*	Port 2	202
*	*	*	*	*	*	*	*	*	*	21	21	Drop	420
*	*	*	*	*	*	*	*	0x806	*	*	*	Local	444
*	*	*	*	*	*	*	*	0x1*	*	*	*	Controller	1

❑ Idle timeout: Remove entry if no packets received for this time

❑ Hard timeout: Remove entry after this time

❑ If both are set, the entry is removed if either one expires.

http://www.cse.wustl.edu/~jain/cse570-15/

Matching

Set Input Port
Ether Src
Ether Dst
Ether Type
Set all others to zero

EtherType =0x8100? — Y → Set VLAN ID / Set VLAN Priority / Use EtherType in VLAN tag for next EtherType Check

N

EtherType =0x0806? — Y → Set IP Src, IP Dst / IP Proto, IP ToS / from within ARP

N

EtherType =0x0800? — Y → Set IP Src, IP Dst / IP Proto, IP ToS

N

Not IP Fragment? — Y → IP Proto =6 or 7 — Y → Set Src Port, Dst Port for L4 fields

N (Not IP Fragment) → Packet lookup using assigned header fields

N (IP Proto =6 or 7) → IP Proto =1? — Y → Use ICMP Type and code for L4 Fields

N (IP Proto =1?) →

Packet lookup using assigned header fields

Match Table 0? — Y → Apply Actions

N

Match Table n? — Y → Apply Actions

N → Send to Controller

Counters

Per Table	Per Flow	Per Port	Per Queue
Active Entries	Received Packets	Received Packets	Transmit Packets
Packet Lookups	Received Bytes	Transmitted Packets	Transmit Bytes
Packet Matches	Duration (Secs)	Received Bytes	Transmit overrun errors
	Duration (nanosecs)	Transmitted Bytes	
		Receive Drops	
		Transmit Drops	
		Receive Errors	
		Transmit Errors	
		Receive Frame Alignment Errors	
		Receive Overrun erorrs	
		Receive CRC Errors	
		Collisions	

Actions

❑ Forward to Physical Port *i* or to *Virtual Port*:

    ➢ **All**: to all interfaces <u>except</u> incoming interface

    ➢ **Controller**: encapsulate and send to controller

    ➢ **Local**: send to its local networking stack

    ➢ **Table**: Perform actions in the flow table

    ➢ **In_port**: Send back to input port

    ➢ **Normal**: Forward using traditional Ethernet

    ➢ **Flood**: Send along minimum spanning tree <u>except</u> the incoming interface

❑ Enqueue: To a particular queue in the port $\Rightarrow$ QoS

❑ Drop

❑ Modify Field: E.g., add/remove VLAN tags, ToS bits, Change TTL

Actions (Cont)

- Masking allows matching only selected fields,
  e.g., Dest. IP, Dest. MAC, etc.

- If header matches an entry, corresponding actions are performed and counters are updated

- If no header match, the packet is queued and
  the header is sent to the controller, which sends a new rule.
  Subsequent packets of the flow are handled by this rule.

- Secure Channel: Between controller and the switch using TLS

- Modern switches already implement flow tables, typically using Ternary Content Addressable Memories (TCAMs)

- Controller can change the forwarding rules if a client moves
  $\Rightarrow$ Packets for mobile clients are forwarded correctly

- Controller can send flow table entries beforehand (**Proactive**) or Send on demand (**Reactive**). OpenFlow allows both models.

Open vSwitch

- Open Source Virtual Switch

- Nicira Concept

- Can Run as a stand alone hypervisor switch or as a distributed switch across multiple physical servers

- Default switch in XenServer 6.0, Xen Cloud Platform and supports Proxmox VE, VirtualBox, Xen KVM

- Integrated into many cloud management systems including OpenStack, openQRM, OpenNebula, and oVirt

- Distributed with Ubuntu, Debian, Fedora Linux. Also FreeBSD

- Intel has an accelerated version of Open vSwitch in its own Data Plane Development Kit (DPDK)

Open vSwitch Features

❑ Inter-VM communication monitoring via:

  ➢ **NetFlow**: Cisco protocol for sampling and collecting traffic statistics (RFC 3954)

  ➢ **sFlow**: Similar to NetFlow by sflow.org (RFC 3176)

  ➢ **Jflow**: Juniper's version of NetFlow

  ➢ **NetStream**: Huawei's version of NetFlow

  ➢ **IPFIX**: IP Flow Information Export Protocol (RFC 7011) - IETF standard for NetFlow

  ➢ SPAN, RSPAN: Remote Switch Port Analyzer – port mirroring by sending a copy of all packets to a monitor port

  ➢ **GRE-tunneled mirrors**: Monitoring device is remotely connected to the switch via a GRE tunnel

Open vSwitch Features (Cont)

- Link Aggregation Control Protocol (LACP)
- IEEE 802.1Q VLAN
- IEEE 802.1ag Connectivity Fault Management (CFM)
- Bidirectional Forwarding Detection (BFD) to detect link faults (RFC 5880)
- IEEE 802.1D-1998 Spanning Tree Protocol (STP)
- Per-VM traffic policing
- OpenFlow
- Multi-table forwarding pipeline
- IPv6
- GRE, VXLAN, IPSec tunneling
- Kernel and user-space forwarding engine options

