# Configuring Wireshark



## Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

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

### Related Topics

Feature History and Information for Troubleshooting Software Configuration

## Prerequisites for Wireshark

- Wireshark is supported on Supervisor Engine 7-E, Supervisor Engine 7L-E, Catalyst 3850, Catalyst 3650, Wireless LAN Controller 5700 Series, Catalyst 4500X-16, and Catalyst 4500X-32.
- IP Base image or IP Services image is required for Embedded Wireshark.

# Restrictions for Wireshark

- Starting in Cisco IOS Release XE 3.3.0(SE), global packet capture on Wireshark is not supported.
- Capture filters are not supported.
- The CLI for configuring Wireshark requires that the feature be executed only from EXEC mode. Actions that usually occur in configuration submode (such as defining capture points), are handled at the EXEC mode instead. All key commands are not NVGEN'd and are not synchronized to the standby supervisor in NSF and SSO scenarios.
- Packets captured in the output direction of an interface might not reflect the changes made by controller rewrite (includes TTL, VLAN tag, CoS, checksum, MAC addresses, DSCP, precedent, UP, etc.).
- Limiting circular file storage by file size is not supported.

### Wireless Packet Capture

- The only form of wireless capture is a CAPWAP tunnel capture.
- When capturing CAPWAP tunnels, no other interface types can be used as attachment points on the same capture point.
- Capturing multiple CAPWAP tunnels is supported.
- Core filters are not applied and should be omitted when capturing a CAPWAP tunnel.
- To capture a CAPWAP data tunnel, each CAPWAP tunnel is mapped to a physical port and an appropriate ACL will be applied to filter the traffic.
- To capture a CAPWAP non-data tunnel, the switch is set to capture traffic on all ports and apply an appropriate ACL to filter the traffic.
- To capture a CAPWAP tunnel, no matter what kind, the controller will be told to capture the traffic on all physical ports and apply an appropriate ACL to filter the traffic.

### Configuration Limitations

- Starting from Cisco IOS release 16.1, as many as 8 capture points can be defined, but only one can be active at a time. You must stop one before you can start the other.
- You cannot use VRFs, management ports, and private VLANs as attachment points.
- Only one ACL (IPv4, IPv6 or MAC) is allowed in a Wireshark class map.
- Wireshark cannot capture packets on a destination SPAN port.
- Wireshark stops capturing packets when one of the attachment points (interfaces) attached to a capture point stops working. For example, if the device that is associated with an attachment point is unplugged from the controller. To resume capturing packets, restart manually.
- CPU-injected packets are considered control plane packets and are not captured on an interface egress capture.

- MAC ACL is only used for non-IP packets such as ARP. It will not be supported on a Layer 3 port or SVI.
- IPv6-based ACLs are not supported in VACL.
- Layer 2 EtherChannels are not supported. Individual members like GigabitEthernet 1 and GigabitEthernet 2 are supported on Layer 3 EtherChannels.
- ACL logging and Wireshark are incompatible. After Wireshark is activated, it takes priority. All traffic, including traffic captured by ACL logging on ports, will be redirected to Wireshark. We recommend that you deactivate ACL logging before you start Wireshark. Otherwise, Wireshark traffic is contaminated by ACL logging traffic.
- Wireshark does not capture packets dropped by floodblock.
- If you capture both PACL and RACL on the same port, only one copy is sent to the CPU. If you capture a DTLS-encrypted CAPWAP interface, two copies are sent to Wireshark, one encrypted and the other decrypted. The same behavior will occur if we capture a Layer 2 interface carrying DTLS-encrypted CAPWAP traffic. The core filter is based on the outer CAPWAP header.
- Control plane packets are not rate limited and do not impact performance. Use filters to limit control plane packet capture.

# Information About Wireshark

## Wireshark Overview

Wireshark is a packet analyzer program, formerly known as Ethereal, that supports multiple protocols and presents information in a text-based user interface.

The ability to capture and analyze traffic provides data on network activity. Prior to Cisco IOS Release XE 3.3.0(SE), only two features addressed this need: SPAN and debug platform packet. Both have limitations. SPAN is ideal for capturing packets, but can only deliver them by forwarding them to some specified local or remote destination; it provides no local display or analysis support.

So the need exists for a traffic capture and analysis mechanism that is applicable to both hardware and software forwarded traffic and that provides strong packet capture, display, and analysis support, preferably using a well known interface.

Wireshark dumps packets to a file using a well known format called .pcap, and is applied or enabled on individual interfaces. You specify an interface in EXEC mode along with the filter and other parameters. The Wireshark application is applied only when you enter a **start** command, and is removed only when Wireshark stops capturing packets either automatically or manually.

**Note** The current version of Wireshark installed on the switch is 1.10.8.

# Capture Points

A capture point is the central policy definition of the Wireshark feature. The capture point describes all of the characteristics associated with a given instance of Wireshark: which packets to capture, where to capture them from, what to do with the captured packets, and when to stop. Capture points can be modified after creation, and do not become active until explicitly activated with a **start** command. This process is termed activating the capture point or starting the capture point. Capture points are identified by name and can also be manually or automatically deactivated or stopped.

Multiple capture points can be defined, but only one can be active at a time. You need to stop one before you can start the other.

In case of stacked systems, the capture point is activated on the active member. A switchover will terminate any active packet capture session and it will have to be restarted.

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Attachment Points

An attachment point is a point in the logical packet process path associated with a capture point. An attachment point is an attribute of the capture point. Packets that impact an attachment point are tested against capture point filters; packets that match are copied and sent to the associated Wireshark instance of the capture point. A specific capture point can be associated with multiple attachment points, with limits on mixing attachment points of different types. Some restrictions apply when you specify attachment points of different types. Attachment points are directional (input or output or both) with the exception of the Layer 2 VLAN attachment point, which is always bidirectional.

In case of stacked systems, the attachment points on all stack members are valid. EPC captures the packets from all the defined attachment points. However these packets are processed only on the active member.

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
Example: Simple Capture and Display, on page 31

# Filters

Filters are attributes of a capture point that identify and limit the subset of traffic traveling through the attachment point of a capture point, which is copied and passed to Wireshark. To be displayed by Wireshark, a packet must pass through an attachment point, as well as all of the filters associated with the capture point.

A capture point has the following types of filters:

- Core system filter—The core system filter is applied by hardware, and its match criteria is limited by hardware. This filter determines whether hardware-forwarded traffic is copied to software for Wireshark purposes.
- Display filter—The display filter is applied by Wireshark. Packets that fail the display filter are not displayed.

### Core System Filter

You can specify core system filter match criteria by using the class map or ACL, or explicitly by using the CLI.

**Note** When specifying CAPWAP as an attachment point, the core system filter is not used.

In some installations, you need to obtain authorization to modify the controller configuration, which can lead to extended delays if the approval process is lengthy. This can limit the ability of network administrators to monitor and analyze traffic. To address this situation, Wireshark supports explicit specification of core system filter match criteria from the EXEC mode CLI. The disadvantage is that the match criteria that you can specify is a limited subset of what class map supports, such as MAC, IP source and destination addresses, ether-type, IP protocol, and TCP/UDP source and destination ports.

If you prefer to use configuration mode, you can define ACLs or have class maps refer capture points to them. Explicit and ACL-based match criteria are used internally to construct class maps and policy maps.

Note The ACL and class map configuration are part of the system and not aspects of the Wireshark feature.

### Display Filter

With the display filter, you can direct Wireshark to further narrow the set of packets to display when decoding and displaying from a .pcap file.

### Related Topics

Additional References, on page 46

# Actions

Wireshark can be invoked on live traffic or on a previously existing .pcap file. When invoked on live traffic, it can perform four types of actions on packets that pass its display filters:

- Captures to buffer in memory to decode and analyze and store
- Stores to a .pcap file
- Decodes and displays
- Stores and displays

When invoked on a .pcap file only, only the decode and display action is applicable.

# Storage of Captured Packets to Buffer in Memory

Packets can be stored in the capture buffer in memory for subsequent decode, analysis, or storage to a .pcap file.

The capture buffer can be in linear or circular mode. In linear mode, new packets are discarded when the buffer is full. In circular mode, if the buffer is full, the oldest packets are discarded to accommodate the new packets. Although the buffer can also be cleared when needed, this mode is mainly used for debugging network traffic. However, it is not possible to only clear the contents of the buffer alone without deleting it. Stop the current captures and restart the capture again for this to take effect.

**Note** If you have more than one capture that is storing packets in a buffer, clear the buffer before starting a new capture to avoid memory loss.

# Storage of Captured Packets to a .pcap File

**Note** When WireShark is used on switches in a stack, packet captures can be stored only on flash or USB flash devices connected to the active switch.

For example, if flash1 is connected to the active switch, and flash2 is connected to the secondary switch, only flash1 can be used to store packet captures.

Attempts to store packet captures on devices other than flash or USB flash devices connected to the active switch will probably result in errors.

Wireshark can store captured packets to a .pcap file. The capture file can be located on the following storage devices:

- Controller on-board flash storage (flash:)
- USB drive (usbflash0:)

**Note** Attempts to store packet captures on unsupported devices or devices not connected to the active switch will probably result in errors.

