



Configuring SISF-Based Device Tracking

- [Information About Switch Integrated Security Features, on page 1](#)
- [How to Configure SISF, on page 20](#)
- [Configuration Examples for SISF, on page 30](#)
- [Feature History and Information for SISF, on page 35](#)

Information About Switch Integrated Security Features

Overview

Switch Integrated Security Features (SISF) is a framework developed to optimize security in Layer 2 domains. It merges the IP Device Tracking (IPDT) and *certain* IPv6 first-hop security (FHS) functionality¹, to simplify the migration from IPv4 to IPv6 stack or a dual-stack.

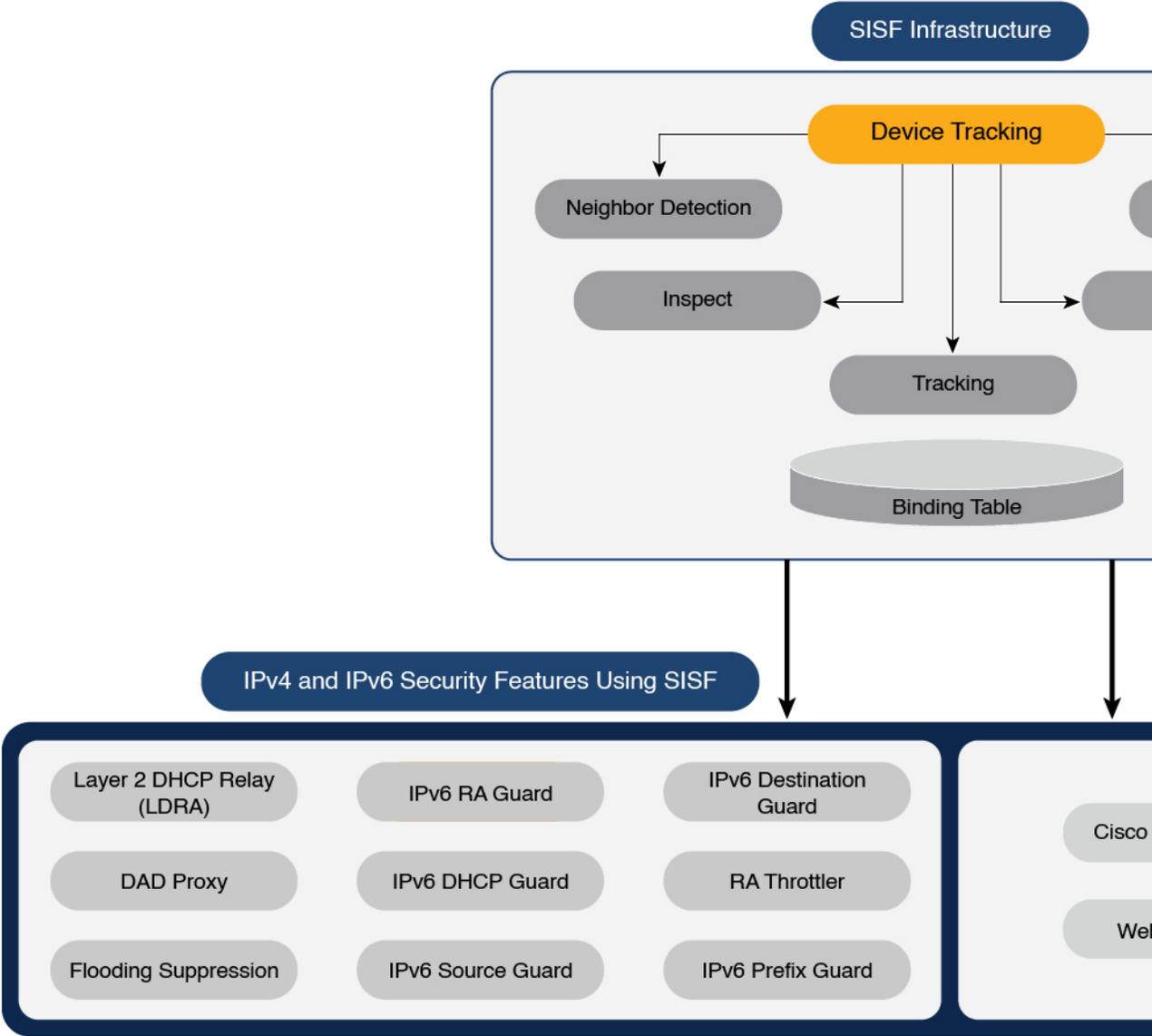
The SISF infrastructure provides a unified database that is used by:

- IPv6 FHS features: IPv6 Router Advertisement (RA) Guard, IPv6 DHCP Guard, Layer 2 DHCP Relay, IPv6 Duplicate Address Detection (DAD) Proxy, Flooding Suppression, IPv6 Source Guard, IPv6 Destination Guard, RA Throttler, and IPv6 Prefix Guard.
- Features like Cisco TrustSec, IEEE 802.1X, Locator ID Separation Protocol (LISP), Ethernet VPN (EVPN), and Web Authentication, which act as clients for SISF.

The following figure illustrates this:

¹ IPv6 Snooping Policy, IPv6 FHS Binding Table Content, and IPv6 Neighbor Discovery Inspection

Figure 1: SISF Framework



Note The terms “SISF” “device-tracking” and “SISF-based device-tracking” are used interchangeably in this document and refer to the same feature. Neither term is used to mean or should be confused with the legacy IPDT or IPv6 Snooping features.

Understanding the SISF Infrastructure

This section explains the various elements of the SISF infrastructure as shown in the [Overview, on page 1](#).

The Binding Table

The SISF infrastructure is built around the binding table. The binding table contains information about the hosts that are connected to the ports of a switch and the IP and MAC address of these hosts. This helps to create a physical map of all the hosts that are connected to a switch.

Each entry in a binding table provides the following information about a connected host:

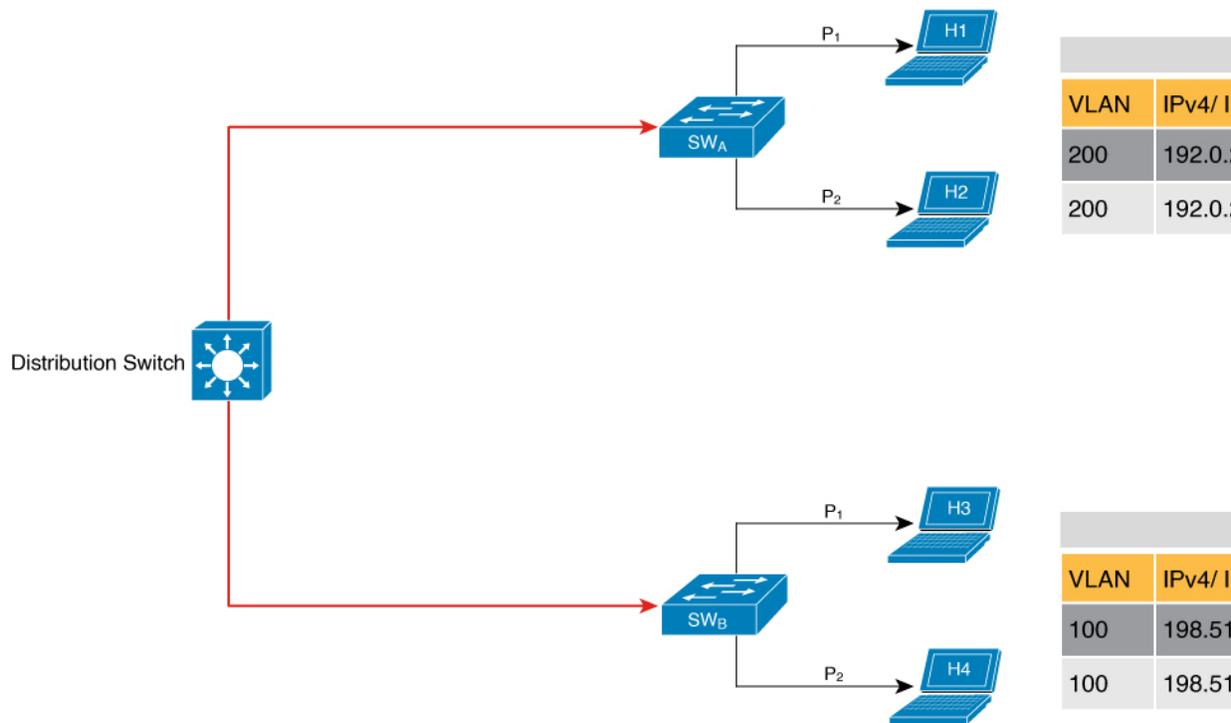
- IPv4 or IPv6 address of the host.
- MAC address of the host. The same MAC address may be linked to an IPv4 and IPv6 address.
- The interface or port on the switch that the host is connected to, and the associated VLAN.
- The state of the entry, which indicates the reachability of the entry.

The following figure shows a simple network topology and a representative binding table for each access switch in the network. SW_A and SW_B are the two access switches in the network. The two access switches are connected to the same distribution switch. H1, H2, H3, H4 are the hosts.

This is an example of a distributed binding table, that is, each access switch in the network has its own table. An alternative set-up could be one centralised binding table on the distribution switch with the entries of SW_A and SW_B.

Having a distributed or a centralised binding table is a key design choice in the process of implementing SISF in your network and is covered in greater detail in the [Understanding Policy Parameters, on page 7](#) section in this chapter.

Figure 2: Binding Table



States and Lifetime of a Binding Table Entry

The state of an entry indicates if the host is reachable or not. The stable states of a binding table entry are: REACHABLE, DOWN, and STALE. When changing from one state to another, an entry may have other temporary or transitional states such as: VERIFY, INCOMPLETE, and TENTATIVE.

How long an entry remains in a given state is determined by its lifetime and by whether or not the entry is validated successfully. The lifetime of an entry can be policy-driven or configured globally.

To configure the REACHABLE, DOWN, and STALE lifetimes, enter the following command in global configuration mode:

```
device-tracking binding { reachable-lifetime { seconds | infinite } | stale-lifetime { seconds | infinite }
| down-lifetime { seconds | infinite } }
```

State: Reachable

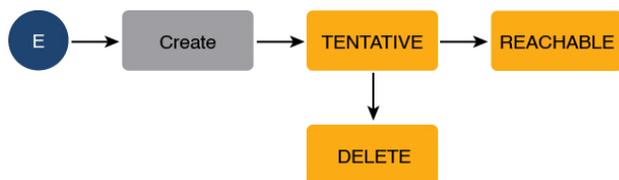
If an entry has this state, it means the host (IP and MAC address) from which a control packet was received, is a verified and valid host. A reachable entry has a default lifetime of 5 minutes. You can also configure a duration. By configuring a reachable-lifetime, you specify how long a host can remain in a REACHABLE state, after the last incoming control packet from that host.

