Configuring SIP and TEL URL Support

The Configuring SIP and TEL URL Support feature enables Cisco gateways to direct incoming calls to a voice application based on the Uniform Resource Locator (URL) and Tcl Interactive Voice Response (IVR) 2.0 and VoiceXML applications to place outbound calls to a SIP or TEL URL. This feature expands call-control capabilities by allowing voice applications to use URL destinations and by implementing dialing plans using SIP or TEL URLs, rather than telephone numbers.

Note In this document, the terms uniform resource identifier (URI) and URL are used interchangeably.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Feature Information for Configuring SIP and TEL URL Support" section on page 31.

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

Contents

Prerequisites for Configuring SIP and TEL URL Support

- Install Cisco IOS Release 12.3(4)T or a later release on the voice gateway.
- Configure basic voice application functionality on the gateway. For information, see "Configuring Basic Functionality for Tcl and VoiceXML Applications".
- Write a Tcl IVR 2.0 script or VoiceXML document that implements Session Iniation Protocol (SIP) or telephone (TEL) URL support. For information, see the *Tcl IVR API Version 2.0 Programmer's Guide* or *Cisco VoiceXML Programmer's Guide.*

Restrictions for Configuring SIP and TEL URL Support

- Access is provided only to headers in the SIP Invite, Subscribe, and Notify messages and H.323 Setup message.
- SIP and TEL URLs are not supported destinations for H.450 or SIP call transfers or call forwarding using Tcl IVR or VoiceXML applications.
- A destination URL can have a maximum length of 1024 bytes.
- H.323 rejects a call if the URL is more than 512 characters because H.225 limits the URL length to 512 bytes.

Information About SIP and TEL URL Support

To configure voice application enhancements on a Cisco gateway, you should understand the following concepts:

- SIP and TEL URL Support for Voice Applications, page 2
- Benefits of SIP and TEL URL Support, page 3

SIP and TEL URL Support for Voice Applications

Cisco IOS Release 12.3(4)T allows Tcl IVR 2.0 and VoiceXML applications to place calls to SIP and TEL URL destinations in addition to E.164 telephone numbers. For outgoing calls, the voice application provides the destination as either an URL or an E.164 number. Incoming calls are matched to a dial peer based on the URL, triggering the voice application that is configured in the dial peer. New dial-peer matching rules are implemented for URLs. The standard dial-peer matching rules apply to telephone number destinations. VoiceXML applications can use the transfer tag to place calls to URL destinations. Tcl IVR 2.0 scripts can do a leg setup to a URL instead of a telephone number.

For configuration information, use the following references:

- To create a voice class that defines the criteria for matching a URL, see the "Configuring a Voice Class URI" section on page 3.
- To assign the URL voice class to an inbound or outbound dial peer, see the "Configuring an Inbound Dial Peer to Match on a URI" section on page 7 or the "Configuring an Outbound Dial Peer for URI Destinations" section on page 15.
- For a description of the new dial-peer matching rules for URLs, see to the **destination uri** command and **incoming uri** command in the *Cisco IOS Voice Command Reference*.

- To enable voice applications to read and pass SIP headers, see the *Cisco IOS SIP Configuration Guide*.
- To specify SIP and TEL URLs in a Tcl IVR 2.0 script or VoiceXML document, see the *Tcl IVR API Version 2.0 Programmer's Guide* or *Cisco VoiceXML Programmer's Guide*.

Benefits of SIP and TEL URL Support

- Enhances call-control features by supporting SIP and TEL URL destinations for a VoiceXML transfer or Tcl leg setup. Incoming calls can be directed to voice applications based on a SIP or TEL URL, and outbound calls can be placed using a URL to route the call.
- Complies with the W3C VoiceXML 2.0 Working Draft requirement for TEL URL support, enabling additional information such as the account number to be provided when a call is transferred to or from a VoiceXML application.
- Allows dialing plans to be implemented using SIP URLs instead of telephone numbers, simplifying the management of SIP telephony networks.

How to Configure SIP and TEL URL Support

This section contains the following procedures:

- Configuring a Voice Class URI, page 3 (required)
- Configuring an Inbound Dial Peer to Match on a URI, page 7 (required)
- Verifying Dial-Peer Configuration for an Incoming URI Based Dial-Peer Match, page 12 (optional)
- Configuring an Outbound Dial Peer for URI Destinations, page 15 (required)
- Verifying Dial-Peer Configuration for an Outgoing URI, page 16 (optional)

Configuring a Voice Class URI

To configure a URI voice class and assign it to an inbound or outbound dial peer, choose one of the following tasks, depending on whether a call has a SIP or TEL URI:

- Configuring a Voice Class SIP URI, page 3
- Configuring a Voice Class TEL URL, page 6

You define sets of related dial-peer attributes in voice classes. The Configuring SIP and TEL URL Support feature introduces a new type of voice class for URI attributes. A URI voice class defines the criteria for matching the URI in a call. You can configure a voice class to match either the entire URI or specific fields within the URI, but, not both. You then assign the voice class to an inbound or outbound dial peer. Calls are matched to the dial peer based on the URI.

Configuring a Voice Class SIP URI

This section describes how to configure a voice class to match on a SIP URI.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **voice class uri** *tag* **sip**
4. **pattern** *uri-pattern*
5. **host** *hostname-pattern* or **host** {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***domain-name* | *hostname-pattern*}
6. **user-id** *username-pattern*
7. **phone context** *context-pattern*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *tag* **sip** **Example:** `Router(config)# voice class uri ab200 sip`	Enter voice class configuration mode for a SIP URI.
Step 4	**pattern** *uri-pattern* **Example:** `Router(config-voice-uri-class)# pattern ^408`	(Optional) Specifies a regular expression pattern for matching the entire TEL URI. **Note** When you configure the **pattern** command, you cannot configure the **phone number** or **phone context** command, because the **pattern** command matches the entire URI.

Step 5	**host** *hostname-pattern*
	or
	host {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***domain-name*}
	
	Example:
	`Router(config-voice-uri-class)# host server1`
	or
	`Router(config-voice-uri-class)# host ipv4:10.0.0.0`
		(Optional) Specifies a regular expression pattern for matching the hostname field in the SIP URI.
		• Only one instance of the command can be configured at any given time.
		Specifies the **host** command by assigning an IPv4 address, IPv6 address, or Domain Name Server (DNS) name. You can specify up to ten instances of this command.
		Note You can use either the **host** *hostname* or **host** {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***domain-name*} command.

