SIP TLS Support on CUBE

The Cisco Unified Border Element (CUBE) supports SIP-to-SIP calls with Transport Layer Security (TLS). TLS provides privacy and data integrity of SIP signaling messages between two applications that communicate. CUBE uses TLS to secure SIP signaling messages. TLS is layered on top of a reliable transport protocol such as TCP. CUBE can be configured at both the global and dial-peer levels for allowing TLS to establish sessions with remote endpoints.

Feature Information for SIP TLS Support on CUBE

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

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

Feature Name	Releases	Feature Information
Server Name Indication (SNI)	Cisco IOS XE Amsterdam 17.3.1a	Support for Server Name Indication (SNI), a TLS extension that allows a TLS client to indicate the name of the server that it is trying connect during the initial TLS handshake process.
Command—**voice class tls-profile** *tag*	Cisco IOS XE Amsterdam 17.3.1a	Support for voice class TLS profile configuration. The *tag* associates voice class TLS profile configuration to the command **crypto signaling**.
Server identity validation through Common Name (CN) and Subject Alternate Name (SAN)	Cisco IOS XE Gibraltar Release 16.11.1a	Support for server identity validation through Common Name (CN) and Subject Alternate Name (SAN) fields in the server certificate.

Feature Name	Releases	Feature Information
Elliptical Curve Ciphers	Cisco IOS XE Gibraltar Release 16.10.1a	Support for configuring Elliptic Curve for a TLS session.
Change in the default SIP TLS Versions support on CUBE	Cisco IOS XE 16.9.1	Behavior of the command **transport tcp tls** is modified. In the earlier releases, TLS version v1.0, v1.1 and v1.2 were enabled by default. From this release onwards, only versions v1.1 and v1.2 are enabled by default. TLS version v1.0 is excluded.
SIP TLS Version 1.2 Support on CUBE	Cisco IOS 15.6(1)T Cisco IOS XE 3.17S	Support is provided for SIP-to-SIP calls with Transport Layer Security (TLS) version 1.2. The following cipher suites are introduced for release Cisco IOS 15.6(1)T: • TLS_DHE_RSA_WITH_AES_128_CBC_SHA1 • TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 • TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 • TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 • TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 The following commands were introduced or modified: **transport tcp tls**, **crypto signaling default trustpoint cube**, and **srtp (voice)**.
SIP TLS Version 1.0 Support on CUBE	Cisco IOS 12.4(6)T	Support is provided for SIP-to-SIP calls with Transport Layer Security (TLS) version 1.0. The following cipher suites are introduced for release Cisco IOS 12.4(6)T : • SSL_RSA_WITH_RC4_128_MD5 • TLS_RSA_WITH_AES_128_CBC_SHA The following commands were introduced or modified: **transport tcp tls** and **crypto signaling default trustpoint cube**.

Restrictions

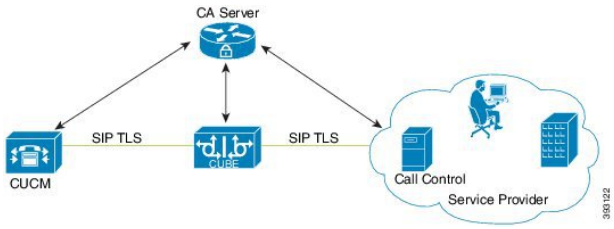
- ECDSA ciphers are not supported on TLS version 1.0.

Information About SIP TLS Support on CUBE

Deployment

The following figure illustrates an example of CUBE with SIP TLS connections.

Figure 1: CUBE with SIP TLS connections



In a typical deployment, CUBE is placed between CUCM and the service provider. These devices are authenticated and enrolled with a Certificate Authority (CA) server that issues certificates. The CA server can be Cisco or a third party entity. When a call is made, a TLS handshake is initiated between CUCM 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 successful TLS handshake, the devices establish a SIP session between the service provider and CUBE. Keys exchanged during the TLS handshake process are used to encrypt or decrypt all SIP signaling messages.

Note The use of PKI on the Cisco IOS software requires that the clock on the devices be synchronized with the network time to ensure proper validation of certificates.

