

Unified Border Element SIP TLS Configuration Example

Document ID: 100446

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Configure

- Network Diagram
- RFC Support for TLS in CUBE
- Configuration Steps
- TLS Implementation Notes
- Sample Configurations

Verify

Troubleshoot

NetPro Discussion Forums – Featured Conversations

Related Information

Introduction

The Cisco Unified Border Element (CUBE) supports Session Initiation Protocol (SIP) to SIP calls with Transport Layer Security (TLS). TLS provides privacy and data integrity of SIP signaling messages between two applications that communicate. TLS is layered on top of a reliable transport protocol such as TCP.

TLS on CUBE can be configured on a per-leg basis in order to allow a TLS to non-TLS SIP call. Similarly, CUBE uses IPsec in order to secure signaling and support calls from H.323 to SIP with the H.323 leg, while the SIP leg uses TLS.

Prerequisites

Requirements

Ensure that you meet these requirements before you attempt this configuration:

- Basic knowledge of how to configure and use Cisco IOS voice (such as dial-peers)
- Basic knowledge of how to configure and use Cisco Unified Border Element (CUBE)
- Familiarity with basic security concepts such as encryption, certification, certificate authorities, PKI (keys), and authentication

Components Used

The information in this document is based on these software and hardware versions:

- Cisco Unified Border Element release on an ISR that uses Cisco IOS release 12.4T
- Cisco IOS router configured as a certificate authority (CA)

The information in this document was created from the devices in a specific lab environment. All of the

devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to the Cisco Technical Tips Conventions for more information on document conventions.

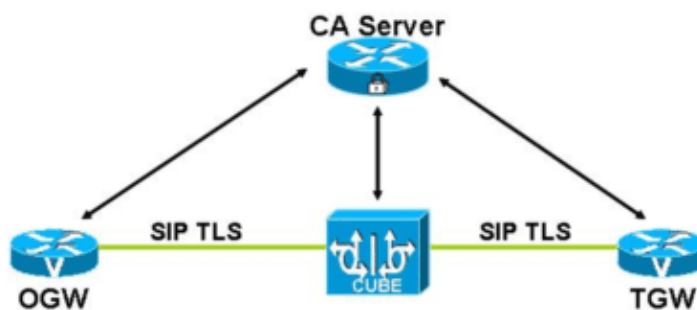
Configure

In this section, you are presented with the information to configure the features described in this document.

Note: Use the Command Lookup Tool (registered customers only) to obtain more information on the commands used in this section.

Network Diagram

This figure shows an example of CUBE with SIP TLS connections.



- The originating gateway (OGW), terminating gateway (TGW), and CUBE devices authenticate and enroll with a CA server. The certificates are signed by the CA server.
- When a call is made, a TLS handshake is initiated between the devices (for example, OGW and CUBE) and the IOS PKI infrastructure is used to exchange certificates signed by a common trusted CA during the handshake.
- During the TLS handshake, a dynamically generated symmetric key and cipher algorithms are negotiated between the devices.
- After the TLS handshake is successful, the devices establish a SIP session between them. Keys exchanged during the TLS handshake process are used to encrypt or decrypt all SIP signaling messages. The URI scheme `sips:` is used for SIP TLS messages.

RFC Support for TLS in CUBE

The cipher suites required for TLS as per SIP RFC 3261 include:

- `TLS_RSA_WITH_AES_128_CBC_SHA` (mandatory)
- `TLS_RSA_WITH_3DES_EDE_CBC_SHA` (optional) Required for network servers (such as proxies and redirect servers for backward compatibility)

Only the `TLS_RSA_WITH_AES_128_CBC_SHA` suite is applicable to CUBE and is supported. Similarly, the TLS implementation in CUBE supports only the mandatory cipher suites of RFC 2246.

The SIP protocol uses a peer-to-peer model. Therefore, CUBE can be either the server or client of a TLS connection and implements both sides. CUBE always performs mutual authentication when it is the server

side.