Step 6	**user-id** *username-pattern* **Example:** `Router(config-voice-uri-class)# user-id elmo`	(Optional) Specifies a regular expression pattern for matching the username field in the SIP URI.
Step 7	**phone context** *context-pattern* **Example:** `Router(config-voice-uri-class)# phone context 408`	(Optional) Specifies that only calls with a matching phone-context field in the URI are selected. **Note** You must use the **phone context** command with either the **host** or **user-id** command or both. Using this command alone does not result in any matches on the voice class.
Step 8	**end** **Example:** `Router(config-voice-uri-class)# end`	Exits voice class URI configuration mode and enters privileged EXEC mode.

What to Do Next

- To use the URI voice class to handle incoming calls, continue with the "Configuring an Inbound Dial Peer to Match on a URI" section on page 7.
- To use the URI voice class to handle outbound calls, continue with the "Configuring an Outbound Dial Peer for URI Destinations" section on page 15.

Configuring a Voice Class TEL URL

This task describes how to configure a voice class to match a TEL URI.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class uri** *tag* **tel**
4. **pattern** *uri-pattern*
5. **phone number** *phone-number-pattern*
6. **phone context** *context-pattern*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *tag* **tel** **Example:** `Router(config)# voice class uri ab200 tel`	Enters voice class URI configuration mode for a TEL URI.
Step 4	**pattern** *uri-pattern* **Example:** `Router(config-voice-uri-class)# pattern ^408`	(Optional) Specifies a regular expression pattern for matching the entire TEL URI. **Note** If you use the **pattern** command you cannot use the **phone number** or **phone context** command in the following steps because the **pattern** command matches the entire URI.
Step 5	**phone number** *phone-number-pattern* **Example:** `Router(config-voice-uri-class)# phone number ^8555`	(Optional) Specifies a regular expression pattern for matching the phone number field in the TEL URI.
Step 6	**phone context** *context-pattern* **Example:** `Router(config-voice-uri-class)# phone context 555`	(Optional) Specifies that only calls with a matching phone-context field are selected. **Note** You must use the **phone context** command with the **phone number** command. Using this command alone does not result in any matches on the voice class.
Step 7	end **Example:** `Router(config-voice-uri-class)# end`	Exits voice class configuration mode and enters privileged EXEC mode.

Configuring an Inbound Dial Peer to Match on a URI

To configure an inbound dial peer to match an URI, choose one of the following tasks, depending on whether the dial peer uses SIP or H.323.

- Configuring an Inbound Dial Peer to Match the URI in H.323 Calls, page 8
- Configuring an Inbound Dial-Peer to Match the URI on SIP Calls, page 9

Prerequisites

- Enable SIP header passing. For information, see the *Cisco IOS SIP Configuration Guide*, Release 15.1
- Write a Tcl IVR 2.0 script or VoiceXML document that accepts a SIP or TEL URI. For information, see the *Tcl IVR API Version 2.0 Programmer's Guide* or *Cisco VoiceXML Programmer's Guide*.

Configuring an Inbound Dial Peer to Match the URI in H.323 Calls

This task describes how to configure an inbound dial peer to match the URI in a H.323 call.

SUMMARY STEPS

1. enable
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **service** *name*
5. **incoming uri** {**called** | **calling**} *tag*
6. **end**

DETAILED STEPS.

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 2 voip`	Enters dial-peer configuration mode for a VoIP dial peer.
Step 4	**service** *name* **Example:** `Router(config-dial-peer)# service vapp1`	Associates an application with this VoIP dial peer. **Note** When you configure the **pattern** command, you cannot configure the **phone number** or **phone context** command, because the **pattern** command matches the entire URI.
Step 5	**incoming uri** {**called** | **calling**} *tag* **Example:** `Router(config-dial-peer)# incoming uri called ab400`	Specifies the URI voice class that matches calls to this dial peer based on the destination URI or source URI in the H.225 message. **Note** This URI voice class must already be configured by using the **voice class uri** command. **Note** Incoming calls with a URI that matches the configured voice class are linked to the dial peer and to the VoiceXML or Tcl IVR application that you configured in Step 4.
Step 6	end **Example:** `Router(config-dial-peer)# end`	Exits the dial peer configuration mode and enters privileged EXEC mode.

Configuring an Inbound Dial-Peer to Match the URI on SIP Calls

This task describes how to configure an inbound dial-peer to match the URI in a SIP call.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class uri** *tag* **sip**
4. **host** *hostname-pattern* or **host** {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***dns-address* | *hostname-pattern*}
5. **exit**

6. **dial-peer voice** *tag* **voip**
7. **session protocol sipv2**
8. **incoming uri** {**from** | **request** | **to** | **via**} *tag*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *tag* **sip'** **Example:** `Router(config)# voice class uri ab200 sip`	Creates a voice class for matching dial peers to a SIP and enters voice URI class configuration mode.

Step 4	**host** *hostname-pattern*
	or
	host {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***domain-name*}
	
	Example:
	`Router(config-voice-uri-class)# host server1`
	or
	`Router(config-voice-uri-class)# host ipv4:10.0.0.0`
		(Optional) Specifies a regular expression pattern for matching the hostname field in the SIP URI.
		• This command can have a single instance only.
		Specifies the host command by assigning the IPv4 address, IPv6 address, or DNS name.
		• You can specify up to ten instances of this command.
		Note You can use either the **host** *hostname* or **host** {**ipv4:***ipv4-address* | **ipv6:***ipv6-address* | **dns:***domain-name*} command.

Step 5	**exit** **Example:** `Router(config-voice-uri-class)# exit`	Enters global configuration mode.
Step 6	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 6000 voip`	Enters dial peer voice configuration mode.
Step 7	**session protocol sipv2** **Example:** `Router(config-dial-peer)# session protocol sipv2`	Configures SIP as the session protocol type.
Step 8	**incoming uri** {**from** | **request** | **to** | **via**} *tag* **Example:** `Router(config-dial-peer)# incoming uri via ab200`	Specifies the voice class used to match a VoIP dial peer to the URI of an incoming call.
Step 9	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and enters privileged EXEC mode.

