SIP Configuration Flexibility

Cisco Unified Border Element (SP Edition) offers flexibility in configuring the following features of a Session Initiation Protocol (SIP) adjacency:

- OPTIONS Support
- Rewriting from header on non-REGISTER requests
- Rewriting to header on non-REGISTER requests
- Auto-detecting NAT
- Routing on wildcard domains

Note For Cisco IOS XR Software Release , this feature is supported in the unified model only.

Cisco Unified Border Element (SP Edition) was formerly known as Integrated Session Border Controller and may be commonly referred to in this document as the session border controller (SBC).

For a complete description of the commands used in this chapter, refer to the *Cisco Unified Border Element (SP Edition) Command Reference: Unified Model* at:

http://www.cisco.com/en/US/docs/ios/sbc/command/reference/sbcu_book.html.

For information about all Cisco IOS commands, use the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or a Cisco IOS master commands list.

Feature History for SIP Configuration Flexibility

Release	Modification
Cisco IOS XR Software Release	This feature was introduced on the Cisco IOS XR along with support for the unified model.
Cisco IOS XE Release 3.6S	The Via Header Passthrough feature was added.

Contents

This module contains the following sections:

Restrictions for Implementing SIP Configuration Flexibility

The restrictions for implementing SIP configuration flexibility are listed per feature in this chapter.

Information About SIP Configuration Flexibility

This section contains the following subsections:

OPTIONS Support

By default, Cisco Unified Border Element (SP Edition) blocks the OPTIONS method from passing through, but users can configure Cisco Unified Border Element (SP Edition) on a per-adjacency basis to pass or block the OPTIONS method by using whitelists and blacklists.

Restrictions for OPTIONS Support

- Cisco Unified Border Element (SP Edition) strips out SDP blocks from messages when it allows the OPTIONS method to pass through. This limits what the SIP endpoints can exchange.
- The SBC-SIG does not send the Accept and Allow headers on any methods, including OPTIONS.
- Cisco Unified Border Element (SP Edition) allows only the 100Rel and Replaces tags of the Supported header to pass through, while the other tags of this header are controlled by whitelists and blacklists.

Rewriting From Header on Non-Register Requests

With this feature, users can configure Cisco Unified Border Element (SP Edition) on a per-adjacency basis to control whether it rewrites the hostport section of the From header on Non-Register Requests to the outbound SIP adjacency address or port. If Cisco Unified Border Element (SP Edition) is configured to allow the From header to pass through without it being rewritten, then Cisco Unified Border Element (SP Edition) allows the entire header to pass through without changing it. The only exception occurs with the Tag parameter; Cisco Unified Border Element (SP Edition) assigns a different value to this parameter before passing it through.

Restrictions for Rewriting From Header on Non-REGISTER Requests

- This feature is not applicable for REGISTER requests.
- This feature may only work in a limited way with the Rewrite-Register feature.

- If the From header contains a Tel URI, then Cisco Unified Border Element (SP Edition) does not rewrite the header since it does not have a hostport.
- Depending on the number of headers, options and SIP whitelist profiles, Cisco Unified Border Element (SP Edition) limits the size of the From header that it allows to pass through to approximately 1000 bytes.

Rewriting To Header on Non-REGISTER Requests

The default behavior of Cisco Unified Border Element (SP Edition) is to rewrite the hostport section of the To header on Non-Register Requests to be the outbound SIP adjacency address and port. It also removes any associated parameters. With this feature, users can configure the SBC on a per-adjacency basis to pass the To header through unchanged.

Auto-detecting NAT

With the addition of a new configuration field to the SIP adjacency, it is now possible for users to specify if Cisco Unified Border Element (SP Edition) must auto-detect whether a NAT is in use on that adjacency. If Cisco Unified Border Element (SP Edition) is configured to auto-detect NAT, then for each request that it receives, Cisco Unified Border Element (SP Edition) determines whether a NAT is in use for that endpoint. If Cisco Unified Border Element (SP Edition) determines that NAT is in use, then Cisco Unified Border Element (SP Edition) stores the bindings for that request and uses them when sending a response. Additionally, Cisco Unified Border Element (SP Edition) stores and reuses bindings for REGISTER requests for subsequent Dialog-forming and Out-of-dialog requests.

