OpenFlow

This module describes how to enable and configure OpenFlow on a device.

Prerequisites for OpenFlow

The device must be booted up in OpenFlow mode. (OpenFlow mode is enabled when you configure the **boot mode openflow** command on a device. All ports will be in this mode, and the device will not support any regular Cisco IOS XE features.)

Restrictions for OpenFlow

- When enabling OpenFlow mode on a device, erase all prior configurations, and delete the *vlan.dat* and *stby-vlan.dat* files from the flash filesystem.
- When the device is in Openflow mode, do not enable other control plane protocols, Border Gateway Protocol (BGP), Spanning Tree Protocol (STP), Port Channels, StackWise Virtual, and so on that work when the device is in normal mode.

Information About OpenFlow

The following sections provide more information about the feature.

OpenFlow Overview

OpenFlow is a specification from the Open Networking Foundation (ONF) that defines a flow-based forwarding infrastructure and a standardized application-programmatic interface. OpenFlow allows a controller to direct the forwarding functions of a device through a secure channel.

OpenFlow is the protocol between a controller (control plane) and an Ethernet switch (data plane). The switch has flow tables arranged in a pipeline. Flows are rules to examine packets that reach these tables.

An OpenFlow agent on the switch communicates with the controller using the OpenFlow protocol. The agent supports both OpenFlow 1.0 (wire protocol 0x1) and OpenFlow 1.3 (wire protocol 0x4). It can have up to eight controller connections. These connections are not preserved across a switchover, and the controller will have to reconnect to the agent after a switchover.

The OpenFlow implementation on Cisco Catalyst 9400 Series Switches is stateless, and nonstop forwarding (NSF) is not supported. The standby supervisor does not synchronize with the flow database.

OpenFlow Controller

The OpenFlow controller is an entity that interacts with an OpenFlow switch using the OpenFlow protocol. In most cases, a controller is a software that manages many OpenFlow logical switches. Controllers offer a centralized view of the network, and enable administrators to dictate to the underlying systems (switches and routers) on how to handle the network traffic. A controller typically runs on a Linux server, and must have IP connectivity to OpenFlow-capable switches.

A controller manages a switch, and inserts and deletes the flows on the switch. These flows support a subset of OpenFlow 1.3 and 1.0 *match* and *action* criteria.

The switch connects to the controller using the management port. The management port is in the management virtual routing and forwarding (VRF) instance, and hence provides a secure connection to the controller. To connect a controller to the switch, configure the IP address and port number on which the controller can be reached.

Flow Management

A flow entry is an element in a flow table that is used to match and process packets. It contains priority levels for matching precedence, a set of match fields for matching packets, a set of instructions to apply, and packet and byte counters. A timeout is also associated with each flow (a hard timeout or an inactivity timeout), which is used to automatically remove flows.

A maximum of 32 flow tables are supported.

Each flow provides the following information:

- Priority: High-priority flows are matched first. A flow update requires all the flows to be prioritized based on the configured priority.
- Match fields: A part of a flow entry against which a packet is matched. Match fields can match the various packet header fields. If no match information is provided for a field, a wildcard is used.
- Action: An operation that acts on a packet.

OpenFlow Pipeline

An OpenFlow pipeline is a set of linked flow tables that provide matching, forwarding, and packet modification in an OpenFlow switch. A port is where packets enter and exit the pipeline.

Packets are received on an ingress port and processed by the OpenFlow pipeline that forwards it to output ports. The packet ingress port is owned by the packet throughout the pipeline, and represents the port on which the packet was received into the switch. Note that the ingress port can also be used as a match field in a flow.

Flow actions allow packets to be sent to subsequent tables in the pipeline for further processing, and allow information to be communicated between tables. Pipeline processing stops when the action associated with a matching flow entry does not specify the next table. At this point, the packet is usually modified and forwarded. The packet can also be dropped.

Flow tables of an OpenFlow switch are sequentially numbered, starting from 0. Pipeline processing always starts by matching the packet against flow entries of flow table 0. Other flow tables can be used depending on the outcome of the match and actions in the first table, which could result in matching the packet against flow entries in subsequent tables.