Verifying Dial-Peer Configuration for an Incoming URI Based Dial-Peer Match

Perform this task to display information about the dial-peer configuration for an incoming URI based dial-peer match. These **show** commands need not be entered in any specific order.

SUMMARY STEPS

1. **show voice class uri** [*tag* | **summary**]
2. **show dial-peer voice**
3. **show dialplan incall uri sip** {*from* | *request* | *to* | *via*} **uri**

DETAILED STEPS

Step 1 **show voice class uri** [*tag* | **summary**]

Use the **show voice class uri summary** command to display summary information about the configured URI voice classes. The following sample output shows the configured host instances under the URI voice class:

```
Router# show voice class uri 1
Voice URI class: 1
SNMP status = Active
Schema = sip
user-id =
host =
phone context =
Host instances:
 ipv4:10.0.0.0
 ipv6:[2001:0DB8:0:1:FFFF:1234::5]
 dns:ogw.example.com
 exam.example.com
```

Step 2 **show dial-peer voice**

Use the **show dial-peer voice** command to display information for voice dial peers. The following sample output shows the voice class URI 2 being set into dial peer 2:

```
Router# show dial-peer voice 2
VoiceOverIpPeer2
        description = `',
        tag = 2, destination-pattern = `',
        URI classes:
        Incoming (Request) =
        Incoming (Via) = 1
        Incoming (To) =
        Incoming (From) =
        Destination
```

Step 3 **show dialplan incall uri sip** {*from* | *request* | *to* | *via*} **uri**

Use the **show dialplan incall uri** to display the dial peer that is matched for a specific URI in an incoming voice call. The following sample output shows the dial peer that is matched for an incoming call, based on the selected URI:

```
Router# show dialplan incall uri sip via sip:9.13.38.83
 Inbound VoIP dialpeer matching based on SIP URI's

VoiceOverIpPeer100
        peer type = voice, system default peer = FALSE, information type = voice,
        description = `',
```

```
tag = 100, destination-pattern = `',
voice reg type = 0, corresponding tag = 0,
allow watch = FALSE
answer-address = `', preference=0,
CLID Restriction = None
CLID Network Number = `'
CLID Second Number sent
CLID Override RDNIS = disabled,
rtp-ssrc mux = system
source carrier-id = `',target carrier-id = `',
source trunk-group-label = `',target trunk-group-label = `',
numbering Type = `unknown'
group = 100, Admin state is up, Operation state is up,
incoming called-number = `', connections/maximum = 0/unlimited,
DTMF Relay = disabled,
modem transport = system,
URI classes:
    Incoming (Request) =
    Incoming (Via) = 1
    Incoming (To) =
    Incoming (From) =
    Destination =
huntstop = disabled,
in bound application associated: 'DEFAULT'
out bound application associated: ''
dnis-map =
permission :both
incoming COR list:maximum capability
outgoing COR list:minimum requirement
outgoing LPCOR:
Translation profile (Incoming):
Translation profile (Outgoing):
incoming call blocking:
translation-profile = `'
disconnect-cause = `no-service'
advertise 0x40 capacity_update_timer 25 addrFamily 4 oldAddrFamily 4
mailbox selection policy: none
type = voip, session-target = `',
technology prefix:
settle-call = disabled
ip media DSCP = ef, ip media rsvp-pass DSCP = ef
ip media rsvp-fail DSCP = ef, ip signaling DSCP = af31,
ip video rsvp-none DSCP = af41,ip video rsvp-pass DSCP = af41
ip video rsvp-fail DSCP = af41,
ip defending Priority = 0, ip preemption priority = 0
ip policy locator voice:
ip policy locator video:
UDP checksum = disabled,
session-protocol = sipv2, session-transport = system,
req-qos = best-effort, acc-qos = best-effort,
req-qos video = best-effort, acc-qos video = best-effort,
req-qos audio def bandwidth = 64, req-qos audio max bandwidth = 0,
req-qos video def bandwidth = 384, req-qos video max bandwidth = 0,
RTP dynamic payload type values: NTE = 101
Cisco: NSE=100, fax=96, fax-ack=97, dtmf=121, fax-relay=122
       CAS=123, TTY=119, ClearChan=125, PCM switch over u-law=0,
       A-law=8, GSMAMR-NB=117 iLBC=116, AAC-ld=114, iSAC=124
       lmr_tone=0, nte_tone=0
       h263+=118, h264=119
       G726r16 using static payload
       G726r24 using static payload
RTP comfort noise payload type = 19
fax rate = voice,   payload size =  20 bytes
fax protocol = system
```

```
        fax-relay ecm enable
        Fax Relay ans enabled
        Fax Relay SG3-to-G3 Enabled (by system configuration)
        fax NSF = 0xAD0051 (default)
        codec = g729r8,   payload size =  20 bytes,
        video codec = None
        voice class codec = `'
        voice class sip session refresh system
        voice class sip rsvp-fail-policy voice post-alert mandatory keep-alive interval
30
        voice class sip rsvp-fail-policy voice post-alert optional keep-alive interval 30
        voice class sip rsvp-fail-policy video post-alert mandatory keep-alive interval
30
        voice class sip rsvp-fail-policy video post-alert optional keep-alive interval 30
        text relay = disabled
        Media Setting = forking (disabled) flow-through (global)
        Expect factor = 10, Icpif = 20,
        Playout Mode is set to adaptive,
        Initial 60 ms, Max 1000 ms
        Playout-delay Minimum mode is set to default, value 40 ms
        Fax nominal 300 ms
        Max Redirects = 1, signaling-type = cas,
        VAD = enabled, Poor QOV Trap = disabled,
        Source Interface = NONE
        voice class sip url = system,
        voice class sip tel-config = system,
        voice class sip rel1xx = system,
        tvoice class sip outbound-proxy = system,
        voice class sip asserted-id = system,
        voice class sip privacy = system,
        voice class sip e911 = system,
        voice class sip history-info = system,
        voice class sip pass-thru headers = system,
        voice class sip pass-thru content unsupp = system,
        voice class sip pass-thru content sdp = system,
        voice class sip anat = system,
        voice class sip g729 annexb-all = system,
        voice class sip early-offer forced = system,
        voice class sip negotiate cisco = system,
        voice class sip reset timer expires 183 = system,
        voice class sip block 180 = system,
        voice class sip block 181 = system,
        voice class sip block 183 = system,
        voice class sip preloaded-route = system,
        voice class sip random-contact = system,
        voice class sip random-request-uri validate = system,
        voice class sip call-route p-called-party-id = system,
        voice class sip call-route history-info = system,
        voice class sip privacy-policy send-always = system,
        voice class sip privacy-policy passthru = system,
        voice class sip privacy-policy strip history-info = system,
        voice class sip privacy-policy strip diversion = system,
        voice class sip bandwidth audio  = system,
        voice class sip bandwidth video  = system,
        voice class sip error-code-override options-keepalive failure = system,
        voice class sip encap clear-channel  = system,
        voice class sip map resp-code 181 = system,
        voice class sip bind control = system,
        voice class sip bind media = system,
        voice class sip authenticate redirecting-number = system,
        redirect ip2ip = disabled
        local peer = false
        probe disabled,
        Secure RTP: system (use the global setting)
```

```
        voice class perm tag = `'
        Time elapsed since last clearing of voice call statistics never
        Connect Time = 0, Charged Units = 0,
        Successful Calls = 0, Failed Calls = 0, Incomplete Calls = 0
        Accepted Calls = 0, Refused Calls = 0,
        Last Disconnect Cause is "",
        Last Disconnect Text is "",
        Last Setup Time = 0.
        Last Disconnect Time = 0.
Matched:    Digits: 0
Target:
```