When configuring a Wireshark capture point, you can associate a filename. When the capture point is activated, Wireshark creates a file with the specified name and writes packets to it. If the file already exists at the time of creation of the capture point, Wireshark queries you as to whether the file can be overwritten. If the file already exists at the time of activating the capture point, Wireshark will overwrite the existing file. Only one capture point may be associated with a given filename.

If the destination of the Wireshark writing process is full, Wireshark fails with partial data in the file. You must ensure that there is sufficient space in the file system before you start the capture session. With Cisco IOS Release IOS XE 3.3.0(SE), the file system full status is not detected for some storage devices.

You can reduce the required storage space by retaining only a segment, instead of the entire packet. Typically, you do not require details beyond the first 64 or 128 bytes. The default behavior is to store the entire packet.

To avoid possible packet drops when processing and writing to the file system, Wireshark can optionally use a memory buffer to temporarily hold packets as they arrive. Memory buffer size can be specified when the capture point is associated with a .pcap file.

# Packet Decoding and Display

Wireshark can decode and display packets to the console. This functionality is possible for capture points applied to live traffic and for capture points applied to a previously existing .pcap file.

**Note** Decoding and displaying packets may be CPU intensive.

Wireshark can decode and display packet details for a wide variety of packet formats. The details are displayed by entering the **monitor capture name start** command with one of the following keyword options, which place you into a display and decode mode:

- brief—Displays one line per packet (the default).
- detailed—Decodes and displays all the fields of all the packets whose protocols are supported. Detailed modes require more CPU than the other two modes.
- (hexadecimal) dump—Displays one line per packet as a hexadecimal dump of the packet data and the printable characters of each packet.

When you enter the **capture** command with the decode and display option, the Wireshark output is returned to Cisco IOS and displayed on the console unchanged.

### Live Traffic Display

Wireshark receives copies of packets from the core system. Wireshark applies its display filters to discard uninteresting packets, and then decodes and displays the remaining packets.

### .pcap File Display

Wireshark can decode and display packets from a previously stored .pcap file and direct the display filter to selectively displayed packets.

## Packet Storage and Display

Functionally, this mode is a combination of the previous two modes. Wireshark stores packets in the specified .pcap file and decodes and displays them to the console. Only the core filters are applicable here.

## Wireshark Capture Point Activation and Deactivation

After a Wireshark capture point has been defined with its attachment points, filters, actions, and other options, it must be activated. Until the capture point is activated, it does not actually capture packets.

Before a capture point is activated, some functional checks are performed. A capture point cannot be activated if it has neither a core system filter nor attachment points defined. Attempting to activate a capture point that does not meet these requirements generates an error.*

**Note** *When performing a wireless capture with a CAPWAP tunneling interface, the core system filter is not required and cannot be used.

The display filters are specified as needed.

After Wireshark capture points are activated, they can be deactivated in multiple ways. A capture point that is storing only packets to a .pcap file can be halted manually or configured with time or packet limits, after which the capture point halts automatically.

When a Wireshark capture point is activated, a fixed rate policer is applied automatically in the hardware so that the CPU is not flooded with Wireshark-directed packets. The disadvantage of the rate policer is that you cannot capture contiguous packets beyond the established rate even if more resources are available.

The set packet capture rate is 1000 packets per sec (pps). The 1000 pps limit is applied to the sum of all attachment points. For example, if we have a capture session with 3 attachment points, the rates of all 3 attachment points added together is policed to 1000 pps.

**Note** Policer is not supported for control-plane packet capture. When activating control-plane capture points, you need to be extra cautious, so that it does not flood the CPU.

## Wireshark Features

This section describes how Wireshark features function in the controller environment:

- If port security and Wireshark are applied on an ingress capture, a packet that is dropped by port security will still be captured by Wireshark. If port security is applied on an ingress capture, and Wireshark is applied on an egress capture, a packet that is dropped by port security will not be captured by Wireshark.
- Packets dropped by Dynamic ARP Inspection (DAI) are not captured by Wireshark.

- If a port that is in STP blocked state is used as an attachment point and the core filter is matched, Wireshark will capture the packets that come into the port, even though the packets will be dropped by the switch.
- Classification-based security features—Packets that are dropped by input classification-based security features (such as ACLs and IPSG) are not caught by Wireshark capture points that are connected to attachment points at the same layer. In contrast, packets that are dropped by output classification-based security features are caught by Wireshark capture points that are connected to attachment points at the same layer. The logical model is that the Wireshark attachment point occurs after the security feature lookup on the input side, and symmetrically before the security feature lookup on the output side.

  On ingress, a packet goes through a Layer 2 port, a VLAN, and a Layer 3 port/SVI. On egress, the packet goes through a Layer 3 port/SVI, a VLAN, and a Layer 2 port. If the attachment point is before the point where the packet is dropped, Wireshark will capture the packet. Otherwise, Wireshark will not capture the packet. For example, Wireshark capture policies connected to Layer 2 attachment points in the input direction capture packets dropped by Layer 3 classification-based security features. Symmetrically, Wireshark capture policies attached to Layer 3 attachment points in the output direction capture packets dropped by Layer 2 classification-based security features.
- Routed ports and switch virtual interfaces (SVIs)—Wireshark cannot capture the output of an SVI because the packets that go out of an SVI's output are generated by CPU. To capture these packets, include the control plane as an attachment point.
- VLANs—Starting with Cisco IOS Release 16.1, when a VLAN is used as a Wireshark attachment point, packet capture is supported on L2 and L3 in both input and output directions.
- Redirection features—In the input direction, features traffic redirected by Layer 3 (such as PBR and WCCP) are logically later than Layer 3 Wireshark attachment points. Wireshark captures these packets even though they might later be redirected out another Layer 3 interface. Symmetrically, output features redirected by Layer 3 (such as egress WCCP) are logically prior to Layer 3 Wireshark attachment points, and Wireshark will not capture them.
- SPAN—Wireshark cannot capture packets on interface configured as a SPAN destination.
- SPAN—Wireshark is able to capture packets on interfaces configured as a SPAN source in the ingress direction, and may be available for egress direction too.
- You can capture packets from a maximum of 1000 VLANs at a time, if no ACLs are applied. If ACLs are applied, the hardware will have less space for Wireshark to use. As a result, the maximum number of VLANs than can be used for packet capture at a time will be lower. Using more than 1000 VLANs tunnels at a time or extensive ACLs might have unpredictable results. For example, mobility may go down.

**Note** Capturing an excessive number of attachment points at the same time is strongly discouraged because it may cause excessive CPU utilization and unpredictable hardware behavior.

### Wireless Packet Capture in Wireshark

- Wireless traffic is encapsulated inside CAPWAP packets. However, capturing only a particular wireless client's traffic inside a CAPWAP tunnel is not supported when using the CAPWAP tunnel as an attachment point. To capture only a particular wireless client's traffic, use the client VLAN as an attachment point and formulate the core filter accordingly.

- Limited decoding of inner wireless traffic is supported. Decoding of inner wireless packets inside encrypted CAPWAP tunnels is not supported.
- No other interface type can be used with the CAPWAP tunneling interface on the same capture point. A CAPWAP tunneling interface and a Level 2 port cannot be attachment points on the same capture point.
- You cannot specify a core filter when capturing packets for Wireshark via the CAPWAP tunnel. However, you can use the Wireshark display filters for filtering wireless client traffic against a specific wireless client.
- You can capture packets from a maximum of 135 CAPWAP tunnels at a time if no ACLs are applied. If ACLs are applied, the hardware memory will have less space for Wireshark to use. As a result, the maximum number of CAPWAP tunnels than can be used for packet capture at a time will be lower. Using more than 135 CAPWAP tunnels at a time or unsing extensive ACLs might have unpredictable results. For example, mobility may go down.

**Note** Capturing an excessive number of attachment points at the same time is strongly discouraged because it may cause excessive CPU utilization and unpredictable hardware behavior.

# Guidelines for Wireshark

- During Wireshark packet capture, hardware forwarding happens concurrently.
- Before starting a Wireshark capture process, ensure that CPU usage is moderate and that sufficient memory (at least 200 MB) is available.
- If you plan to store packets to a storage file, ensure that sufficient space is available before beginning a Wireshark capture process.
- The CPU usage during Wireshark capture depends on how many packets match the specified conditions and on the intended actions for the matched packets (store, decode and display, or both).
- Where possible, keep the capture to the minimum (limit by packets, duration) to avoid high CPU usage and other undesirable conditions.
- Because packet forwarding typically occurs in hardware, packets are not copied to the CPU for software processing. For Wireshark packet capture, packets are copied and delivered to the CPU, which causes an increase in CPU usage.

  To avoid high CPU usage, do the following:

  - Attach only relevant ports.
  - Use a class map, and secondarily, an access list to express match conditions. If neither is viable, use an explicit, in-line filter.
  - Adhere closely to the filter rules. Restrict the traffic type (such as, IPv4 only) with a restrictive, rather than relaxed ACL, which elicits unwanted traffic.