Restrictions for Auto-detecting NAT

- Cisco Unified Border Element (SP Edition) can auto-detect NAT only by comparing the Sent-by stopper in the Via header with the remote address and port of the message.
- If the stopper contains a domain name, instead of an IP address, Cisco Unified Border Element (SP Edition) cannot auto-detect whether NAT is in use. In this case, Cisco Unified Border Element (SP Edition) assumes that NAT is in use.
- Auto-detecting NAT is applied only to Out-of-dialog requests or Dialog-forming requests.

Routing on Wildcard Domains

Cisco Unified Border Element (SP Edition) routing policy allows you to use the * character in a text domain name match string. This character can match any number of characters in the called address. For example, *domain.com can match both sip1.domain.com and sip2.domain.com.

Restrictions for Routing on Wildcard Domains

- You can only specify one wildcard character in a given match string.
- This feature applies only to text domain name match rules, and not to dialed digit match rules.

How to Implement SIP Configuration Flexibility

This section contains the steps for implementing SIP configuration flexibility.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *service-name*
3. **sbe**
4. **adjacency sip** *adjacency-name*
5. **passthrough from header**
6. **header-name [contact [add [tls-param]] | from{*passthrough*} | to{*passthrough*}]**
7. **nat force-on**
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *service-name* **Example:** `Router(config)# sbc mysbc`	Enters the mode of an SBC service. • Use the *service-name* argument to define the name of the service.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the mode of the signaling border element (SBE) function of the SBC.
Step 4	**adjacency sip** ***adjacency-name*** **Example:** `Router(config-sbc-sbe)# adjacency sip sipadj`	Enters the mode of an SBE SIP adjacency. • Use the *adjacency-name* argument to define the name of the SIP adjacency.
Step 5	**passthrough from header** **Example:** `Router(config-sbc-sbe-adj-sip)# passthrough from header`	Configures the SIP adjacency to disable From rewrite.

	Command or Action	Purpose
Step 6	**header-name [contact [add [tls-param]] | from {*passthrough*} | to {*passthrough*}]** **Example:** `Router(config-sbc-sbe-adj-sip)# header-name to passthrough`	Configures the SIP adjacency to disable To rewrite.
Step 7	**nat force-on** **Example:** `Router(config-sbc-sbe-adj-sip)# nat force-on`	Configures the SIP adjacency to assume that all endpoints are behind a NAT device. To configure the SIP adjacency to assume that no endpoints are behind a NAT device, use the **nat force-off** command. By default, the SBC autodetects whether the endpoints are behind a NAT device.
Step 8	**exit** **Example:** `Router(config-sbc-sbe-adj-sip)# exit`	Exits the adj-sip mode and returns to the SBE mode.

Via Header Passthrough

The Via Header Passthrough feature enables the SBC to interoperate with certain devices that use the Via header to authenticate other devices, such as a PBX, that do not support SIP authentication. With the introduction of this feature, the SBC can be configured to function as a SIP proxy that is compliant with RFC 3261 and RFC 3581 in its handling of the received parameter and rport parameter, which are two parameters of the Via header.

This section contains the following topics:

- Restrictions for Via Header Passthrough, page 28-5
- Information About Via Header Passthrough, page 28-6
- How to Configure Via Header Passthrough, page 28-6
- Configuration Example: Via Header Passthrough, page 28-7

Restrictions for Via Header Passthrough

The following are the restrictions for the Via Header Passthrough feature:

- This feature does not support the Topology Hiding feature. After the Via Header Passthrough feature is configured, the data that is passed through the SBC includes information about the topology of the network between the sender and the SBC. If you want to protect the network between the sender and the SBC by using the Topology Hiding feature, do not configure the Via Header Passthrough feature.
- The existing restriction on editing Via headers by using the SIP Message Editing feature is still applicable.