OVSDB
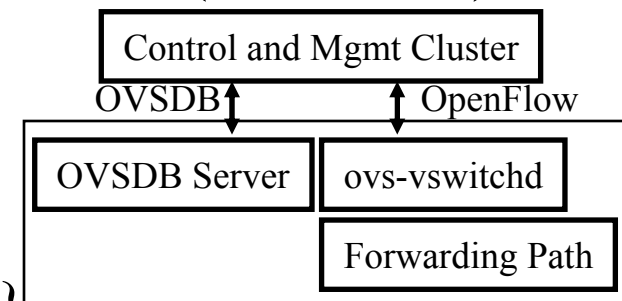
❑ Open vSwitch Database Management Protocol (OVSDB)

❑ Monitoring capability using publish-subscribe mechanisms

❑ Stores both provisioning and operational state

❑ Java Script Object Notation (JSON) used for schema format and for JSON-RPC over TCP for wire protocol (RFC 4627)

<database-schema>

"name": <id>

"version": <version>

"tables": {<id>: <table-schema>,…}

Control and Mgmt Cluster	
OVSDB↕	↕ OpenFlow
OVSDB Server	ovs-vswitchd
	Forwarding Path

❑ RPC Methods: List databases, Get Schema, Update, Lock, …

❑ Open vSwitch project includes open source OVSDB client and server implementations

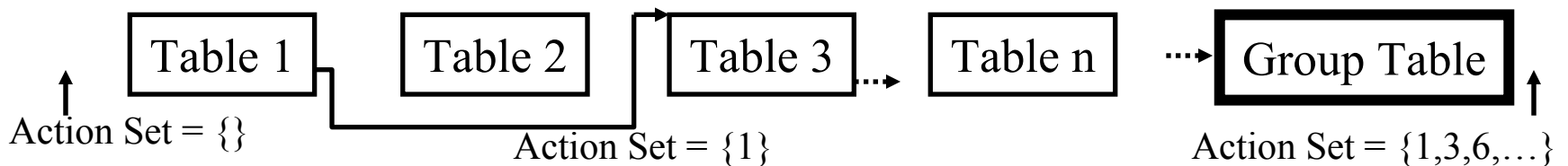Ref: B. Pfaff and B. Davie, "The Open vSwitch Database Management Protocol," IETF draft, Oct 2013, http://tools.ietf.org/html/draft-pfaff-ovsdb-proto-04
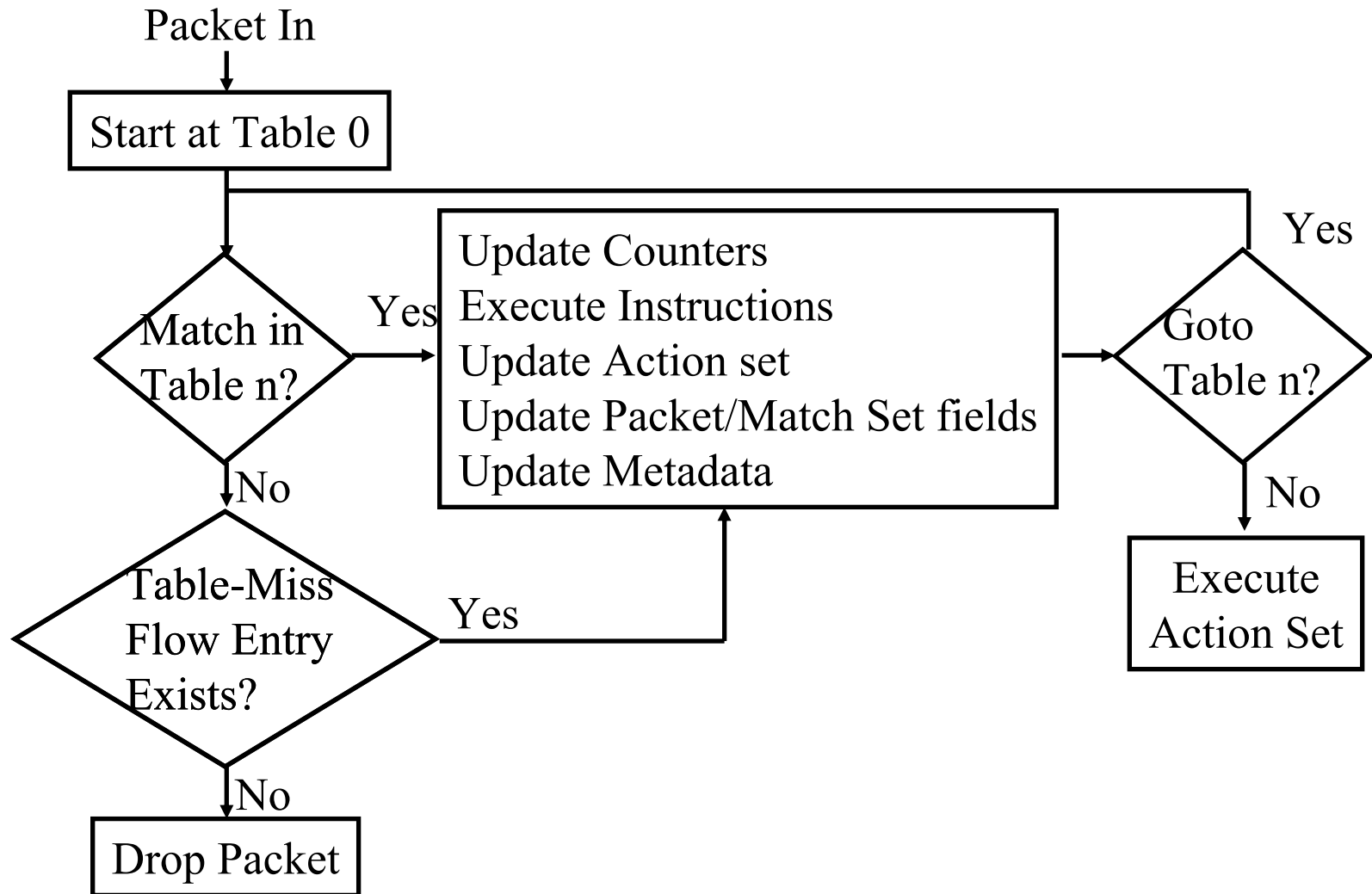
OpenFlow V1.1