Supported Match Fields and Actions

Match Field is a field against which a packet is matched, including packet headers, and the ingress port. A match field can be a wildcard (match any value) and have a bit mask to match selected bits of the field.

Action is an operation that forwards a packet to a port or subsequent tables, or modifies a packet field. Actions can be specified as part of the instructions associated with a flow entry, or an action bucket associated with a group entry. A group entry is a collection of actions that can be shared by multiple flows.

The action specified in one or more flow entries can direct packets to a base action called a group action. The purpose of the group action is to share a set of actions among multiple flows. A group consist of one or more buckets, and in turn, a bucket can have a set of actions (set, pop, or output). Cisco Catalyst 9000 Series Switches support the group types *all* and *indirect*.

The following table lists the supported match fields and actions:

Table 1: Supported Match Fields

Header Field	Prerequisite	Maskable Entrie	Example Value
Ethernet destination MAC address	—	Yes	01:80:c2:00:00:00/ ff:ff:ff:00:00:00 (with mask) de:f3:50:c7:e2:b2 (without mask)
Ethernet source MAC address	—	Yes	0e:00:00:00:00:019 (without mask)
Ethernet type	—	—	ARP (0x0806), IPv4 (0x0800), IPv6 (0x86dd), and so on
VLAN ID	—		0x13f

Header Field	Prerequisite	Maskable Entrie	Example Value
ARP target protocol address	Ethernet type should be set to 0x0806	Yes	—
IP protocol	Ethernet type should be set to 0x0800 or 0x86dd	—	ICMP (0x01), TCP (0x06), UDP (0x11), and so on
IPv4 source address	Ethernet type should be set to 0x0800	Yes	10.0.0.0/24 (with mask)
IPv4 destination address	Ethernet type should be set to 0x0800	Yes	10.0.0.254 (without mask)
IPv6 source address	Ethernet type should be set to 0x08dd	Yes	2001:DB8::1 (without mask)
IPv6 destination address	Ethernet type should be set to 0x08dd	Yes	2001:DB8:0:ABCD::1/48 (with mask)
Neighbor discovery target	Ethernet type should be set to 0x08dd and IP protocol should be set to 0x01	—	ND target
ICMPv6 type	Ethernet type should be set to 0x08dd and IP protocol should be set to 0x01	—	—
UDP/TCP source port	Ethernet type should be set to 0x0800 or 0x86dd and protocol should be set to 0x06 or 0x11	—	—
UDP/TCP destination port	Ethernet type should be set to 0x0800 or 0x86dd and protocol should be set to 0x06 or 0x11	—	—
Incoming interface	—	—	—

Supported Actions

A flow can send a packet to:

- The controller.
- Any interface of the switch (including the incoming interface).
- A subsequent flow table (after Table 0) for another lookup.
- A group.

- The switch CPU for local processing. Only Cisco Discovery Protocol and Link Layer Discovery Protocol (LLDP) packets can be sent for local processing.

A flow can add (push) or remove (pop) a VLAN tag. If a packet is an IP packet, the flow can decrement the Time to Live (TTL) header field.

The ability to modify the packet fields are defined as *Set-Field* action. A flow can also modify the following header fields of a packet:

Table 2: Number of Rewrites Supported on Header Fields

Header Field	Scale
Ethernet destination MAC address	1k
Ethernet source MAC address	256
VLAN ID	4k

Rewrite Fields

In Cisco IOS XE Bengaluru 17.4.1, the support to rewrite the following fields has been added:

Table 3: Rewrite Fields

Field	Scale
ipv4_src	4k
ipv4_dst	4k
icmpv4_type	256
tcp_src/udp_src	4k (shared by both fields)
tcp_dst/udp_dst	4k (shared by both fields)
ip_dscp	64

The IP_DSCP field is part of the IPv4 type of service (ToS) field and the IPv6 traffic class field.

OpenFlow Scale Information

Table 4: Scale Information on Supported Platforms

	Cisco Catalyst 9300 Series Switches	Cisco Catalyst 9400 Series Switches, and Cisco Catalyst 9500 Series Switches	Cisco Catalyst 9500 High-Performance Series Switches
Total number of flows	18K/9K	54K/27K	54K/27K

Flow Operations

This section describes the operations that take place when a flow is sent by the controller to be programmed in the OpenFlow device.