Configuring an Outbound Dial Peer for URI Destinations

This task describes how to configure an outbound dial peer for calls placed to an URI destination.

Prerequisites

Write a Tcl IVR 2.0 script or VoiceXML document that implements call transfer to a SIP or TEL URI. For information, see the *TCL IVR API Version 2.0 Programming Guide* or *Cisco VoiceXML Programmer's Guide*.

Restrictions

- Outbound calls to a SIP URL cannot be placed to a public switched telephone network (PSTN) leg because dial-peer matching does not support this. Outbound calls to a TEL URL can be placed to a PSTN call leg. The telephone number is pulled from the URL and used as the destination number.
- Underscores are not supported in the From header or in the callinfo originationNum field of a SIP URI in a Tcl script or VoiceXML document.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **session target ipv4:***ip-address*
5. **session protocol sipv2**
6. **destination uri** *tag*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**`enable`** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**`configure terminal`** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**`dial-peer voice`** *`tag`* **`voip`** **Example:** `Router(config)# dial-peer voice 2 voip`	Enters dial peer configuration mode for a VoIP dial peer.
Step 4	**`session target ipv4:`***`ip-address`* **Example:** `Router(config-dial-peer)# session target ipv4:10.10.1.1`	(Optional) Specifies the IP address of the terminating gateway **Note** The **session target** command is required when placing a call to a TEL URI; it is optional for a SIP URI. If you do not configure a session target for a SIP URI, the call is sent to the host address in the SIP URI that is passed from the application initiating the outbound call.
Step 5	**`session protocol sipv2`** **Example:** `Router(config-dial-peer)# session protocol sipv2`	(Optional) Specifies the session protocol if you are using SIP for calls between the local and remote gateways. **Note** If you are using H.323 do not configure the **session protocol** command.
Step 6	**`destination uri`** *`tag`* **Example:** `Router(config-dial-peer)# destination uri 700`	Specifies the URI class that links voice calls to this dial peer. **Note** Before configuring the **destination uri** command, you should configure the URI voice class by using the **voice class uri** command.
Step 7	**`end`** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and enters privileged EXEC mode.

Verifying Dial-Peer Configuration for an Outgoing URI

Perform this task to display information about the dial-peer configuration for an outgoing URI. These **show** commands need not be entered in any specific order.

SUMMARY STEPS

1. **show running-config**

2. **show voice class uri** [*tag* | **summary**]
3. **show dialplan uri h323** *uri*

DETAILED STEPS

Step 1 **show running-config**

Use the **show running-config** command to display the configuration of the URI voice class and the outbound dial peer.

The following example shows that voice class 500 is assigned to dial peer 599, which matches the calls that have a pattern beginning with 123 in the outgoing SIP URI:

```
Router# show running-config
.
.
.
voice class uri  500 sip
 pattern 123...
!
!
dial-peer voice 599 voip
 session protocol sipv2
 session target ipv4:10.10.1.1
 destination uri 500
 codec g711ulaw
!
```

Step 2 **show voice class uri** [*tag* | **summary**]

Use the **show voice class uri** command to display the configuration of the voice class:

```
Router# show voice class uri 500

Voice URI class: 500
    Schema = sip
    pattern = 123...
```

To display a list of all voice classes, use the **show voice class uri summary** command.

```
Router# show voice class uri summary
__________________________________________
Class Name                        Schema
------------------------------------------
100                               sip
300                               tel
500                               sip
__________________________________________
```

Step 3 **show dialplan uri**

Use the **show dialplan uri** command to verify that the correct dial peer is matched for the URI.