❑ V1: Perform action on a match. Ethernet/IP only. Single Path Did not cover MPLS, Q-in-Q, ECMP, and efficient Multicast

❑ V1.1 Introduced *Table chaining, Group Tables,* and added *MPLS Label* and *MPLS traffic class* to match fields.

❑ **Table Chaining**: On a match, instruction may be

> Immediate actions: modify packet, update match fields and/or

> Update action set, and/or

> Send match data and action set to *Table n*,

> Go to *Group Table* entry *n*

Controller

OpenFlow

| Secure Channel | Group Table |
| Flow Table | Flow Table |

Table 1 → Table 2 → Table 3 ⋯→ Table n ⋯→ **Group Table**

Action Set = {}

Action Set = {1}

Action Set = {1,3,6,…}

OpenFlow V1.1 (Cont)

Packet In

Start at Table 0

Match in Table n? — Yes → Update Counters / Execute Instructions / Update Action set / Update Packet/Match Set fields / Update Metadata → Goto Table n?

Goto Table n? — Yes (to Start at Table 0) / No → Execute Action Set

Match in Table n? — No → Table-Miss Flow Entry Exists?

Table-Miss Flow Entry Exists? — Yes → (Update Counters box) / No → Drop Packet

OpenFlow V1.1 (Cont)

❑ On a miss, the instruction may be to send packet to controller or continue processing with the sequentially next table

❑ Group Tables: each entry has a variable number of buckets

  ➢ **All**: Execute each bucket. Used for Broadcast, Multicast.

  ➢ **Select**: Execute one *switch selected* bucket. Used for port mirroring. Selection may be done by hashing some fields.

  ➢ **Indirect**: Execute one *predefined* bucket.

  ➢ **Fast Failover**: Execute the first live bucket ⇒ Live port

❑ New Features supported:

  ➢ **Multipath**: A flow can be sent over one of several paths

  ➢ **MPLS**: multiple labels, traffic class, TTL, push/pop labels

  ➢ **Q-in-Q**: Multiple VLAN tags, push/pop VLAN headers

  ➢ **Tunnels**: via virtual ports

Ref: http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf

OpenFlow V1.2

1. **IPv6 Support**: Matching fields include IPv6 source address, destination address, protocol number, traffic class. ICMPv6 type, ICMPv6 code, IPv6 neighbor discovery header fields, and IPv6 flow labels.

2. **Extensible Matches**: Type-Length-Value (TLV) structure. Previously the order and length of match fields was fixed.

3. **Experimenter extensions** through dedicated fields and code points assigned by ONF

OpenFlow 1.3

- **IPv6 extension headers**: Can check if Hop-by-hop, Router, Fragmentation, Destination options, Authentication, Encrypted Security Payload (ESP), unknown extension headers are present
- **MPLS Bottom-of-Stack bit** matching
- **MAC-in-MAC** encapsulation
- **Tunnel ID meta data**: Support for tunnels (VxLAN, …)
- **Per-Connection Event Filtering**: Better filtering of connections to multiple controllers
- Many **auxiliary connections** to the controller allow to exploit parallelism
- Better **capability negotiation**: Requests can span multiple messages
- More general **experimenter capabilities** allowed
- A separate flow entry for **table miss actions**

Ref: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf
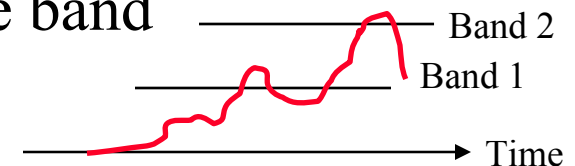
OpenFlow V1.3 (Cont)

❑ **Cookies**: A cookie field with policy identifier is added to messages containing new packets sent to the controller. This helps controller process the messages faster than if it had to search its entire database.