Typically a device has flow tables arranged in a pipeline. The pipeline capabilities information specifies the structure of the pipeline, such as the number of tables or stages, what each stage is capable of doing (match or actions), and the size of each table.

When the controller sends a flow request, the OpenFlow agent verifies whether the flow can be handled by the hardware. It compares the flow against the capabilities of the hardware that are defined when the switch is booted up. If the flow is valid, it is programmed in the appropriate flow table.

If the new pipeline is validated (whether the hardware can support the pipeline), it becomes the new set of capabilities used to check if a flow can be installed or not.

After the pipeline is instantiated and flows are installed, packets are forwarded by the switch. Ingress packets are matched against the flows in each flow table, until the highest-priority matching flow entry is found. Packet matching may be exact (match all fields of the table exactly), or partial (match some or all fields, and fields with bit masks may be partially matched). Packets can be modified or forwarded based on the configured actions. Actions can be applied in the pipeline at any time. An action can determine the next flow table to match, the set of egress ports for the packet, and whether the packet should be routed to the controller.

OpenFlow Table Pipeline

OpenFlow table feature request messages allow an OpenFlow controller to query the capabilities of existing flow tables of an OpenFlow-managed device, or configure these tables to match the supplied configuration.

All the tables can be configured with a subset of the match and action capabilities. Table sizes can also be modified at runtime. When a new flow table configuration is successfully applied, flow entries from old flow tables are removed without any notification. Dynamically configured flow tables are not persistent across reboot. The default pipeline comes up when the device boots up.

While configuring a new flow table based on a request from the OpenFlow controller, ongoing traffic, if any, flowing through the existing flows are dropped.

Breakout Port Support

Breakout ports enable a single 40G Quad Small Form-Factor Pluggable+ (QSFP+) interface to be split into four 10G SFP+ interfaces, and a single 100G QSFP28 interface into four 25G SFP28 interfaces. The breakout port support is available in the OpenFlow mode on platforms that support breakout ports in normal mode. In Cisco IOS XE Bengaluru 17.5.1, the breakout port support is available on Cisco Catalyst 9500 and 9500 High Performance Series Switches.

To view the OpenFlow port number associated with the breakout ports, use the **show openflow switch 1 ports** command. There are no specific rules to calculate the OpenFlow port number from a breakout interface name.

OpenFlow Power over Ethernet

OpenFlow supports Power over Ethernet (PoE). For PoE to work, configure either Cisco Discovery Protocol or LLDP on the device so that Cisco Discovery Protocol packets or LLDP packets are processed (and sent) by the device. Note that no OpenFlow-specific configuration is required for PoE to work with OpenFlow.

On the OpenFlow controller, configure a flow with the *output-to-local* action to ensure that packets are sent to the device CPU for local processing.

For more information about PoE, see the *Configuring POE* chapter.

How to Configure OpenFlow

The following sections provide information about the various OpenFlow configuration tasks.

Enabling OpenFlow Mode on a Device

If the switch is operating in normal mode, we recommend that you configure the **write erase** command to delete the previous configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **boot mode openflow**
4. **exit**
5. **write erase**
6. • **delete flash:***vlan.dat*
   • **delete flash:***stby-vlan.dat*
7. **reload**
8. **enable**
9. **show boot mode**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot mode openflow** **Example:** `Device(config)# boot mode openflow`	Enables OpenFlow forwarding mode.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
Step 5	**write erase** **Example:** `Device# write erase`	Erases all the files in the NVRAM. • We recommend erasing all the files, if the device was operating in normal mode previously.

Step 6	• **delete flash:***vlan.dat*
	• **delete flash:***stby-vlan.dat*
	Example:
	`Device# delete flash:vlan.dat`
	`Device# delete flash:stby-vlan.dat`
		• Deletes the vlan.dat file that stores the VLAN information.
		• Deletes the stby-vlan.dat file, if you have a standby device.

Step 7	**reload** **Example:** `Device# reload`	Reloads the switch and enables OpenFlow forwarding mode for the switch.
Step 8	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 9	**show boot mode** **Example:** `Device# show boot mode`	Displays information about the device's forwarding mode.