- Always limit packet capture to either a shorter duration or a smaller packet number. The parameters of the capture command enable you to specify the following:

  - Capture duration
  - Number of packets captured
  - File size
  - Packet segment size

- Run a capture session without limits if you know that very little traffic matches the core filter.
- You might experience high CPU (or memory) usage if:
  - You leave a capture session enabled and unattended for a long period of time, resulting in unanticipated bursts of traffic.
  - You launch a capture session with ring files or capture buffer and leave it unattended for a long time, resulting in performance or system health issues.
- During a capture session, watch for high CPU usage and memory consumption due to Wireshark that may impact controller performance or health. If these situations arise, stop the Wireshark session immediately.
- Avoid decoding and displaying packets from a .pcap file for a large file. Instead, transfer the .pcap file to a PC and run Wireshark on the PC.
- You can define up to eight Wireshark instances. An active **show** command that decodes and displays packets from a .pcap file or capture buffer counts as one instance. However, only one of the instances can be active.
- Whenever an ACL that is associated with a running capture is modified, you must restart the capture for the ACL modifications to take effect. If you do not restart the capture, it will continue to use the original ACL as if it had not been modified.
- To avoid packet loss, consider the following:
  - Use store-only (when you do not specify the display option) while capturing live packets rather than decode and display, which is an CPU-intensive operation (especially in detailed mode).
  - If you have more than one capture that is storing packets in a buffer, clear the buffer before starting a new capture to avoid memory loss.
  - If you use the default buffer size and see that you are losing packets, you can increase the buffer size to avoid losing packets.
  - Writing to flash disk is a CPU-intensive operation, so if the capture rate is insufficient, you may want to use a buffer capture.
  - The Wireshark capture session always operates in streaming mode at the rate of 1000 pps.
- The streaming capture mode rate is 1000 pps.
- If you want to decode and display live packets in the console window, ensure that the Wireshark session is bounded by a short capture duration.

**Warning** A Wireshark session with either a longer duration limit or no capture duration (using a terminal with no auto-more support using the **term len 0** command) may make the console or terminal unusable.

- When using Wireshark to capture live traffic that leads to high CPU, usage, consider applying a QoS policy temporarily to limit the actual traffic until the capture process concludes.
- All Wireshark-related commands are in EXEC mode; no configuration commands exist for Wireshark.

  If you need to use access list or class-map in the Wireshark CLI, you must define an access list and class map with configuration commands.
- No specific order applies when defining a capture point; you can define capture point parameters in any order, provided that CLI allows this. The Wireshark CLI allows as many parameters as possible on a single line. This limits the number of commands required to define a capture point.
- All parameters except attachment points take a single value. Generally, you can replace the value with a new one by reentering the command. After user confirmation, the system accepts the new value and overrides the older one. A **no** form of the command is unnecessary to provide a new value, but it is necessary to remove a parameter.
- Wireshark allows you to specify one or more attachment points. To add more than one attachment point, reenter the command with the new attachment point. To remove an attachment point, use the **no** form of the command. You can specify an interface range as an attachment point. For example, enter **monitor capture mycap interface GigabitEthernet1/0/1 in** where interface GigabitEthernet1/0/1 is an attachment point.

  If you also need to attach interface GigabitEthernet1/0/2, specify it in another line as follows:

  **monitor capture mycap interface GigabitEthernet1/0/2 in**
- You cannot make changes to a capture point when the capture is active.
- The action you want to perform determines which parameters are mandatory. The Wireshark CLI allows you to specify or modify any parameter prior to entering the **start** command. When you enter the **start** command, Wireshark will start only after determining that all mandatory parameters have been provided.
- If the file already exists at the time of creation of the capture point, Wireshark queries you as to whether the file can be overwritten. If the file already exists at the time of activating the capture point, Wireshark will overwrite the existing file.
- The core filter can be an explicit filter, access list, or class map. Specifying a newer filter of these types replaces the existing one.

  **Note** A core filter is required except when using a CAPWAP tunnel interface as a capture point attachment point.
- You can terminate a Wireshark session with an explicit **stop** command or by entering **q** in automore mode. The session could terminate itself automatically when a stop condition such as duration or packet capture limit is met, or if an internal error occurs, or resource is full (specifically if disk is full in file mode).

- Dropped packets will not be shown at the end of the capture. However, only the count of dropped, oversized packets will be displayed.

## Default Wireshark Configuration

The table below shows the default Wireshark configuration.

| Feature | Default Setting |
|---|---|
| Duration | No limit |
| Packets | No limit |
| Packet-length | No limit (full packet) |
| File size | No limit |
| Ring file storage | No |
| Buffer storage mode | Linear |

# How to Configure Wireshark

To configure Wireshark, perform these basic steps.

1. Define a capture point.
2. (Optional) Add or modify the capture point's parameters.
3. Activate or deactivate a capture point.
4. Delete the capture point when you are no longer using it.

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Defining a Capture Point

The example in this procedure defines a very simple capture point. If you choose, you can define a capture point and all of its parameters with one instance of the **monitor capture** command.

**Note** You must define an attachment point, direction of capture, and core filter to have a functional capture point.

An exception to needing to define a core filter is when you are defining a wireless capture point using a CAPWAP tunneling interface. In this case, you do not define your core filter. It cannot be used.

Follow these steps to define a capture point.

### SUMMARY STEPS

1. **enable**
2. **show capwap summary**
3. **monitor capture** {*capture-name*} {**interface** *interface-type interface-id* | **control-plane**} {**in** | **out** | **both**}
4. **monitor capture** {*capture-name*} [**match** {**any** | **ipv4 any any** | **ipv6**} **any any**}]
5. **show monitor capture** {*capture-name*} [ **parameter**]
6. **show running-config**
7. **copy running-config startup-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **show capwap summary**<br>**Example:**<br>`Controller# show capwap summary` | Displays the CAPWAP tunnels available as attachment points for a wireless capture.<br>**Note** Use this command only if you are using a CAPWAP tunnel as an attachment point to perform a wireless capture. See the CAPWAP example in the examples section. |
| **Step 3** | **monitor capture** {*capture-name*} {**interface** *interface-type interface-id* | **control-plane**} {**in** | **out** | **both**}<br>**Example:**<br>`Controller# monitor capture mycap interface GigabitEthernet1/0/1 in` | Defines the capture point, specifies the attachment point with which the capture point is associated, and specifies the direction of the capture.<br>The keywords have these meanings:<br>• *capture-name*—Specifies the name of the capture point to be defined (mycap is used in the example). Capture Name should be less than or equal to 8 characters. |

| | Command or Action | Purpose |
|---|---|---|
| | | Only alphanumeric characters and underscore (_) is permitted<br>• (Optional) **interface** *interface-type interface-id*—Specifies the attachment point with which the capture point is associated (GigabitEthernet1/0/1 is used in the example).<br>**Note** Optionally, you can define multiple attachment points and all of the parameters for this capture point with this one command instance. These parameters are discussed in the instructions for modifying capture point parameters. Range support is also available both for adding and removing attachment points.<br>Use one of the following for *interface-type*:<br>• **GigabitEthernet**—Specifies the attachment point as GigabitEthernet.<br>• **vlan**—Specifies the attachment point as a VLAN.<br>**Note** Only ingress capture (in) is allowed when using this interface as an attachment point.<br>• **capwap**—Specifies the attachment point as a CAPWAP tunnel.<br>**Note** When using this interface as an attachment point, a core filter cannot be used.<br>• (Optional) **control-plane**—Specifies the control plane as an attachment point.<br>• **in** \| **out** \| **both**—Specifies the direction of capture. |
| **Step 4** | **monitor capture** {*capture-name*} [**match** {**any** \| **ipv4 any any** \| **ipv6**} **any any**}]<br>**Example:**<br>`Controller# monitor capture mycap interface GigabitEthernet1/0/1 in match any` | Defines the core system filter.<br>**Note** When using the CAPWAP tunneling interface as an attachment point, do not perform this step because a core filter cannot be used.<br>The keywords have these meanings:<br>• *capture-name*—Specifies the name of the capture point to be defined (mycap is used in the example).<br>• **match**—Specifies a filter. The first filter defined is the core filter. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note** A capture point cannot be activated if it has neither a core system filter nor attachment points defined. Attempting to activate a capture point that does not meet these requirements generates an error.<br>• **ipv4**—Specifies an IP version 4 filter.<br>• **ipv6**—Specifies an IP version 6 filter. |
| **Step 5** | **show monitor capture** {*capture-name*}[ **parameter**]<br>**Example:**<br>`Controller# show monitor capture mycap parameter`<br>`monitor capture mycap interface GigabitEthernet1/0/1 in`<br>`monitor capture mycap match any` | Displays the capture point parameters that you defined in Step 2 and confirms that you defined a capture point. |
| **Step 6** | **show running-config**<br>**Example:**<br>`Controller# show running-config` | Verifies your entries. |
| **Step 7** | **copy running-config startup-config**<br>**Example:**<br>`Controller# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

