Cisco Unified Border Element (SP Edition)—SPA DSP Services

The shared port adapter (SPA) digital signal processor (DSP) is a single-width, half-height, high-power, SPA module that can be used across multiple Cisco platforms. The SPA DSP is designed for DSP-based voice and video solutions in the SPAs on the Cisco mid-range and high-end routers.

In Cisco IOS XE Release 3.2S, the following SPA DSP features have been deployed on the Cisco ASR 1000 Series Router for the session border controller (SBC):

- Associating SBC configuration with a DSP farm profile.
- Voice transcoding and transrating support using onboard DSP services.
- Dual tone multifrequency (DTMF) interworking using onboard DSP services.
- VoIPv4 and VoIPv6 transcoding and transrating support.
- Transcoding, transrating, and DTMF interworking call control and signaling control.

Cisco Unified Border Element (SP Edition) was earlier known as Integrated Session Border Controller, and is referred to as SBC in this document.

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

Feature History of SPA DSP on the Cisco Unified Border Element (SP Edition)

Release	Modification
Cisco IOS XE Release 3.2S	The SPA DSP onboard services were introduced on the Cisco ASR 1000 Series Routers.
Cisco IOS XE Release 3.3S	The Call Recovery feature was added.
Cisco IOS XE Release 3.8S	The AMR-WB feature was supported on the SBC on the Cisco ASR 1000 Aggregation Services Routers.

Contents

Restrictions

The following restrictions are applicable to a SPA DSP:

- Voice, audio, and video conferencing are not supported.
- HA, system-level In-Service Software Upgrade (ISSU), and Nonstop Forwarding (NSF) are not supported.
- Video codecs are not supported.
- Although Online Insertion and Removal (OIR) is supported, the sessions going through a SPA at the time of removal are lost.
- The Cisco Unified Communications Manager is not supported.

Prerequisites for the SPA DSP Services

The DSP farm definition and SBC configuration and activation must be completed before transcoding the SBC calls. For more information about SPA configuration, see the "Configuring the Cisco DSP SPA for the ASR 1000 Series" chapter in *Cisco ASR 1000 Series Aggregation Services Routers SIP and SPA Software Configuration Guide* at:

http://www.cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/ASR1000/asrcfgdsp.html

Information About the SPA DSP Services

A SPA DSP contains digital signal processors and related hardware to provide voice transcoding capability for the SBC. In addition, Cisco Unified Border Element, Enterprise can use a SPA DSP for simple voice transcoding services.

You can find more information on terminating and generating the RTCP by the SPA-DSP at: http://www.cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/ASR1000/ASRovdsp.html#wp1296621

Transcoding the SBC

SBC transcoding is used for codec translation between two VoIP networks as part of the Data Border Element (DBE) functions. Figure 41-1 shows how a SPA DSP performs codec transcoding for unified SBC and Figure 41-2 shows how a SPA DSP performs codec transcoding for distributed SBC.

Figure 41-1 SPA DSP Transcoding for Unified SBC

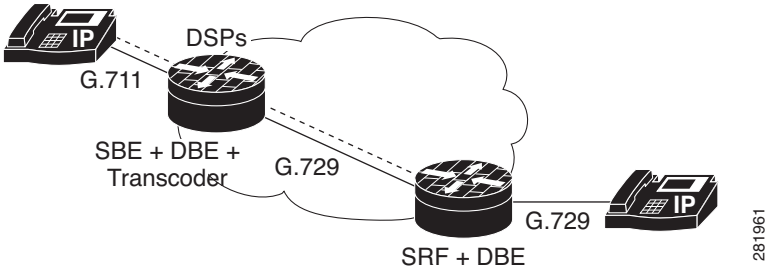
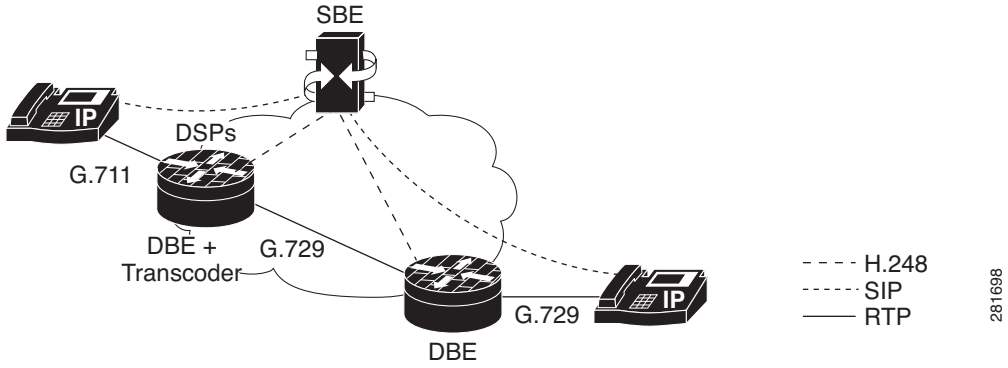


Figure 41-2 SPA DSP Transcoding for Distributed SBC



The SPA DSP allows the translation of one type of media stream or codec to another type of media stream that uses different media encoding and decoding technologies. Other translation activities include:

- Translation between different codecs
- Translation between different packetization settings (transrating)
- DTMF interworking

Transcoding the Distributed SBC

Transcoding is inferred from a Session Description Protocol (SDP) that is used to program a call. Programming terminations in the same call containing different codecs implicitly instruct the distributed SBC to perform transcoding.

Transrating the Distributed SBC

Transrating is inferred from the SDP that is used to program a call. Programming terminations in the same call with different ptime implicitly instruct the distributed SBC to perform transrating.

Note Transrating is supported only for the different rates using the same codec, not across codecs. Therefore, transrating and transcoding cannot be performed simultaneously.

RTP Telephone-Event Codec-to-SIP Interworking

When an RTP packet is marked as DTMF using the telephone-event codec, the RTP packet is removed from the stream. The DBE sends an H.248 message to the signaling border element (SBE), indicating that a DTMF event has occurred, and that the RTP packet should be converted into a SIP DTMF event.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is present in side A of the SDP, but not in side B.
- The dd/etd event is subscribed for side A, but not for side B.

SIP-to-RTP Telephone-Event Codec Interworking

When an endpoint generates a SIP signal, the SIP DTMF signals arrive completely out of band. An endpoint that supports SIP DTMF generates the signals to be sent to the SBE. In turn, the SBE recognizes that this is a DTMF message and sends an H.248 message to the DBE, indicating that a DTMF tone is required to be inserted into the RTP stream. The DBE then inserts the RTP DTMF packets into the audio stream using telephone-event codec.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is present in side B of the SDP, but not in side A.
- The dd/etd event is subscribed for side B, but not for side A.

RTP Telephone-Event Codec-to-RTP In-Band Waveform

After the RTP packet is marked as DTMF using the telephone-event codec, the RTP packet is removed from the stream, and an RTP stream containing the DTMF waveform is sent to the other endpoint.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is present in side A of SDP, but not in side B.
- The dd/etd event is subscribed for side A and side B.