Example

The following is sample output from the **show boot mode** command that shows the device in OpenFlow mode:

```
Device# show boot mode

System initialized in openflow forwarding mode
System configured to boot in openflow forwarding mode
```

What to do next

To go back to normal mode, configure the **no boot mode openflow** command and then reload the device.

Configuring OpenFlow

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **feature openflow**
4. **openflow**

5. **switch 1 pipeline 1**
6. **controller ipv4** *ip-address* **port** *port-number* **vrf** *vrf-name* **security** {**none** | **tls**}
7. **datapath-id** *ID*
8. **tls trustpoint local** *name* **remote** *name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**feature openflow** **Example:** `Device(config)# feature openflow`	Enables the OpenFlow feature.
Step 4	**openflow** **Example:** `Device(config)# openflow`	Enables OpenFlow configuration and enters OpenFlow configuration mode.
Step 5	**switch 1 pipeline 1** **Example:** `Device(config-openflow)# switch 1 pipeline 1`	Configures a logical switch and pipeline, and enters OpenFlow switch configuration mode.
Step 6	**controller ipv4** *ip-address* **port** *port-number* **vrf** *vrf-name* **security** {**none** | **tls**} **Example:** `Device(config-openflow-switch)# controller ipv4 10.2.2.2 port 6633 vrf Mgmt-vrf security tls`	Connects to a controller. • You must configure the **tls trustpoint** command if you have configured TLS as the OpenFlow controller connection security option. • You do not have to configure **tls trustpoint** command if you have not configured any security option for the OpenFlow controller.
Step 7	**datapath-id** *ID* **Example:** `Device(config-openflow-switch)# datapath-id 0x12345678`	(Optional) Sets the OpenFlow logical switch ID. • The *ID* argument specifies the switch ID, which is a hexadecimal value.
Step 8	**tls trustpoint local** *name* **remote** *name* **Example:** `Device(config-openflow-switch)# tls trustpoint local trustpoint1 remote trustpoint1`	(Optional) Configures an OpenFlow switch Transport Layer Security (TLS) trustpoint.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config-openflow-switch)# end`	Exits OpenFlow switch configuration mode and returns to privileged EXEC mode.

Configuring an Interface in OpenFlow Mode

You can configure either a Layer 2 or Layer 3 interface in OpenFlow mode. When using a Layer 3 interface, configure the **no switchport** command in interface configuration mode. Perform the following task when using a Layer 2 interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **feature openflow**
4. **interface** *type number*
5. **switchport mode trunk**
6. **switchport nonnegotiate**
7. **no keepalive**
8. **spanning-tree bpdufilter enable**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**feature openflow** **Example:** `Device(config)# feature openflow`	Enables the OpenFlow feature.
Step 4	**interface** *type number* **Example:** `Device(config)# interface gigabitethernet 1/0/3`	Configures an interface and enters interface configuration mode.
Step 5	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the trunking mode of the Layer 2-switched interface to trunk.

	Command or Action	Purpose
Step 6	**switchport nonnegotiate** **Example:** `Device(config-if)# switchport nonnegotiate`	Specifies that the device will not engage in negotiation protocol on this interface.
Step 7	**no keepalive** **Example:** `Device(config-if)# no keepalive`	Disables keepalive packets.
Step 8	**spanning-tree bpdufilter enable** **Example:** `Device(config-if)# spanning-tree bpdufilter enable`	Enables bridge protocol data unit (BPDU) filtering on the interface.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying OpenFlow

Use these commands to verify your OpenFlow configuration. These commands can be used in any order.

SUMMARY STEPS

1. **enable**
2. **show openflow hardware capabilities**
3. **show openflow switch 1 controller**
4. **show openflow switch 1 ports**
5. **show openflow switch 1 flows list**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

- Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show openflow hardware capabilities**

Displays the hardware capabilities of an OpenFlow device.