### Example

To define a capture point with a CAPWAP attachment point:

```
Controller# show capwap summary

CAPWAP Tunnels General Statistics:
  Number of Capwap Data Tunnels       = 1
  Number of Capwap Mobility Tunnels   = 0
  Number of Capwap Multicast Tunnels  = 0


Name   APName                          Type PhyPortIf Mode      McastIf
------ ------------------------------- ---- --------- --------- -------
Ca0    AP442b.03a9.6715                data Gi3/0/6   unicast   -


Name   SrcIP           SrcPort DestIP          DstPort DtlsEn MTU   Xact
------ --------------- ------- --------------- ------- ------ ----- ----
Ca0    10.10.14.32     5247    10.10.14.2      38514   No     1449  0

Controller# monitor capture mycap interface capwap 0 both
Controller# monitor capture mycap file location flash:mycap.pcap
Controller# monitor capture mycap file buffer-size 1
```

```
Controller# monitor capture mycap start

*Aug 20 11:02:21.983: %BUFCAP-6-ENABLE: Capture Point mycap enabled.on

Controller# show monitor capture mycap parameter
   monitor capture mycap interface capwap  0 in
   monitor capture mycap interface capwap  0 out
   monitor capture mycap file location flash:mycap.pcap buffer-size 1
Controller#
Controller# show monitor capture mycap

Status Information for Capture mycap
  Target Type:
  Interface: CAPWAP,
    Ingress:
 0
    Egress:
 0
   Status : Active
  Filter Details:
    Capture all packets
  Buffer Details:
   Buffer Type: LINEAR (default)
  File Details:
   Associated file name: flash:mycap.pcap
   Size of buffer(in MB): 1
  Limit Details:
   Number of Packets to capture: 0 (no limit)
   Packet Capture duration: 0 (no limit)
   Packet Size to capture: 0 (no limit)
   Packets per second: 0 (no limit)
   Packet sampling rate: 0 (no sampling)
Controller#
Controller# show monitor capture file flash:mycap.pcap
  1   0.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  2   0.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  3   2.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  4   2.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  5   3.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  6   4.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  7   4.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  8   5.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
  9   5.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
 10   6.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
 11   8.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
 Flags=........
 12    9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 13    9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 14    9.225986   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 15    9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 16    9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 17    9.231998   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 18    9.236987   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
 19  10.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
```

```
Flags=........
20  10.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
Flags=........
21  12.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
Flags=........
22  12.239993   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
23  12.244997   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
24  12.244997   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
25  12.250994   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
26  12.256990   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
27  12.262987   10.10.14.2 -> 10.10.14.32  DTLSv1.0 Application Data
28  12.499974 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
Flags=........
29  12.802012   10.10.14.3 -> 10.10.14.255 NBNS Name query NB WPAD.<00>
30  13.000000 00:00:00:00:00:00 -> 3c:ce:73:39:c6:60 IEEE 802.11 Probe Request, SN=0, FN=0,
Flags=........
```

### What to do next

You can add additional attachment points, modify the parameters of your capture point, then activate it, or if you want to use your capture point just as it is, you can now activate it.

**Note** You cannot change a capture point's parameters using the methods presented in this topic.

If the user enters an incorrect capture name, or an invalid/non existing attachment point, the switch will show errors like "*Capture Name should be less than or equal to 8 characters. Only alphanumeric characters and underscore (_) is permitted*" and "*% Invalid input detected at '^' marker*" respectively.

### Related Topics

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Adding or Modifying Capture Point Parameters

Although listed in sequence, the steps to specify values for the parameters can be executed in any order. You can also specify them in one, two, or several lines. Except for attachment points, which can be multiple, you can replace any value with a more recent value by redefining the same option. You will need to confirm interactively when certain parameters already specified are being modified.

Follow these steps to modify a capture point's parameters.

### Before you begin

A capture point must be defined before you can use these instructions.

## SUMMARY STEPS

1. **enable**
2. **monitor capture** {*capture-name*} **match** {**any** | **mac** *mac-match-string* | **ipv4** {**any** | **host** | **protocol**}{**any** | **host**} | **ipv6** {**any** | **host** | **protocol**}{**any** | **host**}}
3. **monitor capture** {*capture-name*} **limit** {[**duration** *seconds*][**packet-length** *size*][**packets** *num*]}
4. **monitor capture** {*capture-name*} **file** {**location** *filename*}
5. **monitor capture** {*capture-name*} **file** {**buffer-size** *size*}
6. **show monitor capture** {*capture-name*}[ **parameter**]
7. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **monitor capture** {*capture-name*} **match** {**any** \| **mac** *mac-match-string* \| **ipv4** {**any** \| **host** \| **protocol**}{**any** \| **host**} \| **ipv6** {**any** \| **host** \| **protocol**}{**any** \| **host**}}<br>**Example:**<br>`Controller# monitor capture mycap match ipv4 any any` | Defines the core system filter (**ipv4 any any**), defined either explicitly, through ACL or through a class map.<br>**Note** If you are defining a wireless capture point using a CAPWAP tunneling interface, this command will have no effect, so it should not be used. |
| **Step 3** | **monitor capture** {*capture-name*} **limit** {[**duration** *seconds*][**packet-length** *size*][**packets** *num*]}<br>**Example:**<br>`Controller# monitor capture mycap limit duration 60 packet-len 400` | Specifies the session limit in seconds (60), packets captured, or the packet segment length to be retained by Wireshark (400). |
| **Step 4** | **monitor capture** {*capture-name*} **file** {**location** *filename*}<br>**Example:**<br>`Controller# monitor capture mycap file location flash:mycap.pcap` | Specifies the file association, if the capture point intends to capture packets rather than only display them.<br>**Note** If the file already exists, you have to confirm if it can be overwritten.<br>**Note** File option does not exist on LAN base license. |
| **Step 5** | **monitor capture** {*capture-name*} **file** {**buffer-size** *size*}<br>**Example:**<br>`Controller# monitor capture mycap file buffer-size 100` | Specifies the size of the memory buffer used by Wireshark to handle traffic bursts. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **show monitor capture** {*capture-name*}[ **parameter**]<br>**Example:**<br>`Controller# show monitor capture mycap parameter`<br>`monitor capture mycap interface GigabitEthernet1/0/1 in`<br>`monitor capture mycap match ipv4 any any`<br>`monitor capture mycap limit duration 60 packet-len 400`<br>`monitor capture point mycap file location bootdisk:mycap.pcap`<br>`monitor capture mycap file buffer-size 100` | Displays the capture point parameters that you defined previously. |
| **Step 7** | **end**<br>**Example:**<br>`Controller(config)# end` | Returns to privileged EXEC mode. |

### Examples

#### Modifying Parameters

##### Associating or Disassociating a Capture File

```
Controller# monitor capture point mycap file location flash:mycap.pcap
Controller# no monitor capture mycap file
```

##### Specifying a Memory Buffer Size for Packet Burst Handling

```
Controller# monitor capture mycap buffer size 100
```

##### Defining an Explicit Core System Filter to Match Both IPv4 and IPv6

```
Controller# monitor capture mycap match any
```

### What to do next

if your capture point contains all of the parameters you want, activate it.

### Related Topics

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Deleting Capture Point Parameters

Although listed in sequence, the steps to delete parameters can be executed in any order. You can also delete them in one, two, or several lines. Except for attachment points, which can be multiple, you can delete any parameter.

Follow these steps to delete a capture point's parameters.

### Before you begin

A capture point parameter must be defined before you can use these instructions to delete it.

## SUMMARY STEPS

1. **enable**
2. **no monitor capture** {*capture-name*} **match**
3. **no monitor capture** {*capture-name*} **limit** [**duration**] [**packet-length**] [**packets**]
4. **no monitor capture** {*capture-name*} **file** [**location**] [**buffer-size**]
5. **show monitor capture** {*capture-name*} [ **parameter**]
6. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **no monitor capture** {*capture-name*} **match**<br>**Example:**<br>`Controller# no monitor capture mycap match` | Deletes all filters defined on capture point (mycap). |
| **Step 3** | **no monitor capture** {*capture-name*} **limit** [**duration**] [**packet-length**] [**packets**]<br>**Example:**<br>`Controller# no monitor capture mycap limit duration packet-len`<br>`Controller# no monitor capture mycap limit` | Deletes the session time limit and the packet segment length to be retained by Wireshark. It leaves other specified limits in place.<br>Deletes all limits on Wireshark. |
| **Step 4** | **no monitor capture** {*capture-name*} **file** [**location**] [**buffer-size**]<br>**Example:**<br>`Controller# no monitor capture mycap file`<br>`Controller# no monitor capture mycap file location` | Deletes the file association. The capture point will no longer capture packets. It will only display them.<br>Deletes the file location association. The file location will no longer be associated with the capture point. However, other defined fille association will be unaffected by this action. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **show monitor capture** {*capture-name*}[ **parameter**]<br>**Example:**<br>`Controller# show monitor capture mycap parameter`<br>`monitor capture mycap interface GigabitEthernet1/0/1 in` | Displays the capture point parameters that remain defined after your parameter deletion operations. This command can be run at any point in the procedure to see what parameters are associated with a capture point. |
| **Step 6** | **end**<br>**Example:**<br>`Controller(config)# end` | Returns to privileged EXEC mode. |

### What to do next

If your capture point contains all of the parameters you want, activate it.