Information About Via Header Passthrough

Certain devices use the Via header to authenticate other devices, such as a PBX, that do not support SIP authentication. The Via Header Passthrough feature enables the SBC to interoperate with the devices that use the Via header to authenticate other devices. In releases prior to Release 3.6.0, the SBC would remove the existing Via headers from an incoming SIP message and add its own Via header before forwarding the message. With the introduction of the Via Header Passthrough feature in Release 3.6.0, the SBC can be configured to allow the existing Via headers to pass through and add its own Via header.

When the Via Header Passthrough feature is configured, the SBC adds its own Via header at the top of the stack of Via headers before forwarding the message. If the remote IP address from which the SBC receives the SIP message differs from the IP address specified in the sent-by address of the header, the SBC sets the received parameter to the actual remote IP address before forwarding the message. At the same time, if the SBC receives a SIP message in which the latest entry in the Via header contains the rport parameter with no value set for it, the SBC sets the value of the rport parameter to the source port of the message. In this scenario, the SBC also adds the received parameter to the Via header, regardless of whether the sent-by address in the Via header matches the IP address from which the message was received.

The Via Header Passthrough feature is configured at the SIP adjacency level. To maintain the Via headers on a message routed through the SBC, the Via Header Passthrough feature must be configured on both the inbound adjacency and the outbound adjacency. If this feature is not configured on either one of these adjacencies, the Via headers are removed from the SIP messages that pass through these adjacencies. Note that the SBC adds its own Via header to the outbound SIP message, regardless of whether the Via Header Passthrough feature is configured.

How to Configure Via Header Passthrough

The following procedure shows how to configure the Via Header Passthrough feature.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **adjacency sip** *adjacency-name*
5. **header-name via passthrough inbound**
6. **header-name via passthrough outbound**
7. **end**
8. **show sbc** *sbc-name* **sbe adjacencies** *adjacency-name* **detail**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mysbc`	Enters the configuration mode of an SBC service. • *sbc-name*—Name of the SBC service.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the configuration mode of the signaling border element (SBE) function of the SBC.
Step 4	**adjacency sip** *adjacency-name* **Example:** `Router(config-sbc-sbe) # adjacency sip adj1`	Enters the mode of an SBE SIP adjacency. • *adjacency-name*—Name of the adjacency.
Step 5	**header-name via passthrough inbound** **Example:** `Router(config-sbc-sbe-adj-sip)# header-name via passthrough inbound`	Specifies that the Via headers on inbound requests for this adjacency must be allowed to pass through.
Step 6	**header-name via passthrough outbound** **Example:** `Router(config-sbc-sbe-adj-sip)# header-name via passthrough outbound`	Specifies that the Via headers on outbound requests for this adjacency must be allowed to pass through.
Step 7	**end** **Example:** `Router(config-sbc-sbe-adj-sip)# end`	Exits the adjacency SIP configuration mode, and returns to the privileged EXEC mode.
Step 8	**show sbc** *sbc-name* **sbe adjacencies** *adjacency-name* **detail** **Example:** `Router# show sbc mySBC sbe adjacencies adj1 detail`	Shows the configuration details of the specified adjacency.

Configuration Example: Via Header Passthrough

The following is a sample configuration of the Via Header Passthrough feature:

```
Router(config)# configure terminal
Router(config)# sbc mySBC
Router(config-sbc)# sbe
```

```
Router(config-sbc-sbe)# adjacency sip adj1
Router(config-sbc-sbe-adj-sip)# header-name via passthrough inbound
Router(config-sbc-sbe-adj-sip)# header-name via passthrough outbound
.
.
.
Router# show sbc mySBC sbe adjacencies adj1 detail
```

The following is a sample output of the **show sbc mySBC sbe adjacencies adj1 detail** command:

```
  Adjacency adj1 (SIP)
    Status:                 Detached
    Signaling address:      0.0.0.0:default
.
.
.
    Contact header parameters: Passthrough
    Inbound Via Passthrough: Allowed
    Outbound Via Passthrough: Allowed
.
.
.
```