Configuration Steps

Configure the CA Server

You can use this command in global configuration mode in order to configure a Cisco IOS router loaded with a crypto image:

```
router(config)#crypto pki server <ca-server-name>
router(cs-server)#no shutdown
```

Notes:

- Use the **ip http server** command in global configuration mode in order to ensure that an HTTP server runs on the router configured as the CA server. This is required since the client trustpoints (CUBE/OGW/TGW) use HTTP in order to receive the certificates from the CA server.
- The clocks in the CA server and the client trustpoints (CUBE/OGW/TGW) must be synchronized. Otherwise, there might be issues with the validity of the certificates issued by the CA server. You can use the **show clock** and **clock set** commands in order to synchronize the clocks on Cisco IOS routers. Alternatively, you can deploy an NTP server in order to synchronize the clocks.

Basic Configuration for CUBE

Use these commands in order to enable CUBE s IP-to-IP gateway functionality. This allows termination of the incoming VoIP call and reorigination of the call with an outbound VoIP dial-peer.

```
voice service voip
  allow-connections h323 to sip
  allow-connections sip to h323
  allow-connections sip to sip
  allow-connections h323 to h323
```

TLS Configuration

Complete these steps in order to configure TLS on CUBE (and other devices like OGW and TGW):

1. Generate an RSA Keypair

Use this command in global configuration mode in order to generate an RSA keypair:

```
router(config)#crypto key generate rsa general-keys label <label> modulus 1024
```

2. Create a PKI Trustpoint (CUBE)

Use this command in global configuration mode in order to create a PKI trustpoint (CUBE):

```
router(config)#crypto pki trustpoint <ca-server-name>
router(ca-trustpoint)#enrollment url <http://ca-server-ip>
router(ca-trustpoint)#rsa keypair <rsa keypair label>
```

3. Authenticate a PKI Trustpoint (CUBE) with the CA Server

Use this command in global configuration mode in order to authenticate a PKI trustpoint (CUBE) with the CA server:

```
router(config)#crypto pki authenticate <ca-server-name>
```

This step triggers the CA server to send its certificate to the trustpoint (CUBE), which should be accepted.

4. Enroll a PKI trustpoint (CUBE) with CA Server

Use this command in global configuration mode:

```
router(config)#crypto pki enroll <ca-server-name>
```

For this step, you must enter a challenge password. The CA server issues two certificates to the trustpoint (CUBE): one to certify the CA server and the other to certify the trustpoint (CUBE). You can check the certificates with the **show run** command.

Full documentation of IOS PKI configuration can be found at:

http://www.cisco.com/en/US/docs/ios/12_4t/secure/configuration/guide/t_pki_en.html

5. Configure TLS as Session Transport

The session transport can be configured to TLS with the **session transport tcp tls** command at either the global level under `voice service voip` or in the appropriate VOIP dial peers.

If the session transport is configured for a VOIP dial peer (incoming or outgoing or both), then TLS transport is used only for the configured leg. TLS transport is supported on a leg-to-leg basis.

6. Configure a Default Trustpoint for the SIP UA

Use this command in `sip-ua` mode in order to configure a default trustpoint for the SIP UA:

```
router(config-sip-ua)#[no] crypto signaling [(remote-addr subnet mask) | default]
trustpoint <label> [strict-cipher]
```

The trustpoint label refers to the CUBE's certificate that is generated with the Cisco IOS PKI commands as part of the enrollment process. *strict-cipher* means that the SIP TLS process uses only those cipher suites that are mandated by the SIP RFC.

Currently, RFC 3261 specifies the TLS_RSA_WITH_AES_128_CBC_SHA and TLS_RSA_WITH_3DES_EDE_CBC_SHA suites. When you use the *strict-cipher* command argument avoids changes to the configuration if SIP should mandate newer ciphers.

The SSL layer in Cisco IOS does not support TLS_RSA_WITH_3DES_EDE_CBC_SHA. Therefore, CUBE actively uses only the TLS_RSA_WITH_AES_128_CBC_SHA suite in strict mode. When *strict-cipher* is not specified, the SIP TLS process uses a larger set of ciphers depending on the support at the SSL layer.

