Revised: November 11, 2025

Flexible NetFlow

Feature history for Flexible NetFlow

This table provides release and platform support information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature name and description	Supported platform
Cisco IOS XE 17.18.1	Flexible NetFlow: Flexible NetFlow is a network traffic monitoring and analysis tool that provides detailed statistics for accounting, network monitoring, and network planning.	Cisco C9350 Series Smart Switches

Understand Flexible NetFlow

Flexible NetFlow is a network traffic monitoring and analysis tool that provides detailed statistics for accounting, network monitoring, and network planning. It enhances network anomaly and security detection by allowing users to define custom flow records based on various packet fields.

Key concepts

- **Flow:** A unidirectional stream of packets sharing the same key values arriving on a source interface.
- **Key:** A specific field within a packet used to identify a flow (example, source/destination IP address, port).

  Key fields are identified by the **match** parameters in a flow record.
- **Nonkey field:** Additional fields of interest gathered for a flow but do not define the flow itself (example, packet counters).

  Nonkey fields are identified by the **collect** parameters in a flow record.
- **Flow record:** A combination of key and nonkey fields that defines what data Flexible NetFlow collects.
- **Flow monitor:** Defines the size of the data to collect for a flow, combining the flow record and exporter with Flexible NetFlow cache information.
- **Flow exporter:** Exports the data gathered by Flexible NetFlow to a remote system (example, NetFlow collector).
- **Flow collector:** Receive, process, store, and analyze NetFlow data.
- **Flow sampler:** Reduces the load on the device by limiting the number of packets or flows selected for analysis.

Benefits of Flexible NetFlow

The benefits of Flexible NetFlow include:

- High-capacity flow recognition, including scalability and aggregation of flow information.
- Enhanced flow infrastructure for security monitoring and DDoS detection and identification.

- New information from packets to adapt flow information to a particular service or operation in the network. The flow information available are customizable.
- Includes a comprehensive IP accounting feature that can replace various accounting functionalities, including IP accounting, Border Gateway Protocol (BGP) Policy Accounting, and persistent caches.

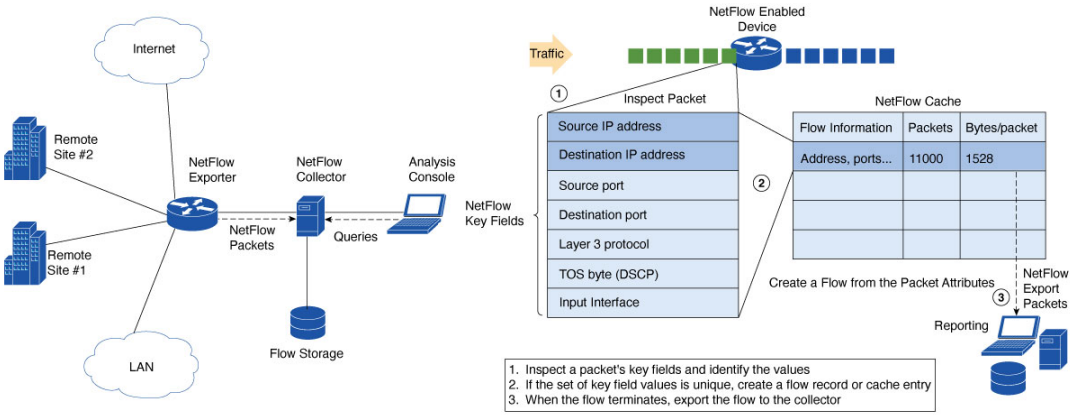
Applications of Flexible NetFlow

Flexible NetFlow helps efficiently understand network behavior by tailoring flow information for various services, as mentioned below:

- Flexible NetFlow enhances NetFlow as a security monitoring tool. For instance, new flow keys can be defined for packet length or MAC address, allowing users to search for a specific type of attack in the network.
- With Flexible NetFlow, track TCP or UDP applications based on the class of service (CoS) in the packets to quickly identify the amount of application traffic sent between hosts.

The figure demonstrates how Flexible NetFlow might be deployed in a network.

Figure 1: Typical Deployment for Flexible NetFlow



Flexible NetFlow components

Flexible NetFlow consists of several components for traffic analysis and data export. It enables user-defined flow records and component structures, which facilitate various configurations on a networking device with minimal commands. Configure each flow monitor using a unique combination of flow record, flow exporter, and cache type. If you change a parameter such as the destination IP address for a flow exporter, it automatically updates all flow monitors that use the flow exporter. The same flow monitor can be utilized with various flow samplers to sample identical types of network traffic, at different rates, and across different interfaces.

Flexible NetFlow components consists of flow records, flow exporters, flow monitors, flow samplers, and target interfaces which work together to define, collect, and export flow data for network traffic analysis.

Flow records

In Flexible NetFlow a combination of key and nonkey fields is called a record. A flow record specifies the keys that Flexible NetFlow uses to identify packets and the additional fields that it gathers for the flow. You can define a flow record with any combination of keys and fields of interest. The device supports a wide set of keys. A flow record also defines the types of counters gathered per flow. You can configure 64-bit packet or byte counters.

Assign Flexible NetFlow records to flow monitors to define the cache used for storing flow data. The device enables these default match fields when you create a flow record:

- **match datalink**: Layer 2 attributes
- **match flow direction**: Fields identifying the direction of flow
- **match interface**: Interface attributes
- **match ipv4**: IPv4 attributes
- **match ipv6**: IPv6 attributes
- **match transport**: Transport layer fields

Flexible NetFlow flow records are user-defined custom templates defined using two key sets of parameters: **match parameters** and **collect parameters**. These records define the specific fields of information to collect about network traffic flows, and the parameters determine the fields that are monitored and exported for network traffic flows.

User-defined records

Records you define for a Flexible NetFlow flow monitor cache are called user-defined records. Flexible NetFlow lets you define your own records for a flow monitor cache. Specify key and nonkey fields to customize data collection to your needs.

Nonkey fields provide extra information about traffic in flows. A change in the value of a nonkey field does not create a new flow. In most cases the values for nonkey fields are taken from only the first packet in the flow. Flexible NetFlow lets you capture counter values like packets as nonkey fields.