❑ **Duration**: Duration field has been added to most stats. Helps compute rates.

❑ Per-flow counters can be disabled to improve performance

❑ Per Flow Meters and meter bands

❑ **Meter**: Switch element that can measure and control the rate of packets/bytes.

  ➢ **Meter Band**: If the packet/byte rate exceeds a pre-defined threshold ⇒ the meter has triggered the band
  ➢ A meter may have multiple bands

Band 2
Band 1
Time

http://www.cse.wustl.edu/~jain/cse570-15/

OpenFlow V1.3 (Cont)

➢ If on triggering a band the meter drops the packet, it is called rate limiter.

➢ Other QoS and policing mechanisms can be designed using these meters

➢ **Per-Flow QoS**: Meters are attached to a flow entry not to a queue or a port.

➢ Multiple flow entries can all point to the same meter.

Match Fields	Priority	Counters	Instructions	Timeouts	Timeouts	Cookie

New Instruction: Meter Meter_ID

Meter ID	Meter Bands	Counters

Band Type	Rate	Counters	Type Specific Arguments

1. Drop
2. Remark DSCP

kb/s
Burst

OpenFlow V1.4

- **Optical ports**: Configure and monitor transmit and receive frequencies of lasers and their power

- **Improved Extensibility**: Type-Length-Value (TLV) encodings at most places $\Rightarrow$ Easy to add new features in future

- **Extended Experimenter Extension API**: Can easily add ports, tables, queues, instructions, actions, etc.

- More information when a packet is sent to controller, e.g., no match, invalid TTL, matching group bucket, matching action, ..

- Controllers can select a subset of flow tables for monitoring

- Switches can **evict** entries of lower importance if table full

- Switches can notify controller if table is getting full

- Atomic execution of a **bundle** of instructions

http://www.cse.wustl.edu/~jain/cse570-15/

OpenFlow V1.4.1

1. **Bundle**: Atomic Instruction Group

   ➢ A group of instructions from the controller that are either all executed or all not executed

   ➢ A bundle may be sent to many switches and then applied at approximately same time on commit request from the controller

2. **Flow Table Monitoring**: Synchronization in a multi-controller system

   ➢ Notify a controller if a set of flow table entries is modified by another controller

3. Bug fixes

http://www.cse.wustl.edu/~jain/cse570-15/

OpenFlow V1.5

1. **Egress Tables**: actions to be done when exiting through a port (encapsulate or decapsulate a packet, tunnels)

2. **Packet Type**: Can now handle non-Ethernet packets, e.g., IP packets

3. **TCP Flags Matching**: Syn, Ack, and Fin may be used to detect beginning and end of a TCP connection