Example:

```
Device# show openflow hardware capabilities
```

```
  Max Interfaces: 1000
  Aggregated Statistics: YES


  Pipeline ID: 1
    Pipeline Max Flows: 2322
    Max Flow Batch Size: 100
    Statistics Max Polling Rate (flows/sec): 10000
    Pipeline Default Statistics Collect Interval: 5

Flow table ID: 0

    Max Flow Batch Size: 100
    Max Flows: 1022
    Bind Subintfs: FALSE
    Primary Table: TRUE
    Table Programmable: TRUE
    Miss Programmable: TRUE
    Number of goto tables: 1
    Goto table id: 1
    Number of miss goto tables: 1
    Miss Goto table id: 1
    Stats collection time for full table (sec): 1

.
.
.
```

Step 3 **show openflow switch 1 controller**

Displays information about the controller connected to the switch.

Example:

```
Device# show openflow switch 1 controller

Logical Switch Id: 1
Total Controllers: 1
Controller: 1
10.10.23.200:6633
Protocol: tcp
VRF: Mgmt-vrf
Connected: Yes
Role: Equal
Negotiated Protocol Version: OpenFlow 1.3
Last Alive Ping: 2018-06-04 17:59:20 PDT
state: ACTIVE
sec_since_connect: 50
```

Step 4 **show openflow switch 1 ports**

Displays information about the ports on an OpenFlow switch.

Example:

```
Device# show openflow switch 1 ports
Logical Switch Id: 1
Port    Interface Name   Config-State     Link-State  Features
   1           Gi1/0/1        PORT_UP        LINK_UP  1GB-FD
   2           Gi1/0/2        PORT_UP        LINK_UP  1GB-FD
   3           Gi1/0/3        PORT_UP        LINK_UP  1GB-FD
   4           Gi1/0/4        PORT_UP        LINK_UP  1GB-FD
   5           Gi1/0/5        PORT_UP      LINK_DOWN  1GB-HD
   6           Gi1/0/6        PORT_UP      LINK_DOWN  1GB-HD
```

```
 7           Gi1/0/7        PORT_UP      LINK_DOWN  1GB-HD
 8           Gi1/0/8        PORT_UP      LINK_DOWN  1GB-HD
 9           Gi1/0/9        PORT_UP        LINK_UP  1GB-FD
10          Gi1/0/10        PORT_UP        LINK_UP  1GB-FD
11          Gi1/0/11        PORT_UP        LINK_UP  1GB-FD
12          Gi1/0/12        PORT_UP        LINK_UP  1GB-FD
13          Gi1/0/13        PORT_UP      LINK_DOWN  1GB-HD
14          Gi1/0/14        PORT_UP      LINK_DOWN  1GB-HD
15          Gi1/0/15        PORT_UP      LINK_DOWN  1GB-HD
16          Gi1/0/16        PORT_UP      LINK_DOWN  1GB-HD
17          Gi1/0/17        PORT_UP      LINK_DOWN  1GB-HD
18          Gi1/0/18        PORT_UP      LINK_DOWN  1GB-HD
19          Gi1/0/19        PORT_UP        LINK_UP  1GB-FD
20          Gi1/0/20        PORT_UP        LINK_UP  1GB-FD
21          Gi1/0/21        PORT_UP        LINK_UP  1GB-FD
22          Gi1/0/22        PORT_UP        LINK_UP  1GB-FD
23          Gi1/0/23        PORT_UP      LINK_DOWN  1GB-HD
24          Gi1/0/24        PORT_UP      LINK_DOWN  1GB-HD
25           Gi1/1/1        PORT_UP      LINK_DOWN  1GB-HD
26           Gi1/1/2        PORT_UP      LINK_DOWN  1GB-HD
27           Gi1/1/3        PORT_UP      LINK_DOWN  1GB-HD
28           Gi1/1/4        PORT_UP      LINK_DOWN  1GB-HD
29           Te1/1/1        PORT_UP      LINK_DOWN  10GB-FD
30           Te1/1/2        PORT_UP      LINK_DOWN  10GB-FD
31           Te1/1/3        PORT_UP      LINK_DOWN  10GB-FD
32           Te1/1/4        PORT_UP      LINK_DOWN  10GB-FD
33           Te1/1/5        PORT_UP      LINK_DOWN  10GB-FD
34           Te1/1/6        PORT_UP      LINK_DOWN  10GB-FD
35           Te1/1/7        PORT_UP      LINK_DOWN  10GB-FD
36           Te1/1/8        PORT_UP      LINK_DOWN  10GB-FD
37           Fo1/1/1        PORT_UP      LINK_DOWN  40GB-FD
38           Fo1/1/2        PORT_UP      LINK_DOWN  40GB-FD
39          Twe1/1/1        PORT_UP      LINK_DOWN  10GB-FD
40          Twe1/1/2        PORT_UP      LINK_DOWN  10GB-FD
```

Step 5 show openflow switch 1 flows list

Displays OpenFlow entries.