You can create user-defined records for applications such as QoS and bandwidth monitoring, application and end user traffic profiling, and security monitoring for DDoS attacks. Flexible NetFlow user-defined records provide the capability to monitor a contiguous section of a packet of a user-configurable size, and use it in a flow record as a key or a nonkey field along with other fields and attributes of the packet. The section may include any Layer 3 data from the packet.

The packet section fields allow you to monitor any packet fields that are not covered by the Flexible NetFlow predefined keys. The ability to analyze packet fields enables more detailed traffic monitoring, facilitates the investigation of DDoS attacks, and supports implementing other security applications such as URL monitoring.

Match parameters

Match parameters define the fields that uniquely identify a flow. Only packets with identical match field values are grouped into the same flow. Configure at least one match parameter for the flow records.

This table describes Flexible NetFlow match parameters.

Table 1: Match Parameters

Command	Purpose
match datalink {**dot1q** | **ethertype** | **mac** | **vlan** }	Specifies a match to datalink or Layer 2 fields. • **dot1q**: Matches to the dot1q field. • **ethertype**: Matches to the ethertype of the packet. • **mac**: Matches to the source or destination MAC fields. • **vlan**: Matches to the VLAN that the packet is located on (input or output).

Command	Purpose
match flow direction	Specifies a match to the flow identifying fields.
match interface {**input** | **output**}	Specifies a match to the interface fields. • **input**: Matches to the input interface. • **output**: Matches to the output interface.
match ipv4 {**destination** | **protocol** | **source** | **tos** | **ttl** | **version**}	Specifies a match to the IPv4 fields. • **destination**: Matches to the IPv4 destination address-based fields. • **protocol**: Matches to the IPv4 protocols. • **source**: Matches to the IPv4 source address based fields. • **tos**: Matches to the IPv4 Type of Service fields. • **ttl**: Matches to the IPv4 time to live fields. • **version**: Matches to the IP version from the IPv4 header.
match ipv6 {**destination** | **hop-limit** | **protocol** | **source** | **traffic-class** | **version** }	Specifies a match to the IPv6 fields. • **destination**: Matches to the IPv6 destination address-based fields. • **hop-limit**: Matches to the IPv6 hop limit fields. • **protocol**: Matches to the IPv6 payload protocol fields. • **source**: Matches to the IPv6 source address based fields. • **traffic-class**: Matches to the IPv6 traffic class. • **version**: Matches to the IP version from the IPv6 header.
match transport {**destination-port** | **igmp** | **icmp** | **source-port**}	Specifies a match to the Transport Layer fields. • **destination-port**: Matches to the transport destination port. • **icmp**: Matches to ICMP fields, including ICMP IPv4 and IPv6 fields. • **igmp**: Matches to IGMP fields. • **source-port**: Matches to the transport source port.
match routing vrf input	Specifies a match to the VRF routing attributes for incoming packets.

Collect parameters

Collect parameters are optional fields that provide additional information about the flow. These fields are collected and included in the exported flow record but do not impact how flows are grouped.

This table describes Flexible NetFlow collect parameters.

Table 2: Collect parameters

Command	Purpose
collect counter packets [**long**]	Collects the total counter fields. • **packets long**: Total number of packets (64 bit counter).
collect timestamp absolute {**first** | **last**}	Collects the fields for the absolute time the first packet was seen or the absolute time the most recent packet was last seen (in milliseconds).
collect transport tcp flags	Collects transport TCP flags. • **ack**: TCP acknowledgement flag • **cwr**: TCP congestion window reduced flag • **ece**: TCP ECN echo flag • **fin**: TCP finish flag • **psh**: TCP push flag • **rst**: TCP reset flag • **syn**: TCP synchronize flag • **urg**: TCP urgent flag **Note** On the device, all TCP flags are collected when you specify to collect transport TCP flags.

Flow exporters

Flow exporters export the data in the flow monitor cache to a remote system, such as a server running NetFlow collector, for analysis and storage. Flow exporters are created as separate entities in the configuration and are assigned to flow monitors to provide data export capability. You can create several flow exporters and assign them to one or more flow monitors to provide several export destinations. You can create one flow exporter and apply it to several flow monitors.

Flexible NetFlow supports Version 9 and Version 10 (IPFIX) export formats. If the export protocol is not configured, the system applies Version 9 export format by default.

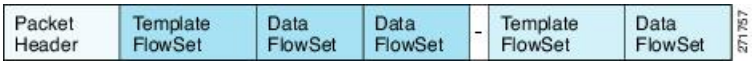
NetFlow data export format version 9

NetFlow primarily produces flow records as output. Several different formats for flow records have evolved as NetFlow has matured. The distinguishing feature of the NetFlow Version 9 export format is that it is template-based. Templates offer a flexible design for the record format. This flexibility enables future improvements to NetFlow services without needing simultaneous updates to the fundamental flow-record format. Using templates provides several key benefits:

- Third-party business partners who produce applications for NetFlow do not have to recompile their applications each time a new feature is added. Instead, they can use an external data file that documents the known template formats.
- New features can be added to NetFlow quickly without breaking current implementations.
- NetFlow is "future-proofed" against new or developing protocols because the Version 9 format can be adapted to provide support for them.

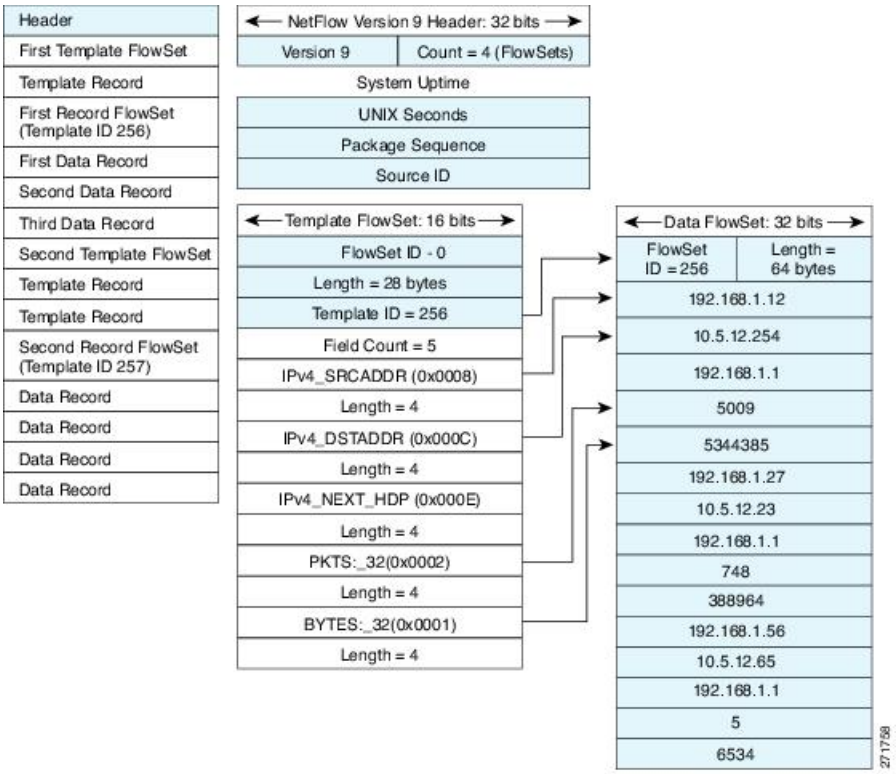
The Version 9 export format includes a packet header and is followed by one or more sets of either template flow or data flow. A template flow set provides a description of the fields that will be present in future data flow sets. These data flow sets may occur later within the same export packet or in subsequent export packets. Template flow and data flow sets can be intermingled within a single export packet, as illustrated in this figure.