If an event is detected before the entry's reachable lifetime expires, then the reachable lifetime is reset.

To qualify for the REACHABLE state, a new entry goes through the process illustrated in the figure below. The switch detects an event (E), such as an incoming control packet from a connected host and creates an entry. Various events cause the creation of an entry, and these are described in the [Binding Table Sources](#) section. The creation of an entry is followed by different transient states, such as TENTATIVE or INCOMPLETE. While in a transitional state, the switch validates and confirms the integrity of the binding entry. If the entry is found to be valid, then the state changes to REACHABLE.

But if an address theft or similar event is detected, then the entry is regarded as invalid and is deleted. For example, if an attacker sends unsolicited neighbor advertisement messages with the same IP as the target IP and its (attacker's) own MAC address to redirect traffic.

Figure 3: Creation of a Reachable Entry

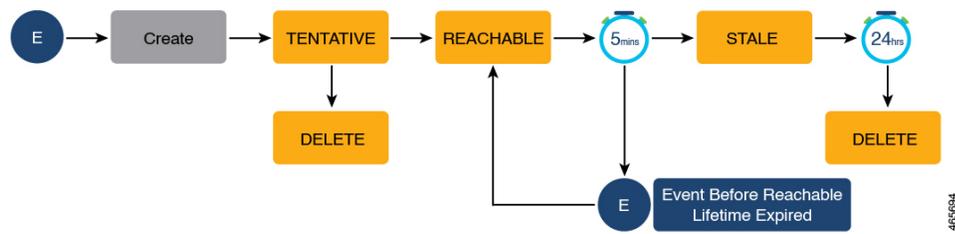


State: Stale

If an entry is in this state it means that the entry's reachable lifetime has expired and the corresponding host is still silent (no incoming packets from the host). A stale entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the STALE state beyond the stale lifetime, is deleted.

This is illustrated in the figure below which depicts the lifecycle of an entry.

Figure 4: Lifecycle of an Entry



State: Down

If an entry is in this state, it means that the host's connecting interface is down. A down entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the DOWN state beyond the down lifetime, is deleted.

Polling a Host and Updating the Binding Table Entry

Polling is a periodic and conditional checking of the host to see the state it is in, whether it is still connected, and whether it is communicating. In addition to determining an entry's state, you can use polling to reconfirm an entry's state.

You can enable polling with the **device-tracking tracking** command in global configuration mode. After you do, you still have the flexibility to turn polling on or off for a particular interface or VLAN. For this, configure the **tracking enable** or **tracking disable** keywords in the policy (the device-tracking configuration mode). When polling is enabled, the switch polls the host at the specified interval, thus reconfirming its reachability for the duration of its reachable lifetime.

When polling is enabled, the switch sends up to three polling requests, after the reachable lifetime expires, at system-determined intervals. You can also configure this interval with the **device-tracking tracking retry-interval seconds** command in global configuration mode.

The figure below depicts the lifecycle of an entry where the host is polled. Default reachable and stale lifetimes, and retry intervals are used in figure:

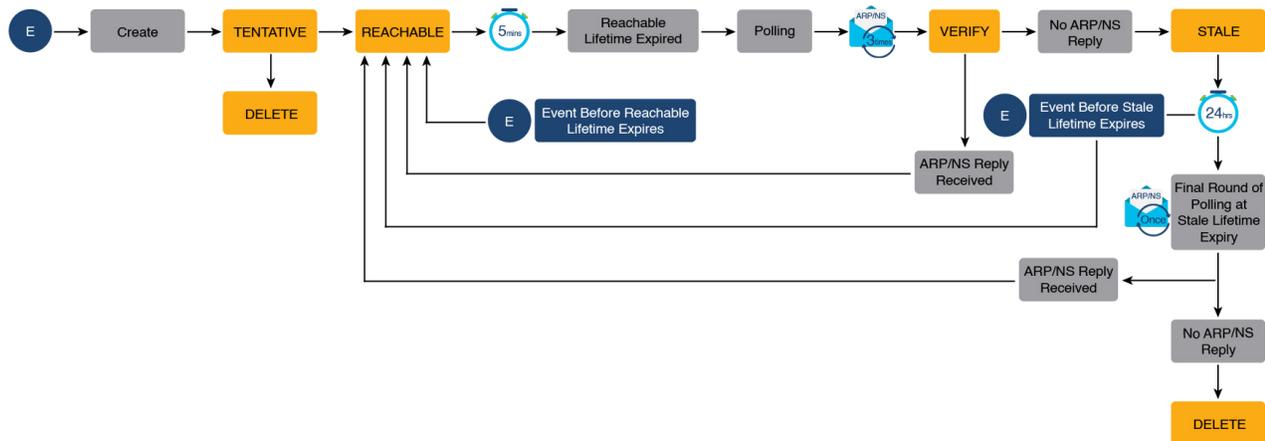
An event (E) is detected and a REACHABLE entry is created.

If an event is detected *during* the reachable lifetime, the reachable lifetime timer is reset.

The switch sends a polling request after the reachable lifetime expires. The switch polls the host up to three times at fixed, system-determined intervals. The polling request may be in the form of a unicast Address Resolution Protocol (ARP) probe or a Neighbor Solicitation message. During this time the state of the entry changes to VERIFY. If a polling response is received (thus confirming reachability of the host), the state of the entry changes back to REACHABLE.

If the switch does not receive a polling response after three attempts, the entry changes to the STALE state. It remains in this state for 24 hours. If an event is detected during the stale lifetime, the state of the entry is changed back to REACHABLE. At expiry of the stale lifetime, the device sends one final polling to ascertain reachability. If this final polling attempt receives a reply, the state of the entry is changed back to REACHABLE. If the final polling attempt does not receive a response, the entry is deleted.

Figure 5: Lifecycle of an Entry Where the Host is Polled



Binding Table Sources

This section describes the sources of information and events that cause the creation and update of a binding table entry.

- Learning events that dynamically populate the binding table:
 - Dynamic Host Configuration Protocol (DHCP) negotiation (DHCP REQUEST, and DHCP REPLY). This includes DHCPv4 and DHCPv6.
 - Address Resolution Protocol (ARP) packets.
 - Neighbor Discovery Protocol (NDP) packets.
 - Multiple Identity Association-Nontemporary Address (IA_NA) and Identity Association-Prefix Delegation (IA_PD).

In some cases, a network device can request and receive more than one IPv6 address from the DHCP server. This may be done to provide addresses to multiple clients of the device, such as when a residential gateway requests addresses to distribute to its LAN clients. When the device sends out a DHCPv6 packet, the packet includes all of the addresses that have been assigned to the device.

When SISF analyzes a DHCPv6 packet, it examines the IA_NA (Identity Association-Nontemporary Address) and IA_PD (Identity Association-Prefix Delegation) components of the packet and extracts each IPv6 address contained in the packet. SISF adds each extracted address to the binding table.

- Configuration of static binding entries.

If there are silent but reachable hosts in the Layer 2 domain, you can create static binding entries to retain binding information even if the host becomes silent.

For this, you configure the following command in global configuration mode: **device-tracking binding vlan** *vlan-id* {*ipv4_address* *ipv6_address* *ipv6_prefix*} {**interface** *interface-type_no*}.



Note In addition to the primary or key events listed above, there is a specific scenario in which a ping can result in a device-tracking entry. If a sender's ARP cache or IPv6 neighbor table doesn't have the target's IP address yet, then a ping triggers an ARP packet for IPv4, or ND packet for IPv6. This can result in a device-tracking entry.

But if the target IP is already in the ARP cache or IPv6 neighbour table, no ARP or ND packet is generated when you ping - in which case SISF cannot learn the IP address.

Device-Tracking

SISF-based device-tracking is disabled by default. You can enable the feature on an interface or VLAN.

When you enable the feature, the binding table is created, followed by subsequent maintenance of the binding table.

The events listed in the [Binding Table Sources, on page 6](#) section act as triggers for SISF-based device-tracking, to track the presence, location, and movement of hosts in the network, to populate and maintain the binding table. For example, if information about a host is learnt by means of an ARP or ND packet, every subsequent ARP or ND packet from the same host acts as an alert for SISF-based device-tracking, to refresh the entry in the binding table, thus indicating if the host is still present in the same location or has moved.

The continuous process of snooping of packets that the switch receives, extraction of device identity (MAC and IP address), and storage of information in the binding table of the switch, ensures binding integrity and maintains the reachability status of the hosts in the binding table.

For information how to enable SISF-based device-tracking, see [How to Configure SISF, on page 20](#).

Device-Tracking Policy

A device-tracking policy is a set of rules that SISF-based device-tracking follows. The policy dictates which events will be listened to, whether a host will be probed, the wait time before the host is probed, and so on. These rules are referred to as policy parameters.



Note The policy must be attached to an interface or VLAN. Only then is the binding table for that interface or VLAN populated - in accordance with policy parameters.

For information about the various ways in which you can create a policy, see [How to Configure SISF, on page 20](#).

To display a policy's settings, use the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

Understanding Policy Parameters

Policy parameters are the keywords available for configuration in the device-tracking configuration mode. Each policy parameter addresses one or more aspects of network security.

This section explains the purpose of *some* of the important policy parameters so you can configure your policy to better suit your requirements.

```

Device(config)# device-tracking policy example_policy
Device(config-device-tracking)# ?
device-tracking policy configuration mode:

  device-role      Sets the role of the device attached to the port
  limit            Specifies a limit
  security-level   setup security level
  tracking          Override default tracking behavior
  trusted-port     setup trusted port

```

For information about all the parameters displayed in the device-tracking configuration mode, see the command reference document of the corresponding release.

Glean versus Guard versus Inspect

When a packet enters the network, SISF extracts the IP and MAC address (the source of the packet) and subsequent action, is dictated by the security-level that is configured in the policy.

Glean, guard, and inspect are the options available under the security-level parameter. Glean is the least secure option, inspect, is moderately secure, and guard, is the most secure.

To configure this parameter in a policy, enter the **security-level** keyword in the device-tracking configuration mode.

Glean

When the security-level is set to **glean**, SISF extracts the IP and MAC address and enters them into the binding table, without any verification. This option therefore does not ensure binding integrity. It may for example, be suited to a set-up where client applications such as IEEE 802.1X or SANET want to only learn about the host and not rely on SISF for authentication.

The only factor that affects the addition of the binding entry for this security-level, is the address count limit. There are separate limits for the maximum number of IPs per port, IPv4 per MAC, and IPv6 per MAC. Entries are rejected once a limit is reached. For more information about this parameter, see [Address Count Limits, on page 17](#).