RTP In-Band Waveform-to-RTP Telephone-Event Codec

After the DTMF is sent as part of the voice waveform, the RTP packets are removed from the stream, and the DBE inserts the a new RTP packet with the payload-type telephone event into the audio stream.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is present in side B of the SDP, but not in side A.
- The dd/etd event is subscribed for side A and side B

SIP-to-RTP In-Band Waveform

After an endpoint generates a SIP signal, the SIP DTMF signals arrive completely out of band. The endpoint that supports SIP DTMF generates the signals to be sent to the SBE. In turn, the SBE recognizes that this is a DTMF message, and sends an H.248 message to the DBE, indicating that a DTMF tone is required to be inserted into the RTP stream. The DBE then inserts a stream containing the DTMF waveform.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is not present on either side A or side B.
- The dd/etd event is subscribed for side B.

RTP In-Band Waveform-to-SIP

When the DTMF is sent as part of the voice waveform, the RTP packets are removed from the stream, and the DBE sends an H.248 message to the SBE, indicating that a DTMF event has occurred, and that the RTP packets should be converted into a SIP DTMF event.

The call must meet the following conditions:

- The telephone-event codec (for RFC 2833) is not present on either side A or side B.
- The dd/etd event is subscribed for side A.

Call Recovery

From Cisco IOS XE Release 3.3S, calls on a partially crashed SPA DSP can be recovered within the call outage time of 2.5s.

When part of a SPA DSP crashes, a crash recovery process runs, and then the RP reprograms the crashed part of the SPA DSP with all calls that were previously on it. For example, a simple transcoding scenario, a-law to u-law transcoding, can represent up to 129 calls that require reprogramming.

Depending on the part of the SPA DSP that crashes, the total recovery time may be longer because it might have to recover more components and also reprogram more calls. However, the entire media path outage time for all the recovered calls is less than 2.5s.

In all cases of the SPA DSP call recovery, the call recovery occurs on the same SPA DSP where the call existed prior to the crash. The calls are not moved to another SPA DSP.

The SPA DSP failure call recovery can be disabled or rendered ineffective if the SPA DSP crash dumps are enabled. It can push the call outage time beyond 2.5s.

The **show voice dsp group all** command indicates when a SPA DSP is undergoing call recovery.

```
Router# show voice dsp group all

Show DSP group all

DSP groups on slot 0 bay 0:
dsp 1:
  State: UP
  HA State : DSP_HA_STATE_PENDING1
  Max signal/voice channel: 43/43
  Max credits: 645
  num_of_sig_chnls_allocated: 43
  Transcoding channels allocated: 43
  Group: FLEX_GROUP_XCODE, complexity: LOW
```

```
Shared credits: 0, reserved credits: 645
Transcoding channels allocated: 24
Credits used (rounded-up): 360
```

Note The **show voice dsp group all** command displays the output **`HA State : DSP_HA_STATE_PENDING1`** only during the recovery process which can be upto a few milliseconds.

AMR-WB Transcoding Support

Adaptive Multi-Rate Wideband (AMR-WB) is a patented speech coding standard based on Adaptive Multi-Rate encoding, using a methodology that is similar to the Algebraic code-excited linear prediction (ACELP). AMR-WB, which was specified by 3GPP, provides improved speech quality due to a wider speech bandwidth of 50 to 7000Hz compared to narrowband speech coders what are in general optimized for Plain old telephone service (POTS) wireline quality of 300 to 3400 Hz.

AMR-WB is codified as G.722.2, an ITU-T standard speech codec, formally known as Wideband coding of speech at around 16 kbps using AMR-WB. G.722.2 AMR-WB is the same codec as the 3GPP AMR-WB.

AMR-WB operates like AMR with nine different bit rates. The lowest bit rate providing excellent speech quality in a clean environment is 12.65 kbps. Higher bit rates are useful in background noise conditions and for music. Also, lower bit rates of 6.60 and 8.85 kbps provide reasonable quality, especially compared to narrowband codecs.

Note The AMR-WB feature requires DSP firmware with AMR-WB codec support.

Table 41-1shows the relationship between the AMR rate mode and bit-rate.

Table 41-1 Relationship Between the AMR Rate Mode and Bit-Rate

Rate Mode	AMR Bit-Rate (kbps)	AMR-WB/G.722.2 Bit-Rate (kbps)
0	4.75	6.60
1	5.15	8.85
2	5.90	12.65
3	6.70	14.25
4	7.40	15.85
5	7.95	18.25
6	10.20	19.85
7	12.20	23.05
8	SID[1]	23.85
9	—	SID

1. SID: Silence Indicator

Configuring the SPA DSP Services for SBC

This section describes the tasks to involved in configuring the SPA DSP services for the SBC:

- Setting Up a SPA DSP for DSP Farm Services, page 41-7
- Configuring a DSP Farm Profile, page 41-8

Setting Up a SPA DSP for DSP Farm Services

Use the following procedure to set up the SPA DSP in the DSP farm mode for the DSP services:

SUMMARY STEPS

1. **configure terminal**
2. **voice-card** *slot number/subslot number*
3. **dsp services dspfarm**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 2	**voice-card** *slot number/subslot number* **Example:** `Router(config)# voice-card 0/2`	Specifies the slot number of the voice card and enters the voice card interface configuration mode.
Step 3	**dsp services dspfarm** **Example:** `Router(config-voicecard)# dsp services dspfarm`	Allows DSP farm services on the SPA DSP voice card.
Step 4	**end** **Example:** `Router(config-voicecard)# end`	Exits the voice card interface configuration mode.

For more information about configuring DSP farm services on a SPA DSP, see the "Configuring the Cisco DSP SPA for ASR 1000 Series" chapter in the *Cisco ASR 1000 Series Aggregation Services Routers SIP and SPA Software Configuration Guide* at:

http://www.cisco.com/en/US/docs/interfaces_modules/shared_port_adapters/configuration/ASR1000/asrcfgdsp.html

Configuring a DSP Farm Profile

Use the following steps to configure a DSP farm profile:

SUMMARY STEPS

1. **configure terminal**
2. **dspfarm profile** *profile-identifier* **{conference | mtp | transcode}**
3. **description** *profile-description-text*
4. **codec** *codec-name*
5. **associate application {cube | sbc | sccp}**
6. **maximum session** *number*
7. **no shutdown**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 2	**dspfarm profile** *profile-identifier* {**conference** | **mtp** | **transcode**} **Example:** `Router(config)# dspfarm profile 20 transcode`	Enables the DSP farm service for the specified DSP farm profile, and enters a DSP farm profile configuration mode. The service options are: • **conference**—Enables conferencing. • **mtp**—Enables media termination point. • **transcode**—Enables transcoding of information. **Note** In Cisco IOS Release 3.2S, only the transcode service is supported.
Step 3	**description** *profile-description-text* **Example:** `Router(config-dspfarm-profile)# description enables transcoding`	Specifies a description for a defined profile.