**Note** If the parameters are deleted when the capture point is active, the switch will show an error "*Capture is active*".

### Related Topics

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Deleting a Capture Point

Follow these steps to delete a capture point.

### Before you begin

A capture point must be defined before you can use these instructions to delete it. You have to stop the capture point before you can delete it.

### SUMMARY STEPS

1. **enable**
2. **no monitor capture** {*capture-name*}
3. **show monitor capture** {*capture-name*}[ **parameter**]
4. **end**
5. **show running-config**

6. **copy running-config startup-config**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **no monitor capture** {*capture-name*}<br>**Example:**<br>`Controller# no monitor capture mycap` | Deletes the specified capture point (mycap). |
| **Step 3** | **show monitor capture** {*capture-name*}[ **parameter**]<br>**Example:**<br>`Controller# show monitor capture mycap parameter`<br>`Capture mycap does not exist` | Displays a message indicating that the specified capture point does not exist because it has been deleted. |
| **Step 4** | **end**<br>**Example:**<br>`Controller(config)# end` | Returns to privileged EXEC mode. |
| **Step 5** | **show running-config**<br>**Example:**<br>`Controller# show running-config` | Verifies your entries. |
| **Step 6** | **copy running-config startup-config**<br>**Example:**<br>`Controller# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

### What to do next

You can define a new capture point with the same name as the one you deleted. These instructions are usually performed when one wants to start over with defining a capture point.

### Related Topics

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33

# Activating and Deactivating a Capture Point

Follow these steps to activate or deactivate a capture point.

### Before you begin

A capture point can be activated even if an attachment point and a core system filter have been defined and the associated filename already exists. In such an instance, the existing file will be overwritten.

A capture point with no associated filename can only be activated to display. When the filename is not specified, the packets are captured into the buffer. Live display (display during capture) is available in both file and buffer modes.

If no display filters are specified, packets are not displayed live, and all the packets captured by the core system filter are displayed. The default display mode is brief.

**Note** When using a CAPWAP tunneling interface as an attachment point, core filters are not used, so there is no requirement to define them in this case.

### SUMMARY STEPS

1. **enable**
2. **monitor capture** {*capture-name*} **start** [**display** [**display-filter** *filter-string*]] [**brief** | **detailed** | **dump**]
3. **monitor capture** {*capture-name*} **stop**
4. **end**
5. **show running-config**
6. **copy running-config startup-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **monitor capture** {*capture-name*} **start** [**display** [**display-filter** *filter-string*]] [**brief** | **detailed** | **dump**]<br>**Example:** | Activates a capture point and filters the display, so only packets containing "stp" are displayed. |

| | Command or Action | Purpose |
|---|---|---|
| | `Controller# monitor capture mycap start display display-filter "stp"` | |
| **Step 3** | **monitor capture** {*capture-name*} **stop**<br>**Example:**<br>`Controller# monitor capture name stop` | Deactivates a capture point. |
| **Step 4** | **end**<br>**Example:**<br>`Controller(config)# end` | Returns to privileged EXEC mode. |
| **Step 5** | **show running-config**<br>**Example:**<br>`Controller# show running-config` | Verifies your entries. |
| **Step 6** | **copy running-config startup-config**<br>**Example:**<br>`Controller# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

### What to do next

While activating and deactivating a capture point, you could encounter a few errors. Here are examples of some of the possible errors.

Missing attachment point on activation

```
Switch#monitor capture mycap match any
Switch#monitor capture mycap start
No Target is attached to capture failed to disable provision featurefailed to remove
policyfailed to disable provision featurefailed to remove policyfailed to disable provision
 featurefailed to remove policy
Capture statistics collected at software (Buffer):
 Capture duration - 0 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0

Unable to activate Capture.
Switch# unable to get action unable to get action unable to get action
Switch#monitor capture mycap interface g1/0/1 both
Switch#monitor capture mycap start
Switch#
*Nov 5 12:33:43.906: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

Missing filter on activation

```
Switch#monitor capture mycap int g1/0/1 both
Switch#monitor capture mycap start
Filter not attached to capture
```

```
Capture statistics collected at software (Buffer):
 Capture duration - 0 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0

Unable to activate Capture.
Switch#monitor capture mycap match any
Switch#monitor capture mycap start
Switch#
*Nov 5 12:35:37.200: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

Attempting to activate a capture point while another one is already active

```
Switch#monitor capture mycap start
PD start invoked while previous run is active Failed to start capture : Wireshark operation
 failure
Unable to activate Capture.
Switch#show monitor capture

Status Information for Capture test
 Target Type:
 Interface: GigabitEthernet1/0/13, Direction: both
 Interface: GigabitEthernet1/0/14, Direction: both
 Status : Active
 Filter Details:
 Capture all packets
 Buffer Details:
 Buffer Type: LINEAR (default)
 Buffer Size (in MB): 10
 File Details:
 Associated file name: flash:cchh.pcap
 Limit Details:
 Number of Packets to capture: 0 (no limit)
 Packet Capture duration: 0 (no limit)
 Packet Size to capture: 0 (no limit)
 Maximum number of packets to capture per second: 1000
 Packet sampling rate: 0 (no sampling)

Status Information for Capture mycap
 Target Type:
 Interface: GigabitEthernet1/0/1, Direction: both
 Status : Inactive
 Filter Details:
 Capture all packets
 Buffer Details:
 Buffer Type: LINEAR (default)
 Buffer Size (in MB): 10
 File Details:
 File not associated
 Limit Details:
 Number of Packets to capture: 0 (no limit)
 Packet Capture duration: 0 (no limit)
 Packet Size to capture: 0 (no limit)
 Maximum number of packets to capture per second: 1000
 Packet sampling rate: 0 (no sampling)
Switch#monitor capture test stop
Capture statistics collected at software (Buffer & Wireshark):
 Capture duration - 157 seconds
 Packets received - 0
 Packets dropped - 0
 Packets oversized - 0

Switch#
*Nov 5 13:18:17.406: %BUFCAP-6-DISABLE: Capture Point test disabled.
```

```
Switch#monitor capture mycap start
Switch#
*Nov 5 13:18:22.664: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
Switch#
```

### Related Topics

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Clearing the Capture Point Buffer

Follow these steps to clear the buffer contents or save them to an external file for storage.

**Note** If you have more than one capture that is storing packets in a buffer, clear the buffer before starting a new capture to avoid memory loss. Do not try to clear buffer on an active capture point.

**Note** Clearing buffer on an active capture point is supported only on Lan Base as this only clears the content. On all other licenses, it deletes the buffer itself, hence cannot be run during active capture.

### SUMMARY STEPS

1. **enable**
2. **monitor capture** {*capture-name*} [**clear** | **export** *filename*]
3. **end**
4. **show running-config**
5. **copy running-config startup-config**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Controller> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **monitor capture** {*capture-name*} [**clear** \| **export** *filename*]<br>**Example:**<br>`Controller# monitor capture mycap clear` | Clear - Completely deletes the buffer.<br>**Note** When the clear command is run,<br>• On Lan base - the command clears the buffer contents without deleting the buffer<br>• On all other licenses - the command deletes the buffer itself.<br>Export - Saves the captured packets in the buffer as well as deletes the buffer. |
| **Step 3** | **end**<br>**Example:**<br>`Controller(config)# end` | Returns to privileged EXEC mode. |
| **Step 4** | **show running-config**<br>**Example:**<br>`Controller# show running-config` | Verifies your entries. |
| **Step 5** | **copy running-config startup-config**<br>**Example:**<br>`Controller# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

### Examples: Capture Point Buffer Handling

#### Exporting Capture to a File

```
Controller# monitor capture mycap export flash:mycap.pcap

Storage configured as File for this capture
```

#### Clearing Capture Point Buffer

```
Controller# monitor capture mycap clear

 Capture configured with file options
```

### What to do next

**Note** If you try to clear the capture point buffer on licenses other than LAN Base, the switch will show an error "*Failed to clear capture buffer : Capture Buffer BUSY*".