The following example shows that dial peer 599 is linked to voice class 500, which matches the specified URI. It also shows that the operational state of the dial peer is up:

```
Router# show dialplan uri sip:123456

 Outbound dialpeer matching based on destination URI

VoiceOverIpPeer599
        peer type = voice, information type = voice,
        description = `',
        tag = 599, destination-pattern = `',
```

```
        answer-address = `', preference=0,
        CLID Restriction = None
        CLID Network Number = `'
        CLID Second Number sent
        source carrier-id = `', target carrier-id = `',
        source trunk-group-label = `',  target trunk-group-label = `',
        numbering Type = `unknown'
        group = 599, Admin state is up, Operation state is up,
        incoming called-number = `', connections/maximum = 0/unlimited,
        DTMF Relay = disabled,
        modem transport = system,
        URI classes:
            Incoming (Request) =
            Incoming (To) =
            Incoming (From) =
            Destination = 500
        huntstop = disabled,
        in bound application associated: 'DEFAULT'
        out bound application associated: ''
        dnis-map =
        permission :both
        incoming COR list:maximum capability
        outgoing COR list:minimum requirement
        Translation profile (Incoming):
        Translation profile (Outgoing):
        incoming call blocking:
        translation-profile = `'
        disconnect-cause = `no-service'
        type = voip, session-target = `10.10.1.1',
        technology prefix:
        settle-call = disabled
        ip media DSCP = ef, ip signaling DSCP = af31, UDP checksum = disabled,
        session-protocol = sipv2, session-transport = system, req-qos = best-ef
        acc-qos = best-effort,
        RTP dynamic payload type values: NTE = 101
        Cisco: NSE=100, fax=96, fax-ack=97, dtmf=121, fax-relay=122
               CAS=123, ClearChan=125, PCM switch over u-law=0,A-law=8
        RTP comfort noise payload type = 19
        fax rate = voice,   payload size =  20 bytes
        fax protocol = system
        fax-relay ecm enable
        fax NSF = 0xAD0051 (default)
        codec = g711ulaw,   payload size =  20 bytes,
        Expect factor = 0, Icpif = 20,
        Playout Mode is set to default,
        Initial 60 ms, Max 300 ms
        Playout-delay Minimum mode is set to default, value 40 ms
        Fax nominal 300 ms
        Max Redirects = 1, signaling-type = ext-signal,
        VAD = enabled, Poor QOV Trap = disabled,
        Source Interface = NONE
        voice class sip url = system,
        voice class sip rel1xx = system,
        voice class perm tag = `'
        Time elapsed since last clearing of voice call statistics never
        Connect Time = 0, Charged Units = 0,
        Successful Calls = 0, Failed Calls = 0, Incomplete Calls = 0
        Accepted Calls = 0, Refused Calls = 0,
        Last Disconnect Cause is "",
        Last Disconnect Text is "",
        Last Setup Time = 0.
Matched:    Digits: 0
Target:
```

Note For a description of the fields shown in this output, see the **show dialplan uri** command in the *Cisco IOS Voice Command Reference.*

Troubleshooting Tips

- Do not use underscores in the From header or in the callinfo originationNum field of a SIP URI in a Tcl script or VoiceXML document. Using underscores in these fields, as shown in the following examples, causes call transfer to fail:

  ```
  set callInfo(originationNum) "sip:firstname_lastname@example.com"

  set headers(From) "sip:firstname_lastname"
  ```

- Use Tcl **puts** commands or VoiceXML log commands in your script to help with debugging. To display the output from these commands, use the **debug voip ivr script** command for Tcl IVR 2.0 scripts, or the **debug vxml puts** command for VoiceXML documents.

  For information on using the Tcl **puts** command, see the *TCL IVR API Version 2.0 Programming Guide*. For information about the VoiceXML log command, see the *Cisco VoiceXML Programmer's Guide*.

Troubleshooting URI Matching

Perform this task to troubleshoot URI matching. These **debug** commands need not be entered in any specific order.

SUMMARY STEPS

1. **debug dialpeer**
2. **debug voice uri**
3. **debug voice ccapi error**

DETAILED STEPS

Step 1 **debug dialpeer**

Use the **debug dialpeer** command to determine whether there is a dial peer with a voice class that matches the URI in the Tcl script or VoiceXML document.

In the following example, the incoming call is matched to dial peer 1000, then the gateway searches through the dial peers looking for a voice class that matches the URI in the outbound script:

```
Router# debug dialpeer

dialpeer detailed info debugging is on

*Mar  1 00:56:40.711: dpAssociateIncomingPeerSPI:
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_INCOMING_DNIS)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match incoming called number; called
(50267)
*Mar  1 00:56:40.711: dpMatchCore:
*Mar  1 00:56:40.711: dpMatchCore: dialstring(50267); expanded string(50267); calling()
```

```
*Mar  1 00:56:40.711: FillTarget: pDest(50267T) pPatn(50267)
*Mar  1 00:56:40.711: MatchNextPeer: peer 1000 matched
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=0 after DP_MATCH_INCOMING_DNIS;
peers (0x62F2E454)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerSPI:
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_INCOMING_DNIS)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match incoming called number; called
(50267)
*Mar  1 00:56:40.711: dpMatchCore:
*Mar  1 00:56:40.711: dpMatchCore: dialstring(50267); expanded string(50267); calling()
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=-1 after DP_MATCH_INCOMING_DNIS;
peers (0x0)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_ANSWER)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match answer address; calling (50006)
*Mar  1 00:56:40.711: dpMatchCore:
*Mar  1 00:56:40.711: dpMatchCore: dialstring(); expanded string(); calling(50006T)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=-1 after DP_MATCH_ANSWER; peers
(0x0)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_ORIGINATE)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match destination pattern; calling
(50006)
*Mar  1 00:56:40.711: dpMatchCore:
*Mar  1 00:56:40.711: dpMatchCore: dialstring(); expanded string(); calling(50006T)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=-1 after DP_MATCH_ORIGINATE;
peers (0x0)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_PORT)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=-1 after DP_MATCH_PORT; peers
(0x0)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_SRC_CARRIER)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=-1 after DP_MATCH_SRC_CARRIER;
peers (0x0)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerSPI:
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match rule (DP_MATCH_INCOMING_DNIS)
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Match incoming called number; called
(50267)
*Mar  1 00:56:40.711: dpMatchCore:
*Mar  1 00:56:40.711: dpMatchCore: dialstring(50267); expanded string(50267); calling()
*Mar  1 00:56:40.711: FillTarget: pDest(50267T) pPatn(50267)
*Mar  1 00:56:40.711: MatchNextPeer: peer 1000 matched
*Mar  1 00:56:40.711: dpAssociateIncomingPeerCore: Result=0 after DP_MATCH_INCOMING_DNIS;
peers (0x62F2E454)
*Mar  1 00:56:49.203: dpMatchPeersMoreArg:
*Mar  1 00:56:49.203: dpMatchPeersCore:
*Mar  1 00:56:49.203: dpMatchPeersCore: Match rule (DP_MATCH_DEST_URI_AND_TGT_CARRIER)
*Mar  1 00:56:49.203: dpMatchPeersCore: Result=-1 after DP_MATCH_DEST_URI_AND_TGT_CARRIER
*Mar  1 00:56:49.203: dpMatchPeersCore: Match rule (DP_MATCH_DEST_AND_TGT_CARRIER)
*Mar  1 00:56:49.203: dpMatchPeersCore: Result=-1 after DP_MATCH_DEST_AND_TGT_CARRIER
*Mar  1 00:56:49.203: dpMatchPeersCore: Match rule (DP_MATCH_DEST_URI)
*Mar  1 00:56:49.203: dpMatchPeersCore: Match Dest. URI; URI
(sip:xxx_no_under@example.com?Subject=Hello&Priority=Urgent&testID=AL_FEAT_SIP_URL_O_RV_11
)
*Mar  1 00:56:49.203: dpMatchCore:
*Mar  1 00:56:49.203: dpMatchCore: dialstring(); expanded string(); calling()
*Mar  1 00:56:49.203: dpMatchPeersCore: Result=-1 after DP_MATCH_DEST_URI
*Mar  1 00:56:49.203: dpMatchPeersCore: Match rule (DP_MATCH_DEST)
*Mar  1 00:56:49.203: dpMatchPeersCore: Result=-1 after DP_MATCH_DEST
*Mar  1 00:56:49.203: dpMatchPeersCore: Match rule (DP_MATCH_TGT_CARRIER)
*Mar  1 00:56:49.203: dpMatchPeersCore: Result=-1 after DP_MATCH_TGT_CARRIER
*Mar  1 00:56:49.203: //-1//TCL2:HN0033E40C:/tcl_PutsCmd:
 STATUS=ls_004
*Mar  1 00:56:49.203:
```