Step 4	**no codec** *codec-name*
	
	Example:
	`Router(config-dspfarm-profile)# codec g711ulaw`
	`Router(config-dspfarm-profile)# codec g711alaw`
	`Router(config-dspfarm-profile)# codec g729ar8`
	`Router(config-dspfarm-profile)# codec g729abr8`
	`Router(config-dspfarm-profile)# codec g729r8`
	`Router(config-dspfarm-profile)# codec g723r63`
	`Router(config-dspfarm-profile)# codec ilbc`
	`Router(config-dspfarm-profile)# codec gsmamr-nb`
	`Router(config-dspfarm-profile)# codec g726r32`
	`Router(config-dspfarm-profile)# codec g729br8`
		Adds codecs or removes the codec from a codec list. The codec must be present in the list of codecs that the SBE is hard-coded to recognize.

Step 5	**associate application** {**cube** | **sbc** | **sccp**} *profile-description-text* **Example:** `Router(config-dspfarm-profile)# associate application sbc`	Associates an application to the profile. The applications that can be associated are: • **cube**—Associates the Cisco Unified Border Element application to a defined profile in the DSP farm. • **sbc**—Associates the SBC application to a defined profile in the DSP farm. • **sccp**—Associates the client control protocol application to a defined profile in the DSP farm. **Note** The **sbc** application keyword is available only when a DSP farm profile transcode service is used.

	Command or Action	Purpose
Step 6	**maximum session** *number* **Example:** `Router(config-dspfarm-profile)# maximum session 300`	Establishes the maximum number of sessions that can be assigned to a defined profile. The maximum number of sessions is dependent upon the number of SPA DSPs in the router, and the codecs configured. For a fully populated Cisco ASR 1013 Series Router with 23 SPA DSPs and only the G711 codec, the maximum number of sessions would be 20769.
Step 7	**no shutdown** **Example:** `Router(config-dspfarm-profile)# no shutdown`	Enables or disables a DSP farm profile.
Step 8	**end** **Example:** `Router(config-dspfarm-profile)# end`	Exits the DSP farm profile.

Configuring the Unified SBC

This section explains the various ways in which to configure the SBC for the SPA DSP voice card:

- Associating the Unified SBC with a DSP Farm Profile, page 41-10
- Configuring the Unified SBC to Enable Transcoding, page 41-11
- Configuring the Unified SBC to Enable Transrating, page 41-17
- Configuring the Unified SBC to Enable SRTP and Transcoding, page 41-22
- Configuring the Unified SBC for Inband DTMF Interworking, page 41-28
- Configuring the Unified SBC to Support AMR-WB, page 41-33

Associating the Unified SBC with a DSP Farm Profile

Association of the SBC to the DSP farm profiles is possible only after the corresponding DSP farm profile is created. Use the **associate dspfarm profile** command in the global configuration mode.

SUMMARY STEPS

1. **show dspfarm** {**all** | dsp | profile}
2. **configure terminal**
3. **sbc** *sbc-name*
4. **associate dspfarm profile** {*profile-number* | **all**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show dspfarm** {**all** | **dsp** | **profile** *profile-identifier*} **Example:** `Router# show dspfarm profile all`	Displays the DSP farm configuration information: • **all**—Displays the DSP farm global information. • **dsp**—Displays information pertaining to all the DSPs. • **profile**—Displays the DSP farm profiles.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enables the global configuration mode.
Step 3	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters the SBC configuration mode.
Step 4	**associate dspfarm profile** {*profile-number* | **all**} **Example:** `Router(config-sbc)# associate dspfarm profile 1`	Associates the SBC to a DSP farm profile: • *profile-number*—Specifies the profile number to be associated. • **all**—Allows the SBC to pick the most appropriate DSP farm profile from the profiles associated to the SBC for the transcoding session.
Step 5	**end** **Example:** `Router(config-sbc-sbe)# end`	Exits the configuration mode.

Configuring the Unified SBC to Enable Transcoding

This task configures the SBC for enabling the transcoding feature.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **cac-policy-set** *policy-set-id*
5. **first-cac-scope** *scope-name*

6. **first-cac-table** ***table-name***
7. **cac-table** ***table-name***
8. **table-type limit** ***list of limit tables***
9. **entry** ***entry-id***
10. **match-value** ***key***
11. **callee-codec-list** *list-name*
12. **caller-codec-list** *list-name*
13. **media police strip | reject | degrade**
14. **action cac-complete**
15. **complete**
16. **cac-policy-set global** *cac-policy-num*
17. **codec-list** *list-name*
18. **codec** *codec-nam*
19. **exit**
20. **codec-list** *list-name*
21. **codec** *codec-nam*
22. **exit**
23. **end**
24. **show sbc** *sbc-name* **sbe call-stats global current5min**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters the SBC configuration mode.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the signaling border element (SBE) function mode of the SBC.
Step 4	**cac-policy-set** *policy-set-id* **Example:** `Router(config-sbc-sbe)# cac-policy-set 1`	Enters the CAC policy set configuration mode within an SBE entity, creating a new policy set, if necessary: • *policy-set-id*—Integer chosen by a user to identify the policy set. The range is from 1 to 2147483647.

	Command or Action	Purpose
Step 5	**first-cac-scope** *scope-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# first-cac-scope dst-adjacency`	Configures the scope at which limits should be initially defined to perform tasks at the admission control stage of the policy. Each CAC policy has a scope that can be applied to it. This CAC policy is applicable on a per call basis. *scope-name* has one of the following values: • **adj-group**—Limits for events from members of the same adjacency group. • **call**—Limits are per single call. • **category**—Limits per category. • **dst-account**—Limits for events sent to the same account. • **dst-adj-group**—Limits for events sent to the same adjacency group. • **dst-adjacency**—Limits for events sent to the same adjacency. • **dst-number**—Limits for events that have the same adjacency number. • **global**—Limits are global and should not be combined with any other option. • **src-account**—Limits for events from the same account. • **src-adj-group**—Limits for events from the same adjacency group. • **arc-adjacency**—Limits for events from the same adjacency. • **src-number**—Limits for events that have the same source number.
Step 6	**first-cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# first-cac-table codec-dst-acc`	Configures the name of the first policy table to be processed. A CAC policy may have many tables configured. To start applying the CAC policy, the first table that is used must be defined: • *table-name*—The admission control table that should be processed first.
Step 7	**cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# cac-table codec-dst-acc`	Enters the CAC table mode for configuration of an admission control table (creating one, if necessary) within the context of an SBE policy set. • *table-name*—Name of the admission control table.