Example 1

The command below configures CUBE to use its trustpoint label **mylabel** when it establishes or accepts a TLS connection with a remote device within the 1.2.3.0 subnet. The cipher suite in this case is the overall set that is supported by the SSL layer on CUBE.

```
crypto signaling remote-addr 1.2.3.0 255.255.255.0 trustpoint mylabel
```

Example 2

The command below configures CUBE to use its trustpoint label **chef** when it establishes or accepts a TLS connection with any remote device unless an individual subnet label configuration is matched.

```
crypto signaling default trustpoint chef
```

Example 3

The command below configures CUBE to use its trustpoint label **mylabel** when it establishes or accepts a TLS connection with a remote device within the 1.2.3.0 subnet. The cipher suite used during the TLS handshake is limited to the TLS_RSA_WITH_AES_128_CBC_SHA suite.

```
crypto signaling remote-addr 1.2.3.0 255.255.255.0 trustpoint mylabel
strict-cipher
```

7. Enabling TLS Listener Port

Use this command in `sip-ua` mode in order to enable the TLS port on TCP 5061 to listen:

```
transport tcp tls
```

8. Configuring SIPS URL Scheme

The `sips:` URL scheme can be configured either under VOIP dial-peer level or at the global level. This command is used for configuring `sips:` in a VOIP dial peer:

```
voice-class sip url sips
```

In order to configure the `sips:` URL scheme under the global level, use this command in `voice service voip sip` mode:

```
voice service voip
sip
url sips
```

The use of the SIPS URL requires all hops in the signaling path to use TLS and SIPS. This becomes important for SRTP as keys are in the SDP and for a secure connection that information should not be sent in clear text. If a proxy receives an INVITE with SIPS (for example, INVITE `sips:123@proxy SIP /2.0`) the proxy must use SIPS for the next hop. When TLS is used with a plain SIP url, there is no guarantee that all hops will use TLS, potentially compromising the end-to-end security of the call.

If a `sips` URL is configured, the transport will automatically be TLS.

TLS Implementation Notes

- Current CUBE operation requires the use of TLS as the transport when secure media is configured (SRTP). A future enhancement may lift this requirement.
- When SRTP is configured to secure the media connection, either TLS or IPsec *must* also be configured to secure the SIP signaling messages. The keys used for SRTP encryption are exchanged via the signaling messages not securing the signaling channel results in the SRTP keys exchanged in clear-text and this negates the security of SRTP for the media connection.
- Current CUBE operation requires the use of the `sips:` URI scheme for a TLS call. A future enhancement may lift this requirement.
- Current CUBE operation has been verified with a single CA server only.

Sample Configurations

CUBE

```
ipipgw
```

```

ipipgw#show run
Building configuration...

Current configuration : 5096 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname ipipgw
!
boot-start-marker
boot system flash c3845-adventerprisek9_ivs-mz.124-3.9.PI3a
boot-end-marker
!
logging buffered 10000000 debugging
no logging console
!
no aaa new-model
!
resource policy
!
ip subnet-zero
ip cef
!
no ip domain lookup
!
voice-card 0
no dspfarm
!
voice service voip
  allow-connections sip to sip
  sip
  url sips
!
crypto pki trustpoint ca-server
  enrollment url http://9.13.46.14:80
  serial-number
  revocation-check crl
  rsakeypair kkp
!
crypto pki certificate chain ca-server
certificate 04
  3082020D 30820176 A0030201 02020104 300D0609 2A864886 F70D0101 04050030
  14311230 10060355 04031309 63612D73 65727665 72301E17 0D303530 39323231
  37333435 315A170D 30363039 32323137 33343531 5A303431 32300F06 03550405
  13084337 33323231 3333301F 06092A86 4886F70D 01090216 1270696E 612D3338
  34352D69 70697067 77312E30 819F300D 06092A86 4886F70D 01010105 0003818D
  00308189 02818100 BBCC2977 637E8E42 17EB7C26 FB2BA0A3 6E1ECECB E01A64F8
  8F18200F 9837E4FA 7D908B3C 1297A4DE A403D315 C7BB96C6 50D95291 0433FA7B
  CB8FFFFD 8FC1C211 CCC7BCA9 140FF942 C3ACF4BC 3EDCE2DC 28FCEA87 AA83629F
  D217F833 A727940A 0BBB8624 3EA9D1EC 1F69228F E1DFC113 243246B7 BF57696C
  2278F5C3 674EE0E1 02030100 01A34F30 4D300B06 03551D0F 04040302 05A0301F
  0603551D 23041830 16801486 7414D5D6 9B8299C1 787211AB 1B265B06 D2B62D30
  1D060355 1D0E0416 0414FED1 97051946 D2F870D8 0DE819C3 AA1F3830 AD35300D
  06092A86 4886F70D 01010405 00038181 00845AB8 F6589AED 17D0BB10 2AEA48AA
  9299C130 4B358EA1 96632C84 0387D2DE 4774C776 6A14F25B 5D062E12 45EF730D
  27D45795 62C17F55 A0428259 B13669BC 022201C7 EB6B7ACF 4C7143FA 8A038301
  CEA17A0B D0662887 26BA8F0E C44410BB 4F982706 11F0D248 77D8A0E5 4417F0F4
  3F993CE3 F62F6BDE BA2DD6BB B843391D 6D
  quit
certificate ca 01
  30820201 3082016A A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  14311230 10060355 04031309 63612D73 65727665 72301E17 0D303530 39323031
  37303335 375A170D 30383039 31393137 30333537 5A301431 12301006 03550403