TLS Cipher Suite Category

Before release Cisco IOS15.6(1)T, CUBE supported TLS v1.0 with the following cipher suites:

- SSL_RSA_WITH_RC4_128_MD5
- TLS_RSA_WITH_AES_128_CBC_SHA

CUBE supports only the mandatory cipher suites for TLS implementation. From Cisco IOS15.6(1)T release onwards, CUBE supports TLS v1.2 which is backward compatible. Following are the cipher suites added:

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA1
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

Following cipher suites are added in the Cisco IOS XE Amsterdam 17.3.1a release:

- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA

Use the **srtp pass-thru** command to globally enable the transparent passthrough of all (supported and unsupported) crypto suites. If SRTP pass-thru feature is enabled, media interworking features such as transcoding, transrating, DTMF interworking, and so on, will not be supported. Ensure that you have symmetric configuration on both the incoming and outgoing dial-peers to avoid media-related issues.

How to Configure SIP TLS Support on CUBE

Configuring SIP TLS on CUBE

Note From IOS XE Release16.6.1 onwards, the key-pair information is encrypted in all the router platforms.

When you downgrade the router from IOS XE version 16.6.1 or a later release to a pre-16.6.1 release, ensure that you disable the key encryption before the downgrade. Otherwise, the downgrade discards the encrypted keys. To disable the encryption, use the command **no service private-config-encryption** in global configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa {general-keys | usage-keys}label** *key-label*[**exportable** |][**modulus** *modulus-size*][**storage** *device:*]
4. **crypto key generate ec keysize {256 | 384}**[**label** *label*][*ec key-label*] *! Applicable only for TLS version 1.2.*
5. **crypto pki trustpoint** *name*
6. **rsakeypair** *key-label* [*key-size [encryption-key-size]*]
7. **eckeypair** *keyname*] *! Applicable only for TLS version 1.2.*
8. **serial-number** [**none**]
9. **ip-address** {*ip-address*|**interface**|**none**]
10. **subject-name** [*x.500-name*]
11. **enrollment** [**mode**][**retry period** *minutes*][**retry count** *number*]**url** *url*[**pem**]
12. **crl optional** or **revocation-check** *method1*[*method2*[*method3*]]
13. **password** *string*
14. **exit**
15. **crypto ca enroll** *name* or **crypto pki enroll** *name*
16. **crypto ca authenticate** *name* or **crypto pki authenticate** *name*
17. **crypto pki import <trustpoint> certificate**

18. **sip-ua**
19. **transport tcp tls** [**v1.0** | **v1.1** | **v1.2**]
20. **crypto signaling**{**remote-addr** *ip address subnet mask* | **default**}[**tls-profile** *tag* | **trustpoint** *trustpoint-name* [**client-vtp** *trustpoint-name*| [{**ecdsa-cipher** [**curve-size** 384] | **strict-cipher**}]| **cn-san-validate** {**server** [**client-vtp** *trustpoint-name* | [{**ecdsa-cipher** [**curve-size 384**] | **strict-cipher**}] }]*! ECDSA ciphers are not supported on TLS version 1.0.*
21. **voice service** {**pots**| **voatm** |**vofr**|**voip**}
22. **transport tcp tls**
23. **url** {**sip**| **sips** |**tel**}
24. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa {general-keys | usage-keys}label** *key-label*[**exportable** |][**modulus** *modulus-size*][**storage** *device:*] **Example:** `Router(config)# crypto key generate rsa general-keys label kp1 exportable`	Generates RSA key pairs. Arguments and keywords are as follows: • **general-keys**—Specifies that the general-purpose key pair should be generated. • **usage-keys**—Specifies that two RSA special-usage key pairs should be generated (that is, one encryption pair and one signature pair) instead of one general-purpose key pair. • **label key-label**—(Optional) Name that is used for an RSA key pair when they are being exported. If a key label is not specified, the fully qualified domain name (FQDN) of the router is used. • **exportable**—(Optional) Specifies that the RSA key pair can be exported to another Cisco device, such as a router. • **modulus modulus-size**—(Optional) IP size of the key modulus in a range 350–2048. If you do not enter the modulus keyword and specify a size, you will be prompted. • **storage device:**—(Optional) Specifies the key storage location. The name of the storage device is followed by a colon (:).