A message showing Result=-1 indicates that a dial peer was not matched. Because this output does not display another dial peer match statement, you know that the gateway failed to find a match.

Step 2 **debug voice uri**

Use the **debug voice uri** command to verify whether there is a voice class that matches the URI in the Tcl script or VoiceXML document.

In the following example, the gateway failed to match the URI in the script to the only configured voice class, 805:

```
Router# debug voice uri

Voice URI debugging is enabled
Router#
*Mar  1 01:09:03.319: vuri_match_class: tag (805)
*Mar  1 01:09:03.319: vuri_match_class_sip: Match with phone context
*Mar  1 01:09:03.319: vuri_match_class_sip: input ()
*Mar  1 01:09:03.319: vuri_match_class_sip: Match with host
*Mar  1 01:09:03.319: vuri_match_class_sip: input (example.com)
*Mar  1 01:09:03.319: vuri_match_class_sip: Match with user-id
*Mar  1 01:09:03.319: vuri_match_class_sip: input (xxx_no_under)
*Mar  1 01:09:03.319: vuri_match_class_sip: Match failed
```

Step 3 **debug voice ccapi error**

Use the **debug voice ccapi error** command to display any errors that might occur during the call transfer:

```
Router# debug voice ccapi error

*Mar  1 01:39:59.319: //32/7FAB684B8022/CCAPI/cc_get_associated_stream: Matched the
                                            Stream CallID 0x21::
*Mar  1 01:40:04.279: //32/7FAB684B8022/CCAPI/cc_api_call_disconnect_done:
cause=28,retry=0,vcCauseCode=0
```

Any output from this command means that the call transfer did not complete, although the reason for the error might not be obvious from the messages.

Configuration Examples for SIP and TEL URL Support

This section provides the examples that show the configuration for an originating and terminating gateway, each configured to handle both SIP and TEL URIs.

- Example: Outbound Dial-Peer Originating Gateway, page 21
- Example: Outbound Dial-Peer Terminating Gateway, page 26

Note For information about reading and passing SIP headers, see the *Cisco IOS SIP Configuration Guide*.

Example: Outbound Dial-Peer Originating Gateway

This following sections provide examples of SIP and TEL URIs with Header passing using the respective protocols in the originating gateway.

- Example: URIs with Header Passing Using SIP Protocol and H.323 Protocol, page 27

Example: URIs with Header Passing Using SIP Protocol and H.323 Protocol

In the following example, when a SIP call comes into the originating gateway, the application sip_headers_tcl asks the caller to enter an account number. After the account number is collected, it is assigned to a header named AccountInfo. The AccountInfo is passed to the leg setup along with other standard and user-defined headers in the destination URL.

The outgoing call matches on dial peer 766, which is configured with the **destination uri** command to match on voice class 766. Voice class 766 is configured with the **voice class uri** command to match destination SIP URI sip:elmo@sip.example.com. The gateway places a call to sip:elmo@sip.example.com.

When the call setup request is received by the gateway, the headers that are passed from the application are included in the SIP INVITE message.

In the following example, when an H.323 call comes into the originating gateway, the application tel_headers_vxml asks the caller to enter an account number. After the account number is collected, it is assigned to a header named AccountInfo. The AccountInfo is passed to the leg setup along with other user-defined headers in the destination URL.

The outgoing call matches on dial peer 767, which is configured with the **destination uri** command to match on voice class 767. Voice class 767 is configured with the **voice class uri** command to match telephone number 555-0100 and a phone context of 408. The voice class can also match against a URL pattern. This example uses the phone number and phone context as a matching criteria. The gateway places a call to a TEL URL with the header of tel:555-0100;phone-context=408;tsp=example.com;Subject=HelloTelVXML;To=oscar@example.com; From=nobody; Priority=urgent'+';AccountInfo='+acctInfo.