Guard

This is the default value for the security-level parameter.

When the security-level is set to **guard**, SISF extracts and verifies the IP and MAC address of packets entering the network. The outcome of the verification determines if a binding entry is added, or updated, or if the packet is dropped and the client is rejected.

The process of verification starts with the search for a matching entry in the database. The database may be centralised or distributed. If a matching entry is not found, a new entry is added.

If a matching entry is found and the points of attachment (MAC, VLAN, or interface) are found to be the same, only the timestamp is updated. If not, the scope of verification is extended to include validation of address ownership. This may include host polling to determine if the change in the point of attachment (a different MAC, or VLAN) is valid. If the change is valid the entry is updated, or if it is a case of theft, the entry is not added to the binding table.

If a binding entry is added or updated, the corresponding client is granted access to the network. If an entry does not pass verification, the corresponding client is rejected.



Note The verification process affects the binding entry and the corresponding incoming packet.

There are differences in the way SISF handles IPv4 and IPv6 control packets of a client that is rejected (entry did not pass verification). If the control packet of the rejected client is IPv6, the packet is dropped. If the control packet of the rejected client is IPv4, the packet is not dropped and allowed to enter the network.

Inspect

Even though security-level **inspect** is available on the CLI, we recommend not using it. The **glean** and **guard** options described above address most use cases and network requirements.

Trusted-Port and Device-Role Switch

The **device-role switch** and **trusted-port** options help you design an efficient and scalable secure zone. When used together, these two parameters help you achieve an efficient distribution of the creation of entries in the binding table. This keeps the binding tables size under control.

The **trusted-port** option: Disables the guard function on configured targets. Bindings learned through a trusted-port have preference over bindings learned through any other port. A trusted port is also given preference in case of a collision while making an entry in the table.

The **device-role** option: Indicates the type of device that is facing the port and this can be a node or a switch. To allow the creation of binding entries for a port, you configure the device as a node. To stop the creation of binding entries, you configure the device as switch.

Configuring the device as a switch is suited to multi-switch set-ups, where the possibility of large device tracking tables is very high. Here, a port facing a device (an uplink trunk port) can be configured to stop creating binding entries, and the traffic arriving at such a port can be trusted, because the switch on the other side of the trunk port will have device-tracking enabled and that will have checked the validity of the binding entry.



Note While there are scenarios where configuring only either one of these options may be suitable, the more common use case is for both the **trusted-port** and **device-role switch** options to be configured on the port - the examples below explain this in detail. Possible scenarios where only either one of these options is suited or required have also been described, at the end of this section.

To configure these parameters in a policy, enter the **trusted-port** and **device-role** keywords in the device-tracking configuration mode.

Example: Using Trusted-Port and Device-Role Switch Options in a Multi-Switch Set-Up

The following example explains how the **device-role switch** and **trusted-port** options help to design an efficient and scalable “secure zone”.

In figure [Figure 6: Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options, on page 10](#) below, SW_A, SW_B, and SW_C are three access switches. They are all connected to a common distribution switch. The only required configuration on the distribution switch in this scenario is to ensure that traffic of any kind is *not* blocked.

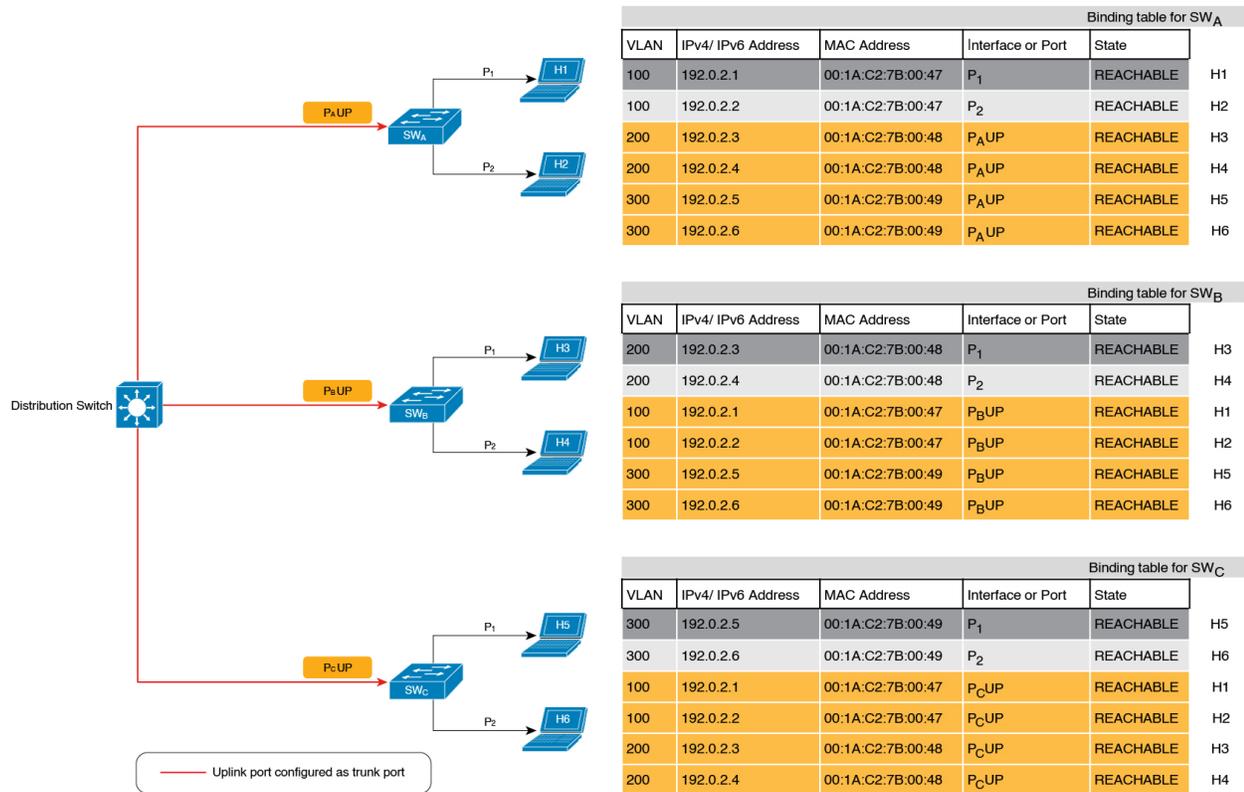
H1, H2, ... H6 are the hosts. Each switch has two directly connected hosts. All hosts are communicating with each other, that is, control packets are being transmitted. All hosts are also within the same VLAN boundary. Each switch is receiving control packets from hosts that are directly connected to it, and also from hosts that are connected to other switches. This means SW_A is receiving control packets from H1, H2, ... H6 similarly with SW_B and SW_C.

For each switch, the entries of directly connected hosts have interface or port P₁ and P₂ in the binding table. Entries originating from hosts that are connected to other switches have interface or port name P_xUP, to show that they have been learned through the uplink port (x represents the corresponding uplink port for each switch). For example, the entries that SW_A has learnt through its uplink port have interface or port name P_AUP and for SW_B it is P_BUP, and so forth.

The end result is that each switch learns and creates binding entries for all hosts in the set-up.

This scenario displays an inefficient use of the binding table, because each host is being validated multiple times, which does not make it more secure than if just one switch validates host. Secondly, entries for the same host in multiple binding tables could mean that the address count limit is reached sooner. After the limit is reached, any further entries are rejected and required entries may be missed this way.

Figure 6: Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options



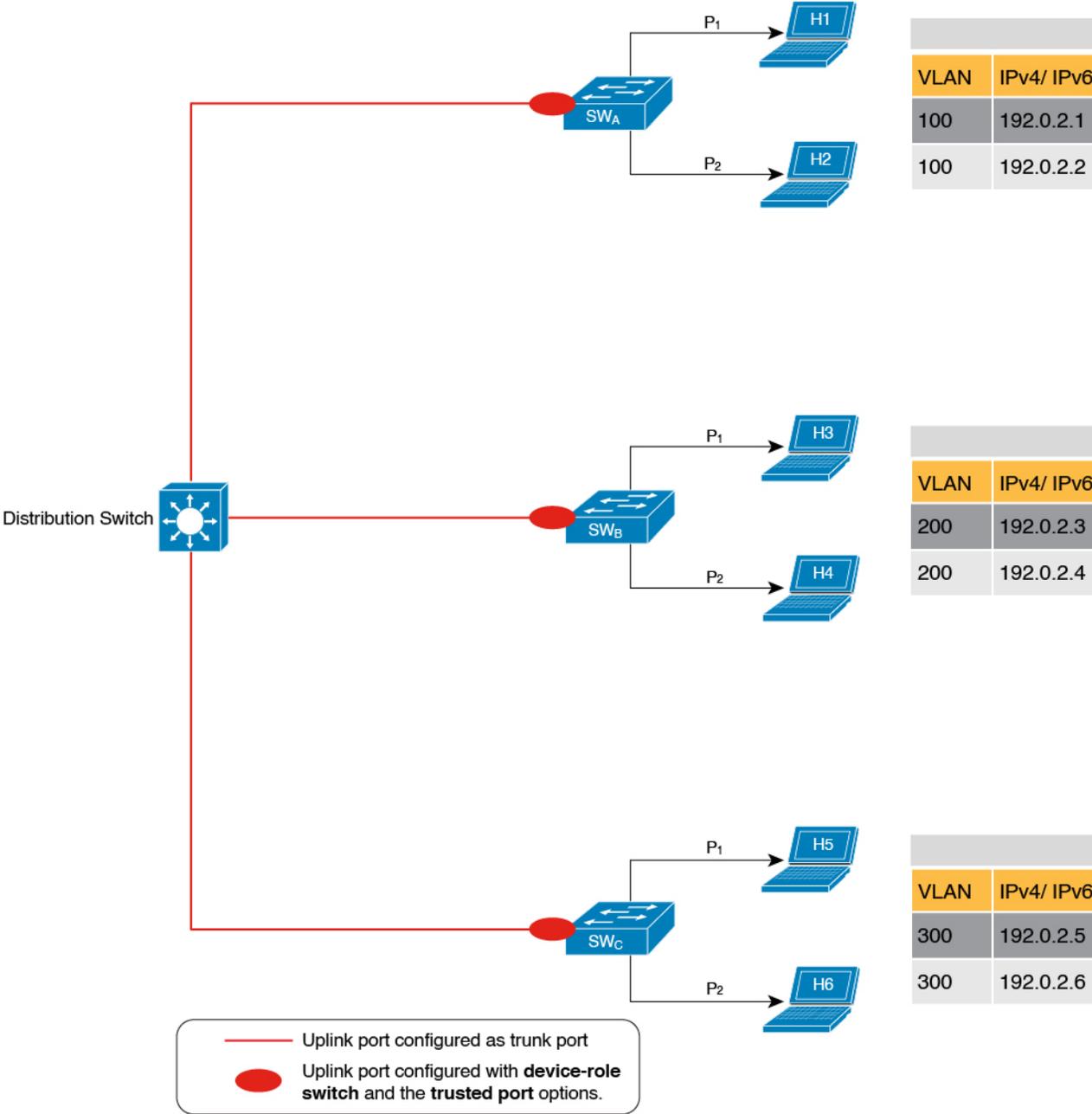
By contrast see figure [Figure 7: Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options](#), on page 12 below. Here when SW_A intercepts the packet of a host that is not attached to it (say H3 which is directly attached to SW_B), it does not create an entry because it detects that H3 is attached to a device that is

configured as a switch (**device-role switch** option) and the uplink port of the switch (where the packet came from) is a trusted port (**trusted-port** option).