	Command or Action	Purpose
		• **kp1**— kp1 is a label name that you select.
Step 4	**crypto key generate ec keysize {256 | 384}**[**label** *label*][*ec key-label*] *! Applicable only for TLS version 1.2.* **Example:** `Router(config)# crypto key generate ec keysize 384 <cr>`	Generates EC key pairs.
Step 5	Required: **crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint cube1`	Declares the trustpoint that your router should use. Argument is as follows: • *name*—Creates a name for the trustpoint that you created. • cube1—Represents the trustpoint name that the user specifies.
Step 6	**rsakeypair** *key-label* [*key-size [encryption-key-size]*] **Example:** `Router(config)# rsakeypair kp1`	Specifies which key pair to associate with the certificate. Arguments are as follows: • *key-label*—Name of the key pair, which is generated during enrollment if it does not exist or if the **auto-enroll regenerate** command is configured. • *key-size*—(Optional) Size of the desired RSA key. If not specified, the existing key size is used. • *encryption-key-size*—(Optional) Size of the second key, which is used to request separate encryption, signature keys, and certificates.
Step 7	**eckeypair** *keyname*] *! Applicable only for TLS version 1.2.* **Example:** `Router(config)# eckeypair mykey`	Generates EC keys for ECDSA cipher suites.
Step 8	**serial-number** [**none**] **Example:** `Router(ca-trustpoint)# serial-number`	Specifies whether the router serial number should be included in the certificate request. Keyword is as follows: • **none**—(Optional) Specifies that a serial number will not be included in the certificate request.
Step 9	**ip-address** {*ip-address*|**interface**|**none**] **Example:** `Router(ca-trustpoint)# ip-address 172.18.197.154`	Specifies a dotted IP address or an interface that will be included as "unstructuredAddress" in the certificate request. Arguments and keyword are as follows: • ip-address—Specifies a dotted IP address that will be included as "unstructuredAddress" in the certificate request.

	Command or Action	Purpose
		• interface—Specifies an interface, from which the router can get an IP address, that will be included as "unstructureAddress" in the certificate request. • **none**—Specifies that an IP address is not to be included in the certificate request.
Step 10	**subject-name** [*x.500-name*] **Example:** `Router(ca-trustpoint)# subject-name CN=172.18.197.154`	Specifies the subject name in the certificate request. Argument is as follows: • **x.500-name**—(Optional) Specifies the subject name that is used in the certificate request.
Step 11	**enrollment** [**mode**][**retry period** *minutes*][**retry count** *number*]**url** *url*[**pem**] **Example:** `Router (ca-trustpoint)# enrollment url http://172.18.193.103`	Specifies the enrollment parameters of a certificate authority (CA). Arguments and keywords are as follows: • **mode**—(Optional) Registration authority (RA) mode, if your CA system provides an RA. By default, RA mode is disabled. • **retry period minutes**—(Optional) Specifies the period in which the router waits before sending the CA another certificate request. The default is 1 minute between retries. (Specify from 1 through 60 minutes.) • **retry count number**—(Optional) Specifies the number of times a router resends a certificate request when it does not receive a response from the previous request. The default is 10 retries. (Specify from 1 through 100 retries.) • **url url**—URL of the file system where your router should send certificate requests. For enrollment method options, see the enrollment url command. • **pem**—(Optional) Adds privacy-enhanced mail (PEM) boundaries to the certificate request.

Step 12	**crl optional** or **revocation-check** *method1*[*method2*[*method3*]]
	Example:
	`Router(ca-trustpoint)# crl optional`
	`or`
	`Router(ca-trustpoint)# revocation-check none`
		Allows the certificates of other peers to be accepted without trying to obtain the appropriate CRL or checks the revocation status of a certificate. Arguments are as follows:
		• *method1 [method2 [method3]]*—Method used by the router to check the revocation status of the certificate.
		Available methods are as follows:
		• **crl**—Certificate checking is performed by a certificate revocation list (CRL). This is the default behavior.
		• **none**—Certificate checking is not required.

	Command or Action	Purpose
		• **ocsp**—Certificate checking is performed by an online certificate status protocol (OCSP). **Note** If the second and the third methods are specified, each method will be used only if the previous method returns an error, such as the server being down.
Step 13	**password** *string* **Example:** `Router(ca-trustpoint)# password password`	(Optional) Specifies the revocation password for the certificate. Argument is as follows: • *string*—Name of the password
Step 14	**exit** **Example:** `Router# exit`	Exists the current mode.

Step 15	**crypto ca enroll** *name* or **crypto pki enroll** *name*
	Example:
	`Router(config)# crypto ca name cube1`
	`or`
	`Router(config)# crypto pki name cube1`
		Obtains the certificates of your router from the certificate authority. The CA server issues two certificates to the trustpoint (CUBE): one to certify the CA server and the other to certify the trustpoint (CUBE). Argument is as follows:
		• *name*—Specifies the name of the CA. Use the same name when you declared the CA using the **crypto pki trustpoint** command.

