

# Konfigurieren von SIP-TLS zwischen CUCM-CUBE/CUBE-SBC und von CA signierten Zertifikaten

## Inhalt

[Einführung](#)

[Voraussetzungen](#)

[Anforderungen](#)

[Verwendete Komponenten](#)

[Konfigurieren](#)

[Netzwerkdiagramm](#)

[Konfiguration](#)

[Überprüfen](#)

—

[Fehlerbehebung](#)

## Einführung

In diesem Dokument wird beschrieben, wie SIP Transport Layer Security (TLS) zwischen Cisco Unified Communication Manager (CUCM) und Cisco Unified Border Element (CUBE) mit Zertifikaten der Zertifizierungsstelle (Certificate Authority, CA) konfiguriert wird.

## Voraussetzungen

Cisco empfiehlt, diese Themen zu kennen.

- SIP-Protokoll
- Sicherheitszertifikate

## Anforderungen

- Datum und Uhrzeit müssen auf den Endpunkten übereinstimmen (es wird empfohlen, dieselbe NTP-Quelle zu verwenden).
- Der CUCM muss sich im gemischten Modus befinden.
- TCP-Konnektivität ist erforderlich (Open port 5061 on any Transit Firewall).
- CUBE muss über die Sicherheits- und Unified Communication K9 (UCK9)-Lizenzen verfügen.

**Hinweis:** Ab Cisco IOS-XE Version 16.10 ist die Plattform auf Smart Licensing umgestiegen.

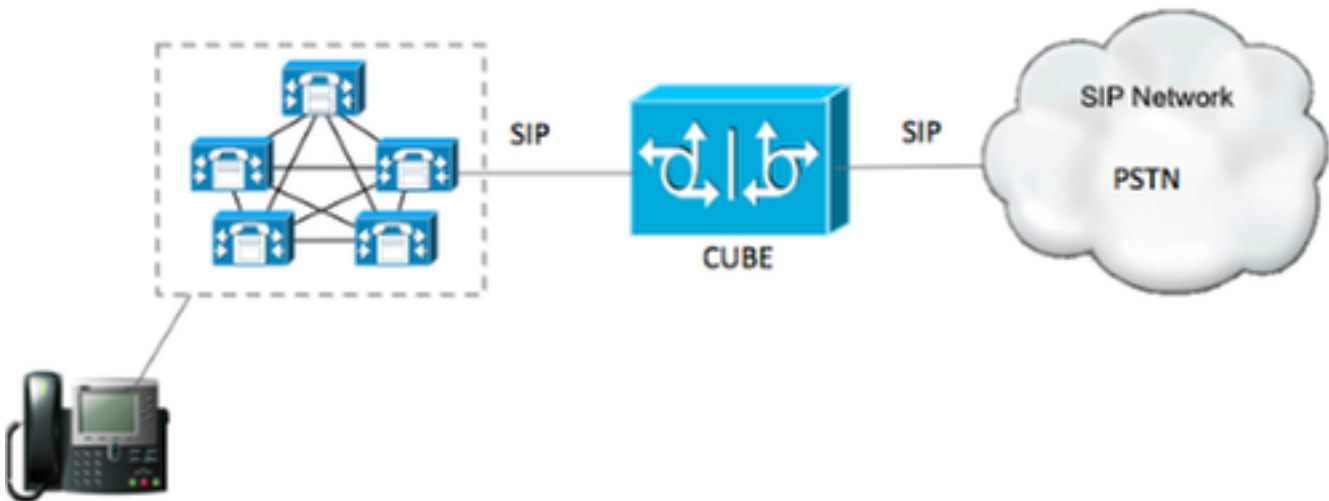
## Verwendete Komponenten

- SIP

- Zertifizierungsstelle unterzeichnete Zertifikate
- Cisco IOS- und IOS-XE-Gateways Versionen 2900/3900/4300/4400/CSR1000v/ASR100X: Über 15,4
- Cisco Unified Communications Manager (CUCM) Versionen: Über 10,5

## Konfigurieren

### Netzwerkdiagramm



### Konfiguration

Schritt 1: Sie erstellen einen RSA-Schlüssel, der mit dem Befehl:

```
Crypto key generate rsa label TestRSAkey exportable modulus 2048
```

Mit diesem Befehl wird ein RSA-Schlüssel mit einer Länge von 2048 Bit erstellt (maximal 4096).