By creating binding entries only on switches where the host appears on an access port (port P_1 and P_2 of each switch), and not creating entries for a host that appears over an uplink port or trusted port (P_x UP), each switch in the set-up validates and makes only the required entries, thus achieving an efficient distribution of the creation of binding table entries.

A second advantage of configuring **device-role switch** and **trusted-port** options in a multi-switch scenario is that it prevents duplicate entries when a host, say H1 moves from one switch to another. H1's IP and MAC binding in the earlier location (let's say SW_A) continues to remain there until it reaches the STALE state. But if H1 moves and connects to a second switch, say SW_C , then SW_A receives a duplicate binding entry through the uplink port. In such a situation, if the uplink port of the second switch (SW_C) is configured as a trusted port, SW_A deletes its stale entry. Further, it doesn't create another new binding entry because the SW_C will already have the latest entry and this entry is trusted.

Figure 7: Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options



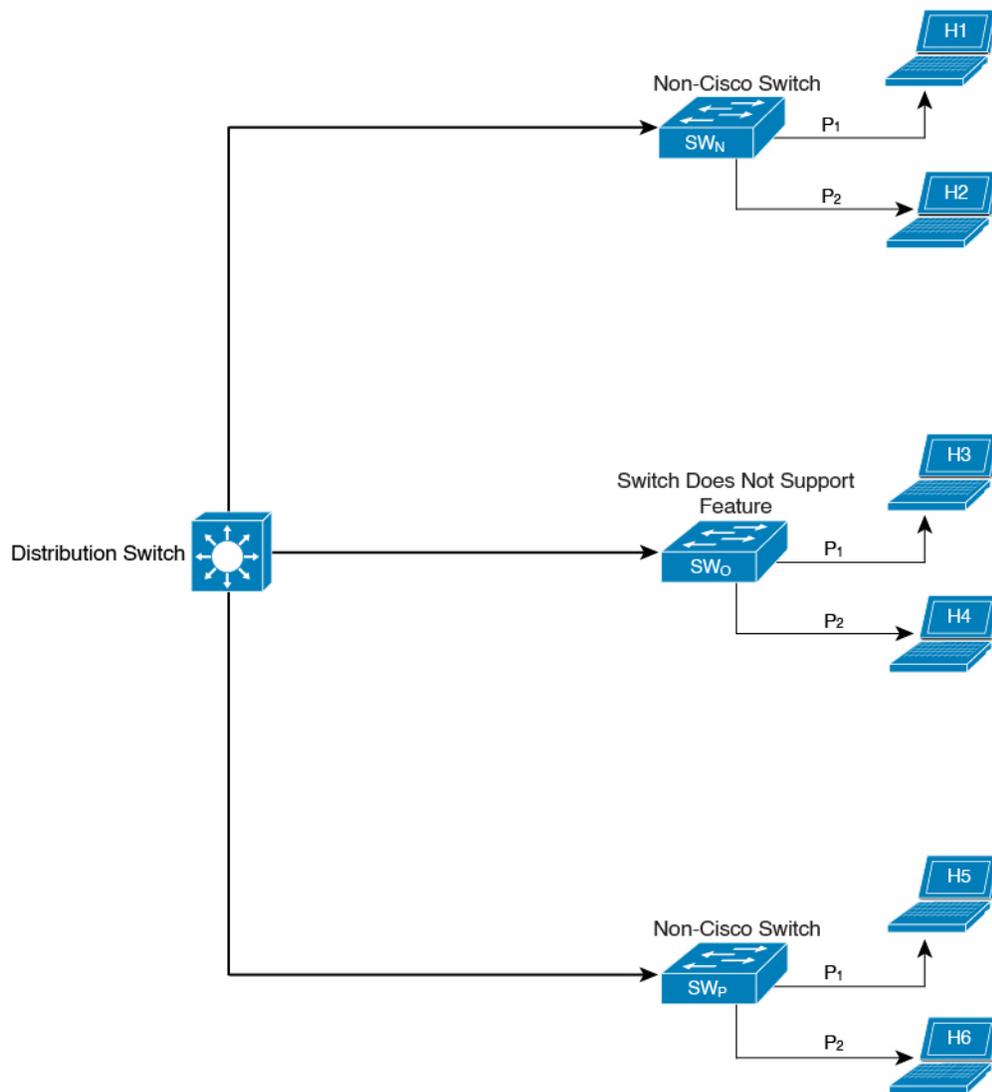
Example: When Not to Use Trusted-Port and Device-Role Switch Options

While the previous example clarifies how a multi-switch set-up with distributed binding tables stands to benefit from the **device-role switch** and **trusted-port** options, it may not suit networks of the following kinds:

- Networks where non-Cisco switches are being used
- Networks where the switch does not support the SISF-based device-tracking feature.

In both cases, we recommended that you not configure the **device-role switch** and **trusted-port** options. Further, we recommended that you maintain a centralised binding table - on the distribution switch. When you do, all the binding entries for all the hosts connected to non-Cisco switches and switches that do not support the feature, are validated by the distribution switch and still secure your network. The figure below illustrates the same.

Figure 8: Centralised Binding Table



VLAN	IPv4/I
100	192.0.0.
100	192.0.0.
200	192.0.0.
200	192.0.0.
300	192.0.0.
300	192.0.0.

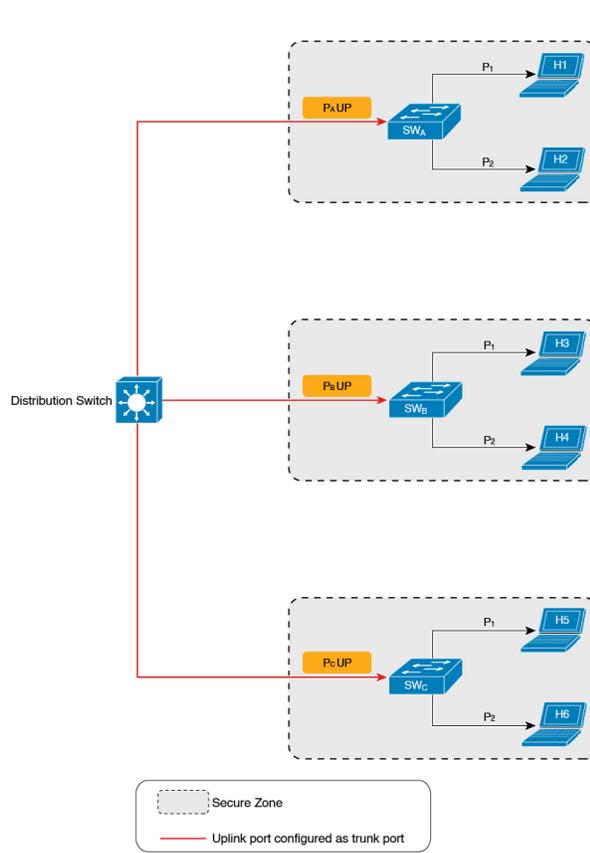
Creating an Efficient and Scalable Secure Zone

By using the **trusted-port** and **device-role switch** options in suitable networks and leaving them out in others, you can achieve an efficient and scalable secure zone.

Secure Zones 1, 2 and 3, display three different set-ups and the secure zone that is established in each case.

Secure Zone:	Figure 9: Secure Zone 1 - Inefficient and Unscalable Secure Zone, on page 15	Figure 10: Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized, on page 16	Figure 11: Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized, on page 17
Scalability:	Unscalable; each switch has entries of all the hosts in the network	Scalable; each switch as entries of only directly connected hosts	Unscalable; the distribution switch has entries of all hosts in the network
Polling and its effect on the network: n = number of hosts m = number of switches total number of polling requests: = n X m	18 polling requests are being sent (6 hosts x 3 switches). Each host is polled by all the switches in the network (in the absence of the trusted-port and device-role switch options). Network load is very high.	6 polling requests are being sent (2 hosts x 1 switch for <i>each</i> switch). Minimal network load. (Polling requests are sent by the local access switches to directly connected hosts, each polling request passes through fewer points in the network.)	6 polling requests are being sent (6 hosts x 1 switch) Network load is higher than secure zone 2, but not as high as secure zone 1. (Polling requests come from the distribution switch and go through the access switch before reaching the host.)
Efficiency:	Inefficient binding table, because the binding table is duplicated on each switch.	Efficient binding table, because each host's binding information is entered only once, and in one binding table and this the binding table of the directly connected switch.	Efficient binding table, because the binding information for each host is entered only once, and this is in the central binding table, which is on the distribution switch.
Recommended Action:	Reapply suitable policies to make the secure zone like secure zone 2	None; this is an efficient and scalable secure zone.	None; this is the best possible secure zone given the type of set-up (where the other switches in the network are either non-Cisco or do not support the feature)

Figure 9: Secure Zone 1 - Inefficient and Unscalable Secure Zone



Binding table for SW_A

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State
100	192.0.2.1	00:1A:C2:7B:00:47	P ₁	REACHABLE
100	192.0.2.2	00:1A:C2:7B:00:47	P ₂	REACHABLE
200	192.0.2.3	00:1A:C2:7B:00:48	P _A UP	REACHABLE
200	192.0.2.4	00:1A:C2:7B:00:48	P _A UP	REACHABLE
300	192.0.2.5	00:1A:C2:7B:00:49	P _A UP	REACHABLE
300	192.0.2.6	00:1A:C2:7B:00:49	P _A UP	REACHABLE