Step 16	**crypto ca authenticate** *name* or **crypto pki authenticate** *name*
	Example:
	`Router(config)# crypto ca authenticate cube1`
	`or`
	`Router(config)# crypto pki authenticate cube1`
		Authenticates the CA (by getting the certificate of the CA). Argument is as follows:
		• *name*—Specifies the name of the CA. This is the same name that is used when the CA was declared with the crypto CA identity command.
		Note This is where you paste the remote root CA certificate (PEM file format).

Step 17	**crypto pki import <trustpoint> certificate**	Imports the certificate given by the CA.
Step 18	**sip-ua** **Example:** `Router(config)# sip-ua`	Enters SIP user-agent configuration mode.
Step 19	**transport tcp tls** [**v1.0** | **v1.1** | **v1.2**] **Example:**	Configures the specified TLS version.

	Command or Action	Purpose
	`Router(config-sip-ua)# transport tcp tls v1.2`	**Note** TLS v1.1 and TLS v1.2 are the default TLS versions that are configured. TLS v1.0 is also supported. However, to configure TLS v1.0, you must explicitly specify the TLS version. For more information on the TLS version configuration, see Transport command.
Step 20	**crypto signaling**{**remote-addr** *ip address subnet mask* | **default**}[**tls-profile** *tag* | **trustpoint** *trustpoint-name* [**client-vtp** *trustpoint-name*| [{**ecdsa-cipher** [**curve-size** 384] | **strict-cipher**}]| **cn-san-validate** {**server** [**client-vtp** *trustpoint-name* | [{**ecdsa-cipher** [**curve-size 384**] | **strict-cipher**}] }]*! ECDSA ciphers are not supported on TLS version 1.0.* **Example:** `Router(config-sip-ua)# crypto signaling default trustpoint cube1`	Configures the SIP gateway to use its trustpoint when it establishes or accepts TLS connection with a remote device with an IP address. The trustpoint label refers to the CUBE's certificate that is generated with the Cisco IOS PKI commands as part of the enrollment proces. **strict-cipher** means that the SIP TLS process uses only those cipher suites that are mandated by the SIP RFC. 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. Keywords and arguments are as follows: • **remote-addr** *address*—Associates an IP address to a trustpoint. • **remote-addr** *subnet mask*—Associates a subnet mask to a trustpoint. • **default**—Configures a default trustpoint. • **trustpoint** *string*—Refers to the SIP gateways certificate generated as part of the enrollment process using Cisco IOS PKI commands • **ecdsa-cipher**—Examples are the following: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384. **Note** **ecdsa-cipher** is applicable only for the **TLS version 1.2** • **curve-size** - configures the specific size of elliptic curves to be used for a TLS session. **384**- configures 384-bit Elliptic Curve. • **strict-cipher**—Examples are the following: TLS_RSA_WITH_AES_128_CBC_SHA,

	Command or Action	Purpose
		TLS_DHE_RSA_WITH_AES_128_CBC_SHA1, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, and TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384. • **cn-san-validate server**- Enables the server identity validation through Common Name (CN) and Subject Alternate Name (SAN) fields in the server certificate during client-side SIP/TLS connections. Validation of the CN and SAN fields of the server certificate ensures that the server-side domain is a valid entity. While setting up a TLS connection to a target server, CUBE validates the domain name that is configured as destination against the CN/SAN fields in the certificate provided by server. The TLS connection is established only if the domain name that is configured as destination, matches with one of the domain names in the CN/SAN fields of the server certificate that is configured. Once you configure **cn-san-validate**{**server**}, validation of the server identity happens for every new TLS connection. • The keyword **tls-profile** *tag* associates all the voice class configurations that are made through the command **voice class tls-profile** *tag*. In addition to all the TLS crypto configuration options available under the command **crypto signaling**, the **voice class tls-profile** *tag* command has a keyword **sni send**. **sni send** enables Server Name Indication (SNI), a TLS extension that allows a TLS client to indicate the name of the server that it is trying connect during the initial TLS handshake process. Only the fully qualified DNS hostname of the server is sent in the client hello. SNI does not support IPv4 and IPv6 addresses in the client hello extension. After receiving a "hello" with the server name from the TLS client, the server uses appropriate certificate in the subsequent TLS handshake process. SNI is supported from TLS 1.2. For more information on associating **voice class tls-profile** *tag* command to **crypto signaling** command, see crypto signaling and voice class tls-profile.