**Related Topics**

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4
Example: Simple Capture and Display, on page 31
Example: Simple Capture and Store, on page 33
Example: Using Buffer Capture, on page 35
Example: Capture Sessions, on page 42
Example: Capture and Store in Lock-step Mode, on page 43
Example: Simple Capture and Store of Packets in Egress Direction, on page 44

# Monitoring Wireshark

The commands in this table are used to monitor Wireshark.

| Command | Purpose |
|---|---|
| **show monitor capture** [*capture-name* ] | Displays the capture point state so that you can see what capture points are defined, what their attributes are, and whether they are active. When capture point name is specified, it displays specific capture point's details. |
| **show monitor capture** [*capture-name* **parameter**] | Displays the capture point parameters. |
| **show capwap summary** | Displays all the CAPWAP tunnels on the controller. Use this command to determine which CAPWAP tunnels are available to use for a wireless capture. |

# Configuration Examples for Wireshark

## Example: Displaying a Brief Output from a .pcap File

You can display the output from a .pcap file by entering:

```
Controller# show monitor capture file flash:mycap.pcap brief
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1 0.000000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=0/0, ttl=254
  2 0.000051000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=0/0, ttl=255 (request in 1)
  3 0.000908000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=1/256, ttl=254
  4 0.001782000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=1/256, ttl=255 (request in 3)
  5 0.002961000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
```

```
seq=2/512, ttl=254
  6 0.003676000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=2/512, ttl=255 (request in 5)
  7 0.004835000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=3/768, ttl=254
  8 0.005579000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=3/768, ttl=255 (request in 7)
  9 0.006850000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=4/1024, ttl=254
 10 0.007586000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=4/1024, ttl=255 (request in 9)
 11 0.008768000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=5/1280, ttl=254
 12 0.009497000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=5/1280, ttl=255 (request in 11)
 13 0.010695000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=6/1536, ttl=254
 14 0.011427000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=6/1536, ttl=255 (request in 13)
 15 0.012728000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=7/1792, ttl=254
 16 0.013458000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=7/1792, ttl=255 (request in 15)
 17 0.014652000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=8/2048, ttl=254
 18 0.015394000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=8/2048, ttl=255 (request in 17)
 19 0.016682000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=9/2304, ttl=254
 20 0.017439000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=9/2304, ttl=255 (request in 19)
 21 0.018655000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=10/2560, ttl=254
 22 0.019385000   10.10.10.1 -> 10.10.10.2   ICMP 114 Echo (ping) reply    id=0x002e,
seq=10/2560, ttl=255 (request in 21)
 23 0.020575000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x002e,
seq=11/2816, ttl=254
 --More‹
```

# Example: Displaying Detailed Output from a .pcap File

You can display the detailed .pcap file output by entering:

```
Controller# show monitor capture file flash:mycap.pcap detailed
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

Frame 1: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface 0
    Interface id: 0
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov  6, 2015 11:44:48.322497000 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1446810288.322497000 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 114 bytes (912 bits)
    Capture Length: 114 bytes (912 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
```

```
    [Protocols in frame: eth:ip:icmp:data]
Ethernet II, Src: Cisco_f3:63:46 (00:e1:6d:f3:63:46), Dst: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)

    Destination: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)
        Address: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: Cisco_f3:63:46 (00:e1:6d:f3:63:46)
        Address: Cisco_f3:63:46 (00:e1:6d:f3:63:46)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 10.10.10.2 (10.10.10.2), Dst: 10.10.10.1 (10.10.10.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not
ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport)
(0x00)
    Total Length: 100
    Identification: 0x04ba (1210)
    Flags: 0x00
        0... .... = Reserved bit: Not set
        .0.. .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 254
    Protocol: ICMP (1)
    Header checksum: 0x8fc8 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 10.10.10.2 (10.10.10.2)
    Destination: 10.10.10.1 (10.10.10.1)
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xe4db [correct]
    Identifier (BE): 46 (0x002e)
    Identifier (LE): 11776 (0x2e00)
    Sequence number (BE): 0 (0x0000)
    Sequence number (LE): 0 (0x0000)
    Data (72 bytes)

0000  00 00 00 00 09 c9 8f 77 ab cd ab cd ab cd ab cd   .......w........
0010  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0020  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0030  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0040  ab cd ab cd ab cd ab cd                           ........
        Data: 0000000009c98f77abcdabcdabcdabcdabcdabcdabcd...
        [Length: 72]

Frame 2: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface 0
    Interface id: 0
```

# Example: Simple Capture and Display

This example shows how to monitor traffic in the Layer 3 interface Gigabit Ethernet 1/0/1:

**Step 1**: Define a capture point to match on the relevant traffic by entering:

```
Controller# monitor capture mycap interface GigabitEthernet1/0/3 in
Controller# monitor capture mycap match ipv4 any any
Controller# monitor capture mycap limit duration 60 packets 50
Controller# monitor capture mycap buffer size 100
```

To avoid high CPU utilization, a low packet count and duration as limits has been set.

**Step 2**: Confirm that the capture point has been correctly defined by entering:

```
Controller# show monitor capture mycap parameter
      monitor capture mycap interface GigabitEthernet1/0/3 in
   monitor capture mycap match ipv4  any any
   monitor capture mycap buffer size 100
   monitor capture mycap limit packets 50 duration 60

Controller# show monitor capture mycap
Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/3, Direction: in
   Status : Inactive
  Filter Details:
   IPv4
    Source IP:  any
    Destination IP:  any
   Protocol: any
  Buffer Details:
   Buffer Type: LINEAR (default)
   Buffer Size (in MB): 100
  File Details:
   File not associated
  Limit Details:
   Number of Packets to capture: 50
   Packet Capture duration: 60
   Packet Size to capture: 0 (no limit)
   Packet sampling rate: 0 (no sampling)
```

**Step 3**: Start the capture process and display the results.

```
Controller# monitor capture mycap start display
 Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1   0.000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030, seq=0/0,
 ttl=254
  2   0.003682   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=1/256, ttl=254
  3   0.006586   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=2/512, ttl=254
  4   0.008941   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=3/768, ttl=254
  5   0.011138   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=4/1024, ttl=254
  6   0.014099   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=5/1280, ttl=254
  7   0.016868   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=6/1536, ttl=254
  8   0.019210   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=7/1792, ttl=254
  9   0.024785   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0030,
seq=8/2048, ttl=254
 --More--
```

**Step 4**: Delete the capture point by entering:

```
Controller# no monitor capture mycap
```

**Note** A **stop** command is not required in this particular case since we have set a limit and the capture will automatically stop once that limit is reached.

For more information on syntax to be used for pcap statistics, refer the "*Additional References*" section.

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Example: Simple Capture and Store

This example shows how to capture packets to a filter:

**Step 1**: Define a capture point to match on the relevant traffic and associate it to a file by entering:

```
Controller# monitor capture mycap interface GigabitEthernet1/0/3 in
Controller# monitor capture mycap match ipv4 any any
Controller# monitor capture mycap limit duration 60 packets 50
Controller# monitor capture mycap file location flash:mycap.pcap
```

**Step 2**: Confirm that the capture point has been correctly defined by entering:

```
Controller# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/3 in
   monitor capture mycap match ipv4  any any
   monitor capture mycap file location flash:mycap.pcap
   monitor capture mycap limit packets 50 duration 60

Controller# show monitor capture mycap

Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/3, Direction: in
   Status : Inactive
  Filter Details:
   IPv4
    Source IP:  any
    Destination IP:  any
   Protocol: any
  Buffer Details:
   Buffer Type: LINEAR (default)
  File Details:
   Associated file name: flash:mycap.pcap
```

```
  Limit Details:
   Number of Packets to capture: 50
   Packet Capture duration: 60
   Packet Size to capture: 0 (no limit)
   Packet sampling rate: 0 (no sampling)
```

**Step 3**: Launch packet capture by entering:

```
Controller# monitor capture mycap start
```

**Step 4**: Display extended capture statistics during runtime by entering:

```
Controller#  show monitor capture mycap capture-statistics
Capture statistics collected at software:
        Capture duration - 15 seconds
        Packets received - 40
        Packets dropped - 0
        Packets oversized - 0
        Packets errored - 0
        Packets sent - 40
        Bytes received - 7280
        Bytes dropped - 0
        Bytes oversized - 0
        Bytes errored - 0
        Bytes sent ¬ 4560
```

**Step 5**: After sufficient time has passed, stop the capture by entering:

```
Controller# monitor capture mycap stop
Capture statistics collected at software (Buffer & Wireshark):
        Capture duration - 20 seconds
        Packets received - 50
        Packets dropped - 0
        Packets oversized - 0
```

**Note** Alternatively, you could allow the capture operation stop automatically after the time has elapsed or the packet count has been met.

The mycap.pcap file now contains the captured packets.

**Step 6**: Display extended capture statistics after stop by entering:

```
Controller#  show monitor capture mycap capture-statistics
Capture statistics collected at software:
        Capture duration - 20 seconds
        Packets received - 50
        Packets dropped - 0
        Packets oversized - 0
        Packets errored - 0
        Packets sent - 50
        Bytes received - 8190
        Bytes dropped - 0
        Bytes oversized - 0
        Bytes errored - 0
        Bytes sent ¬ 5130
```

**Step 7**: Display the packets by entering:

```
Controller# show monitor capture file flash:mycap.pcap
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1 0.000000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=0/0, ttl=254
  2 0.002555000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=1/256, ttl=254
  3 0.006199000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=2/512, ttl=254
  4 0.009199000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=3/768, ttl=254
  5 0.011647000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=4/1024, ttl=254
  6 0.014168000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=5/1280, ttl=254
  7 0.016737000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=6/1536, ttl=254
  8 0.019403000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=7/1792, ttl=254
  9 0.022151000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=8/2048, ttl=254
 10 0.024722000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=9/2304, ttl=254
 11 0.026890000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=10/2560, ttl=254
 12 0.028862000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0031,
seq=11/2816, ttl=254
 --More--
```

For more information on syntax to be used for pcap statistics, refer the "*Additional References*" section.