	Command or Action	Purpose
Step 8	**table-type limit** *list of limit tables* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# table-type limit dst-adjacency`	Configures a new CAC Limit table type in which the criteria used to match the entries must be entered. *list of limit tables* can be one of the following values: • **account**—Compare the name of the account. • **adj-group**—Compare the name of the adjacency group. • **adjacency**—Compare the name of the adjacency. • **all**—No comparison type. All events match this type. • **call-priority**—Compare with call priority. • **category**—Compare the number analysis assigned category. • **dst-account**—Compare the name of the destination account. • **dst-adj-group**—Compare the name of the destination adjacency group. • **dst-adjacency**—Compare the name of the destination adjacency. • **dst-prefix**—Compare the beginning of the dialed digit string. • **event-type**—Compare with CAC policy event types. • **src-account**—Compare the name of the source account. • **src-adj-group**—Compare the name of the source adjacency group. • **src-adjacency**—Compare the name of the source adjacency. • **src-prefix**—Compare the beginning of the calling number string.
Step 9	**entry** *entry-id* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# entry 1`	Enters the CAC table entry mode to modify an entry in an admission control table. • *entry-id*—Specifies the table entry.
Step 10	**match-value** *key* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry) # match-value nava`	Configures the match value of an entry in a CAC Limit table type.
Step 11	**callee-codec-list** ***list-name*** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry) # callee-codec-list PCMU`	Lists the codecs that the callee leg of a call is allowed to use.

	Command or Action	Purpose
Step 12	**caller-codec-list** ***list-name*** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# caller-codec-list PCMA`	Lists the codecs that the caller leg of a call is allowed to use.
Step 13	**media police strip | reject | degrade** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# media police strip`	Configures the manner in which the SBC will handle the media streams that exceed the bandwidth limit for media calls.
Step 14	**action cac-complete** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# action cac-complete`	When an event matches, the CAC policy is considered complete.
Step 15	**complete** **Example:** `Router(config-sbc-sbe-cacpolicy)# complete`	Completes the CAC policy set when you have committed the full set.
Step 16	**cac-policy-set global** *policy-num* **Example:** `Router(config-sbc-sbe)# cac-policy-set global 1`	Activates the global CAC policy set. The CAC policy set must be in a complete state before it can be assigned as the default policy. • *policy-num*—The call policy set number, ranging from 1 to 2147483647. The policy set must be in a complete state before it can be assigned as the default policy.
Step 17	**codec-list** ***list-name*** **Example:** `Router(config-sbc-sbe)# codec-list PCMU`	Creates a codec list, and enters the Codec list configuration mode.
Step 18	**codec** ***codec-name*** **Example:** `Router(config-sbc-sbe-codec-list)# codec PCMU`	Adds a codec to a codec list.
Step 19	**exit** **Example:** `Router(config-sbc-sbe-codec-list)# exit`	Exits the codec list configuration mode.
Step 20	**codec-list** ***list-name*** **Example:** `Router(config-sbc-sbe)# codec-list PCMA`	Creates a codec list, and enters the Codec list configuration mode.
Step 21	**codec** ***codec-name*** **Example:** `Router(config-sbc-sbe-codec-list)# codec PCMA`	Adds a codec to a codec list.

	Command or Action	Purpose
Step 22	**exit** **Example:** `Router(config-sbc-sbe-codec-list)# exit`	Exits the codec list configuration mode.
Step 23	**end** **Example:** `Router(config-sbc-sbe)# end`	Ends the configuration session.
Step 24	**show sbc** *sbc-name* **sbe call-stats global current5min** **Example:** `Router# show sbc mySBC sbe call-stats global current5min`	Lists the statistics for all the calls on the specified SBE.

The following example shows an output of the **show sbc sbe call-stats global current5min** command that lists the count of the active transcoded and transrated calls.

```
Router# show sbc mySBC sbe call-stats global current5min

SBC Service "mySBC"
Statistics for the current 5 mins for global counters
Call count totals:
  Total call attempts =                                    0
  Total active calls =                                     1
  Total active IPv6 calls =                                0
  Total activating calls =                                 0
  Total de-activating calls =                              0
  Total active emergency calls =                           0
  Total active e2 emergency calls =                        0
  Total IMS rx active calls =                              0
  Total IMS rx call renegotiation attempts =               0
  Total SRTP-RTP interworked calls =                       0
  Total active calls not using SRTP =                      1
  Total active transcoded calls =                          1
  Total active transrated calls =                          0
General call failure counters:
  Total call setup failures =                              0
  Total active call failures =                             0
  Total failed call attempts =                             0
  Total failed calls due to update failure =               0
  Total failed calls due to resource failure =             0
  Total failed calls due to congestion =                   0
  Total failed calls due to media failure =                0
  Total failed calls due to signaling failure =            0
  Total failed calls due to IMS rx setup failure =         0
  Total failed calls due to IMS rx renegotiation failure = 0
  Total failed calls due to RTP disallowed on call leg =   0
  Total failed calls due to SRTP disallowed on call leg =  0
```

Configuring the Unified SBC to Enable Transrating

Note Transrating is supported only for different rates using the same codec, not across codecs. Therefore, transrating and transcoding cannot be performed simultaneously.

This section describes how to enable transrating using either of the following methods:

- Transrating Using the Same Codec Policy, page 41-17
- Transrating Using a New Codec Policy, page 41-21

Transrating Using the Same Codec Policy

This task configures the SBC for enabling the transrating using the same codec policy.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **cac-policy-set** *policy-set-id*
5. **first-cac-table** *table-name*
6. **cac-table** *table-name*
7. **table-type {policy-set | limit {*list of limit tables*}}**
8. **entry** *entry-id*
9. **cac-scope** {*list of scope options*}
10. **callee ptime** *0-100*
11. **caller ptime** *0-100*
12. **media police strip | reject | degrade**
13. **action cac complete**
14. **complete**
15. **cac-policy-set global** *cac-policy-num*
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters the SBC configuration mode.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the SBE function mode of the SBC.
Step 4	**cac-policy-set** *policy-set-id* **Example:** `Router(config-sbc-sbe)# cac-policy-set 1`	Enters the CAC policy set configuration mode within an SBE entity, creating a new policy set, if necessary.
Step 5	**first-cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# first-cac-table Transrate`	Configures the name of the first policy table to be processed. A CAC policy may have many tables configured. To start applying the CAC policy, the first table that is used must be defined: • *table-name*—The admission control table that should be processed first.
Step 6	**cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# cac-table Transrate`	Enters the CAC table mode for configuration of an admission control table (creating one, if necessary) within the context of an SBE policy set: • *table-name*—Name of the admission control table.