	Command or Action	Purpose
		Note From Cisco IOS XE Amsterdam 17.3.1a onwards, any new voice class TLS profile configuration option is available only under the command **voice class tls-profile** *tag*. You must perform voice class TLS profile configuration under the command **voice class tls-profile** *tag* and associate it to **crypto signaling** command. For example, **sni send** keyword is available only under the command **voice class tls-profile** *tag*. The **crypto signaling** command continues to support previously existing TLS crypto options. You can use either **voice class tls-profile** *tag* or **crypto signaling** command to configure trustpoint. However, from Cisco IOS XE Amsterdam 17.3.1a onwards, we recommend that you use the command **voice class tls-profile** *tag* to perform TLS profile configurations.
Step 21	**voice service** {**pots**| **voatm** |**vofr**|**voip**} **Example:** `Router(config)# voice service voip`	Specifies a voice encapsulation type and enters voice service VoIP configuration mode.
Step 22	**transport tcp tls** **Example:** `Router(config-voi-sip)# transport tcp tls`	Enters this command in SIP configuration mode to enable the TLS port on TCP 5061 to listen.
Step 23	**url** {**sip**| **sips** |**tel**} **Example:** `Router(config-serv-sip)# url sips`	Configures URLs to either the SIP, SIPS, or TEL format for your VoIP SIP calls. Keywords are as follows: • **sip**—Generate URLs in SIP format for VoIP calls. This is the default. • **sips**—Generate URLs in SIPS format for VoIP calls. • **tel**—Generate URLs in TEL format for VoIP calls. **Note** This SIP gateway is now configured to use TLS with endpoints sharing the same CA.
Step 24	**end** **Example:**	Ends the current mode.

	Command or Action	Purpose
	`Router(conf-serv-sip)# end`	

Verifying SIP TLS Configuration

After a call is made, the **show sip-ua connections tcp tls** command is used to verify whether the transport used for the call is TLS.

Sample output for this command when TLS version is 1.0:

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 Local-Address
  =========== ======= =========== =========== ==============
         5061       1 Established           0   10.64.86.88
======================================================================
```

Sample output for the **show sip-ua connections tcp tls** command when TLS version is 1.2:

Detail Output

```
router# show sip-ua connections tcp tls detail
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 1, recorded for 209.165.201.1:5061
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:209.165.201.1, Connections-Count:2
Remote-Port Conn-Id Conn-State  WriteQ-Size Local-Address TLS-Version

=========== ======= =========== =========== ============= ===========
       5061       3 Established           0             -      TLSv1.2
      36289       2 Established           0             -      TLSv1.2

Cipher                          Curve
==============================  ======
ECDHE-ECDSA-AES256-GCM-SHA384   P-384
ECDHE-ECDSA-AES256-GCM-SHA384   P-384

-------------- SIP Transport Layer Listen Sockets ---------------
  Conn-Id               Local-Address
 ===========    ============================
   0            [0.0.0.0]:5061:
```

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
```