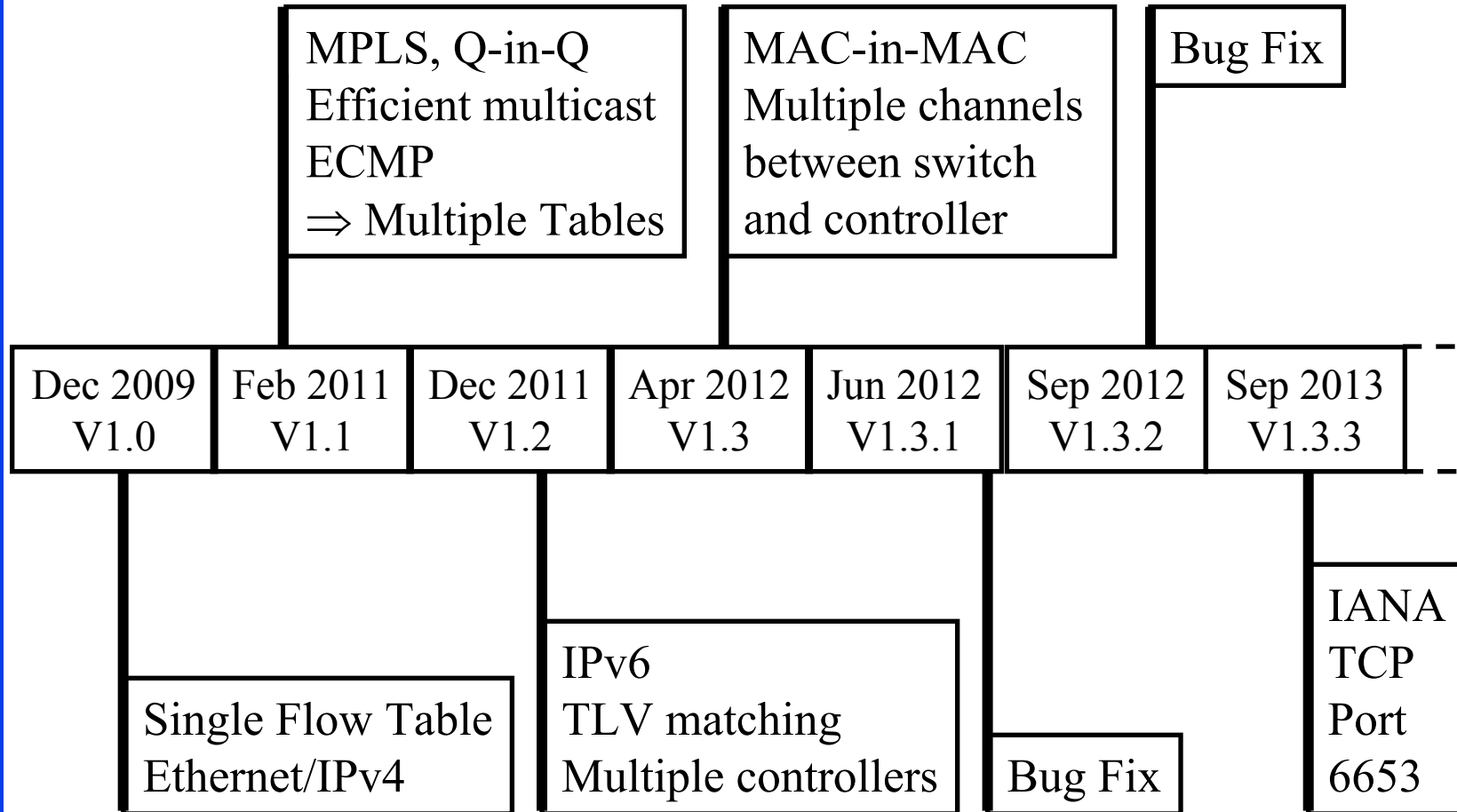OpenFlow V1.5.1: Bug Fixes, March 2015

Washington University in St. Louis
http://www.cse.wustl.edu/~jain/cse570-15/
©2015 Raj Jain

OpenFlow Evolution Summary

| | MPLS, Q-in-Q
Efficient multicast
ECMP
$\Rightarrow$ Multiple Tables | | MAC-in-MAC
Multiple channels
between switch
and controller | | Bug Fix | |

| Dec 2009
V1.0 | Feb 2011
V1.1 | Dec 2011
V1.2 | Apr 2012
V1.3 | Jun 2012
V1.3.1 | Sep 2012
V1.3.2 | Sep 2013
V1.3.3 |

| Single Flow Table
Ethernet/IPv4 | | IPv6
TLV matching
Multiple controllers | Bug Fix | IANA
TCP
Port
6653 |

OpenFlow Evolution Summary (Cont)

Minor Changes	OTN Experimenters Bundles Table full	Egress Tables Non-Ethernet Packets TCP Flags Matching	

Mar 2014 V1.3.4	Mar 2015 V1.3.5	Oct 2013 V1.4	Mar 2015 V1.4.1	Dec 2014 V1.5.0	Mar 2015 V1.5.1

Minor Changes

Instruction Bundles
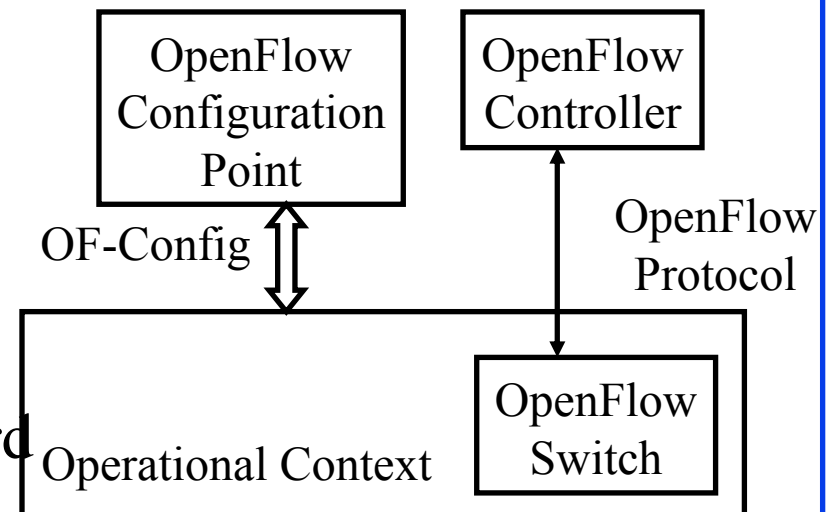Flow Table Monitoring
Multiple controllers

Bug Fix

Bootstrapping

❑ Switches require initial configuration: Switch IP address, Controller IP address, Default gateway

❑ Switches connect to the controller

❑ Switch provides configuration information about ports

❑ Controller installs a rule to forward LLDP packets to controller and then sends, one by one, LLDP packets to be sent out to port i (i=1, 2, …, n) which are forwarded to respective neighbors. The neighbors send the packets back to controller.

❑ Controller determines the topology from LLDP packets

❑ LLDP is a one-way protocol to advertise the capabilities at fixed intervals.