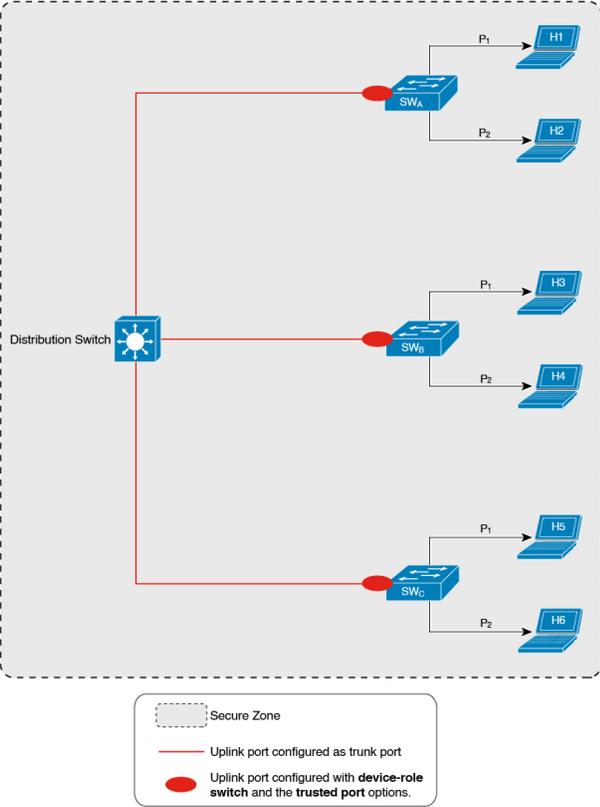
Binding table for SW_B

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State
200	192.0.2.3	00:1A:C2:7B:00:48	P ₁	REACHABLE
200	192.0.2.4	00:1A:C2:7B:00:48	P ₂	REACHABLE
100	192.0.2.1	00:1A:C2:7B:00:47	P _B UP	REACHABLE
100	192.0.2.2	00:1A:C2:7B:00:47	P _B UP	REACHABLE
300	192.0.2.5	00:1A:C2:7B:00:49	P _B UP	REACHABLE
300	192.0.2.6	00:1A:C2:7B:00:49	P _B UP	REACHABLE

Binding table for SW_C

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State
300	192.0.2.5	00:1A:C2:7B:00:49	P ₁	REACHABLE
300	192.0.2.6	00:1A:C2:7B:00:49	P ₂	REACHABLE
100	192.0.2.1	00:1A:C2:7B:00:47	P _C UP	REACHABLE
100	192.0.2.2	00:1A:C2:7B:00:47	P _C UP	REACHABLE
200	192.0.2.3	00:1A:C2:7B:00:48	P _C UP	REACHABLE
200	192.0.2.4	00:1A:C2:7B:00:48	P _C UP	REACHABLE

Figure 10: Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized



Binding table for SW_A

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
100	192.0.2.1	00:1A:C2:7B:00:47	P ₁	REACHABLE	H1
100	192.0.2.2	00:1A:C2:7B:00:47	P ₂	REACHABLE	H2

Binding table for SW_B

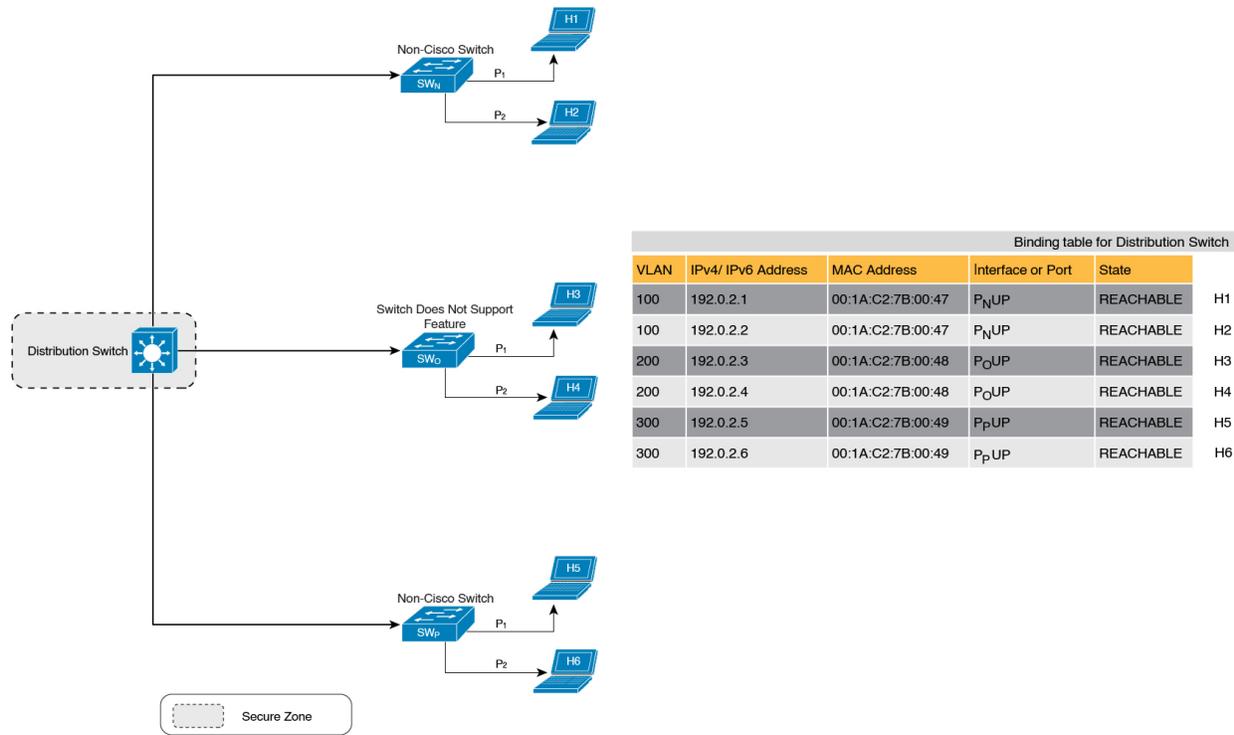
VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
200	192.0.2.3	00:1A:C2:7B:00:48	P ₁	REACHABLE	H3
200	192.0.2.4	00:1A:C2:7B:00:48	P ₂	REACHABLE	H4

Binding table for SW_C

VLAN	IPv4/ IPv6 Address	MAC Address	Interface or Port	State	
300	192.0.2.5	00:1A:C2:7B:00:49	P ₁	REACHABLE	H5
300	192.0.2.6	00:1A:C2:7B:00:49	P ₂	REACHABLE	H6

468701

Figure 11: Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized



When to Use Only Trusted-Port or Only Device-Role Switch

Configuring only **device-role switch** is suited to situations when you want to listen but not learn entries. For example, for Duplicate Address Detection (DAD), or when you want to send IPv6 or Neighbor Solicitation (NS) message on a switch-facing port.

When you configure this option on a switch port (or interface), SISF-based device-tracking treats the port as a trunk port, implying that the port is connected to other switches. It does not matter whether the port is actually a trunk port or not. Therefore, when NS packets or queries are sent to switches in the network for new entry validation, only the secure ports (ports where the **device-role switch** is configured) receive the packet or query. This safeguards the network. If the command is not configured on any port, a general broadcast of the query is sent.

Configuring only **trusted-port** is suited to situations where an access port should be configured as a trusted port. If an access port is connected to a DHCP server or a similar service that the switch is consuming, configuring an access port as a trusted port ensures that the service is not disrupted because traffic from such a port is trusted. This also widens the secure zone, to include the access port.

Address Count Limits

The address count limit parameter specifies limits for the number of IP and MAC addresses that can be entered in a binding table. The purpose of these limits is to contain the size of the binding table based on the number of known and expected hosts, thus enabling you to take pre-emptive action against rogue hosts or IPs in the network.

At a policy level there are separate limits for the number of IP addresses per port, the number of IPv4 addresses per MAC, and IPv6 addresses per MAC. You can configure or change only the number of IP addresses per port.

IP per Port

The IP per port option is the total number of IP addresses allowed for a port. The address can be IPv4 or IPv6. When the limit is reached, no further IP addresses (i.e., entries) are added to the binding table.

To configure this parameter in a policy, enter the **limit address-count** *ip-per-port* keyword in the device-tracking configuration mode. If you configure a limit that is lower than the currently configured one, then the new (lower) limit is applicable only to new entries. An existing entry remains in the binding table and goes through its binding entry lifecycle.

IPv4 per MAC and IPv6 per MAC

The number of IPv4 addresses that can be mapped to one MAC address and the number of IPv6 addresses that can be mapped to one MAC address. When the limit is reached, no further entries can be added to the binding table, and traffic from new hosts will be dropped.



Note The IPv4 per MAC limit and the IPv6 per MAC limit that is effective on an interface or VLAN is as defined in the policy that is applied. If the policy does not specify a limit, this means that a limit does not exist. You cannot change or configure a limit for IPv4 per MAC or IPv6 per MAC for any kind of policy (programmatic, or custom policy, or default policy).

Enter the **show device-tracking policy** *policy name* to check if a limit exists. The following is sample output of a policy where an IPv4 per MAC and an IPv6 per MAC limit exists:

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN
Policy LISP-DT-GUARD-VLAN configuration:
  security-level guard (*)
  <output truncated>