SIP TLS Configuration Examples

Example: SIP TLS Configuration

```
show running-config
Building configuration...
```

```
Current configuration : 10894 bytes
!
! Last configuration change at 23:19:20 IST Wed Aug 19 2015
! NVRAM config last updated at 20:25:45 IST Tue Aug 18 2015
!
version 15.6
service timestamps debug datetime msec localtime show-timezone
service timestamps log datetime msec localtime show-timezone
no service password-encryption
!
hostname CUBE
!
boot-start-marker
boot system flash:ctestimg
boot-end-marker
!
aqm-register-fnf
!
logging queue-limit 1000
logging buffered 9999999
no logging rate-limit
no logging console
!
no aaa new-model
ethernet lmi ce
clock timezone IST 5 30
!
!
!
!
!
ip traffic-export profile 1 mode capture
  bidirectional
  incoming access-list 123
  outgoing access-list 123
!
!
!
!
no ip domain lookup
ip cef
no ipv6 cef
!
multilink bundle-name authenticated
!
!
!
!
!
!
crypto pki trustpoint ecdsacert1
enrollment terminal pem
subject-name cn=plutododsn
revocation-check none
eckeypair myeckey
!
crypto pki trustpoint selfsign
enrollment selfsigned
subject-name cn=plutododsn
revocation-check none
rsakeypair selfsign
!
crypto pki trustpoint ccm155RSA
```

```
enrollment terminal
revocation-check none
!
!
crypto pki certificate chain ecdsacert1
certificate 07
  30820248 308201CD A0030201 02020107 300A0608 2A8648CE 3D040303 30593112
  30100603 5504030C 09706C75 746F3164 6F64310C 300A0603 55040B0C 03544143
  310E300C 06035504 0A0C0543 6973636F 310B3009 06035504 06130242 45311830
  16060355 04070C0F 74757269 6E672D65 7865632D 6C6E7830 1E170D31 35303831
  38313235 3431345A 170D3136 30383137 31323534 31345A30 36311330 11060355
  0403130A 706C7574 6F646F64 736E311F 301D0609 2A864886 F70D0109 02161063
  656E7472 616C6973 65645F72 74723230 76301006 072A8648 CE3D0201 06052B81
  04002203 62000446 4E28C72B 9A66C344 7D6EB2C7 51CE17F3 D125D12B 7043A98B
  F21825DF 0621A34D 3119E23F DB2A5ACE 1C744F17 789450F5 1071E504 DA7DC56C
  CDA24A8B 5318F11B EBA618A1 4BE2C66A 27857932 48485192 74923495 E762E3B4
  5BDCDBD0 BC6B1FA3 818B3081 88301D06 03551D0E 04160414 BE2A3FDE F3CDA85E
  A0EC7EA1 A47F3AEB 6B16D388 301F0603 551D2304 18301680 1460CAB8 AF1250CF
  BB00C516 ACEEAF72 FDB6198D 6C303606 082B0601 05050701 01042A30 28302606
  082B0601 05050730 01861A68 7474703A 2F2F2031 37332E33 392E3537 2E38333A
  38303830 2F300E06 03551D0F 0101FF04 04030207 80300A06 082A8648 CE3D0403
  03036900 30660231 00977017 6DCE34A4 3B0F78CF 2C69C7AD 2123B5F9 C10999E7
  A3316D34 43E9C928 8FBF42A4 84583017 856D513D C5B66547 1E023100 AEF7EFE8
  48AC2C81 884E8C8D 421A9B11 3177582D DBE9973F D67EA687 0CF08620 375628D0
  F5F7FDFA 53052711 E493A754
              quit
certificate ca 00
  3082023B 308201C1 A0030201 02020100 300A0608 2A8648CE 3D040303 30593112
  30100603 5504030C 09706C75 746F3164 6F64310C 300A0603 55040B0C 03544143
  310E300C 06035504 0A0C0543 6973636F 310B3009 06035504 06130242 45311830
  16060355 04070C0F 74757269 6E672D65 7865632D 6C6E7830 1E170D31 35303830
  36303934 3730345A 170D3136 30383035 30393437 30345A30 59311230 10060355
  04030C09 706C7574 6F31646F 64310C30 0A060355 040B0C03 54414331 0E300C06
  0355040A 0C054369 73636F31 0B300906 03550406 13024245 31183016 06035504
  070C0F74 7572696E 672D6578 65632D6C 6E783076 30100607 2A8648CE 3D020106