Ref: S. Sharma, et al., "Automatic Bootstrapping of OpenFlow Networks," 19th IEEE Workshop on LANMAN, 2013, pp. 1-6, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6528283 (Available to subscribers only)

OpenFlow Configuration Protocol (OF-Config)

❑ **OpenFlow Control Point**: Entity that configures OpenFlow switches

❑ **OF-Config**: Protocol used for configuration and management of OpenFlow Switches.
Assignment of OF controllers so that switches can initiate connections to them:

> ➢ IP address of controller
> ➢ Port number at the controller
> ➢ Transport protocol: TLS or TCP
> ➢ Configuration of queues (min/max rates) and ports
> ➢ Enable/disable receive/forward speed, media on ports

OpenFlow Configuration Point	OpenFlow Controller

OF-Config ↕   OpenFlow Protocol ↕

Operational Context   OpenFlow Switch

Ref: Cisco, "An Introduction to OpenFlow," Feb 2013,
http://www.cisco.com/web/solutions/trends/open_network_environment/docs/cisco_one_webcastan_introduction_to_openflowfebruary142013.pdf
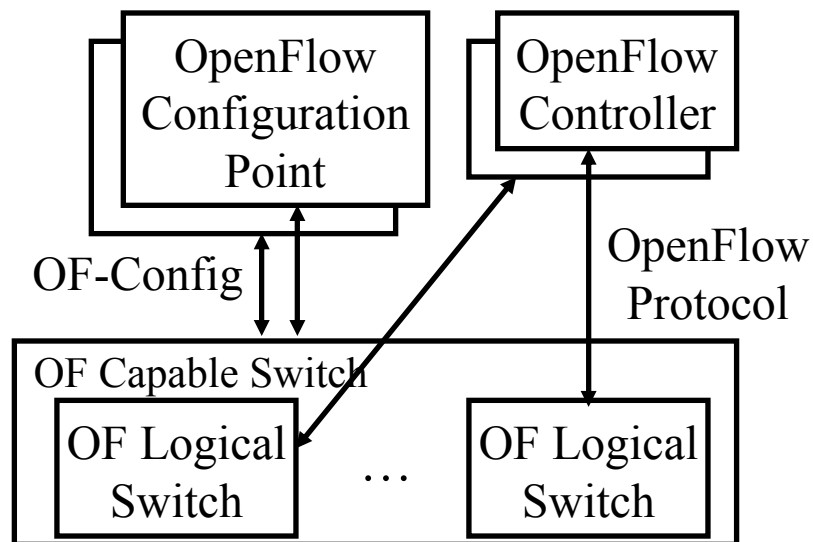
OF-Config (Cont)

❑ A physical switch = one or more logical switches each controlled by an OF Controller

❑ OF-Config allows configuration of logical switches.

Ref: ONF, "OpenFlow Management and Configuration Protocol (OF-Config 1.1.1)," March 23, 2013,
https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1-1-1.pdf

OF-Config Concepts

- **OF Capable Switch**: Physical OF switch. Can contain one or more OF logical switches.

- **OpenFlow Configuration Point**: configuration service

- **OF Controller**: Controls logical switch via OF protocol

- **Operational Context**: OF logical switch

- **OF Queue**: Queues of packets waiting for forwarding

- **OF Port**: forwarding interface. May be physical or logical.

- **OF Resource**: ports, queues, certificates, flow tables and other resources of OF capable switches assigned to a logical switch

- **Datapath ID**: 64-ID of the switch. Lower 48-bit = Switch MAC address, Upper 16-bit assigned by the operator

OF-Config Evolution

- ❑ V1.0 (Jan 2012): Based on OpenFlow V1.2
    - ➢ Assign controllers to logical switches
    - ➢ Retrieve logical switch configurations
    - ➢ Configure ports and queues
- ❑ V1.1 (May 2012): Based on OpenFlow V1.3
    - ➢ Configuration of certificates
    - ➢ Capability Discovery: Retrieve logical switch capabilities
    - ➢ Configure logical tunnels (VXLAN, NVGRE, …)
- ❑ V1.1.1 (Jan 2013): Bug Fix. Versioning support
- ❑ V1.2 (2014):
    - ➢ OF-Config version negotiation
    - ➢ Assigning resources to logical switches

Ref: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config1dot0-final.pdf
https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.1.pdf
https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1-1-1.pdf
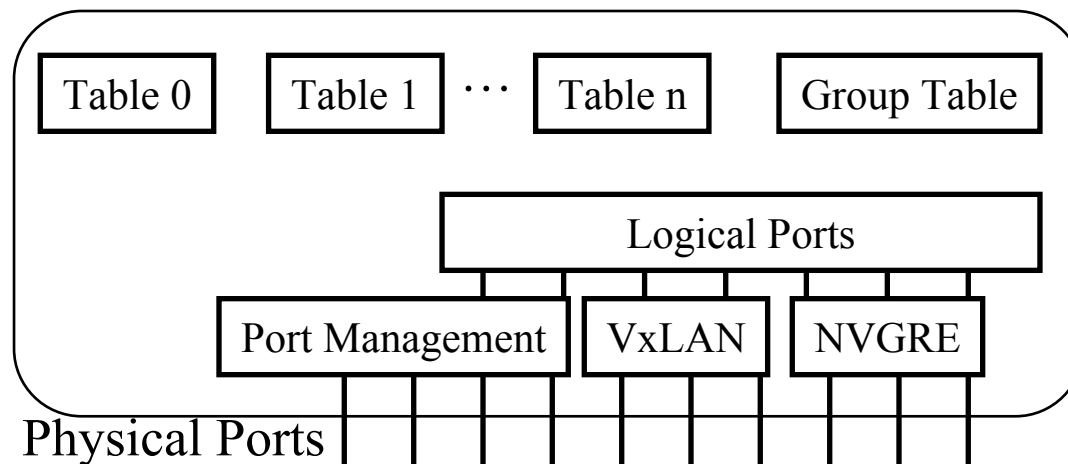http://www.cse.wustl.edu/~jain/cse570-15/