Command or Action	Purpose
Step 7 **table-type {policy-set | limit** {*list of limit tables*}} **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# table-type policy-set`	Configures the table type of a CAC table within the context of an SBC policy set. *list of limit tables* can be one of the following values: • **account**—Compare the name of the account. • **adj-group**—Compare the name of the adjacency group. • **adjacency**—Compare the name of the adjacency. • **all**—No comparison type. All events match this type. • **call-priority**—Compare with call priority. • **category**—Compare the number analysis assigned category. • **dst-account**—Compare the name of the destination account. • **dst-adj-group**—Compare the name of the destination adjacency group. • **dst-adjacency**—Compare the name of the destination adjacency. • **dst-prefix**—Compare the beginning of the dialed digit string. • **event-type**—Compare with CAC policy event types. • **src-account**—Compare the name of the source account. • **src-adj-group**—Compare the name of the source adjacency group. • **src-adjacency**—Compare the name of the source adjacency. • **src-prefix**—Compare the beginning of the calling number string. Features can be enabled or disabled per adjacency group through CAC configuration the same way this is done per individual adjacencies. The adj-group table type matches either the source adjacency group or the destination adjacency group. When the **policy-set** keyword is specified, use the **cac-scope** command to configure the scope within each of the entries in which limits are applied in a CAC Policy Set table.
Step 8 **entry** *entry-id* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# entry 1`	Enters the CAC table entry mode to create or modify an entry in an admission control table.

	Command or Action	Purpose

Step 9	**cac-scope {***list of scope options***}**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# cac-scope call`
		Enables the selection of a scope at which CAC limits are applied within each entry in a Policy Set table.
		
		list of scope options—Specifies one of the following strings used to match events:
		• *account*—Events that are from the same account.
		• *adjacency*—Events that are from the same adjacency.
		• *adj-group*—Events that are from members of the same adjacency group.
		• *call*—Scope limits are per single call.
		• *category*—Events that have the same category.
		• *dst-account*—Events that are sent to the same account.
		• *dst-adj-group*—Events that are sent to the same adjacency group.
		• *dst-adjacency*—Events that are sent to the same adjacency.
		• *dst-number*—Events that have the same destination.
		• *global*—Scope limits are global.
		• *src-account*—Events that are from the same account.
		• *src-adj-group*—Events that are from the same adjacency group.
		• *src-adjacency*—Events that are from the same adjacency.
		• *src-number*—Events that have the same source number.

Step 10	**callee ptime** *<0-100>*
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# callee ptime 30`
		Configures the packetization time on the callee side that is forced for calls using this CAC entry.
		
		By default, 0 ms is configured, which means no transrating occurs.

Step 11	**caller ptime** *<0-100>*
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# caller ptime 40`
		Configures the packetization time on the caller side that is forced for calls using this CAC entry.
		
		By default, 0 ms is configured, which means no transrating occurs.

Step 12	**media police strip | reject | degrade**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# media police strip`
		Configures the manner in which the SBC handles the media streams that exceed the bandwidth limit for media calls.

Step 13	**action cac-complete**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# action cac complete`
		When an event matches, this CAC policy is complete.

	Command or Action	Purpose
Step 14	**complete** **Example:** `Router(config-sbc-sbe-cacpolicy)# complete`	Completes the CAC policy set when you have committed the full set.
Step 15	**cac-policy-set global** *policy-num* **Example:** `Router(config-sbc-sbe)# cac-policy-set global 1`	Activates the global CAC policy set. The CAC policy set must be in a complete state before it can be assigned as the default policy. • *policy-num*—The call policy set number, ranging from 1 to 2147483647. The policy set must be in a complete state before it can be assigned as the default policy.
Step 16	**end** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# end`	Exits the CAC configuration mode and returns to privileged EXEC mode.

Transrating Using a New Codec Policy

This task configures the SBC for enabling the transrating feature. This is an alternative mechanism to that described in the ?$paranum>Transrating Using the Same Codec Policy? section on page 41-17 section for configuring transrating.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **codec list** *list-name*
5. **policy {minimum | transrating}**
6. **codec** *codec-name* **packetization-period** *packet-period* **[priority** *priority-value***]**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters into the SBC configuration mode.

	Command or Action	Purpose
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the SBE function mode of the SBC.
Step 4	**codec list** *list-name* **Example:** `Router(config-sbc-sbe)# codec list PCMU`	Creates the codec list and enters the codec list mode.
Step 5	**policy** {**minimum** | **transrating**} **Example:** `Router(config-sbc-sbe-codec-list)# policy minimum`	Configures the packetization period policy that is to be specified as either of the following: • **minimum**—Packetization period is the minimum. • **transrating**—Packetization period is transrating.
Step 6	**codec** *codec-name* **packetization-period** *packet-period* [**priority** *priority-value*] **Example:** `Router(config-sbc-sbe-codec-list)# codec PCMU packetization-period 200 priority 1`	Adds a codec to a codec list, and sets a minimum packetization period and priority value for the codec.
Step 7	**end** **Example:** `Router(config-sbc-sbe-codec-list)# end`	Exits the CAC configuration mode, and returns to privileged EXEC mode.

Configuring the Unified SBC to Enable SRTP and Transcoding

Although Secure Real-time Transport Protocol (SRTP) is independent of transcoding, both can be configured to be used simultaneously.

This task configures the unified SBC to enable the SRTP and transcoding features.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **cac-policy-set** *policy-set-id*
5. **first-cac-table** *table-name*
6. **cac-table** *table-name*

7. **table-type {policy-set | limit {*list of limit tables*}}**
8. **entry** *entry-id*
9. **cac-scope** {*list of scope options*}
10. **srtp support allow**
11. **srtp caller forbid | mandate | allow | prefer**
12. **srtp callee forbid | mandate | allow | prefer**
13. **srtp interworking forbid | allow**
14. **srtp media interworking forbid | allow**
15. **action next-table** *goto-table-name*
16. **exit**
17. **exit**
18. **cac-table** *table-name*
19. **table-type limit** *list of limit tables*
20. **entry** *entry-id*
21. **match-value** *key*
22. **callee-codec-list** *list-name*
23. **action cac-complete**
24. **complete**
25. **cac-policy-set global** *cac-policy-num*
26. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters into the SBC configuration mode.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the SBE function mode of the SBC.
Step 4	**cac-policy-set** *policy-set-id* **Example:** `Router(config-sbc-sbe)# cac-policy-set 3`	Enters the CAC policy set configuration mode within an SBE entity, creating a new policy set, if necessary.

	Command or Action	Purpose
Step 5	**first-cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# first-cac-table C3`	Configures the name of the first policy table to be processed. A CAC policy may have many tables configured. To start applying the CAC policy, the first table that is used must be defined: • *table-name*—The admission control table that should be processed first.
Step 6	**cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# cac-table C3`	Enters the CAC table mode for configuration of an admission control table (creating one, if necessary) within the context of an SBE policy set: • *table-name*—Name of the admission control table.