  052B8104 00220362 0004D2EE C8BE0015 AE8DF590 3F0A8955 C1B3D80F 99B3CE51
  241719ED 4D733BDC 061F92D0 36899A05 71E515B9 A86306B4 6DC49D66 87843054
  71E3151B 293971A2 94B14074 893BB537 09D4BC9C BF57E3DC AD5FA66B 590DA475
  B303068C 66899963 763CA35D 305B300C 0603551D 13040530 030101FF 300B0603
  551D0F04 04030201 06301D06 03551D0E 04160414 60CAB8AF 1250CFBB 00C516AC
  EEAF72FD B6198D6C 301F0603 551D2304 18301680 1460CAB8 AF1250CF BB00C516
  ACEEAF72 FDB6198D 6C300A06 082A8648 CE3D0403 03036800 30650230 390E60B9
  9AF19940 B0898320 AE96D8CB 52FB3B75 CE599444 EA3DBAC1 F4517F13 B96C26CB
  3B719834 A99AF174 6EF9E35D 0231008E 337B0A8F 864F32D4 C85CC7CC 585FCD8B
  6F5F4BCE 3B0313D1 E3B76598 0D9E43EB B11EFCF5 8C76318C 0F835560 0CD149
              quit
crypto pki certificate chain selfsign
certificate self-signed 01
  30820235 3082019E A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  36311330 11060355 0403130A 706C7574 6F646F64 736E311F 301D0609 2A864886
  F70D0109 02161063 656E7472 616C6973 65645F72 74723230 1E170D31 35303831
  38313434 3234355A 170D3230 30313031 30303030 30305A30 36311330 11060355
  0403130A 706C7574 6F646F64 736E311F 301D0609 2A864886 F70D0109 02161063
  656E7472 616C6973 65645F72 74723230 819F300D 06092A86 4886F70D 01010105
  0003818D 00308189 02818100 A01400F8 9A599812 F5CC7347 1F9E223C E395073B
  9138C777 7EAEA997 5EA3B937 4B858866 2A022ADA 7D29C4C6 DC9B01EB 0E9E77DF
  782B099F 8F701221 A11C8B81 D82AB7F3 DBC1FFCB 809FC745 3FC6BD87 725F6B66
  EBEBBD78 6597DDFB 700D3DA6 73C52342 568670EA 1DEB6619 2ED5EC71 99B2612A
  BEC9B76E 38C442D9 DB9C2293 02030100 01A35330 51300F06 03551D13 0101FF04
  05300301 01FF301F 0603551D 23041830 1680141D 5971FE06 1D126AA3 6767DBA6
  C30E2EF0 2C044430 1D060355 1D0E0416 04141D59 71FE061D 126AA367 67DBA6C3
  0E2EF02C 0444300D 06092A86 4886F70D 01010505 00038181 0033BC90 8AF1DFBD
  B03AE032 ABBD80B7 7418402B 0BFB9E0B 341CB523 7077570C CD495BE3 47A1B35B
  C878C693 A491B433 37BA1170 45F1DF85 9BC22CA8 94E25907 F91C7B75 450B0DE1
```

```
  76AC2C6B 5517F42A 46260F76 4A1DF81F 733A14FE 918F43F4 9BABAA49 227B5014
  986044E7 8E98E373 7A361696 F0AD3ACC C9B101DF 2F80CCF7 E3
              quit
crypto pki certificate chain ccm155RSA
certificate ca 4E23E56C7339CC679FD444D77F7A463F
  308203AB 30820293 A0030201 0202104E 23E56C73 39CC679F D444D77F 7A463F30
  0D06092A 864886F7 0D01010B 0500306A 310B3009 06035504 06130249 4E310E30
  0C060355 040A0C05 63697363 6F310D30 0B060355 040B0C04 73727467 31143012
  06035504 030C0B50 4C55544F 2D435543 4D313112 30100603 5504080C 096B6172
  6E617461 6B613112 30100603 5504070C 0962616E 67616C6F 7265301E 170D3135
  30383034 31333431 35315A17 0D323030 38303231 33343135 305A306A 310B3009
  06035504 06130249 4E310E30 0C060355 040A0C05 63697363 6F310D30 0B060355
  040B0C04 73727467 31143012 06035504 030C0B50 4C55544F 2D435543 4D313112
  30100603 5504080C 096B6172 6E617461 6B613112 30100603 5504070C 0962616E
  67616C6F 72653082 0122300D 06092A86 4886F70D 01010105 00038201 0F003082
  010A0282 010100CC 39112782 D93A3501 8913EBEA 42522D27 E2C58D3F 4FC896D2
  8F38F4A5 7CCC2519 9683142A 6B203E9F C7C92673 85D5A940 99B20FBD CC8F97D1
  F42C1580 D34B8831 3BA74AE0 79AC0C74 E7BFAFCE 4D23F106 3D4EA333 16BA4768
  66C5561C 5CE19946 DA731D9E 6E743FA0 5F25E445 8E5B6789 64076291 7E5EB0DA
  C482074E 56DA6841 245EEB96 F44C900D 85C5EDEC 32E89675 BC934EC3 8C0FC7D8