Figure 2: Version 9 export packet



NetFlow Version 9 periodically exports the template data so the NetFlow collector will understand what data is to be sent and also export the data flow set for the template. The key advantage to Flexible NetFlow is that the user configures a flow record, which is effectively converted to a Version 9 template and then forwarded to the collector. This figure provides a detailed example of the NetFlow Version 9 export format, including the header, template flow, and data flow sets.

Figure 3: Detailed example of the NetFlow Version 9 export format



NetFlow data export format version 10 (IPFIX)

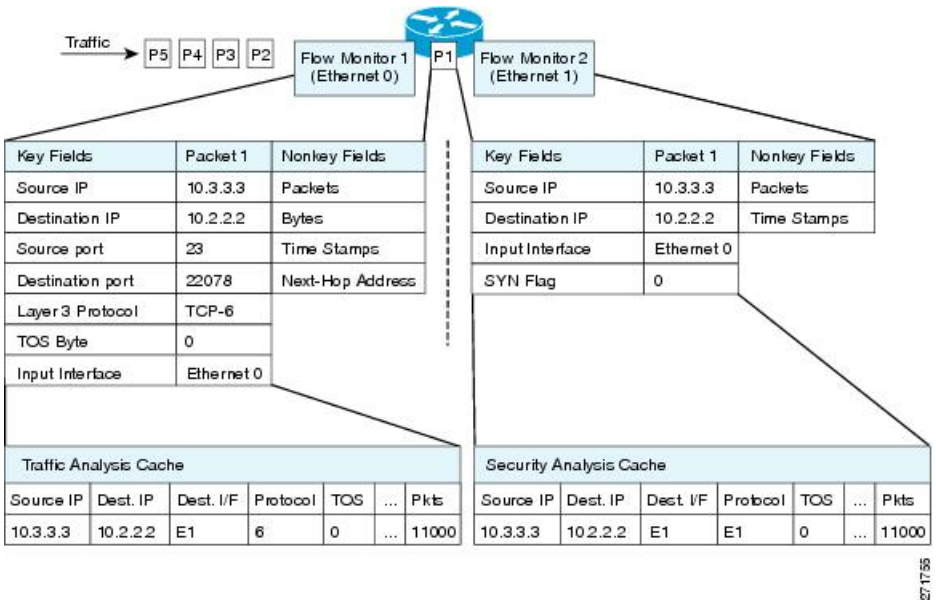
Internet Protocol Flow Information Export (IPFIX) or version 10 is an Export Protocol that collects and exports user defined flow records. IPFIX is an IETF standard (RFC 5153 and 7011) based on NetFlow version 9. The IPFIX format maintains the same principles of separate templates and records as NetFlow version 9. For IPFIX exporting protocol, the default destination port is 4739, the DSCP value is 0, and TTL is 255.

Flow monitors

Flow monitors are a component of Flexible NetFlow which are applied to interfaces to monitor network traffic. The system collects flow data from network traffic and stores it in the flow monitor cache based on the flow record fields. The device supports up to 8 feature profiles which is defined by flow monitor parameters such as cache size and inactive timeout.

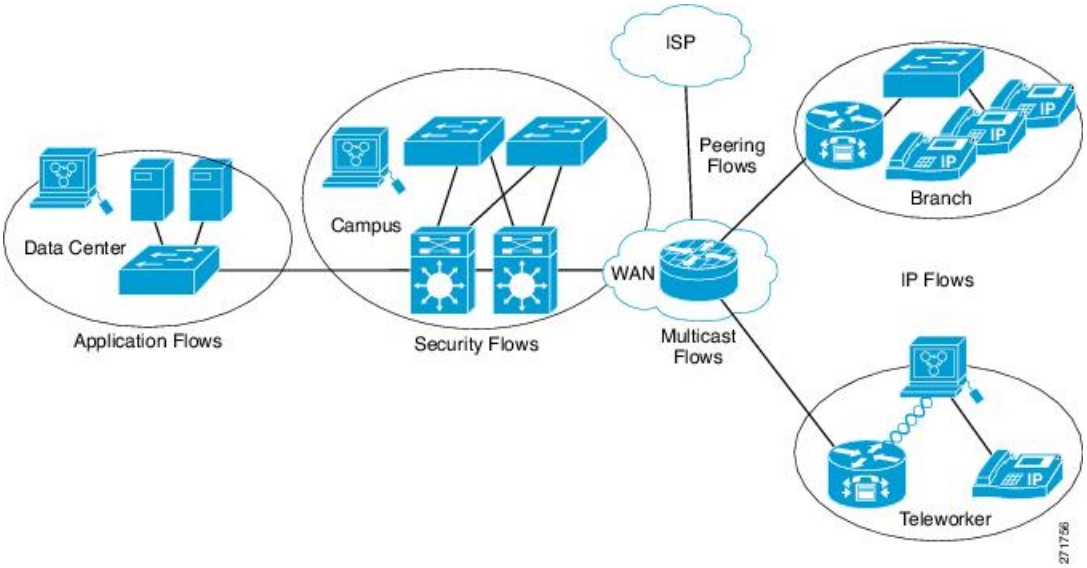
Flexible NetFlow enables different types of traffic analysis on identical data. The diagram shows how packet 1 is analyzed with a standard traffic analysis record on the input interface and a security analysis record on the output interface.