	Command or Action	Purpose
Step 7	**table-type {policy-set | limit** {*list of limit tables*}} **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# table-type policy-set`	Configures the table type of a CAC table within the context of an SBC policy set. *list of limit tables* can be one of the following values: • **account**—Compare the name of the account. • **adj-group**—Compare the name of the adjacency group. • **adjacency**—Compare the name of the adjacency. • **all**—No comparison type. All events match this type. • **call-priority**—Compare with call priority. • **category**—Compare the number analysis assigned category. • **dst-account**—Compare the name of the destination account. • **dst-adj-group**—Compare the name of the destination adjacency group. • **dst-adjacency**—Compare the name of the destination adjacency. • **dst-prefix**—Compare the beginning of the dialed digit string. • **event-type**—Compare with CAC policy event types. • **src-account**—Compare the name of the source account. • **src-adj-group**—Compare the name of the source adjacency group. • **src-adjacency**—Compare the name of the source adjacency. • **src-prefix**—Compare the beginning of the calling number string. Features can be enabled or disabled per adjacency group through CAC configuration the same way this is done per individual adjacency. The adj-group table type matches on either the source adjacency group or the destination adjacency group. When the policy-set keyword is specified, use the **cac-scope** command to configure the scope within each entry in which limits are applied in a CAC Policy Set table.
Step 8	**entry** *entry-id* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# entry 1`	Enters the mode to create or modify an entry in an admission control table.

	Command or Action	Purpose

Step 9	**cac-scope {***list of scope options***}**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# cac-scope global`
		Choose a scope at which CAC limits are applied within each entry in a Policy Set table.
		
		list of scope options—Specifies one of the following strings used to match events:
		• *account*—Events that are from the same account.
		• *adjacency*—Events that are from the same adjacency.
		• *adj-group*—Events that are from members of the same adjacency group.
		• *call*—Scope limits are per single call.
		• *category*—Events that have the same category.
		• *dst-account*—Events that are sent to the same account.
		• *dst-adj-group*—Events that are sent to the same adjacency group.
		• *dst-adjacency*—Events that are sent to the same adjacency.
		• *dst-number*—Events that have the same destination.
		• *global*—Scope limits are global.
		• *src-account*—Events that are from the same account.
		• *src-adj-group*—Events that are from the same adjacency group.
		• *src-adjacency*—Events that are from the same adjacency.
		• *src-number*—Events that have the same source number.

Step 10	**srtp support allow**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# srtp support allow`
		Configures SRTP support.

Step 11	**srtp caller forbid | mandate | allow | prefer**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# srtp caller mandate`
		Configures SRTP for the caller side of the call with one of the following SRTP settings:
		• **forbid**—SRTP is not supported on the caller side of the call.
		• **mandate**—SRTP is mandatory on the caller side of the call.
		• **allow**—SRTP is optional on the caller side of the call.
		• **prefer**—SRTP is preferred on this adjacency. Both RTP and SRTP are accepted inbound, but only SRTP is offered outbound.

	Command or Action	Purpose
Step 12	**srtp callee forbid | mandate | allow | prefer** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# srtp callee mandate`	Configures SRTP for the callee side of the call: • **forbid**—SRTP is not supported on the callee side of the call. • **mandate**—SRTP is mandatory on the callee side of the call. • **allow**—SRTP is optional on the callee side of the call. • **prefer**—SRTP is preferred on this adjacency. Both RTP and SRTP are accepted inbound, but only SRTP is offered outbound.
Step 13	**srtp interworking forbid | allow** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# srtp interworking allow`	Configures SRTP-to-RTP interworking. • **forbid**—Prohibits SRTP-to-RTP interworking on a call. • **allow**—Allows SRTP-to-RTP interworking on a call.
Step 14	**srtp media interworking forbid | allow** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# srtp media interworking allow`	Configures SRTP-to-RTP media interworking. • **forbid**—Prohibits SRTP-to-RTP media interworking on a call. • **allow**—Allows SRTP-to-RTP media interworking on a call.
Step 15	**action next-table** *goto-table-name* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# action next-table xcode`	Configures the action to be taken when the routing entry is chosen. • *goto-table-name*—Specifies the next routing table to be processed when an event matches the entry.
Step 16	**exit** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# exit`	Exits the CAC table entry configuration mode.
Step 17	**exit** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# exit`	Exits the CAC table configuration mode.
Step 18	**cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# cac-table xcode`	Enters the CAC table mode for configuration of an admission control table (creating one, if necessary) within the context of an SBE policy set. • *table-name*—Name of the admission control table.
Step 19	**table-type limit** *list of limit tables* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# table-type limit dst-adjacency`	Configures the limit of the table types to be matched by the **match-value** command. For the example provided here, use the following table type: • *dst-adjacency*—Compares the name of the destination adjacency.

	Command or Action	Purpose
Step 20	**entry** *entry-id* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# entry 1`	Enters the CAC table entry mode to modify an entry in an admission control table. • *entry-id*—Specifies the table entry.
Step 21	**match-value** *key* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# match-value nav4B`	Configures the match-value of an entry in a Call Admission Control (CAC) Limit table: • *key*—Specifies the keyword used to match events. The format of the key is determined by the table-type limit.
Step 22	**callee-codec-list** *list-name* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# callee-codec-list PCMU`	Lists the codecs that the callee leg of a call is allowed to use: • *list-name*—Specifies the name of the codec list. The maximum size is 30 characters.
Step 23	**action cac-complete** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# action cac-complete`	When the event matches, this CAC policy is complete.
Step 24	**complete** **Example:** `Router(config-sbc-sbe-cacpolicy)# complete`	Completes the CAC policy set when you have committed the full set.
Step 25	**cac-policy-set global** *policy-num* **Example:** `Router(config-sbc-sbe)# cac-policy-set global 3`	Activates the global CAC policy set. The CAC policy set must be in a complete state before it can be assigned as the default policy. • *policy-num*—The call policy set number, ranging from 1 to 2147483647. The policy set must be in a complete state before it can be assigned as the default policy.
Step 26	**end** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# end`	Exits the CAC configuration mode and returns to privileged EXEC mode.

Configuring the Unified SBC for Inband DTMF Interworking

A SPA DSP can be used to detect the DTMF tones, called inband, that are played in the real-time transport protocol (RTP) stream. Inband DTMF interworking uses SPA DSP resources, and can be used for plain calls and transcoded calls.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

SUMMARY STEPS

1. **configure terminal**
2. **sbc** *sbc-name*
3. **sbe**
4. **cac-policy-set** *policy-set-id*
5. **first-cac-table** *table-name*
6. **cac-table** *table-name*
7. **table-type** {**policy-set** | **limit** {***list of limit tables***}}
8. **entry** *entry-id*
9. **cac-scope** {*list of scope options*}
10. **callee inband-dtmf-mode always**
11. **caller inband-dtmf-mode never**
12. **action next-table** *goto-table-name*
13. **complete**
14. **cac-policy-set global** *cac-policy-num*
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enables global configuration mode.
Step 2	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC`	Creates the SBC service on the SBC, and enters into the SBC configuration mode.
Step 3	**sbe** **Example:** `Router(config-sbc)# sbe`	Enters the SBE function mode of the SBC.