  02BBCC06 93EE3698 A8B44527 93A73391 9C71869D BDEB96BF 06D68AC0 D47D810E
  FCAB3C8F 13BC3D62 02591976 CD49436E 3E2D5B20 079A031E 3FDDEC1C DFBF8261
  3CC5C6AF 7C6FC79C 0234D266 6C508DD7 CC72C8C6 239372F6 7D7CF5CD B56FFB26
  DB4122E2 01E15F02 03010001 A34D304B 300B0603 551D0F04 04030202 B4301D06
  03551D25 04163014 06082B06 01050507 03010608 2B060105 05070302 301D0603
  551D0E04 1604142D DF3CC8F3 57F44974 38D8E8E8 20B15658 9C17F430 0D06092A
  864886F7 0D01010B 05000382 01010038 060F1AC3 C3938667 8A3A0513 B5B2CE16
  0DC6BAE5 5B1D6DD7 CEB68832 F92A4270 5FC7EC97 7AAFB2AA 4FA288DD 66A94AB4
  A466CA7E F974B9B8 630FAC21 AB95C3BB ECB7A082 AB0343BE 2F89399D AD94D4A5
  6B477B44 88FB94BF FEE2E571 4917D0BB 2A5733B5 4F1F58BA CCCE710F 64365B39
  3F1F9E8F 81A1B71B 61BD51EB C45A2FAD CA743432 A61C19AB E6C4B5F1 6E673A38
  53421ECE 992505BD 5BAAF32A 954E37EA FE03B725 283A7F19 374A87E9 891E4E60
  B8399050 0902EA25 99FD2A26 2BD3A2E9 74F01C53 EFB3D4D6 654D064E 56878F6C
  21D8D184 88C24AD9 E655B78E 12EDB7EE 696B9B77 3E73A3F0 10DEBDF2 3CDF2BC9
  606700D1 2D42389C EEE43B56 22977A
              quit
voice-card 0
dspfarm
dsp services dspfarm
!
!
!
voice service voip
no ip address trusted authenticate
allow-connections sip to sip
fax protocol t38 version 0 ls-redundancy 0 hs-redundancy 0 fallback none
sip
  bind control source-interface GigabitEthernet0/1
  bind media source-interface GigabitEthernet0/1
  asymmetric payload full
  srtp negotiate cisco
!
!
!
!
voice iec syslog
!
!
!
!
mta send mail-from username $s$
license udi pid CISCO2921/K9 sn FGL1538116L
hw-module pvdm 0/0
!
```

```
!
!
no memory lite
!
redundancy
!
!
!
!
!
interface Embedded-Service-Engine0/0
no ip address
shutdown
!
interface GigabitEthernet0/0
ip address 9.45.38.192 255.255.0.0
shutdown
duplex auto
speed auto
!
interface GigabitEthernet0/1
ip address 10.64.86.177 255.255.255.0
ip traffic-export apply 1 size 5000000
duplex auto
speed auto
  no clns route-cache
!
interface GigabitEthernet0/2
no ip address
shutdown
duplex auto
speed auto
!
ip forward-protocol nd
!
ip http server
no ip http secure-server
!
ip rtcp report interval 9000
ip route 0.0.0.0 0.0.0.0 10.64.86.1
ip route 10.0.0.0 255.0.0.0 10.64.86.1
!
!
!
access-list 123 permit udp any any
access-list 123 permit tcp any any
!
control-plane
!
call treatment on
!
!
!
!
!
mgcp behavior rsip-range tgcp-only
mgcp behavior comedia-role none
mgcp behavior comedia-check-media-src disable
mgcp behavior comedia-sdp-force disable
!
mgcp profile default
!
sccp local GigabitEthernet0/1
sccp ccm 10.64.86.154 identifier 1 version 7.0
```

```
!
!
!
dial-peer voice 1 voip
destination-pattern 6003
session protocol sipv2
session target ipv4:10.64.86.206:5061
session transport tcp tls
incoming called-number 7003
codec g711ulaw
!
dial-peer voice 2 voip
destination-pattern 7003
session protocol sipv2
session target ipv4:10.64.86.206:5061
session transport tcp tls
incoming called-number 6003
codec g711ulaw
!
!
sip-ua
 transport tcp tls v1.2
connection-reuse
crypto signaling default trustpoint ecdsacert1 ecdsa-cipher
!
!
!
gatekeeper
shutdown
!
!
!
line con 0
exec-timeout 0 0
speed 115200
line aux 0
line 2
no activation-character
no exec
transport preferred none
transport output pad telnet rlogin lapb-ta mop udptn v120 ssh
stopbits 1
line vty 0 4
login
transport input none
!
!
end
```