Figure 4: Example of Using Two Flow Monitors to Analyze the Same Traffic



The figure presents a more complex example of applying different types of flow monitors with custom records.

Figure 5: Complex Example: Various Flow Monitors Using Custom Records



The default cache type is normal, where entries age out only on inactive timeout. On active timeout, only collect field statistics are updated. When a cache entry ages out, it is removed from the cache and exported via any configured exporters.

Flow samplers

A flow sampler is a separate component created in a device configuration. Flow samplers reduce the load on the device running Flexible NetFlow by limiting the number of packets or flows selected for analysis.

Flow sampling improves device performance at the expense of monitoring accuracy. When you apply a sampler to a flow monitor, the overhead load on the device from running the flow monitor is reduced because the number of packets or flows the flow monitor must analyze is reduced. Reducing the number of packets or flows analyzed by the flow monitor reduces the accuracy of the information in the flow monitor cache.

Target interface

Target interface refers to the interface where NetFlow can be attached. The target interface is part of the configuration process when setting up NetFlow to collect, analyze, and export traffic statistics.

Layer 2, IPv4, and IPv6 traffic types are supported as multiple user-defined caches (flow monitors). Multiple flow monitors of different traffic types can be applied for a given interface and direction, but multiple flow monitors of the same traffic type cannot be applied for a given interface and direction.

Unicast, multicast, and broadcast streams are supported for both ingress and egress directions.

Both Layer 2 and Layer 3 physical interfaces along with the following logical interfaces are supported:

- Port-channel interfaces (L2 and/or L3)
- Sub-interfaces
    - Physical sub-interfaces
    - Port-channel sub-interfaces
- SVI (Switch Virtual Interface – interface vlan)

- VLAN ID (vlan configuration)

Supported Flexible NetFlow fields

These tables list supported Flexible NetFlow (FNF) fields for different traffic types and directions.

Flexible NetFlow also supports a set of key fields known as the 5-tuple flows. These fields represent unidirectional TCP and UDP sessions and include IPv4 and IPv6 source and destination addresses, Layer 4 protocol, source port, and destination port.

 **Note**

The VLAN field length is not accounted for if present in the packet.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key or collect fields - common							
Interface input	Yes	—	Yes	—	Yes	—	If you apply a flow monitor in the input direction, use the **match** keyword and use the input interface as a key field.
Interface output	—	Yes	—	Yes	—	Yes	If you apply a flow monitor in the output direction, use the **match** keyword and use the output interface as a key field.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Flow direction	Yes	Yes	Yes	Yes	Yes	Yes	
Key fields - datalink							
Ethertype	Yes	Yes	—	—	—	—	Supported only for a switch port.
VLAN input	Yes	—	—	—	—	—	Supported only for a switch port.
VLAN output	—	Yes	—	—	—	—	Supported only for a switch port.
dot1q VLAN input	Yes	—	—	—	—	—	Supported only for a switch port.
dot1q VLAN output	—	Yes	—	—	—	—	Supported only for a switch port.

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
dot1q priority	Yes	Yes	—	—	—	—	Supported only for a switch port.
MAC source address input	Yes	Yes	—	—	—	—	Supported only for a switch port.
MAC source address output	—	—	—	—	—	—	
MAC destination address input	Yes	—	—	—	—	—	Supported only for a switch port.
MAC destination address output	—	Yes	—	—	—	—	Supported only for a switch port.
Key fields - IPv4							
IPv4 version	—	—	Yes	Yes	—	—	
IPv4 TOS	—	—	Yes	Yes	—	—	
IPv4 protocol	—	—	Yes	Yes	—	—	Must use if any of src/dest port, ICMP code/type, IGMP type or TCP flags are used.
IPv4 TTL	—	—	Yes	Yes	—	—	
IPv4 source address	—	—	Yes	Yes	—	—	
IPv4 destination address	—	—	Yes	Yes	—	—	
ICMP IPv4 type	—	—	Yes	Yes	—	—	
ICMP IPv4 code	—	—	Yes	Yes	—	—	
IGMP type	—	—	Yes	Yes	—	—	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key fields - IPv6							
IPv6 version	—	—	—	—	Yes	Yes	Same as IP version.
IPv6 protocol	—	—	—	—	Yes	Yes	Same as IP protocol. Must use if any of src/dest port, ICMP code/type, IGMP type or TCP flags are used.
IPv6 source address	—	—	—	—	Yes	Yes	
IPv6 destination address	—	—	—	—	Yes	Yes	
IPv6 traffic-class	—	—	—	—	Yes	Yes	Same as IP TOS.
IPv6 hop-limit	—	—	—	—	Yes	Yes	Same as IP TTL.
ICMP IPv6 type	—	—	—	—	Yes	Yes	
ICMP IPv6 code	—	—	—	—	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Key fields - transport common							
Routing VRF input	—	—	Yes	Yes	Yes	Yes	
Source port	—	—	Yes	Yes	Yes	Yes	
Destination port	—	—	Yes	Yes	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Collect fields							
Packets long	Yes	Yes	Yes	Yes	Yes	Yes	

Field	Layer 2 In	Layer 2 Out	IPv4 In	IP v4 Out	IPv6 In	IPv6 Out	Notes
Timestamp absolute first	Yes	Yes	Yes	Yes	Yes	Yes	
Timestamp absolute last	Yes	Yes	Yes	Yes	Yes	Yes	
TCP flags	Yes	Yes	Yes	Yes	Yes	Yes	Collects all flags.

Default settings

This table lists the Flexible NetFlow default settings for the device.

Table 3: Default Flexible NetFlow Settings

Setting	Default
Flow active timeout	1800 seconds
Flow inactive timeout	15 seconds

Supported Flexible NetFlow features

This section describes the features supported by Flexible NetFlow.

Bridged NetFlow on a VLAN

Bridged NetFlow on a VLAN enables monitoring and collection of NetFlow data for traffic that is bridged within a VLAN. Unlike traditional routed NetFlow, which captures traffic as it is routed between Layer 3 interfaces, bridged NetFlow captures traffic at Layer 2 within a VLAN, making it useful for environments where traffic does not leave the VLAN but needs to be monitored.