	Command or Action	Purpose
Step 4	**cac-policy-set** *policy-set-id* **Example:** `Router(config-sbc-sbe)# cac-policy-set 4`	Enters the CAC policy set configuration mode within an SBE entity, creating a new policy set, if necessary.
Step 5	**first-cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# first-cac-table C4`	Configures the name of the first policy table to be processed. A CAC policy may have many tables configured. To start applying the CAC policy, the first table that is used must be defined: • *table-name*—The admission control table that should be processed first.
Step 6	**cac-table** *table-name* **Example:** `Router(config-sbc-sbe-cacpolicy)# cac-table C4`	Enters the CAC table mode for configuration of an admission control table (creating one, if necessary) within the context of an SBE policy set: • *table-name*—Name of the admission control table.

	Command or Action	Purpose
Step 7	**table-type {policy-set | limit** {*list of limit tables*}} **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# table-type policy-set`	Configures the table type of a CAC table within the context of an SBC policy set. *list of limit tables* can be one of the following values: • **account**—Compare the name of the account. • **adj-group**—Compare the name of the adjacency group. • **adjacency**—Compare the name of the adjacency. • **all**—No comparison type. All events match this type. • **call-priority**—Compare with call priority. • **category**—Compare the number analysis assigned category. • **dst-account**—Compare the name of the destination account. • **dst-adj-group**—Compare the name of the destination adjacency group. • **dst-adjacency**—Compare the name of the destination adjacency. • **dst-prefix**—Compare the beginning of the dialed digit string. • **event-type**—Compare with CAC policy event types. • **src-account**—Compare the name of the source account. • **src-adj-group**—Compare the name of the source adjacency group. • **src-adjacency**—Compare the name of the source adjacency. • **src-prefix**—Compare the beginning of the calling number string. Features can be enabled or disabled per adjacency group through CAC configuration the same way this is done per individual adjacency. The adj-group table type matches either the source adjacency group or destination adjacency group. When the policy-set keyword is specified, use the **cac-scope** command to configure the scope within each entry at which limits are applied in a CAC Policy Set table.
Step 8	**entry** *entry-id* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable)# entry 1`	Enters the CAC table entry mode to create or modify an entry in an admission control table.

	Command or Action	Purpose

Step 9	**cac-scope {***list of scope options***}**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# cac-scope global`
		Choose the scope at which CAC limits are applied within each entry in a Policy Set table.
		
		list of scope options—Specifies one of the following strings used to match events:
		• *account*—Events that are from the same account.
		• *adjacency*—Events that are from the same adjacency.
		• *adj-group*—Events that are from members of the same adjacency group.
		• *call*—Scope limits are per single call.
		• *category*—Events that have the same category.
		• *dst-account*—Events that are sent to the same account.
		• *dst-adj-group*—Events that are sent to the same adjacency group.
		• *dst-adjacency*—Events that are sent to the same adjacency.
		• *dst-number*—Events that have the same destination.
		• *global*—Scope limits are global
		• *src-account*—Events that are from the same account.
		• *src-adj-group*—Events that are from the same adjacency group.
		• *src-adjacency*—Events that are from the same adjacency.
		• *src-number*—Events that have the same source number.

Step 10	**callee inband-dtmf-mode {always | inherit | maybe | never}**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# callee inband-dtmf-mode always`
		Configures the DTMF inband mode for the callee side.
		• **always**—The inband DTMF tones are always in use by the endpoint.
		• **inherit**—The inband DTMF mode for the endpoint is not affected by this CAC entry.
		• **maybe**—The inband DTMF tones are used by the endpoint unless signaling indicates that an alternative format for DTMF is in use.
		• **never**—The endpoint never uses inband DTMF.

Step 11	**caller inband-dtmf-mode {always | inherit | maybe | never}**
	
	Example:
	`Router(config-sbc-sbe-cacpolicy-cactable-entry)`
	`# caller inband-dtmf-mode never`
		Configures the DTMF inband mode for the caller side.
		• **always**—The inband DTMF tones are always in use by the endpoint.
		• **inherit**—The inband DTMF mode for the endpoint is not affected by this CAC entry.
		• **maybe**—The inband DTMF tones are used by the endpoint unless signaling indicates that an alternative format for DTMF is in use.
		• **never**—The endpoint never uses inband DTMF.

	Command or Action	Purpose
Step 12	**action next-table** *goto-table-name* **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# action next-table xcode`	Configures the action to take when this routing entry is chosen. • *goto-table-name*—Specifies the next routing table to be processed when an event matches the entry.
Step 13	**complete** **Example:** `Router(config-sbc-sbe-cacpolicy)# complete`	Completes the CAC policy set when you have committed the full set.
Step 14	**cac-policy-set global** *policy-num* **Example:** `Router(config-sbc-sbe)# cac-policy-set global 4`	Activates the global CAC policy set. The CAC policy set must be in a complete state before it can be assigned as the default policy. • *policy-num*—The call policy set number, ranging from 1 to 2147483647. The policy set must be in a complete state before it can be assigned as the default policy.
Step 15	**end** **Example:** `Router(config-sbc-sbe-cacpolicy-cactable-entry)# end`	Exits the CAC configuration mode and returns to privileged EXEC mode.

Configuring the Unified SBC to Support AMR-WB

This section explains how to configure the Unified SBC to support AMR-WB.

SUMMARY STEPS

1. **configure terminal**
2. **dspfarm profile** *profile-identifier* **transcode**
3. **codec amr-wb**
4. **sbc** *sbc-name*
5. **associate dspfarm profile** *profile-identifier*
6. **activate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 2	**dspfarm profile** *profile-identifier* **transcode** **Example:** `Router(config)# dspfarm profile 20 transcode`	Enters the DSP farm profile configuration mode, and defines a profile for DSP farm services.
Step 3	**codec amr-wb** **Example:** `Router(config-dspfarm-profile)#codec amr-wb`	Specifies the AMR-WB codec in the DSP farm profile.
Step 4	**sbc** *sbc-name* **Example:** `Router(config)# sbc mySBC dbe`	Enters the mode of an SBC service.
Step 5	**associate dspfarm profile** *profile-identifier* **Example:** `Router(config-sbc-dbe))# associate profile 20`	Associates a DSP farm profile to a Cisco Call Manager group.
Step 6	**activate** **Example:** `Router(config-sbc-dbe)# activate`	Initiates the DBE service of the SBC.