```

```

13096361 2D736572 76657230 819F300D 06092A86 4886F70D 01010105 0003818D
00308189 02818100 BE7F0760 70D3B5C3 923D59FB C10AED17 71C6F477 7580851A
282FFAEB 43B918A1 2D867C1B 63963B36 F779FE18 D5DFFDB6 5E436276 459FC5EA
A729C386 CDDD922B 2A0439AE 68A5F4C4 3B05F168 5BB93EF2 DF737F11 0BA3F5EB
3E62F423 CB5364D3 C39CCA09 8ADECBFF 4C0515A6 0750A283 ABA39ED2 F5866B98
D3361C1A B88AA62B 02030100 01A36330 61300F06 03551D13 0101FF04 05300301
01FF300E 0603551D 0F0101FF 04040302 0186301F 0603551D 23041830 16801486
7414D5D6 9B8299C1 787211AB 1B265B06 D2B62D30 1D060355 1D0E0416 04148674
14D5D69B 8299C178 7211AB1B 265B06D2 B62D300D 06092A86 4886F70D 01010405
00038181 00AC7DAF 0DF589CA C6175EC0 8F976C5F E08C3C91 85282FFA 94EE6F30
02EEE5B9 E60198ED 643151E0 CCE192FA A352BA3D 8BC5C006 EF89CFCF 59DA9B12
D729102C 3D6ADC3C 09931B96 3F1FB48C C0A85FDB 4F9A7C16 028673C3 91786D57
9D7C1016 62F9D4E9 78FED276 0C404815 B1FE3A11 4D215FCF 573536B4 477ECDB7
7060E221 31
quit
!
interface GigabitEthernet0/0
ip address 9.13.46.12 255.255.255.0
duplex auto
speed auto
media-type rj45
negotiation auto
!
interface GigabitEthernet0/1
no ip address
shutdown
duplex auto
speed auto
media-type rj45
negotiation auto
!
ip classless
ip route 0.0.0.0 0.0.0.0 9.13.46.1
!
ip http server
no ip http secure-server
!
no cdp log mismatch duplex
!
control-plane
!
call treatment on
!
dial-peer voice 1 voip
session protocol sipv2
incoming called-number 9000
codec g711ulaw
!
dial-peer voice 2 voip
destination-pattern 9000
session protocol sipv2
session target ipv4:9.13.46.200
codec g711ulaw
!
dial-peer voice 3 voip
session protocol sipv2
incoming called-number 4000
codec g711ulaw
!
dial-peer voice 4 voip
destination-pattern 4000
session protocol sipv2
session target ipv4:9.13.32.75
codec g711ulaw
!
dial-peer voice 5 voip

```

```

destination-pattern 5000
session protocol sipv2
session target ipv4:9.13.0.10
codec g711alaw
!
dial-peer voice 7 voip
destination-pattern 9999
session protocol sipv2
session target ipv4:9.13.2.36
codec g711alaw
!
dial-peer voice 12 pots
destination-pattern 8400
!
dial-peer voice 10 voip
destination-pattern 50000
session protocol sipv2
session target ipv4:9.13.2.150
codec g711alaw
!
dial-peer voice 11 voip
session protocol sipv2
session transport tcp tls
incoming called-number 8004
codec g711ulaw
!
dial-peer voice 13 voip
destination-pattern 8004
session protocol sipv2
session target ipv4:9.13.2.70
codec g711ulaw
!
dial-peer voice 20 voip
destination-pattern 4444
session target ipv4:9.13.46.111
codec g711ulaw
!
dial-peer voice 21 voip
incoming called-number 4444
codec g711ulaw
!
sip-ua
retry invite 10
crypto signaling default trustpoint ca-server
!
gatekeeper
shutdown
!
line con 0
stopbits 1
line aux 0
stopbits 1
line vty 0 4
login
!
scheduler allocate 20000 1000
!
end

```

IOS CA-Server

ca-server

```

ca-server#show run
Building configuration...