This feature is particularly beneficial in networks where significant traffic exists between devices within the same VLAN or subnet, such as in campus networks, data centers, or virtualized environments.

Flexible NetFlow ingress and egress VRF

Use the Flexible NetFlow Ingress VRF feature to collect the virtual routing and forwarding (VRF) ID from incoming packets on a device by applying an input flow monitor having a flow record that collects the VRF ID as a key field.

Use the Flexible Netflow Egress VRF feature to collect the VRF ID from outgoing packets on a device by applying an output flow monitor having a flow record that collects the VRF ID as a key field.

Layer 2 and Layer 3 Flexible NetFlow

The Flexible Netflow Layer 2 feature enables collecting statistics for Layer 2 fields such as MAC addresses and VLAN IDs from traffic. This feature focuses on monitoring traffic within a local network or broadcast domain.

The Flexible Netflow Layer 3 feature, on the other hand, captures only routed traffic and not switched traffic. It is one of the most commonly used applications of Flexible NetFlow.

Stateful Switchover

Flexible NetFlow Stateful Switchover (SSO) feature allows for high availability and reliability in environments where NetFlow is used for traffic monitoring and analysis. This functionality ensures that NetFlow data collection and export processes continue seamlessly in the event of a failover between redundant supervisors.

This feature is enabled by default when redundancy is enabled and the mode of operation is set to SSO by enabling the **mode sso** command in redundancy configuration mode.

Prerequisites for Flexible NetFlow

Here are the prerequisites for Flexible NetFlow:

Key fields

Be familiar with the Flexible NetFlow key fields as they are defined in these commands.

- **match flow**
- **match interface**
- **match** {**ipv4** | **ipv6**}
- **match routing**
- **match transport**

Nonkey fields

Be familiar with the Flexible NetFlow nonkey fields as they are defined in these commands.

- **collect counter**
- **collect timestamp absolute**
- **collect transport**

IPv4 and IPv6 traffic

- The networking device must be configured for IPv4 and IPv6 routing.

Restrictions for Flexible NetFlow

Here are the restrictions for Flexible NetFlow:

- Flexible NetFlow is supported on Layer 2 and Layer 3 port-channel interfaces, but not on member ports.
- Flexible NetFlow on Layer 2 and VLAN can learn multicast traffic but Flexible NetFlow on Layer 3 does not learn multicast traffic flows.
- Traditional NetFlow accounting is not supported.
- Byte counter is not supported.

- Network Based Application Recognition (NBAR) Flexible NetFlow record and a regular Flexible NetFlow record cannot be configured at the same time.
- Multiple flow monitors of the same traffic type cannot be applied to a given interface and direction.
- Flexible NetFlow export is not supported on the Ethernet management port, GigabitEthernet 0/0.
- Source Group Tag (SGT) and Destination Group Tag (DGT) fields are not supported.
- NetFlow records do not support MultiProtocol Label Switching-enabled (MPLS-enabled) interfaces.
- When Flexible NetFlow and Network Address Translation (NAT) are configured on an interface,
    - Flexible NetFlow will display and export the actual flow details; but not the translated flow details. Application-level gateway (ALG) flow details are not part of the actual flow details that are exported.
    - If the ALG traffic gets translated through the CPU, Flexible NetFlow will display and export the translated flow details for the ALG traffic.
- The match on interface name in flow monitor attached to an egress SVI and subinterface gives the physical interface name and not SVI or subinterface name.
- A maximum of 8 NetFlow profiles are supported on an ASIC including ingress and egress. NetFlow profile is defined by flow monitor parameters such as cache size and inactive timeout. All the flow monitors using same flow monitor parameters share the NetFlow profile on ingress and egress. Example, if two flow monitors share the same NetFlow profiles then cache size will be shared among these two flow monitors.
- Flexible NetFlow on Layer 3 interfaces learns only routed traffic.
- **match datalink vlan output** command is not supported in datalink flow monitor for egress Flexible NetFlow.
- Control packets like ARP and CDP are not learnt by Flexible NetFlow.
- Flow monitor parameters such as active timeout, inactive timeout, and cache size cannot be modified if the flow monitor is attached to an interface.

Configure Flexible Netflow

To configure Flexible Netflow, follow these general steps:

1. Create a flow record by specifying keys and non-key fields to the flow.
2. Create an optional flow exporter by specifying the protocol and transport destination port, destination, and other parameters.
3. Create a flow monitor based on the flow record and flow exporter.
4. Create an optional sampler.
5. Apply the flow monitor to a Layer 2 port, Layer 3 port, or VLAN.

Additionally, you can configure other Flexible Netflow features.

Create and modify user-defined flow record

Follow this task to configure a customized flow record for your needs.

Customized flow records are used to analyze traffic data for a specific purpose. A customized flow record must have at least one **match** criterion for use as the key field and typically has at least one **collect** criterion for use as a nonkey field.

Customized flow records can be configured in numerous ways to fulfill specific traffic analysis needs. This task shows the steps that are used to create one of the possible permutations. You can modify these steps to create a customized flow record that suits your requirements.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *record-name* **Example:** `Device(config)# flow record FLOW-RECORD-1`	Creates a flow record and enters Flexible NetFlow flow record configuration mode. • This command also allows you to modify an existing flow record.
Step 4	**description** *description* **Example:** `Device(config-flow-record)# description Used for basic traffic analysis`	(Optional) Creates a description for the flow record.
Step 5	**match** {**ipv4** | **ipv6**} {**destination** | **source**} **address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	Configures a key field for the flow record. **Note** This example configures the IPv4 destination address as a key field for the record.
Step 6	Repeat Step 5 as needed until all desired key fields are configured for the record.	—
Step 7	**end** **Example:** `Device(config-flow-record)# end`	Exits Flexible NetFlow flow record configuration mode and returns to privileged EXEC mode.
Step 8	**show flow record** *record-name* **Example:** `Device# show flow record FLOW_RECORD-1`	(Optional) Displays the current status of the specified flow record.
Step 9	**show running-config flow record** *record-name* **Example:** `Device# show running-config flow record FLOW_RECORD-1`	(Optional) Displays the configuration of the specified flow record.

Create a flow exporter

You can create a flow export to define the export parameters for a flow.

Note

Each flow exporter supports only one destination. If you want to export the data to multiple destinations, you must configure multiple flow exporters and assign them to the flow monitor.