Configuration Examples of the SPA DSP Services for the SBC

This section contains the following examples:

Example: Enabling DSP Farm Service on the SPA DSP

The following example shows how to enable DSP farm services on the SPA DSP:

```
enable
configure terminal
voice-card 0/2
dsp services dspfarm
end
```

Example: Configuring a DSP Farm Profile

The following example shows how to configure a DSP farm profile:

```
enable
configure terminal
dspfarm profile 1 transcode
description enables transcoding
    codec g711ulaw
 codec g711alaw
 codec g729ar8
 codec g729abr8
 codec g729r8
 codec g723r63
 codec ilbc
 codec gsmamr-nb
 codec g726r32
 codec g729br8
associate application sbc
maximum session 300
end
```

Example: Viewing a DSP Farm Profile Configuration and Status

After a DSP farm profile is created, use the **show** command to display a DSP farm profile configuration and status. The following examples show the output of the **show** commands:

```
Router# show running-config
!
voice-card 2/0
no dspfarm
dsp services dspfarm
!
dspfarm profile 20 transcode
codec g711ulaw
codec g711alaw
codec g729r8
codec g729ar8
codec g729br8
codec g729abr8
rsvp
maximum sessions 5
associate application SBC
!

Router# show dspfarm profile 20

Dspfarm Profile Configuration
Profile ID = 20, Service = TRANSCODING, Resource ID = 1
Profile Description :
Profile Admin State : UP
Profile Operation State : ACTIVE
Application : SBC Status : ASSOCIATED
Resource Provider : FLEX_DSPRM Status : UP
Number of Resource Configured : 5
Number of Resource Available : 5
Codec Configuration
Codec : g729abr8, Maximum Packetization Period : 60
Codec : g711alaw, Maximum Packetization Period : 30
Codec : g711ulaw, Maximum Packetization Period : 30
Codec : g729r8, Maximum Packetization Period : 60
```

```
Codec : g729ar8, Maximum Packetization Period : 60
Codec : g729br8, Maximum Packetization Period : 60
RSVP : ENABLED
!

Router# show dspfarm all

DSPFARM Configuration Information:
Admin State: UP, Oper Status: ACTIVE - Cause code: NONE
Transcoding Sessions: 0(Avail: 0), Conferencing Sessions: 2 (Avail: 2)
Trans sessions for mixed-mode conf: 0 (Avail: 0), RTP Timeout: 600
Connection check interval 600 Codec G729 VAD: ENABLED
Total number of active session(s) 0, and connection(s) 0
SLOT DSP CHNL STATUS USE   TYPE SESS-ID CONN-ID PKTS-RXED PKTS-TXED
0   0   1   UP  FREE conf - -  -   -
0   0   2   UP  FREE conf- -   -   -
0   0   3   UP  FREE conf - -  -   -
0   0   4   UP  FREE conf - -  -   -
0   0   5   UP  FREE conf - -  -   -
0   0   6   UP  FREE conf - -  -   -
```

Configuration Examples of Unified SBC

This section contains the following examples:

- Example: Associating the Unified SBC with a DSP Farm Profile, page 41-36
- Example: Configuring the Unified SBC to Enable Transcoding, page 41-36
- Example: Configuring the Unified SBC to Enable Transrating, page 41-37
- Example: Configuring the Unified SBC to Enable SRTP and Transcoding, page 41-38
- Example: Configuring the Unified SBC for In-Band DTMF Interworking, page 41-38
- Example: Configuring the Unified SBC to Support AMR-WB, page 41-39

Example: Associating the Unified SBC with a DSP Farm Profile

The following example shows how to associate the Unified SBC with a DSP farm profile:

```
enable
 configure terminal
  sbc mySBC
  associate dspfarm profile 1
  end
```

Example: Configuring the Unified SBC to Enable Transcoding

The following example shows how to configure the unified SBC to enable transcoding.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

```
enable
 configure terminal
  sbc mySBC
   sbe
   cac-policy-set 1
   first-cac-scope dst-adjacency
   first-cac-table codec-dst-acc
   cac-table codec-dst-acc
   table-type limit dst-adjacency
    entry 1
    match-value nava
    caller-codec-list PCMU
    callee-codec-list PCMA
    media police strip
    action cac-complete
    complete
   cac-policy-set global 1
 codec-list PCMU
 codec PCMU
 exit
 codec-list PCMA
 codec PCMA
 exit
end
```

Example: Configuring the Unified SBC to Enable Transrating

Note Transrating is supported only for different rates using the same codec, not across codecs. Therefore, transrating and transcoding cannot be performed simultaneously.

The following example shows how to configure the unified SBC for enabling the transrating feature using the same codec policy:

```
enable
 configure terminal
  sbc mySBC
   sbe
   cac-policy-set 2
   first-cac-table Transrate
   cac-table Transrate
   table-type policy-set
   entry 1
    cac-scope call
    callee ptime 30
    caller ptime 20
    media police strip
    action cac complete
    complete
   cac-policy-set global 2
   end
```

The following example shows how to configure the Unified SBC for enabling the transrating feature using the same codec policy:

```
enable
 configure terminal
  sbc MySBC
   sbe
    codec list PCMU
```

```
 policy transrating
 codec PCMU packetization-period 200 priority 1
 end
```

Example: Configuring the Unified SBC to Enable SRTP and Transcoding

The following example shows how to configure SBC to enable the SRTP and transcoding features.

```
enable
 configure terminal
  sbc mySBC
   sbe
   cac-policy-set 3
   first-cac-table C3
   cac-table c3
   table-type policy-set
   entry 1
    cac-scope global
    srtp support allow
    srtp caller mandate
    srtp callee mandate
    srtp interworking allow
    srtp media interworking allow
    action next-table xcode
    exit
   exit
  cac-table xcode
  table-type limit dst-adjacency
  entry 1
   match-value nav4b
   callee-codec-list PCMU
   action cac-complete
   complete
  cac-policy-set global 3
  end
```

Example: Configuring the Unified SBC for In-Band DTMF Interworking

The following example shows how to configure the unified SBC for inband DTMF transmission.

Note The **caller** and **callee** commands have been used in this procedure. In some scenarios, the **branch** command can be used as an alternative to the **caller** and **callee** command pair. The **branch** command has been introduced in Release 3.5.0. See the ?$paranum>Configuring Directed Nonlimiting CAC Policies? section on page 7-37 for information about this command.

```
enable
 configure terminal
  sbc mySBC
  sbe
   cac-policy-set 4
   first-cac-table c4
   cac-table c4
   table-type policy-set
   entry 1
    cac-scope global
    callee inband-dtmf-mode always
    caller inband-dtmf-mode never
```

```
   action next-table xcode
   exit
  exit
 cac-table xcode
 table-type limit dst-adjacency
 entry 1
  match-value spab
  callee-codec-list PCMU
  action cac-complete
  complete
 cac-policy-set global 4
  end
```

Example: Configuring the Unified SBC to Support AMR-WB

The following example shows how to configure the Unified SBC to support AMR-WB:

```
enable
configure terminal
sbc mySBC
sbe
cac-policy-set 1
first-cac-scope dst-adjacency
first-cac-table codec-dst-acc
cac-table codec-dst-acc
table-type limit dst-adjacency
entry 1
match-value nava
caller-codec-list AMRWB
callee-codec-list PCMA
media police strip
action cac-complete
complete
cac-policy-set global 1
codec-list AMRWB
codec AMR-WB
exit
codec-list PCMA
codec PCMA
exit
```