OpenFlow Notification Framework

❑ **Notification**: Event triggered messages, e.g., link down

❑ **Publish/subscribe model**: Switch = publisher. OpenFlow controller or OpenFlow config points, and others can subscribe. They will be notified about the events they subscribe.

❑ Use **ITU-T M.3702** Notifications: Attribute value change, Communication alarm, Environmental alarm, Equipment alarm, QoS alarm, Processing error alarm, Security alarm, State change, Object creation and deletion

❑ **Pre-existing Notifications**: Do not fit in the framework but will be recognized.

  ➢ OpenFlow: Packet-in, Flow removed, Port Status, Error, Hello, Echo request, Echo reply, Experimenter

  ➢ OpenFlow Config: OpenFlow logical switch instantiation, OpenFlow capability switch capability change, Successful OpenFLow session establishment, Failed OpenFlow session establishment, Port failure or recovery

Implementation Issues

❑ 40+ matching fields in a flow

❑ Multiple tables, each with a large number of flow entries

❑ Instructions and actions for each table

❑ Need VXLAN, NVGRE, etc. support

❑ For a large network, flow level programming can take a long time

Table 0	Table 1	...	Table n	Group Table

Logical Ports

Port Management	VxLAN	NVGRE

Physical Ports

Ref: R. Oshana and S. Addepalli, "Networking Trends- Software Defined Networking, Network Virtualization and Cloud Orchestration," Asia Power Arch. Conf, Oct 2012, https://www.power.org/wp-content/uploads/2012/10/13.-FSL-SDN-Openflow-and-Cloud-computing-UPD_Rob-Oshana.pdf

OpenFlow: Future Work Items

- Each controller has its own way to program. Need a common standard "Northbound API" (ONF NBI group)

- No standard API for communication between controllers of overlapping domain $\Rightarrow$ Need an East-West API

- Ability to continue operation when the controller is down

- Flow $\Rightarrow$ Decide once, use many times $\Rightarrow$ Performance

  - But does not help non-flow based request/response apps

- Need API to encrypt data plane packets, to inject packets, to instantiate a service, such as a firewall, IDS, on the switch

- Need to program an abstract view, e.g., source to destination, without knowing the physical network

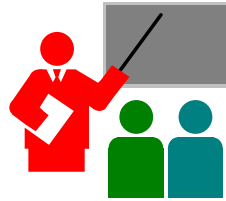Ref: http://onrc.stanford.edu/research_modern_sdn_stack.html

Ref: T. Nadeau and K. Gray, "SDN," O'Reilly, 2013, 384 pp, ISBN:978-1-449-34230-2

Summary

1. Four planes of Networking: Data, Control, Management, Service

2. OpenFlow separates control plane and moves it to a central controller $\Rightarrow$ Simplifies the forwarding element

3. Switches match incoming packets with flow entries in a table and handle it as instructed. The controller supplies the flow tables and other instructions.

4. Many hardware and software based switches including Open vSwitch

5. OpenFlow has been extended to IPv4, MPLS, IPv6, and Optical Network. But more work ahead.

Reading List

- S. Azodolmolky, "Software Defined Networking with OpenFlow," Packt Publishing, October 2013, 152 pp., ISBN:978-1-84969-872-6 (Safari Book)

- T. Nadeau and K. Gray, "SDN," O'Reilly, 2013, 384 pp, ISBN:978-1-449-34230-2B (Safari Book)

- Pfaff and B. Davie, "The Open vSwitch Database Management Protocol," IETF draft, Oct 2013, http://tools.ietf.org/html/draft-pfaff-ovsdb-proto-04

References

❑ N. McKeown, et al., ``OpenFlow: Enabling Innovation in Campus Networks,'' ACM SIGCOMM CCR, Vol. 38, No. 2, April 2008, pp. 69-74.