You can export to a destination using IPv4 address.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**flow exporter** *name* **Example:** `Device(config)# flow exporter ExportTest`	Creates a flow exporter and enters flow exporter configuration mode.
Step 4	**description** *string* **Example:** `Device(config-flow-exporter)# description ExportV9`	(Optional) Describes this flow record as a maximum 63-character string.
Step 5	**destination** {*ipv4-address*} **Example:** `Device(config-flow-exporter)# destination 192.0.2.1 (IPv4 destination)`	Sets the IPv4 destination address or hostname for this exporter.
Step 6	**dscp** *value* **Example:** `Device(config-flow-exporter)# dscp 0`	(Optional) Specifies the differentiated services codepoint value. The range is from 0 to 63. The default is 0.
Step 7	**source** *interface type interface number* **Example:** `Device(config-flow-exporter)# source gigabitEthernet1/0/1`	(Optional) Specifies the interface to use to reach the NetFlow collector at the configured destination. **Note** The Flow Exporter does not support unnumbered IP interfaces as source interfaces. The following interfaces can be configured as source: • **Auto Template**: Auto-Template interface • **Capwap**: CAPWAP tunnel interface

	Command or Action	Purpose
		• **GigabitEthernet**: Gigabit Ethernet IEEE 802 • **GroupVI**: Group virtual interface • **Internal Interface**: Internal interface • **Loopback**: Loopback interface • **Null**: Null interface • **Port-channel**: Ethernet channel of interface • **TenGigabitEthernet**: 10-Gigabit Ethernet • **Tunnel**: Tunnel interface • **Vlan**: Catalyst VLANs
Step 8	**transport udp** *number* **Example:** `Device(config-flow-exporter)# transport udp 200`	(Optional) Specifies the UDP port to use to reach the NetFlow collector.
Step 9	**ttl** *seconds* **Example:** `Device(config-flow-exporter)# ttl 210`	(Optional) Configures the time-to-live (TTL) value for datagrams sent by the exporter. The range is from 1 to 255 seconds. The default is 255.
Step 10	**export-protocol** {**netflow-v9**} **Example:** `Device(config-flow-exporter)# export-protocol netflow-v9`	Specifies the version of the NetFlow export protocol used by the exporter.
Step 11	**end** **Example:** `Device(config-flow-record)# end`	Returns to privileged EXEC mode.
Step 12	**show flow exporter** [**name** *record-name*] **Example:** `Device# show flow exporter ExportTest`	(Optional) Displays information about NetFlow flow exporters.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Create a customized flow monitor

Each flow monitor has a separate cache assigned to it. Flow monitor parameters are called NetFlow profiles such as cache size and inactive time out. Each flow monitor requires a record to define the contents and layout of its cache entries. Advanced users can create customized formats using the **flow record** command.

- If you want to use a customized record, you must create the customized record before you can perform this task.

- If you want to add a flow exporter to the flow monitor for data export, you must create the exporter before you can complete this task.

Note

You must use the **no ip flow monitor** command to remove a flow monitor from all of the interfaces to which you have applied it before you can modify the parameters for the **record** command on the flow monitor.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow monitor** *monitor-name* **Example:** `Device(config)# flow monitor FLOW-MONITOR-1`	Creates a flow monitor and enters Flexible NetFlow flow monitor configuration mode. • This command also allows you to modify an existing flow monitor.
Step 4	**description** *description* **Example:** `Device(config-flow-monitor)# description Used for basic ipv4 traffic analysis`	(Optional) Creates a description for the flow monitor.
Step 5	**record** {*record-name* | **netflow-original** | **netflow** {**ipv4** | **ipv6**} *record* [**peer**]} **Example:** `Device(config-flow-monitor)# record FLOW-RECORD-1`	Specifies the record for the flow monitor.

Step 6	**cache** {**timeout** {**active** | **inactive** | **update** | **rate-limit**} *seconds* | **type normal** }
	Example:
	`Device(config-flow-monitor)# cache type normal`
	`Device(config-flow-monitor)# cache timeout active`
		(Optional) Modifies the flow monitor cache parameters such as timeout values, and the cache type. Associates a flow cache with the specified flow monitor.

Step 7	Repeat Step 6 to complete cache parameter modifications for this flow monitor, as necessary.	—
Step 8	**exporter** *exporter-name* **Example:** `Device(config-flow-monitor)# exporter EXPORTER-1`	(Optional) Specifies the name of an exporter that was created previously.

	Command or Action	Purpose
Step 9	**end** **Example:** `Device(config-flow-monitor)# end`	Exits Flexible NetFlow flow monitor configuration mode and returns to privileged EXEC mode.
Step 10	**show flow monitor** [[**name**] *monitor-name* [**cache** [**format** {**csv** | **record** | **table**}]] [**statistics**]] **Example:** `Device# show flow monitor FLOW-MONITOR-2 cache`	(Optional) Displays the status and statistics for a Flexible NetFlow flow monitor.
Step 11	**show running-config flow monitor** *monitor-name* **Example:** `Device# show running-config flow monitor FLOW_MONITOR-1`	(Optional) Displays the configuration of the specified flow monitor.
Step 12	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Create a flow sampler

Perform this required task to configure and enable a flow sampler.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sampler** *sampler-name* **Example:** `Device(config)# sampler SAMPLER-1`	Creates a sampler and enters sampler configuration mode. • This command also allows you to modify an existing sampler.
Step 4	**description** *description* **Example:** `Device(config-sampler)# description Sample at 50%`	(Optional) Creates a description for the flow sampler.
Step 5	**mode** {**random**} **1 out-of** *window-size* **Example:** `Device(config-sampler)# mode random 1 out-of 2`	Specifies the sampler mode and the flow sampler window size. • The range for the *window-size* argument is from 0 to 1024.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Device(config-sampler)# exit`	Exits sampler configuration mode and returns to global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Specifies an interface and enters interface configuration mode.
Step 8	{**ip** | **ipv6**} **flow monitor** *monitor-name* [[**sampler**] *sampler-name*] {**input** | **output**} **Example:** `Device(config-if)# ip flow monitor FLOW-MONITOR-1 sampler SAMPLER-1 input`	Assigns the created flow monitor and sampler to the interface, enabling sampling.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 10	**show sampler sampler-name** **Example:** `Device# show sampler SAMPLER-1`	Displays the status and statistics of the flow sampler that you configured and enabled.