When the call setup request is received by the gateway, the destination URL with headers and parameters is passed in the setup message as part of the destination address.

```
!
version x.x
service timestamps debug datetime msec localtime
service timestamps log datetime msec localtime
no service password-encryption
service internal
!
hostname 10.7.102.32
!
no logging buffered
enable password lab
!
username 1111
username 2222 password 0 2222
!
!
resource-pool disable
!
aaa new-model
!
!
aaa authentication login h323 local group radius
aaa authorization exec h323 group radius
aaa accounting connection h323 start-stop group radius
aaa session-id common
ip subnet-zero
ip ftp username dump
ip ftp password dump123
ip host px1-sun 10.0.1.0
ip host rtsp-ws 10.1.1.0
ip host dev 10.3.0.1
```

```
ip host px1-exm.example.com 10.0.1.1
ip host example.com 10.1.1.2
!
!
isdn switch-type primary-5ess
isdn voice-call-failure 0
!
!
voice service voip
 sip
!
!
voice class uri  766 sip
 pattern elmo@sip.example.com*
!
voice class uri  767 tel
 phone number 767....
 phone context 408
!
!
!
!
!
no voice hpi capture buffer
no voice hpi capture destination
!
!
ivr prompt memory 16384
ivr record memory system 256000
ivr record memory session 256000
ivr asr-server rtsp://nuance-asr/recognizer
ivr tts-server rtsp://nuance-asr/synthesizer
ivr record memory system 256000
ivr record memory session 256000
ivr asr-server rtsp://nuance-asr/recognizer
ivr tts-server rtsp://nuance-asr/synthesizer
rtsp client session history duration 1
rtsp client session history records 1
http client cache memory pool 0
http client cache memory file 10000
http client cache refresh 30
http client connection timeout 10
http client response timeout 10
fax interface-type modem
mta receive maximum-recipients 0
call-history-mib retain-timer 0
call-history-mib max-size 0
!
controller T1 0
 framing esf
 clock source line primary
 linecode b8zs
 cablelength short 133
 pri-group timeslots 1-24
!
controller T1 1
 framing esf
 clock source line secondary 1
 linecode b8zs
 cablelength short 133
 pri-group timeslots 1-24
!
controller T1 2
 framing esf
```

```
 clock source line secondary 2
 linecode b8zs
 cablelength short 133
 pri-group timeslots 1-24
!
controller T1 3
 framing esf
 clock source line secondary 3
 linecode b8zs
 cablelength short 133
 pri-group timeslots 1-24
!
gw-accounting h323
gw-accounting h323 vsa
gw-accounting voip
!
!
interface Ethernet0
 ip address 209.165.201.1 255.255.255.224
 ip helper-address 10.1.201.30
 no ip route-cache
 no ip mroute-cache
 no cdp enable
!
interface Serial0
 no ip address
 shutdown
 clockrate 2015232
 no fair-queue
 no cdp enable
!
interface Serial1
 no ip address
 shutdown
 clockrate 2015232
 no fair-queue
 no cdp enable
!
interface Serial2
 no ip address
 shutdown
 clockrate 2015232
 no fair-queue
 no cdp enable
!
interface Serial3
 no ip address
 shutdown
 clockrate 2015232
 no fair-queue
 no cdp enable
!
interface Serial0:23
 no ip address
 no logging event link-status
 dialer-group 1
 isdn switch-type primary-5ess
 isdn incoming-voice modem
 isdn disconnect-cause 1
 fair-queue 64 256 0
 no cdp enable
!
interface Serial1:23
 no ip address
```

```
 no logging event link-status
 isdn switch-type primary-5ess
 no cdp enable
!
interface Serial2:23
 no ip address
 no logging event link-status
 isdn switch-type primary-5ess
 no cdp enable
!
interface Serial3:23
 no ip address
 no logging event link-status
 isdn switch-type primary-5ess
 no cdp enable
!
interface FastEthernet0
 no ip address
 shutdown
 duplex auto
 speed auto
 no cdp enable
!
ip default-gateway 10.255.255.255
ip classless
ip route 209.165.200.20 255.255.255.224
no ip http server
ip pim bidir-enable
!
!
no cdp run
!
!
radius-server retransmit 3
radius-server authorization permit missing Service-Type
call rsvp-sync
!
application
 service sip_headers_tcl tftp://dev/demo/TCL/scripts/sip_headers.tcl
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dev/demo/AUDIO/en/
 !
 service sip_headers_vxml tftp://dev/demo/VXML/scripts/sip_headers.vxml
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dev/demo/AUDIO/en/
 !
 service tel_headers_tcl tftp://dev/demo/TCL/scripts/tel_headers.tcl
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dev/demo/AUDIO/en/
 !
 service tel_headers_vxml tftp://dev/demo/VXML/scripts/tel_headers.vxml
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dev/demo/AUDIO/en/
 !
!
voice-port 0:D
!
voice-port 1:D
!
voice-port 2:D
```

```
!
voice-port 3:D
!
mgcp modem passthrough voip mode ca
no mgcp timer receive-rtcp
!
mgcp profile default
!
dial-peer cor custom
!
!
dial-peer voice 1 pots
 service sip_headers_tcl
 incoming called-number 52948
 port 0:D
!
dial-peer voice 2 pots
 application tel_headers_vxml
 incoming called-number 52950
 port 0:D
!
dial-peer voice 767 voip
 session target ipv4:10.0.0.1
 destination uri 767
 codec g711ulaw
!
dial-peer voice 766 voip
 session protocol sipv2
 session target ipv4:10.0.0.1
 destination uri 766
 codec g711ulaw
!
dial-peer voice 7671234 voip
 service get_headers_vxml out-bound
 destination-pattern .......
 session protocol sipv2
 session target ipv4:10.0.0.1
 codec g711ulaw
!
sip-ua
 sip-server ipv4:10.0.1.1
!
!
line con 0
 exec-timeout 0 0
line aux 0
line vty 0 4
!
exception core-file special3
exception dump 10.7.100.1
end
```

Example: Outbound Dial-Peer Terminating Gateway

This section provides examples of SIP and TEL URIs with Header passing using the respective protocols in the terminating gateway.

- Example: URIs with Header Passing Using SIP Protocol and H.323 Protocol, page 27

Example: URIs with Header Passing Using SIP Protocol and H.323 Protocol

In the following example, when the call arrives at the terminating gateway and dial peer 766 is matched, the gateway stores all headers received in the incoming INVITE message so these can be accessed by the application.

The inbound dial peer can be configured to match the request-URI, or the "To" or "From" header in the incoming INVITE message. This example uses the request-URI for matching. The incoming call matches on dial peer 766, which is configured with the **incoming uri request** command to match on voice class 766. Voice class 766 is configured to match the incoming SIP request-URI sip:elmo@sip.example.com.

When the call is handed to the application configured in the inbound dial peer, get_headers_tcl, this Tcl application can read any header that is part of the incoming INVITE message.

In the following example, when the call arrives at the terminating gateway and dial peer 767 is matched, the gateway stores the incoming URI so it can be accessed by the application.

The inbound dial peer can be configured to match the entire TEL URL pattern, the E.164 number portion, or the phone context of the TEL URL. This example uses the phone number and phone context for matching. The incoming call matches on dial peer 767, which is configured with the **incoming uri called** command to match on voice class 767. Voice class 767 is configured to match the incoming called URL with the header of tel:555-0100;phone-context=408;tsp=example.com;Subject=HelloTelVXML;To=oscar@example.com;From=nobody;Priority=urgent'+';AccountInfo='+acctInfo.