The following sample output displays a flow that is available in Table 0, where *match any* goes to Table 1. (match any means that all the packets go to Table 1.) In Table 1, the destination MAC address 00:00:01:00:00:01 is matched, and the output port is set to 36.

Example:

```
Device# show openflow switch 1 flows list

Logical Switch Id: 1
Total flows: 8

Flow: 1 Match: any Actions: goto_table:1, Priority: 9000, Table: 0, Cookie: 0x1,
Duration: 2382.117s, Packets: 34443, Bytes: 3359315

Flow: 2 Match: any Actions: drop, Priority: 0, Table: 0, Cookie: 0x0,
Duration: 2382.118s, Packets: 294137, Bytes: 28806211

Flow: 3 Match: any Actions: drop, Priority: 0, Table: 1, Cookie: 0x0,
Duration: 2382.118s, Packets: 34443, Bytes: 3359315

Flow: 4 Match: dl_dst=00:00:01:00:00:01 Actions: output:36, Priority: 9000,
```

```
Table: 1, Cookie: 0x1, Duration: 2382.116s, Packets: 0, Bytes: 0
```

Configuration Examples for OpenFlow

Example: Enabling OpenFlow on a Device

The following example shows how to enable OpenFlow:

```
Device> enable
Device# configure terminal
Device(config)# boot mode openflow
Device(config)# exit
Device# write erase
Device# delete flash:vlan.dat
Device# reload
Device> enable
Device# show boot mode
```

Example: Configuring OpenFlow

The following example shows how to configure OpenFlow:

```
Device# configure terminal
Device(config)# feature openflow
Device(config)# openflow
Device(config-openflow)# switch 1 pipeline 1
Device(config-openflow-switch)# controller ipv4 10.2.2.2 port 6633 vrf Mgmt-vrf security
tls
Device(config-openflow-switch)# datapath-id 0x12345678
Device(config-openflow-switch)# tls trustpoint local trustpoint1 remote trustpoint1
Device(config-openflow-switch)# end
```

Additional References

Related Documents

Related Topic	Document Title
OpenFlow commands	*Programmability Command Reference*
Open Network Foundation	https://www.opennetworking.org/

Related Topic	Document Title
Faucet OpenFlow controller	• https://faucet.nz/ • https://docs.faucet.nz/en/latest/
PoE	• "Configuring Power over Ethernet" on Cisco Catalyst 9300 Series Switches • "Configuring PoE" on Cisco Catalyst 9400 Series Switches

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for OpenFlow

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5: Feature Information for OpenFlow

Feature Name	Release	Feature Information
OpenFlow	Cisco IOS XE Fuji 16.9.1	OpenFlow is a Software-Defined Network (SDN) standard. It defines a communication protocol in SDN environments that enables an SDN controller to directly interact with the forwarding plane of network devices such as switches and routers. This feature was implemented on the following platforms: • Catalyst 9300 Series Switches • Catalyst 9400 Series Switches • Catalyst 9500 Series Switches • Catalyst 9500 Series High Performance Switches
	Cisco IOS XE Gibraltar 16.10.1	Table feature message support on Catalyst 9500 Series High Performance Switches was implemented.
OpenFlow Power over Ethernet	Cisco IOS XE Gibraltar 16.12.1	PoE is supported on OpenFlow ports. This feature was implemented on the following platforms: • Catalyst 9300 Series Switches • Catalyst 9400 Series Switches
OpenFlow Breakout Port Support	Cisco IOS XE Bengaluru 17.5.1	Breakout cables enable a single 40G Quad Small Form-Factor Pluggable+ (QSFP+) interface to be split into four 10G SFP+ interfaces, and a single 100G QSFP28 interface into four 25G SFP28 interfaces. This feature was introduced on the following platforms: • Catalyst 9500 and 9500 High Performance Series Switches