  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable

<output truncated>
```

Overall Address Count Limit Considerations

- The limits do not have a hierarchy, but the threshold that is set for each limit affects the others.

For example, if the IP per port limit is 100, and the IPv4 per MAC limit is one. The limit is reached with a single host's IPv4-MAC binding entry. No further entries are allowed even though the port has a provision for 99 more IP addresses.

- Address count limits and the security-level parameter.

For information about how the address count limits interact with the security-level parameter **glean**, see [Glean versus Guard versus Inspect, on page 8](#).

When the security-level parameter is **guard**, reaching an address count limit results in a rejection of the entry. This has the following effect on the incoming packet:

- If the incoming packet is IPv4, the packet is allowed to go through even though the entry is rejected.
 - If the incoming packet is IPv6, a rejected entry means that the packet is also dropped.
- Global and policy-level limits

The limits configured with the **device-tracking binding max-entries** command are at the global level, the limits configured with the **limit address-count** command in the device-tracking configuration mode are for a policy, which is at the interface or VLAN level.

If a policy-level value *and* a globally configured value exists, the creation of binding entries is stopped when *a* limit is reached - this can be any one of the global values or the policy-level value.

If only globally configured values exist, the creation of binding entries is stopped when *a* limit is reached.

If only a policy-level value exists, the creation of binding entries is stopped when the policy-level limit is reached.

Tracking

The tracking parameter involves tracking of hosts in the network. In section [Polling a Host and Updating the Binding Table Entry, on page 5](#) above, this is referred to as "polling". It also describes polling behaviour in detail.

To configure polling parameters at the global level, enter the **device-tracking tracking** command in global configuration mode. After you configure this command you still have the flexibility to turn polling on or off, for individual interfaces and VLANs. For this you must enable or disable polling in the policy.

To enable polling in a policy, enter the **tracking enable** keywords in the device-tracking configuration mode. By default, polling is disabled in a policy.

Guidelines for Policy Creation

- If multiple policies are available on a given target, a system-internal policy priority determines which policy takes precedence.

A manually created policy has the highest priority. When you want to override the settings of a programmatically created policy, you can create a custom policy, so it has higher priority.

- The parameters of a programmatically created policy cannot be changed. You can configure certain attributes of a custom policy.

Guidelines for Applying a Policy

- Multiple policies can be attached to the same VLAN.
- If a programmatic policy is attached to a VLAN and you want to change policy settings, create a custom device-tracking policy and attach it to the VLAN.
- When multiple policies with different priorities are attached to the same VLAN, the settings of the policy with the highest priority are effective. The exceptions here are the limit address-count for IPv4 per mac and limit address-count for IPv6 per mac settings - the settings of the policy with the lowest priority are effective.

- When a device-tracking policy is attached to an interface under a VLAN, the policy settings on the interface take precedence over those on its VLAN; exceptions here are the values for limit address-count for IPv4 per mac and limit address-count for IPv6 per mac, which are aggregated from the policy on both the interface and VLAN.
- A policy cannot be removed unless the device tracking client feature configuration is removed.

How to Configure SISF

SISF or SISF-based device-tracking, is disabled by default. You enable it by defining a device-tracking policy and attaching the policy to a specific target. The target could be an interface or a VLAN. There are multiple ways to define a policy and no single method is a preferred or recommended one - use the option that suits your requirements.

Method of Enabling SISF	Applicable Configuration Tasks	Result
Option 1: Manually, by using interface configuration commands to create and apply the default policy to a target.	Applying the Default Device Tracking Policy to a Target, on page 21	Automatically applies the default device tracking policy to the specified target. The default policy is a built-in policy with default settings; you cannot change any of the attributes of the default policy. See Option 2 if you want to configure device tracking policy attributes.
Option 2: Manually, by using global configuration commands to create a custom policy and applying the custom policy to a target.	<ol style="list-style-type: none"> 1. Creating a Custom Device Tracking Policy with Custom Settings, on page 22 2. Attach the custom policy to an interface or VLAN: Attaching a Device Tracking Policy to an Interface, on page 26 OR Attaching a Device Tracking Policy to a VLAN, on page 27 	Creates a custom policy with the name and policy parameters you configure, and attaches the policy to the specified target.

Method of Enabling SISF	Applicable Configuration Tasks	Result
Option 3: Programmatically, by configuring the snooping command.	Enter the ip dhcp snooping vlan <i>vlan</i> command in global configuration mode. See section <i>Example: Programmatically Enabling SISF by Configuring DHCP Snooping</i> in Example: Programmatically Enabling SISF-Based Device Tracking in Cisco IOS XE Fuji 16.9.x , on page 30.	When you configure the command, the system automatically creates policy <code>DT-PROGRAMMATIC</code> . Use this method if you want to enable SISF-based device tracking for these clients: IEEE 802.1X, Web authentication, Cisco TrustSec, IP Source Guard, and SANET.
Option 4: Programmatically, by configuring Locator ID Separation Protocol (LISP).	See sections <i>Example: Programmatically enabling SISF by Configuring LISP [LISP-DT-GUARD-VLAN]</i> and <i>Example: Programmatically enabling SISF by Configuring LISP [LISP-DT-GLEAN-VLAN]</i> in Example: Programmatically Enabling SISF-Based Device Tracking in Cisco IOS XE Fuji 16.9.x , on page 30.	When you configure LISP, the system automatically creates policy <code>LISP-DT-GUARD-VLAN</code> or <code>LISP-DT-GLEAN-VLAN</code> .
Option 5: Programmatically, by configuring EVPN VLAN.	See section <i>Example: Programmatically Enabling SISF by EVPN on VLAN</i> in Example: Programmatically Enabling SISF-Based Device Tracking in Cisco IOS XE Fuji 16.9.x , on page 30.	When you configure EVPN on VLAN, the system automatically creates policy <code>evpn-sisf-policy</code> .
Option 6: Migrating from legacy IPDT and IPv6 Snooping.	Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device Tracking , on page 27 IPDT, IPv6 Snooping, and SISF-Based Device Tracking CLI Compatibility , on page 29	Convert legacy IPDT and IPv6 Snooping configuration to the SISF-based device-tracking commands.

Applying the Default Device Tracking Policy to a Target

Beginning in privileged EXEC mode, follow these steps to apply the default device tracking policy to an interface or VLAN:

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: Device# configure terminal	Enters the global configuration mode.
Step 2	Specify an interface or a VLAN <ul style="list-style-type: none"> • interface <i>interface</i> • vlan configuration <i>vlan_list</i> Example: Device(config)# interface gigabitethernet 1/1/4 OR Device(config)# vlan configuration 333	interface <i>type number</i> —Specifies the interface and enters the interface configuration mode. The device tracking policy will be attached to the specified interface. vlan configuration <i>vlan_list</i> —Specifies the VLANs and enters the VLAN feature configuration mode. The device tracking policy will be attached to the specified VLAN.
Step 3	device-tracking Example: Device(config-if)# device-tracking OR Device(config-vlan-config)# device-tracking	Enables SISF-based device tracking and attaches the default policy it to the interface or VLAN. The default policy is a built-in policy with default settings; none of the attributes of the default policy can be changed.
Step 4	exit Example: Device(config-if)# exit OR Device(config-vlan-config)# exit	Exits configuration mode.
Step 5	show device-tracking policy <i>policy-name</i> Example: Device# show device-tracking policy default	Displays device-tracking policy configuration, and all the targets it is applied to.

Creating a Custom Device Tracking Policy with Custom Settings

Beginning in privileged EXEC mode, follow these steps to create and configure a device tracking policy:

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: Device# configure terminal	Enters the global configuration mode.

	Command or Action	Purpose
<p>Step 2</p>	<p>[no] device-tracking policy <i>policy-name</i></p> <p>Example:</p> <pre>Device(config)# device-tracking policy example_policy</pre>	<p>Creates the policy and enters the device-tracking configuration mode.</p>
<p>Step 3</p>	<p>[data-glean default destination-glean device-role distribution-switch exit limit no prefix-glean protocol security-level tracking trusted-port vpc]</p> <p>Example:</p> <pre>Device (config-device-tracking)# destination-glean log-only</pre>	<p>Enter the question mark (?) at the system prompt to obtain a list of available options in this mode. You can configure the following for both IPv4 and IPv6:</p> <ul style="list-style-type: none"> • (Optional) data-glean—Enables learning of addresses from a data packet snooped from a source inside the network and populates the binding table with the data traffic source address. Enter one of these options: <ul style="list-style-type: none"> • log-only—Generates a syslog message upon data packet notification • recovery—Uses a protocol to enable binding table recovery. Enter NDP or DHCP. • (Optional) default—Sets the policy attribute to its default value. You can set these policy attributes to their default values: data-glean, destination-glean, device-role, limit, prefix-glean, protocol, security-level, tracking, trusted-port. • (Optional) destination-glean—Populates the binding table by gleaning data traffic destination address. Enter one of these options: <ul style="list-style-type: none"> • log-only—Generates a syslog message upon data packet notification • recovery—Uses a protocol to enable binding table recovery. Enter DHCP. • (Optional) device-role—Sets the role of the device attached to the port. It can be a node or a switch. Enter one of these options: <ul style="list-style-type: none"> • node—Configures the attached device as a node. This is the default option. • switch—Configures the attached device as a switch.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • (Optional) distribution-switch—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect. • exit—Exits the device-tracking policy configuration mode. • limit address-count—Specifies an address count limit per port. The range is 1 to 32000. • no—Negates the command or sets it to defaults. • (Optional) prefix-glean—Enables learning of prefixes from either IPv6 Router Advertisements or from DHCP-PD. You have the following option: <ul style="list-style-type: none"> • (Optional) only—Gleans only prefixes and not host addresses. • (Optional) protocol—Sets the protocol to glean; by default, all are gleaned. Enter one of these options: <ul style="list-style-type: none"> • arp [prefix-list <i>name</i>]—Gleans addresses in ARP packets. Optionally, enter the name of prefix-list that is to be matched. • dhcp4 [prefix-list <i>name</i>]—Glean addresses in DHCPv4 packets. Optionally, enter the name of prefix-list that is to be matched. • dhcp6 [prefix-list <i>name</i>]—Glean addresses in DHCPv6 packets. Optionally, enter the name of prefix-list that is to be matched. • ndp [prefix-list <i>name</i>]—Glean addresses in NDP packets. Optionally, enter the name of prefix-list that is to be matched. • udp [prefix-list <i>name</i>]—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • (Optional) security-level—Specifies the level of security enforced by the feature. Enter one of these options: <ul style="list-style-type: none"> • glean—Gleans addresses passively. • guard—Inspects and drops un-authorized messages. This is the default. • inspect—Gleans and validates messages. • (Optional) tracking—Specifies a tracking option. Enter one of these options: <ul style="list-style-type: none"> • disable [stale-lifetime [<i>1-86400-seconds</i> infinite]] —Turns off device-tracking. Optionally, you can enter the duration for which the entry is kept inactive before deletion, or keep it permanently inactive. • enable [reachable-lifetime [<i>1-86400-seconds</i> infinite]] —Turns on device-tracking. Optionally, you can enter the duration for which the entry is kept reachable, or keep it permanently reachable. • (Optional) trusted-port—Sets up a trusted port. Disables the guard on applicable targets. Bindings learned through a trusted port have preference over bindings learned through any other port. A trusted port is given preference in case of a collision while making an entry in the table. • (Optional) vpc—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect.