**Step 8**: Delete the capture point by entering:

```
Controller# no monitor capture mycap
```

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Example: Using Buffer Capture

This example shows how to use buffer capture:

**Step 1**: Launch a capture session with the buffer capture option by entering:

```
Controller# monitor capture mycap interface GigabitEthernet1/0/3 in
Controller# monitor capture mycap match ipv4 any any
```

```
Controller# monitor capture mycap buffer circular size 1
Controller# monitor capture mycap start
```

**Step 2**: Determine whether the capture is active by entering:

```
Controller# show monitor capture mycap
Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/3, Direction: in
   Status : Active
  Filter Details:
   IPv4
    Source IP:  any
    Destination IP:  any
   Protocol: any
  Buffer Details:
   Buffer Type: CIRCULAR
   Buffer Size (in MB): 1
  File Details:
   File not associated
  Limit Details:
   Number of Packets to capture: 0 (no limit)
   Packet Capture duration: 0 (no limit)
   Packet Size to capture: 0 (no limit)
   Maximum number of packets to capture per second: 1000
   Packet sampling rate: 0 (no sampling)
```

**Step 3**: Display extended capture statistics during runtime by entering:

```
Controller# show monitor capture mycap capture-statistics
Capture statistics collected at software:
        Capture duration - 88 seconds
        Packets received - 1000
        Packets dropped - 0
        Packets oversized - 0
        Packets errored - 0
        Packets sent - 1000
        Bytes received - 182000
        Bytes dropped - 0
        Bytes oversized - 0
        Bytes errored - 0
        Bytes sent - 114000
```

**Step 4**: Stop the capture by entering:

```
Controller# monitor capture mycap stop
Capture statistics collected at software (Buffer):
        Capture duration - 2185 seconds
        Packets received - 51500
        Packets dropped - 0
        Packets oversized - 0
```

**Step 5**: Display extended capture statistics after stop by entering:

```
Controller# show monitor capture mycap capture-statistics
Capture statistics collected at software:
        Capture duration - 156 seconds
        Packets received - 2000
        Packets dropped - 0
        Packets oversized - 0
        Packets errored - 0
        Packets sent - 2000
        Bytes received - 364000
        Bytes dropped - 0
```

```
        Bytes oversized - 0
        Bytes errored - 0
        Bytes sent - 228000
```

**Step 6**: Determine whether the capture is active by entering:

```
Controller# show monitor capture mycap
Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/3, Direction: in
   Status : Inactive
  Filter Details:
   IPv4
    Source IP:  any
    Destination IP:  any
   Protocol: any
  Buffer Details:
   Buffer Type: CIRCULAR
   Buffer Size (in MB): 1
  File Details:
   File not associated
  Limit Details:
   Number of Packets to capture: 0 (no limit)
   Packet Capture duration: 0 (no limit)
   Packet Size to capture: 0 (no limit)
   Maximum number of packets to capture per second: 1000
   Packet sampling rate: 0 (no sampling)
```

**Step 7**: Display the packets in the buffer by entering:

```
Controller# show monitor capture mycap buffer brief
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1   0.000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40057/31132, ttl=254
  2   0.000030   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40058/31388, ttl=254
  3   0.000052   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40059/31644, ttl=254
  4   0.000073   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40060/31900, ttl=254
  5   0.000094   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40061/32156, ttl=254
  6   0.000115   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40062/32412, ttl=254
  7   0.000137   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40063/32668, ttl=254
  8   0.000158   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40064/32924, ttl=254
  9   0.000179   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40065/33180, ttl=254
 10   0.000200   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40066/33436, ttl=254
 11   0.000221   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40067/33692, ttl=254
 12   0.000243   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0038,
seq=40068/33948, ttl=254
 --More--
```

Notice that the packets have been buffered.

**Step 8**: Display the packets in other display modes.

```
Controller# show monitor capture mycap buffer detailed
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

Frame 1: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface 0
    Interface id: 0
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov  6, 2015 18:10:06.297972000 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1446833406.297972000 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 114 bytes (912 bits)
    Capture Length: 114 bytes (912 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ip:icmp:data]
Ethernet II, Src: Cisco_f3:63:46 (00:e1:6d:f3:63:46), Dst: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)

    Destination: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)
        Address: Cisco_31:f1:c6 (00:e1:6d:31:f1:c6)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: Cisco_f3:63:46 (00:e1:6d:f3:63:46)
        Address: Cisco_f3:63:46 (00:e1:6d:f3:63:46)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 10.10.10.2 (10.10.10.2), Dst: 10.10.10.1 (10.10.10.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not
ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport)
(0x00)
    Total Length: 100
    Identification: 0xabdd (43997)
    Flags: 0x00
        0... .... = Reserved bit: Not set
        .0.. .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 254
    Protocol: ICMP (1)
    Header checksum: 0xe8a4 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 10.10.10.2 (10.10.10.2)
    Destination: 10.10.10.1 (10.10.10.1)
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xa620 [correct]
    Identifier (BE): 56 (0x0038)
    Identifier (LE): 14336 (0x3800)
    Sequence number (BE): 40057 (0x9c79)
    Sequence number (LE): 31132 (0x799c)
    Data (72 bytes)

0000  00 00 00 00 0b 15 30 63 ab cd ab cd ab cd ab cd   ......0c........
0010  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0020  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
```

```
0030  ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ..............
0040  ab cd ab cd ab cd ab cd                     ........
       Data: 000000000b153063abcdabcdabcdabcdabcdabcdabcd...
       [Length: 72]

Frame 2: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface 0
```

```
Controller# show monitor capture mycap buffer dump
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

0000  00 e1 6d 31 f1 c6 00 e1 6d f3 63 46 08 00 45 00   ..m1....m.cF..E.
0010  00 64 ab dd 00 00 fe 01 e8 a4 0a 0a 0a 02 0a 0a   .d..............
0020  0a 01 08 00 a6 20 00 38 9c 79 00 00 00 00 0b 15   ..... .8.y......
0030  30 63 ab cd ab cd ab cd ab cd ab cd ab cd ab cd   0c..............
0040  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0050  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0060  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0070  ab cd                                            ..

0000  00 e1 6d 31 f1 c6 00 e1 6d f3 63 46 08 00 45 00   ..m1....m.cF..E.
0010  00 64 ab de 00 00 fe 01 e8 a3 0a 0a 0a 02 0a 0a   .d..............
0020  0a 01 08 00 a6 1d 00 38 9c 7a 00 00 00 00 0b 15   .......8.z......
0030  30 65 ab cd ab cd ab cd ab cd ab cd ab cd ab cd   0e..............
0040  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0050  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0060  ab cd ab cd ab cd ab cd ab cd ab cd ab cd ab cd   ................
0070  ab cd
```

**Step 9**: Clear the buffer by entering:

```
Controller# monitor capture mycap clear
```

**Note** NOTE - Clearing the buffer deletes the buffer along with the contents.

**Note** If you require the buffer contents to be displayed, run the clear commands after show commands.

**Step 10**: Restart the traffic, wait for 10 seconds, then display the buffer contents by entering:

**Note** We cannot run show from buffer during an active capture. Capture should be stopped before running show from buffer. We can however run a show on a pcap file during an active capture in both file and buffer mode. In file mode, we can display the packets in the current capture session's pcap file as well when the capture is active.

```
Controller# monitor capture mycap start
Switch#  show monitor capture mycap

Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/3, Direction: in
   Status : Active
  Filter Details:
```

```
   IPv4
    Source IP:  any
    Destination IP:  any
   Protocol: any
  Buffer Details:
   Buffer Type: CIRCULAR
   Buffer Size (in MB): 1
  File Details:
   File not associated
  Limit Details:
   Number of Packets to capture: 0 (no limit)
   Packet Capture duration: 0 (no limit)
   Packet Size to capture: 0 (no limit)
   Maximum number of packets to capture per second: 1000
   Packet sampling rate: 0 (no sampling)
```

**Step 11**: Stop the packet capture and display the buffer contents by entering:

```
Controller# monitor capture mycap stop
Capture statistics collected at software (Buffer):
 Capture duration - 111 seconds
 Packets received - 5000
 Packets dropped - 0
 Packets oversized - 0
```

**Step 12**: Determine whether the capture is active by entering:

```
Controller# show monitor capture mycap
Status Information for Capture mycap
 Target Type:
  Interface: GigabitEthernet1/0/3, Direction: in
 Status : Inactive
Filter Details:
 IPv4
  Source IP: any
  Destination IP: any
 Protocol: any
Buffer Details:
 Buffer Type: CIRCULAR
 Buffer Size (in MB): 1
File Details:
 File not associated
Limit Details:
 Number of Packets to capture: 0 (no limit)
 Packet Capture duration: 0 (no limit)
 Packet Size to capture: 0 (no limit)
 Maximum number of packets to capture per second: 1000
 Packet sampling rate: 0 (no sampling)
```

**Step 13**: Display the packets in the buffer by entering:

```
Controller# show monitor capture mycap buffer brief
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1 0.000000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=0/0, ttl=254
  2 0.000030000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=1/256, ttl=254
  3 0.000051000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=2/512, ttl=254
  4 0.000072000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=3/768, ttl=254
  5 0.000093000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=4/1024, ttl=254
```

```
  6 0.000114000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=5/1280, ttl=254
  7 0.000136000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=6/1536, ttl=254
  8 0.000157000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=7/1792, ttl=254
  9 0.000178000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=8/2048, ttl=254
 10 0.000199000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=9/2304, ttl=254
 11 0.000220000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=10/2560, ttl=254
 12 0.000241000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=11/2816, ttl=254
 --More‹
```

**Step 14**: Store the buffer contents to the mycap.pcap file in the internal flash: storage device by entering:

```
Controller# monitor capture mycap export flash:mycap.pcap
Exported Successfully
```

**Note** The current implementation of export is such that when the command is run, export is "started" but not complete when it returns the prompt to the user. So we have to wait for a message display on the console from Wireshark before it can run a display of packets in the file.