Apply a flow to an interface

You can apply a flow monitor and an optional sampler to an interface.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type* **Example:** `Device(config)# interface GigabitEthernet1/0/1`	Enters interface configuration mode and configures an interface.
Step 4	{**ip flow monitor** | **ipv6 flow monitor** | **datalink flow monitor**} *name* [**sampler** *name*] {**input** | **output**} **Example:** `Device(config-if)# ip flow monitor MonitorTest input`	Associates an IPv4, IPv6, and datalink flow monitor, along with an optional sampler, to the interface for input or output packets. • **ip flow monitor**: Enables Flexible NetFlow to monitor IPv4 traffic.

	Command or Action	Purpose
		• **ipv6 flow monitor**: Enables Flexible NetFlow to monitor IPv6 traffic. • **datalink flow monitor**: Enables Flexible NetFlow to monitor non-IP traffic. **Note** You can associate multiple monitors to an interface in both input and output directions.
Step 5	**end** **Example:** `Device(config-flow-monitor)# end`	Returns to privileged EXEC mode.
Step 6	**show flow interface** [*interface-type number*] **Example:** `Device# show flow interface`	(Optional) Displays information about NetFlow on an interface.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configure a bridged NetFlow on a VLAN

Apply a flow monitor and an optional sampler to a VLAN.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**vlan** [**configuration**] *vlan-id* **Example:** `Device(config)# vlan configuration 30`	Configures a VLAN and enters VLAN or VLAN configuration mode.
Step 4	**ip flow monitor** *monitor name* [**sampler** *sampler name*] {**input**} **Example:** `Device(config-vlan-config)# ip flow monitor MonitorTest input`	Associates a flow monitor and an optional sampler to the VLAN for input packets.

	Command or Action	Purpose
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configure Flexible Netflow ingress and egress VRF

Perform this task to configure the collection of VRF ID from incoming packets on a device by applying an input or output flow monitor.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *record-name* **Example:** `Device(config)# flow record rm_1`	Creates a flow record and enters Flexible NetFlow flow record configuration mode. • This command also allows you to modify an existing flow record.
Step 4	**match routing vrf input** **Example:** `Device(config-flow-record)# match routing vrf input`	Configures the virtual routing and forwarding (VRF) ID for incoming as a key field.
Step 5	**match** {**ip** | **ipv6**} {**destination** | **source**} **address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	Configures a key field for the flow record. **Note** This example configures the IPv4 destination address as a key field for the record.
Step 6	Repeat Step 5 as needed until all desired key fields are configured for the record.	—
Step 7	**exit** **Example:** `Device(config-flow-record)# end`	Exits Flexible NetFlow flow record configuration mode and returns to global configuration mode.
Step 8	**flow monitor** *monitor-name* **Example:** `Device(config)# flow monitor mm_1`	Creates a flow monitor and enters Flexible NetFlow flow monitor configuration mode. • This command also allows you to modify an existing flow monitor.

	Command or Action	Purpose
Step 9	**record** {*record-name* | **netflow-original** | **netflow** {**ipv4** | **ipv6**} *record* [**peer**]} **Example:** `Device(config-flow-monitor)# record rm_1`	Specifies the record for the flow monitor.
Step 10	**exit** **Example:** `Device(config-flow-record)# end`	Exits Flexible NetFlow flow record configuration mode and returns to global configuration mode.
Step 11	**interface GigabitEthernet** *interface-id* **Example:** `Device(config)# interface GigabitEthernet 1/0/1`	Creates an interface and enters interface configuration mode.
Step 12	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding green`	Associates the VRF instance with the interface.
Step 13	**ip address** *ip-address* **Example:** `Device(config-if)# ip address 172.16.2.2 255.255.255.252`	Sets an IP address for the interface.
Step 14	**ip flow monitor** *monitor-name* {**input** | **output**} **Example:** `Device(config-if)# ip flow monitor mm_1 input`	Enables a Flexible NetFlow flow monitor for IPv4 traffic. **Note** This example enables a flow monitor for IPv4 traffic that the device is receiving.
Step 15	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configure Layer 2 NetFlow

Define Layer 2 keys in Flexible NetFlow records to capture flows in Layer 2 interfaces.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**flow record** *name* **Example:** `Device(config)# flow record L2_record`	Enters flow record configuration mode.
Step 4	**match datalink** {**dot1q** |**ethertype** | **mac** | **vlan**} **Example:** `Device(config-flow-record)# match datalink ethertype`	Specifies the Layer 2 attribute as a key.
Step 5	**end** **Example:** `Device(config-flow-record)# end`	Returns to privileged EXEC mode.
Step 6	**show flow record** [*name*] **Example:** `Device# show flow record`	(Optional) Displays information about NetFlow on an interface.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configure Layer 3 NetFlow

Define Layer 3 keys in Flexible NetFlow records to capture flows in Layer 3 interfaces.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.
Step 3	**flow record** *name* **Example:** `Device(config)# flow record L3_record`	Enters flow record configuration mode.
Step 4	**match ipv4 destination address** **Example:** `Device(config-flow-record)# match ipv4 destination address`	Specifies the Layer 3 attribute as a key.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-flow-record)# end`	Returns to privileged EXEC mode.
Step 6	**show flow record** [*name*] **Example:** `Device# show flow record`	(Optional) Displays information about NetFlow on an interface.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuration examples

Refer this section for configuration examples of Flexible NetFlow.

Example: Configure a flow and apply to an interface

This example shows how to create a flow and apply it to an interface:

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.

Device(config)# flow export export1
Device(config-flow-exporter)# destination 10.0.101.254
Device(config-flow-exporter)# transport udp 2055
Device(config-flow-exporter)# exit
Device(config)# flow record record1
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match ipv4 protocol
Device(config-flow-record)# match transport source-port
Device(config-flow-record)# match transport destination-port
Device(config-flow-record)# collect counter
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# exit
Device(config)# flow monitor monitor1
Device(config-flow-monitor)# record record1
Device(config-flow-monitor)# exporter export1
Device(config-flow-monitor)# exit
Device(config)# interface tenGigabitEthernet 1/0/1
Device(config-if)# ip flow monitor monitor1 input
Device(config-if)# end
```

Example: Configure a bridged NetFlow on a VLAN

This example shows how to configure a bridged NetFlow on a VLAN:

```
Device# configure terminal
Device(config)# vlan configuration 30
```

```
Device(config-vlan-config)# ip flow monitor MonitorTest input
Device(config-vlan-config)# end
```

Example: Configure Flexible NetFlow for ingress VRF

This example configures the collection of the VRF ID from incoming packets on a device by applying an input flow monitor having a flow record that collects the VRF ID as a key field.