Step 4	end Example: Device(config-device-tracking)# exit	Exits configuration mode.
Step 5	show device-tracking policy <i>policy-name</i> Example:	Displays the device-tracking policy configuration.

	Command or Action	Purpose
	Device# <code>show device-tracking policy example_policy</code>	

What to do next

Attach the policy to an interface or VLAN.

Attaching a Device Tracking Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach a device tracking policy to an interface:

Procedure

	Command or Action	Purpose
Step 1	<code>configure terminal</code> Example: Device# <code>configure terminal</code>	Enters the global configuration mode.
Step 2	<code>interface interface</code> Example: Device (config)# <code>interface gigabitethernet 1/1/4</code>	Specifies an interface and enters the interface configuration mode.
Step 3	<code>[no] device-tracking attach-policy policy name</code> Example: Device (config-if)# <code>device-tracking attach-policy example_policy</code>	Attaches the device tracking policy to the interface. Note SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed.
Step 4	<code>end</code> Example: Device# <code>end</code>	Returns to the privileged EXEC mode.
Step 5	<code>show device-tracking policies [interface interface]</code> Example: Device# <code>show device-tracking policies interface gigabitethernet 1/1/4</code>	Displays policies that match the specified interface type and number.

Attaching a Device Tracking Policy to a VLAN

Beginning in privileged EXEC mode, follow these steps to attach a device-tracking policy to VLANs across multiple interfaces:

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: Device# <code>configure terminal</code>	Enters the global configuration mode.
Step 2	vlan configuration <i>vlan_list</i> Example: Device(config)# <code>vlan configuration 333</code>	Specifies the VLANs to which the device tracking policy will be attached; enters the VLAN interface configuration mode.
Step 3	[no] device-tracking attach-policy <i>policy_name</i> Example: Device(config-vlan-config)# <code>device-tracking attach-policy</code> <code>example_policy</code>	Attaches the device tracking policy to the specified VLANs across all switch interfaces. Note SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed.
Step 4	do show device-tracking policies <i>vlan</i> <i>vlan-ID</i> Example: Device(config-vlan-config)# <code>do show</code> <code>device-tracking policies vlan 333</code>	Verifies that the policy is attached to the specified VLAN, without exiting the VLAN interface configuration mode.

Migrating from Legacy Commands to SISF-Based Device-Tracking Commands

Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device Tracking

Starting with Cisco IOS XE Denali 16.1.1, the existing IPv6 snooping and IP Device Tracking (IPDT) commands have corresponding SISF-based device-tracking commands that allow you to apply your configuration to both IPv4 and IPv6 address families.

After you have upgraded from a Cisco IOS XE 3.x.x release to a Cisco IOS XE 16.x.x release, enter the **device-tracking upgrade-cli** to convert legacy IPDT and IPv6 Snooping commands to SISF-based device tracking commands. After you run the command, only the new device-tracking commands are available on your device and the legacy commands are not supported.

Based on the legacy configuration that exists on your device, the **device-tracking upgrade-cli** command upgrades your CLI differently. Consider the following configuration scenarios and the corresponding migration results before you migrate your existing configuration.



Note You cannot configure a mix of the old IPDT and IPv6 snooping CLI with the new SISF-based device-tracking CLI.

Only IPDT Configuration Exists

If your device has only IPDT configuration, running the **device-tracking upgrade-cli** command converts the configuration to use the new SISF policy that is created and attached to the interface. You can then update this SISF policy.

If you continue to use the legacy commands, this restricts you to operate in a legacy mode where only the legacy IPDT and IPv6 snooping commands are available on the device.

Only IPv6 Snooping Configuration Exists

On a device with existing IPv6 snooping configuration, the old IPv6 Snooping commands are available for further configuration. The following options are available:

- (Recommended) Use the **device-tracking upgrade-cli** command to convert all your legacy configuration to the new SISF-based device tracking commands. After conversion, only the new device tracking commands will work on your device.
- Use the legacy IPv6 Snooping commands for your future configuration and do not run the **device-tracking upgrade-cli** command. With this option, only the legacy IPv6 Snooping commands are available on your device, and you cannot use the new SISF-based device tracking CLI commands.

Both IPDT and IPv6 Snooping Configuration Exist

On a device that has both legacy IPDT configuration and IPv6 snooping configuration, you can convert legacy commands to the SISF-based device tracking CLI commands. However, note that only one snooping policy can be attached to an interface, and the IPv6 snooping policy parameters override the IPDT settings.



Note If you do not migrate to the new SISF-based commands and continue to use the legacy IPv6 snooping or IPDT commands, your IPv4 device tracking configuration information may be displayed in the IPv6 snooping commands, as the SISF-based device tracking feature handles both IPv4 and IPv6 configuration. To avoid this, we recommend that you convert your legacy configuration to SISF-based device tracking commands.

No IPDT or IPv6 Snooping Configuration Exists

If your device has no legacy IP Device Tracking or IPv6 Snooping configurations, you can use only the new SISF-based device tracking commands for all your future configuration. The legacy IPDT commands and IPv6 snooping commands are not available.



Note Starting from Cisco IOS XE Denali 16.3.1, the **ip dhcp snooping vlan** *vlan* command creates a device tracking policy programmatically, to support the IEEE 802.1X, web authentication, Cisco TrustSec and IPSG features. The programmatically created policy tracks both IPv4 and IPv6 clients. Ensure that this command is configured, if you are using any of the aforementioned features.

IPDT, IPv6 Snooping, and SISF-Based Device Tracking CLI Compatibility

Table [Table 1: IPDT → IPv6 Snooping Commands, on page 29](#) displays legacy IPDT and then the IPv6 snooping commands they are converted to - if the **device-tracking upgrade-cli** command (global configuration mode) is NOT executed.

Table [Table 2: IPDT → SISF Commands, on page 30](#) displays legacy IPDT and then the SISF-based device-tracking commands that the system converts them to, if you have executed the **device-tracking upgrade-cli** command.

Table 1: IPDT → IPv6 Snooping Commands

Legacy IP Device Tracking (IPDT)	IPv6 Snooping Command (Starting from Cisco IOS XE Denali 16.3.7 and all later Cisco IOS XE 16.x.x releases).
ip device tracking probe count	Set to the default value, and cannot be changed.
ip device tracking probe delay	Set to the default value, and cannot be changed ² .
ip device tracking probe interval	ipv6 neighbor binding reachable-lifetime ³
ip device tracking probe use-svi	Set to the default behavior, and cannot be changed.
ip device tracking probe auto-source [fallback <i>host-ip-address subnet-mask</i>] [override]	ipv6 neighbor tracking auto-source [fallback <i>host-ip-address subnet-mask</i>] [override]
ip device tracking trace-buffer	Not supported
ip device tracking maximum n	ipv6 snooping policy <i>IPDT_MAX_n</i> [limit <i>address-count</i>]
ip device tracking maximum 0	Not supported
clear ip device tracking all	Not supported

² Until Cisco IOS XE Denali 16.3.6 and in Cisco IOS XE Everest 16.5.1a, the system incorrectly converts the **ip device tracking probe delay** command to **ipv6 neighbor binding reachable-lifetime**. Starting from Cisco IOS XE Denali 16.3.7 (except in Cisco IOS XE Everest 16.5.x), this is set to the default value and cannot be changed.

³ Until Cisco IOS XE Denali 16.3.6 and in Cisco IOS XE Everest 16.5.1a, the system incorrectly converts the **ip device tracking probe interval** command to **ipv6 snooping tracking retry-interval**. Starting from Cisco IOS XE Denali 16.3.7 (except in Cisco IOS XE Everest 16.5.x), this is correctly converted to **ipv6 neighbor binding reachable-lifetime**.

Table 2: IPDT → SISF Commands

Legacy IP Device Tracking (IPDT)	SISF-Based Device-Tracking After SISF Conversion (Starting from Cisco IOS XE Denali 16.3.7 and all later Cisco IOS XE 16.x.x releases).
ip device tracking probe count	Set to the default value, and cannot be changed.
ip device tracking probe delay	Set to the default value, and cannot be changed ⁴ .
ip device tracking probe interval	device-tracking binding reachable-lifetime ⁵
ip device tracking probe use-svi	Set to the default behaviour and cannot be changed.
ip device tracking probe auto-source [fallback <i>host-ip-address</i> <i>subnet-mask</i>] [override]	device-tracking tracking auto-source [fallback <i>host-ip-address</i> <i>subnet-mask</i>] [override]
ip device tracking trace-buffer	Not supported.
ip device tracking maximum n	device-tracking snooping policy <i>IPDT_MAX_n</i> [limit <i>address-count</i>]
ip device tracking maximum 0	Not supported.
clear ip device tracking all	Not supported.

⁴ Until Cisco IOS XE Denali 16.3.6 and in Cisco IOS XE Everest 16.5.1a, the system incorrectly converts the **ip device tracking probe delay** command to **device-tracking binding reachable-lifetime**. Starting from Cisco IOS XE Denali 16.3.7 (except in Cisco IOS XE Everest 16.5.x), this is set to the default value, and cannot be changed.

⁵ Until Cisco IOS XE Denali 16.3.6 and in Cisco IOS XE Everest 16.5.1a, the system incorrectly converts the **ip device tracking probe interval** command to **device-tracking tracking retry-interval**. Starting from Cisco IOS XE Denali 16.3.7 (except in Cisco IOS XE Everest 16.5.1a), this is correctly converted to **device-tracking binding reachable-lifetime**.

Configuration Examples for SISF

Example: Programmatically Enabling SISF-Based Device Tracking in Cisco IOS XE Fuji 16.9.x

The sample output in the examples show the different settings of programmatically created policies.

Device tracking client: LISP on VLAN

After you configure LISP, enter the **show device-tracking policy** command in privileged EXEC mode, to display the `LISP-DT-GUARD-VLAN` policy that is created and the corresponding settings.

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN
Policy LISP-DT-GUARD-VLAN configuration:
```

```

security-level guard (*)
device-role node
gleaning from Neighbor Discovery
gleaning from DHCP
gleaning from ARP
gleaning from DHCP4
NOT gleaning from protocol unkn
limit address-count for IPv4 per mac 4 (*)
limit address-count for IPv6 per mac 12 (*)
tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target      Type      Policy                               Feature      Target range
vlan 10     VLAN     LISP-DT-GUARD-VLAN                 Device-tracking  vlan all
note:
Binding entry Down timer: 10 minutes (*)
Binding entry Stale timer: 30 minutes (*)