When the call is handed to the application configured in the inbound dial peer, get_headers_vxml, this VoiceXML application can read any header which is part of the incoming called URI received in the setup indication.

```
!
version x.x
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service internal
!
hostname as5300-09
!
enable secret 5 $1$KRsb$cFAQFOylLr9j5Fof.eLgx1
enable password lab
!
!
!
resource-pool disable
clock timezone PDT -8
clock calendar-valid
!
ip subnet-zero
no ip domain lookup
ip domain name fieldlabs.example.com
ip host dcl1server 10.7.108.2
ip host px1-sun 10.14.99.1
ip host dirt 192.168.254.254
ip host jurai 192.168.254.254
ip host dclserver 10.7.108.2
ip host dcl2server 10.7.112.2
ip host ts 10.7.100.1
!
!
isdn switch-type primary-5ess
isdn voice-call-failure 0
```

```
!
!
voice service voip
 sip
  header-passing
!
!
voice class uri  766 sip
 pattern elmo@sip.example.com*
!
voice class uri  767 tel
 phone number 767....
 phone context 408
!
!
!
!
!
no voice hpi capture buffer
no voice hpi capture destination
!
!
ivr record memory system 100000
ivr record memory session 100000
ivr record memory system 100000
ivr record memory session 100000
fax interface-type modem
mta receive maximum-recipients 0
!
controller T1 0
 framing esf
 clock source line primary
 linecode b8zs
 pri-group timeslots 1-24
!
controller T1 1
 framing sf
 clock source line secondary 1
 linecode ami
!
controller T1 2
 framing sf
 linecode ami
!
controller T1 3
 framing sf
 linecode ami
!
!
!
interface Ethernet0
 ip address 209.165.200.225 255.255.255.224
 ip helper-address 192.168.254.254
 no ip route-cache
 no ip mroute-cache
 no cdp enable
!
interface Serial0:23
 no ip address
 dialer-group 1
 isdn switch-type primary-5ess
 isdn incoming-voice modem
 fair-queue 64 256 0
 no cdp enable
```

```
!
interface FastEthernet0
 ip address 209.165.201.28 255.255.255.224
 no ip route-cache
 no ip mroute-cache
 duplex half
 speed 10
 no cdp enable
!
ip default-gateway 10.7.0.1
ip classless
ip route 209.168.201.1 255.255.255.224 10.165.196.1
ip route 209.168.201.5 255.255.255.224 10.165.0.1
no ip http server
ip pim bidir-enable
!
!
no cdp run
!
!
call rsvp-sync
!
application
 service get_headers_tcl tftp://dev/demo/TCL/scripts/get_headers.tcl
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dirt/cchiu/AUDIO/en/
 !
 service voice get_headers_vxml tftp://dev/demo/VXML/scripts/get_headers.vxml
  paramspace english language en
  paramspace english index 1
  paramspace english location tftp://dirt/cchiu/AUDIO/en/
 !
!
voice-port 0:D
!
mgcp ip qos dscp cs5 media
mgcp ip qos dscp cs3 signaling
!
mgcp profile default
!
dial-peer cor custom
!
!
!
dial-peer voice 1 pots
 service test
 incoming called-number 52950
 port 0:D
!
dial-peer voice 767 voip
 service get_headers_vxml
 session target ipv4:10.0.0.1
 incoming uri called 767
 codec g711ulaw
!
dial-peer voice 766 voip
 service get_headers_tcl
 session protocol sipv2
 session target ipv4:10.0.0.0
 incoming uri request 766
 codec g711ulaw
!
dial-peer voice 2 pots
```

```
 destination-pattern 767....
 port 0:D
 prefix 9767
!
sip-ua
!
!
line con 0
 exec-timeout 0 0
 logging synchronous
line aux 0
line vty 0 4
 password lab
 login
!
scheduler interval 1000
end
```

Where to Go Next

- To configure properties for audio files, see "Configuring Audio File Properties for Tcl and VoiceXML Applications".
- To configure voice recording using a VoiceXML application, see "Configuring VoiceXML Voice Store and Forward".
- To configure properties for speech recognition or speech synthesis, see "Configuring ASR and TTS Properties".
- To configure a VoiceXML fax detection application, see "Configuring Fax Detection for VoiceXML".
- To configure telephony call-redirect features for voice applications, see "Configuring Telephony Call-Redirect Features".
- To configure session interaction for a Tcl IVR 2.0 application, see "Configuring Tcl IVR 2.0 Session Interaction".
- To monitor and troubleshoot voice applications, see "Monitoring and Troubleshooting Voice Applications".

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS Voice commands	Cisco IOS Voice Command Reference
Overview of Cisco Tcl IVR and VoiceXML Applications	*Cisco IOS Tcl IVR and VoiceXML Application Guide*

MIBs

MIB	MIBs Link
• CISCO-VOICE-DIAL-CONTROL-MIB • CISCO-VOICE-DNIS-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring SIP and TEL URL Support

Table 11-1 lists the features in this module and provides links to specific configuration information.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Note Table 11-1 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.

Table 11-1 Feature Information for Configuring SIP and TEL URL Support

Feature Name	Releases	Feature Information
SIP and TEL URL support	12.3(4)T 12.3(14)T	SIP and TEL URL support enables Cisco gateways to direct incoming calls to a voice application based on the URL and Tcl IVR 2.0 and VoiceXML applications to place outbound calls to a Session Initiation Protocol (SIP) or telephone (TEL) URL. It expands call-control capabilities by allowing voice applications to use URL destinations and dialing plans to be implemented using SIP or TEL URLs rather than telephone numbers. The following sections provide information about this feature: • Configuring an Inbound Dial Peer to Match on a URI, page 7 • Configuring an Outbound Dial Peer for URI Destinations, page 15
Support Inbound Dial-peer Match Based on Remote IP Address on SIP Trunks	15.1(2)T	The Support Inbound Dial-peer Match Based on Remote IP Address on SIP Trunks feature supports the expansion of inbound dial-peer matching logic to include matching based on the source IP address of inbound signaling on a SIP trunk. This feature enables enforcement of specific call-treatment, security, and routing policies on each SIP trunk. The following section provide information about this feature: • Configuring an Inbound Dial-Peer to Match the URI on SIP Calls, page 9 The following commands were introduced or modified: **dial-peer voice, voice-class uri**.