```

```
Current configuration : 2688 bytes
!
! Last configuration change at 17:11:41 UTC Tue Sep 20 2005
! NVRAM config last updated at 16:57:43 UTC Tue Sep 20 2005
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname ca-server
!
boot-start-marker
boot system flash c2800nm-adventerprisek9_ivs-mz.124-3.9.PI3a
boot-end-marker
!
no aaa new-model
!
resource policy
!
ip subnet-zero
!
ip cef
!
voice-card 0
no dspfarm
!
crypto pki server ca-server
grant auto
!
crypto pki trustpoint ca-server
revocation-check crl
rsa-keypair ca-server
!
crypto pki certificate chain ca-server
certificate ca 01
 30820201 3082016A A0030201 02020101 300D0609 2A864886 F70D0101 04050030
14311230 10060355 04031309 63612D73 65727665 72301E17 0D303530 39323031
37303335 375A170D 30383039 31393137 30333537 5A301431 12301006 03550403
13096361 2D736572 76657230 819F300D 06092A86 4886F70D 01010105 0003818D
00308189 02818100 BE7F0760 70D3B5C3 923D59FB C10AED17 71C6F477 7580851A
282FFAEB 43B918A1 2D867C1B 63963B36 F779FE18 D5DFFDB6 5E436276 459FC5EA
A729C386 CDDD922B 2A0439AE 68A5F4C4 3B05F168 5BB93EF2 DF737F11 0BA3F5EB
3E62F423 CB5364D3 C39CCA09 8ADECBFF 4C0515A6 0750A283 ABA39ED2 F5866B98
D3361C1A B88AA62B 02030100 01A36330 61300F06 03551D13 0101FF04 05300301
01FF300E 0603551D 0F0101FF 04040302 0186301F 0603551D 23041830 16801486
7414D5D6 9B8299C1 787211AB 1B265B06 D2B62D30 1D060355 1D0E0416 04148674
14D5D69B 8299C178 7211AB1B 265B06D2 B62D300D 06092A86 4886F70D 01010405
00038181 00AC7DAF 0DF589CA C6175EC0 8F976C5F E08C3C91 85282FFA 94EE6F30
02EEE5B9 E60198ED 643151E0 CCE192FA A352BA3D 8BC5C006 EF89CFCF 59DA9B12
D729102C 3D6ADC3C 09931B96 3F1FB48C C0A85FDB 4F9A7C16 028673C3 91786D57
9D7C1016 62F9D4E9 78FED276 0C404815 B1FE3A11 4D215FCF 573536B4 477ECDB7
7060E221 31
quit
!
interface FastEthernet0/0
ip address 9.13.46.14 255.255.255.0
duplex auto
speed auto
!
interface FastEthernet0/1
no ip address
shutdown
duplex auto
speed auto
```

```

!
ip classless
ip route 0.0.0.0 0.0.0.0 9.13.46.1
!
ip http server
no ip http secure-server
!
no cdp log mismatch duplex
!
control-plane
!
gatekeeper
 shutdown
!
line con 0
line aux 0
line vty 0 4
 login
!
scheduler allocate 20000 1000
!
end

```

Verify

After a call is made, this **show** command can be used in order to verify whether the transport used for the call is TLS:

```

router#show sip-ua connections tcp tls ?
      brief Show summary of connections
      detail Show detail connection information

```

Sample output for this command is shown in these examples:

Example 1: Detail Output

```

=====
router#show sip-ua connections tcp tls detail
Total active connections      : 1
No. of send failures         : 0
No. of remote closures       : 3
No. of conn. failures        : 0
No. of inactive conn. ageouts : 0
Max. tls send msg queue size of 0, recorded for 0.0.0.0:0
TLS client handshake failures : 0
TLS server handshake failures : 0

-----Printing Detailed Connection Report-----
Note:
** Tuples with no matching socket entry
  - Do 'clear sip <tcp[tls]/udp> conn t ipv4:<addr>:<port>'
    to overcome this error condition
++ Tuples with mismatched address/port entry
  - Do 'clear sip <tcp[tls]/udp> conn t ipv4:<addr>:<port> id <connid>'
    to overcome this error condition

Remote-Agent:9.13.46.12, Connections-Count:1
Remote-Port Conn-Id Conn-State WriteQ-Size
=====
          5061          1 Established          0
=====