❑ ONF, "The OpenFlow Timeline," http://openflownetworks.com/of_timeline.php

❑ Open Data Center Alliance Usage Model: Software Defined Networking Rev 1.0," http://www.opendatacenteralliance.org/docs/Software_Defined_Networking _Master_Usage_Model_Rev1.0.pdf

❑ R. Oshana and S. Addepalli, "Networking Trends- Software Defined Networking, Network Virtualization and Cloud Orchestration," Asia Power Arch. Conf, Oct 2012, https://www.power.org/wp-content/uploads/2012/10/13.-FSL-SDN-Openflow-and-Cloud-computing-UPD_Rob-Oshana.pdf

❑ ONF, **Technical Library** (includes all OpenFlow, OF-Config, and other specifications), https://www.opennetworking.org/sdn-resources/technical-library

References (Cont)

- http://www.openvswitch.org/

- http://www.projectfloodlight.org/indigo/

- http://flowforwarding.github.io/LINC-Switch/

- http://github.com/CPqD/openflow-openwrt

- http://cpqd.github.io/ofsoftswitch13/

- http://sourceforge.net/projects/xorplus

Wikipedia Links

❑ http://en.wikipedia.org/wiki/OpenFlow

❑ http://en.wikipedia.org/wiki/Software-defined_networking

❑ http://en.wikipedia.org/wiki/Network_Functions_Virtualization

❑ http://en.wikipedia.org/wiki/Forwarding_plane

❑ http://en.wikipedia.org/wiki/NetFlow

❑ http://en.wikipedia.org/wiki/IP_Flow_Information_Export

❑ http://en.wikipedia.org/wiki/SFlow

❑ http://en.wikipedia.org/wiki/Northbound_interface

❑ http://en.wikipedia.org/wiki/Big_Switch_Networks

Wikipedia Links (Optional)

❑ http://en.wikipedia.org/wiki/Open_Data_Center_Alliance

❑ http://en.wikipedia.org/wiki/Virtual_Extensible_LAN

❑ http://en.wikipedia.org/wiki/Optical_Transport_Network

❑ http://en.wikipedia.org/wiki/Automatically_switched_optical_network

❑ http://en.wikipedia.org/wiki/Wavelength-division_multiplexing

❑ http://en.wikipedia.org/wiki/IEEE_802.1ad

❑ http://en.wikipedia.org/wiki/Transport_Layer_Security

❑ http://en.wikipedia.org/wiki/OpenStack

❑ http://en.wikipedia.org/wiki/IPv6_packet

❑ http://en.wikipedia.org/wiki/ICMPv6

❑ http://en.wikipedia.org/wiki/Multiprotocol_Label_Switching

Acronyms

- ACL   Access Control List
- API   Application Programming Interface
- ARP   Address Resolution Protocol
- ASICs   Application Specific Integrated Circuit
- BFD   Bidirectional Forwarding Detection
- BUM   Broadcast, Unknown, and Multicast
- CFM   Connectivity Fault Management
- CPU   Central Processing Unit
- DFCA   Dynamic Frequency Channel Allocation
- DSCP   Differentiated Service Control Point
- ECMP   Equal Cost Multipath
- ESP   Encrytec Security Payload
- FCAPS   Fault, Configuration, Accounting, Performance and Security
- GRE   Generic Routing Encapsulation
- ICMP   Internet Control Message Protocol
- ID   Identifier

Acronyms (Cont)

- IDS   Intrusion Detection System
- IEEE   Institution of Electrical and Electronic Engineers
- IETF   Internet Engineering Task Force
- IGMP   Internet Group Multicast Protocol
- IP   Internet Protocol
- IPFIX   IP Flow Information Export Protocol
- IPSec   IP Security
- IPv4   Internet Protocol version 4
- IPv6   Internet Protocol version 6
- JSON   Java Script Object Notation
- KVM   Kernel-based Virtual Machine
- LACP   Link Aggregation Control Protocol
- LLDP   Link Layer Discovery Protocol
- MAC   Media Access Control
- MAN   Metropolitan Area Network
- MPLS   Multiprotocol Label Switching

Acronyms (Cont)

- NFV   Network Function Virtualization
- NVGRE   Network Virtualization using Generic Routing Encapsulation
- OF   OpenFlow
- ONF   Open Networking Foundation
- openQRM   Open Qlusters Resource Manager
- OpenWRT   Open WRT54G (Linksys product name) software
- OSPF   Open Shortest Path First
- OTN   Optical Transport Network
- OVSDB   Open vSwitch Database
- PIM-SM   Protocol Independent Multicast - Sparse Mode
- PIM   Protocol Independent Multicast
- QoS   Quality of Service
- RAN   Radio area networks
- RFC   Request for Comments
- RIP   IGMP, IPv6, PIM-SM
- RIP   Routing Information Protocol

Acronyms (Cont)

- RPC   Remote Procedure Call
- RSPAN   Remote Switch Port Analyzer
- SDN   Software Defined Network
- SPAN   Switch Port Analyzer
- SSL   Secure Socket Layer
- STP   Spanning Tree Protocol
- TCAM   Ternary Content Addressable Memory
- TCP   Transmission Control Protocol
- TLS   Transport Level Security
- TLV   Type-Length-Value
- ToS   Type of Service
- TTL   Time to Live
- TTP   Table Typing Patterns
- UDP   User Datagram Protocol
- VLAN   Virtual Local Area Network
- VM   Virtual Machine
- VxLAN   Virtual Extensible Local Area Network
- WG   Working Group