```

Device tracking client: LISP on VLAN

After you configure LISP, enter the **show device-tracking policy** command in privileged EXEC mode, to display the LISP-DT-GLEAN-VLAN policy that is created and the corresponding settings:

```

Device# show device-tracking policy LISP-DT-GLEAN-VLAN
Policy LISP-DT-GLEAN-VLAN configuration:
security-level glean (*)
device-role node
gleaning from Neighbor Discovery
gleaning from DHCP
gleaning from ARP
gleaning from DHCP4
NOT gleaning from protocol unkn
limit address-count for IPv4 per mac 4 (*)
limit address-count for IPv6 per mac 12 (*)
tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target      Type      Policy                               Feature      Target range
vlan 10     VLAN     LISP-DT-GLEAN-VLAN                 Device-tracking  vlan all
note:
Binding entry Down timer: 10 minutes (*)
Binding entry Stale timer: 30 minutes (*)

```

Device tracking client: EVPN on VLAN

After you configure EVPN, enter the **show device-tracking policy** command in privileged EXEC mode, to display the evpn-sisf-policy policy that is created and the corresponding settings that are made:

```

Device# show device-tracking policy evpn-sisf-policy
Policy evpn-sisf-policy configuration:
security-level glean (*)
device-role node
gleaning from Neighbor Discovery
gleaning from DHCP
gleaning from ARP
gleaning from DHCP4
NOT gleaning from protocol unkn
tracking enable
Policy evpn-sisf-policy is applied on the following targets:
Target      Type      Policy                               Feature      Target range
vlan 10     VLAN     evpn-sisf-policy                   Device-tracking  vlan all
note:

```

Example: Mitigating the IPv4 Duplicate Address Problem

```
Binding entry Down timer: 24 hours (*)
Binding entry Stale timer: 24 hours (*)
```

Device tracking clients: IEEE 802.1X, Web Authentication, Cisco TrustSec, IPSG

Configure the **ip dhcp snooping vlan** *vlan* command in global configuration mode to enable device-tracking for the IEEE 802.1X, web authentication, Cisco TrustSec, and IPSG features. Enter the **show device-tracking policy** command in privileged EXEC mode, to display the `DT-PROGRAMMATIC` policy that is created and the corresponding settings that are made:

```
Device# configure terminal
Device(config)# ip dhcp snooping vlan 10
Device(config)# end
Device# show device-tracking policy DT-PROGRAMMATIC
Policy DT-PROGRAMMATIC configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 1 (*)
  tracking enable
Policy DT-PROGRAMMATIC is applied on the following targets:
Target      Type      Policy          Feature          Target range
vlan 10     VLAN     DT-PROGRAMMATIC Device-tracking   vlan all

note:
Binding entry Down timer: 24 hours (*)
Binding entry Stale timer: 24 hours (*)
```

Identifying the Active Policy When Multiple Policies are Applied to a Target

This example shows you how to identify the active policy when multiple policies are attached to the same VLAN.

In this example, two policies are attached to VLAN 10; LISP-DT-GUARD-VLAN is the active policy.

```
Device# show device-tracking policies
Target      Type      Policy          Feature          Target range
vlan 10     VLAN     DT-PROGRAMMATIC Device-tracking   vlan all
vlan 10     VLAN     LISP-DT-GUARD-VLAN Device-tracking   vlan all

Device# show device-tracking capture-policy vlan 10
HW Target vlan 10 HW policy signature 0001DF9F policies#:2 rules 14 sig 0001DF9F
SW policy DT-PROGRAMMATIC feature Device-tracking -

SW policy LISP-DT-GUARD-VLAN feature Device-tracking - Active
```

Example: Mitigating the IPv4 Duplicate Address Problem

For an IPv4 device-tracking entry, its reachability is verified by sending a SISF probe to its end-node, which is an ARP request message. Selection of the source IP address for this ARP probe follows these rules:

- If an SVI is configured on the VLAN, the IPv4 address of the SVI is selected. Please ensure that the SVI IP address is unique in the subnet.

- If SVI does not exist and the **device-tracking tracking auto-source** [**fallback** *host-ip mask*] [**override**] command is configured, source IP is selected according to the [table](#) below.
- Otherwise, all zeros address (0.0.0.0) is selected.

This example show how you can tackle the Duplicate IP Address 0.0.0.0 error message problem encountered by clients that run Microsoft Windows:

Configure the **device-tracking tracking auto-source** command. This command determines the source IP and MAC address used in the Address Resolution Packet (ARP) request sent by the switch to probe a client, in order to maintain its entry in the device-tracking table. The purpose, is to avoid using 0.0.0.0 as source IP address.



Note Configure the **device-tracking tracking auto-source** command only when a switch virtual interface (SVI) is not configured. You do not have to configure it when a SVI is configured with an IPv4 address on the VLAN.

Command	Action (In order to select source IP and MAC address for device tracking ARP probe)	Notes
device-tracking tracking auto-source	<ul style="list-style-type: none"> • Set source to VLAN SVI if present. • Look for IP and MAC binding in device-tracking table from same subnet. • Use 0.0.0.0 	We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping.
device-tracking tracking auto-source override	<ul style="list-style-type: none"> • Set source to VLAN SVI if present • Use 0.0.0.0 	Not recommended when there is no SVI.
device-tracking tracking auto-source fallback 0.0.0.X 255.255.255.0	<ul style="list-style-type: none"> • Set source to VLAN SVI if present. • Look for IP and MAC binding in device-tracking table from same subnet. • Compute source IP from client IP using host bit and mask provided. Source MAC is taken from the MAC address of the switchport facing the client*. 	<p>We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping.</p> <p>The computed IPv4 address must not be assigned to any client or network device.</p>

Command	Action (In order to select source IP and MAC address for device tracking ARP probe)	Notes
device-tracking tracking auto-source fallback 0.0.0.X 255.255.255.0 override	<ul style="list-style-type: none"> Set source to VLAN SVI if present. Compute source IP from client IP using host bit and mask provided*. Source MAC is taken from the MAC address of the switchport facing the client*.	

* Depending on the client IP address, an IPv4 address has to be reserved for the source IP.

A reserved source IPv4 address = (client-ip and mask) | host-ip

- Client IP = 192.0.2.25
- Source IP = (192.0.2.25 and 255.255.255.0) | (0.0.0.1) = 192.0.2.1

IP address 192.0.2.1 should not be assigned to any client or network device.

Example: Disabling IPv6 Device Tracking on a Target

By default, SISF-based device tracking supports both IPv4 and IPv6. The following configuration examples show how you can disable IPv6 device tracking if you have to:

Disabling IPv6 device tracking when the target is attached to a custom policy:

```
Device(config)# device-tracking policy example-policy
Device(config-device-tracking)# no protocol ndp
Device(config-device-tracking)# no protocol dhcp6
Device(config-device-tracking)# end
```



Note In the Cisco IOS XE Denali 16.3.x release, you cannot disable IPv6 device tracking for a programmatically created policy.

Example: Enabling IPv6 for SVI on VLAN (To Mitigate the Duplicate Address Problem)

For an IPv6 device-tracking entry, its reachability is verified by sending an SISF probe to its end-node, which is a neighbor solicitation message. Selection of the source IP address for this neighbor solicitation probe follows these rules:

- If an SVI is configured on the VLAN, the link-local IPv6 address of the SVI is selected. Please ensure that the SVI IP address is unique in the subnet.

- Otherwise, all zeros address (0:0:0:0:0:0:0:0) is selected.

When IPv6 is enabled in the network and a switched virtual interface (SVI) is configured on a VLAN, we recommend that you add the following to the SVI configuration. This enables the SVI to acquire a link-local address automatically; this address is used as the source IP address of the SISF probe, thus preventing the duplicate IP address issue.

```
Device(config)# interface vlan 10
Device(config-if)# ipv6 enable
Device(config-if)# end
```

Example: Configuring a Multi-Switch Network to Stop Creating Binding Entries from a Trunk Port

In a multi-switch network, SISF-based device tracking provides the capability to distribute binding table entries between switches running the feature. Binding entries are only created on the switches where the host appears on an access port. No entry is created for a host that appears over a trunk port. This is achieved by configuring a policy with the **trusted-port** and **device-role switch** options, and attaching it to the trunk port.



Note Both, the **trusted-port**, and **device-role switch** options, must be configured in the policy.

Further, we recommended that you apply such a policy on a port facing a device, which also has SISF-based device tracking enabled.

```
Device> enable
Device# configure terminal
Device(config)# device-tracking policy example_trusted_policy
Device(config-device-tracking)# device-role switch
Device(config-device-tracking)# trusted-port
Device(config-device-tracking)# exit
Device(config)# interface gigabitethernet 1/0/25
Device(config-if)# device-tracking attach-policy example_trusted_policy
Device(config-if)# end
```

Example: Avoiding a Short Device-Tracking Binding Reachable Time

When migrating from an older release, the following configuration may be present:

```
device-tracking binding reachable-time 10
```

Remove this by entering the **no** version of the command.

Feature History and Information for SISF

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

Release	Modification
Cisco IOS XE Denali 16.1.1	<p>This feature was introduced.</p> <p>SISF-Based Device-Tracking tracks the presence, location, and movement of end-nodes in the network. The feature snoops traffic received by the switch, extracts device identity (MAC and IP address), and stores them in a binding table. Other features (called device tracking clients) depend on the accuracy of this information to operate properly.</p> <p>Both IPv4 and IPv6 are supported.</p> <p>SISF-based device-tracking is disabled by default.</p>
Cisco IOS XE Denali 16.3.7	<p>Correction in the system conversion of IPv6 snooping commands and SISF-based device-tracking commands.</p> <p>IPDT → IPv6 Snooping conversion corrections:</p> <ul style="list-style-type: none"> • Until Cisco IOS XE Denali 16.3.6, the system incorrectly converts the ip device tracking probe delay command to ipv6 neighbor tracking retry-interval. Starting from Cisco IOS XE Denali 16.3.7, this is set to the default value and cannot be changed. • Until Cisco IOS XE Denali 16.3.6, the system incorrectly converts the ip device tracking probe interval command to ipv6 neighbor tracking retry-interval. Starting from Cisco IOS XE Denali 16.3.7, this is correctly converted to ipv6 snooping tracking retry-interval <p>IPDT → SISF conversion corrections:</p> <ul style="list-style-type: none"> • Until Cisco IOS XE Denali 16.3.6 the system incorrectly converts the ip device tracking probe delay command to device-tracking binding reachable-lifetime. In the specified releases, you can still use this command, but to only configure the reachable-lifetime of an entry. Starting from Cisco IOS XE Denali 16.3.7, this is set to the default value and cannot be changed. • Until Cisco IOS XE Denali 16.3.6, the system incorrectly converts the ip device tracking probe interval command to device-tracking tracking retry-interval. Starting from Cisco IOS XE Denali 16.3.7, this is correctly converted to device-tracking binding reachable-lifetime.
Cisco IOS XE Everest 16.6.1	<p>Option to change parameters of DT_PROGRAMMATIC</p> <p>Starting with this release, you can change certain settings of the programmatically created device tracking policy: DT_PROGRAMMATIC, in the device tracking configuration mode (config-device-tracking)).</p>

Release	Modification
Cisco IOS XE Fuji 16.9.1	<p data-bbox="675 291 1520 420">Policy priority: Support for policy priority was introduced. Priority is determined by how the policy is created. A manually created policy has the highest priority. This enables you to apply policy settings that are different from policies that are generated programmatically.</p> <p data-bbox="675 436 1490 527">Additional device tracking clients: More device tracking client features were introduced. The programmatic policy created by each device tracking client differs.</p> <p data-bbox="675 546 1442 609">Change for programmatically created policies: The option to change the parameters of <i>any</i> programmatic policy was deprecated.</p>