```

Example 2: Brief Output

```
=====  
router#show sip-ua connections tcp tls brief  
Total active connections      : 2  
No. of send failures         : 0  
No. of remote closures      : 0  
No. of conn. failures       : 0  
No. of inactive conn. ageouts : 0  
Max. tls send msg queue size of 0, recorded for 0.0.0.0:0  
TLS client handshake failures : 0  
TLS server handshake failures : 0  
=====
```

Alternatively, the **debug ccsip messages** command can be used to verify the `Via:` header for TLS is included. This output is a sample INVITE request of a call that uses SIP TLS and the `sips:` URI scheme:

```
INVITE sips:777@172.18.203.181 SIP/2.0  
Via: SIP/2.0/TLS 172.18.201.173:5060;branch=z9hG4bK2C419  
From: <sips:333@172.18.201.173>;tag=581BB98-1663  
To: <sips:5555555@172.18.197.154>  
Date: Wed, 28 Dec 2005 18:31:38 GMT  
Call-ID: EB5B1948-770611DA-804F9736-BFA4AC35@172.18.201.173  
Remote-Party-ID: "Bob" <sips:+14085559999@1.2.3.4>  
Contact: <sips:123@host>  
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, COMET, REFER, SUBSCRIBE, NOTIFY, INFO  
Max-Forwards: 70  
Cseq: 104 INVITE  
Expires: 60  
Timestamp: 730947404  
Content-Length: 298  
Content-Type: application/sdp  
  
v=0  
o=CiscoSystemsSIP-GW-UserAgent 8437 1929 IN IP4 172.18.201.173  
s=SIP Call  
c=IN IP4 1.1.1.1  
t=0 0  
m=audio 18378 RTP/AVP 0 19  
c=IN IP4 1.1.1.1  
a=rtpmap:0 PCMU/8000  
a=rtpmap:19 CN/8000  
a=ptime:20
```

Troubleshoot

Some troubleshooting tips for TLS calls include:

- In order to allow the CA server to issue certificates to the trustpoints, make sure that the IOS router that is configured as a CA server has HTTP enabled (command **ip http server**).
- The clock on CA Server and the trustpoints must be synchronized.
- If the TLS handshake fails between two devices (for example, OGW and CUBE), check the validity of the certificates on the devices. The **debug crypto pki** command can be used to troubleshoot issues during the TLS handshake.
- Sometimes when the devices (for example, OGW and CUBE) are on different subnets, there may an issue of TCP Window size negotiation that causes these errors: *I/O Send Error* and *I/O Read Error*. This issue can be resolved with the **ip tcp path-mtu-discovery** command on both devices. This issue might happen after a successful TLS handshake.
- The `clear sip-ua connections` command in sip-ua mode can be used to clear TLS connections.

```
Router#clear sip-ua tcp [tls] connections <id <conn id> | target
```

<ipv4:ip address:port>

The **tls** option appears after **tcp** since TLS rides on top of TCP. This command works like the existing clear commands for TCP and UDP.

NetPro Discussion Forums – Featured Conversations

Networking Professionals Connection is a forum for networking professionals to share questions, suggestions, and information about networking solutions, products, and technologies. The featured links are some of the most recent conversations available in this technology.

NetPro Discussion Forums – Featured Conversations for Voice
Service Providers: Voice over IP
Voice & Video: Voice over IP
Voice & Video: IP Telephony
Voice & Video: IP Phone Services for End Users
Voice & Video: Unified Communications
Voice & Video: IP Phone Services for Developers
Voice & Video: General

Related Information

- **Voice Technology Support**
- **Voice and Unified Communications Product Support**
- **Recommended Reading: Troubleshooting Cisco IP Telephony**
- **Technical Support & Documentation – Cisco Systems**

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2007 – 2008 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

Updated: Jan 07, 2008

Document ID: 100446