```
Device> enable
Device# configure terminal
Device(config)# flow record rm_1
Device(config-flow-record)# match routing vrf input
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# collect counter
Device(config-flow-record)# exit

Device(config)# flow monitor mm_1
Device(config-flow-record)# record rm_1
Device(config-flow-record)# exit

Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# ip vrf forwarding green
Device(config-if)# ip address 172.16.2.2 255.255.255.252
Device(config-if)# ip flow monitor mm_1 input
Device(config-if)# end
```

Example: Configure Flexible NetFlow for egress VRF

This example configures the collection of the VRF ID from outgoing packets on a device by applying an output flow monitor having a flow record that collects the VRF ID as a key field.

```
Device> enable
Device# configure terminal
Device(config)# flow record rm_1
Device(config-flow-record)# match routing vrf input
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# collect counter
Device(config-flow-record)# exit

Device(config)# flow monitor mm_1
Device(config-flow-record)# record rm_1
Device(config-flow-record)# exit

Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# ip vrf forwarding green
Device(config-if)# ip address 172.16.2.2 255.255.255.252
Device(config-if)# ip flow monitor mm_1 output
Device(config-if)# end
```

Example: Configure Layer 2 NetFlow

This example shows how to configure Layer 2 NetFlow:

```
Device# configure terminal
Device(config)# flow record L2_record
Device(config-flow-record)# match datalink ethertype
Device(config-flow-record)# end
```

Example: Configure Layer 3 NetFlow

This example shows how to configure Layer 3 NetFlow:

```
Device# configure terminal
Device(config)# flow record L3_record
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# end
```

Monitor Flexible NetFlow

The commands in the following table can be used to monitor Flexible NetFlow.

Table 4: Flexible NetFlow Monitoring Commands

Command	Purpose
show redundancy [**clients** | **counters** | **history** | **switchover history** | **states**]	View information about SSO configuration information.
show flow exporter [**broker** | **export-ids** | **name** | *name* | **statistics** | **templates**]	View information about NetFlow flow exporters and statistics.
show flow interface	View information about NetFlow interfaces.
show flow monitor [**name** *exporter-name*]	View information about NetFlow flow monitors and statistics.
show flow monitor statistics	View the statistics for the flow monitor
show flow monitor cache format {**table** | **record** | **csv**}	View the contents of the cache for the flow monitor, in the format specified.
show flow record [**name** *record-name*]	View information about NetFlow flow records.
show sampler [**broker** | **name** | *name*]	View information about NetFlow samplers.

Example: Monitor IPv4 ingress traffic

This example shows how to monitor IPv4 ingress traffic (int g1/0/11 sends traffic to int g1/0/36 and int g3/0/11).

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# flow record  fr-1
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match interface input
Device(config-flow-record)# collect counter
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# exit

Device(config)# flow exporter fe-ipfix6
Device(config-flow-exporter)# destination 2001:0:0:24::10
Device(config-flow-exporter)# source Vlan106
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
```

```
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-ipfix
Device(config-flow-exporter)# description IPFIX format collector 100.0.0.80
Device(config-flow-exporter)# destination 100.0.0.80
Device(config-flow-exporter)# dscp 30
Device(config-flow-exporter)# ttl 210
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-1
Device(config-flow-exporter)# destination 10.5.120.16
Device(config-flow-exporter)# source Vlan105
Device(config-flow-exporter)# dscp 32
Device(config-flow-exporter)# ttl 200
Device(config-flow-exporter)# transport udp 2055

Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow monitor fm-1
Device(config-flow-monitor)# exporter fe-ipfix6
Device(config-flow-monitor)# exporter fe-ipfix
Device(config-flow-monitor)# exporter fe-1
Device(config-flow-monitor)# cache timeout inactive 60
Device(config-flow-monitor)#  cache timeout active 180
Device(config-flow-monitor)# record fr-1
Device(config-flow-monitor)# end

Device# show running-config interface g1/0/11
Device# show running-config interface g1/0/36
Device# show running-config interface g3/0/11
Device# show flow monitor fm-1 cache format table
```

Example: Monitor IPv4 egress traffic

This example shows how to monitor IPv4 egress traffic.

```
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# flow record  fr-1 out
Device(config-flow-record)# match ipv4 source address
Device(config-flow-record)# match ipv4 destination address
Device(config-flow-record)# match interface output
Device(config-flow-record)# collect counter
Device(config-flow-record)# collect timestamp absolute first
Device(config-flow-record)# collect timestamp absolute last
Device(config-flow-record)# exit

Device(config)# flow exporter fe-1
Device(config-flow-exporter)# destination 10.5.120.16
Device(config-flow-exporter)# source Vlan105
Device(config-flow-exporter)# dscp 32
Device(config-flow-exporter)# ttl 200
Device(config-flow-exporter)# transport udp 2055
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-ipfix6
```

```
Device(config-flow-exporter)# destination 2001:0:0:24::10
Device(config-flow-exporter)# source Vlan106
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow exporter fe-ipfix
Device(config-flow-exporter)# description IPFIX format collector 100.0.0.80
Device(config-flow-exporter)# destination 100.0.0.80
Device(config-flow-exporter)# dscp 30
Device(config-flow-exporter)# ttl 210
Device(config-flow-exporter)# transport udp 4739
Device(config-flow-exporter)# export-protocol ipfix
Device(config-flow-exporter)# template data timeout 240
Device(config-flow-exporter)# exit

Device(config)# flow monitor fm-1-output
Device(config-flow-monitor)# exporter fe-1
Device(config-flow-monitor)# exporter fe-ipfix6
Device(config-flow-monitor)# exporter fe-ipfix
Device(config-flow-monitor)# cache timeout inactive 50
Device(config-flow-monitor)#  cache timeout active 120
Device(config-flow-monitor)# record fr-1-out
Device(config-flow-monitor)# end

Device# show flow monitor fm-1-output cache format table
```