**Step 15**: Display capture packets from the file by entering:

```
Controller#  show monitor capture file flash:mycap.pcap
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1 0.000000000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=0/0, ttl=254
  2 0.000030000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=1/256, ttl=254
  3 0.000051000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=2/512, ttl=254
  4 0.000072000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=3/768, ttl=254
  5 0.000093000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=4/1024, ttl=254
  6 0.000114000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=5/1280, ttl=254
  7 0.000136000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=6/1536, ttl=254
  8 0.000157000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=7/1792, ttl=254
  9 0.000178000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=8/2048, ttl=254
 10 0.000199000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=9/2304, ttl=254
 11 0.000220000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=10/2560, ttl=254
 12 0.000241000   10.10.10.2 -> 10.10.10.1   ICMP 114 Echo (ping) request  id=0x0039,
seq=11/2816, ttl=254
 --More--
```

**Step 16**: Delete the capture point by entering:

```
Controller# no monitor capture mycap
```

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Example: Capture Sessions

```
Controller# monitor capture mycap start display display-filter "stp"
  0.000000 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
  2.000992 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
   2.981996 20:37:06:cf:08:b6 -> 20:37:06:cf:08:b6 LOOP Reply
  4.000992 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
  6.000000 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
  7.998001 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
  9.998001 20:37:06:cf:08:b6 -> 01:80:c2:00:00:00 STP Conf. Root = 32768/100/20:37:06:ce:f0:80
   Cost = 0  Port = 0x8136
Capture test is not active Failed to Initiate Wireshark
Controller# show monitor capture mycap parameter
   monitor capture mycap control-plane both
   monitor capture mycap match any
   monitor capture mycap file location flash:mycap1.1 buffer-size 90
   monitor capture mycap limit duration 10

Controller# no monitor capture mycap file
Controller# monitor capture mycap start display display-filter "udp.port == 20002" dump
 Please associate capture file/buffer
Unable to activate Capture.

Controller# monitor capture mycap start display display-filter "udp.port == 20002"
 Please associate capture file/buffer
Unable to activate Capture.

Controller# monitor capture mycap start display detailed
 Please associate capture file/buffer
Unable to activate Capture.
```

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22

Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Example: Capture and Store in Lock-step Mode

This example captures live traffic and stores the packets in lock-step mode.

**Note** The capture rate might be slow for the first 15 seconds. If possible and necessary, start the traffic 15 seconds after the capture session starts.

**Step 1**: Define a capture point to match on the relevant traffic and associate it to a file by entering:

```
Controller# monitor capture mycap interface GigabitEthernet1/0/1 in
Controller# monitor capture mycap match ipv4 any any
Controller# monitor capture mycap limit duration 60 packets 100
Controller# monitor capture mycap file location flash:mycap.pcap buffer-size 64
```

**Step 2**: Confirm that the capture point has been correctly defined by entering:

```
Controller# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/1 in
   monitor capture mycap file location flash:mycap.pcap buffer-size 64
   monitor capture mycap limit packets 100 duration 60

Controller# show monitor capture mycap

Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/1, Direction: in
   Status : Inactive
  Filter Details:
   Filter not attached
  Buffer Details:
   Buffer Type: LINEAR (default)
  File Details:
   Associated file name: flash:mycap.pcap
   Size of buffer(in MB): 64
  Limit Details:
   Number of Packets to capture: 100
   Packet Capture duration: 60
   Packet Size to capture: 0 (no limit)
   Packets per second: 0 (no limit)
   Packet sampling rate: 0 (no sampling)
```

**Step 3**: Launch packet capture by entering:

```
Controller# monitor capture mycap start
A file by the same capture file name already exists, overwrite?[confirm]
Turning on lock-step mode

Controller#
*Oct 14 09:35:32.661: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

**Step 4**: Display the packets by entering:

```
Controller# show monitor capture file flash:mycap.pcap
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  0.000000    10.1.1.30 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  1.000000    10.1.1.31 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  2.000000    10.1.1.32 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  3.000000    10.1.1.33 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  4.000000    10.1.1.34 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  5.000000    10.1.1.35 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  6.000000    10.1.1.36 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  7.000000    10.1.1.37 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  8.000000    10.1.1.38 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  9.000000    10.1.1.39 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
```

**Step 5**: Delete the capture point by entering:

```
Controller# no monitor capture mycap
```

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27
How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Example: Simple Capture and Store of Packets in Egress Direction

This example shows how to capture packets to a filter:

**Step 1**: Define a capture point to match on the relevant traffic and associate it to a file by entering:

```
Controller# monitor capture mycap interface Gigabit 1/0/1 out match ipv4 any any
Controller# monitor capture mycap limit duration 60 packets 100
Controller# monitor capture mycap file location flash:mycap.pcap buffer-size 90
```

**Step 2**: Confirm that the capture point has been correctly defined by entering:

```
Controller# show monitor capture mycap parameter
   monitor capture mycap interface GigabitEthernet1/0/1 out
   monitor capture mycap match ipv4  any any
   monitor capture mycap file location flash:mycap.pcap buffer-size 90
   monitor capture mycap limit packets 100 duration 60

Controller# show monitor capture mycap

Status Information for Capture mycap
  Target Type:
   Interface: GigabitEthernet1/0/1, Direction: out
   Status : Inactive
  Filter Details:
   IPv4
    Source IP:  any
    Destination IP:  any
```

```
  Protocol: any
 Buffer Details:
  Buffer Type: LINEAR (default)
 File Details:
  Associated file name: flash:mycap.pcap
  Size of buffer(in MB): 90
 Limit Details:
  Number of Packets to capture: 100
  Packet Capture duration: 60
  Packet Size to capture: 0 (no limit)
  Packets per second: 0 (no limit)
  Packet sampling rate: 0 (no sampling)
```

**Step 3**: Launch packet capture by entering:

```
Controller# monitor capture mycap start
A file by the same capture file name already exists, overwrite?[confirm]
Turning on lock-step mode

Controller#
*Oct 14 09:35:32.661: %BUFCAP-6-ENABLE: Capture Point mycap enabled.
```

**Note** Allow the capture operation stop automatically after the time has elapsed or the packet count has been met. When you see the following message in the output, will know that the capture operation has stopped:

```
*Oct 14 09:36:34.632: %BUFCAP-6-DISABLE_ASYNC: Capture Point mycap disabled. Rea
son : Wireshark Session Ended
```

The mycap.pcap file now contains the captured packets.

**Step 4**: Display the packets by entering:

```
Controller# show monitor capture file flash:mycap.pcap
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  0.000000    10.1.1.30 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  1.000000    10.1.1.31 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  2.000000    10.1.1.32 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  3.000000    10.1.1.33 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  4.000000    10.1.1.34 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  5.000000    10.1.1.35 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  6.000000    10.1.1.36 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  7.000000    10.1.1.37 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  8.000000    10.1.1.38 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
  9.000000    10.1.1.39 -> 20.1.1.2     UDP Source port: 20001  Destination port: 20002
```

**Step 5**: Delete the capture point by entering:

```
Controller# no monitor capture mycap
```

### Related Topics

Defining a Capture Point, on page 14
Adding or Modifying Capture Point Parameters, on page 18
Deleting Capture Point Parameters, on page 21
Deleting a Capture Point, on page 22
Activating and Deactivating a Capture Point, on page 24
Clearing the Capture Point Buffer, on page 27

How to Configure Wireshark, on page 13
Capture Points, on page 4
Attachment Points, on page 4

# Additional References

### Related Documents

| Related Topic | Document Title |
|---|---|
| General Packet Filtering | For general packet filtering, refer to:<br>Display Filter Reference |

### Error Message Decoder

| Description | Link |
|---|---|
| To help you research and resolve system error messages in this release, use the Error Message Decoder tool. | https://www.cisco.com/cgi-bin/Support/Errordecoder/index.cgi |

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| None | - |

### MIBs

| MIB | MIBs Link |
|---|---|
| All the supported MIBs for this release. | To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

### 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.<br><br>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.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

### Related Topics

Filters, on page 5

# Feature History and Information for WireShark

| Release | Modification |
|---|---|
| Cisco IOS XE 3.3SE | This feature was introduced. |