Schritt 2: Erstellen Sie mithilfe der folgenden Befehle einen Vertrauenspunkt, um unser Zertifikat mit CA-Signatur zu speichern:

```
Crypto pki trustpoint CUBE_CA_CERT
  serial-number none
  fqdn none
  ip-address none
  subject-name cn=ISR4451-B.cisco.lab !(this has to match the router's hostname
[hostname.domain.name])
  revocation-check none
  rsakeypair TestRSAkey !(this has to match the RSA key you just created)
```

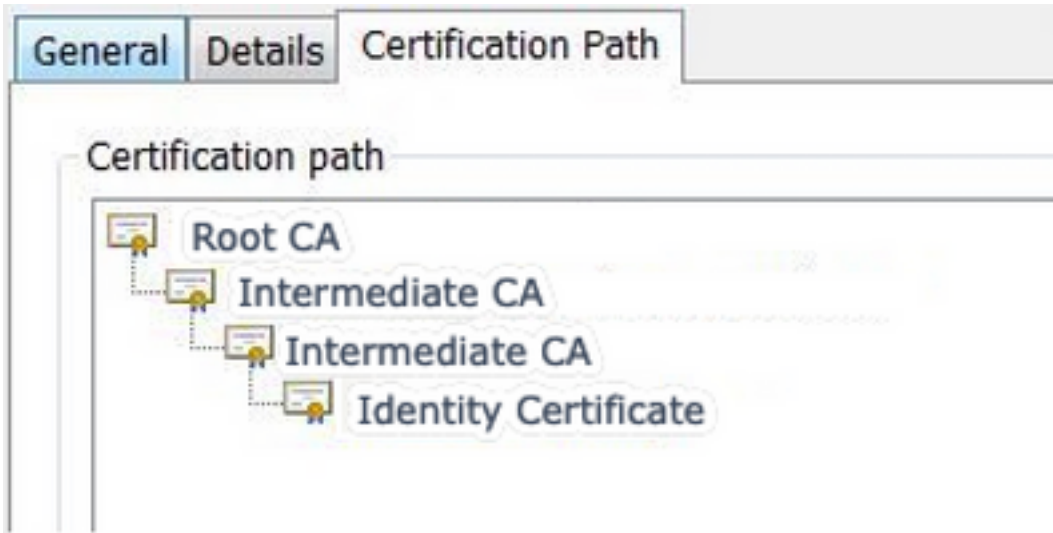
Schritt 3: Nachdem Sie unser Trustpoint eingerichtet haben, erstellen Sie jetzt unsere CSR-Anfrage mit den folgenden Befehlen:

```
Crypto pki enroll CUBE_CA_CERT
```

Beantworten Sie die Fragen auf dem Bildschirm, kopieren Sie dann die CSR-Anfrage, speichern Sie sie in einer Datei und senden Sie sie dann an die Zertifizierungsstelle.

Schritt 4: Sie müssen herausfinden, ob die Stammzertifizierungskette Zwischenzertifikate besitzt. Falls keine Zwischenzertifikatbehörden vorhanden sind, fahren Sie mit Schritt 7 fort, andernfalls fahren Sie mit Schritt 6 fort.

Schritt 5: Erstellen Sie einen Vertrauenspunkt, um das Root-Zertifikat zu besitzen. Erstellen Sie außerdem einen Vertrauenspunkt, um eine zwischengeschaltete Zertifizierungsstelle zu speichern, bis die Zertifizierungsstelle das CUBE-Zertifikat signiert (siehe Bild unten).



In diesem Beispiel ist die 1. Ebene die Root-Zertifizierungsstelle, die 2. Ebene die erste Zwischenstufe der Zertifizierungsstelle, die 3. Ebene ist die Zertifizierungsstelle, die unser CUBE-Zertifikat signiert. Daher müssen Sie einen Vertrauenspunkt erstellen, um die ersten 2 Zertifikate mit diesen Befehlen zu besitzen.

```
Crypto pki trustpoint Root_CA_CERT
Enrollment terminal pem
Revocation-check none
```

```
Crypto pki authenticate Root_CA_CERT
Paste the X.64 based certificate here
```

```
Crypto pki trustpoint Intermediate_CA
Enrollment terminal
Revocation-check none
```

```
Crypto pki authenticate Intermediate_CA
```

Schritt 6: Nach Erhalt des Zertifikats der Zertifizierungsstelle authentifizieren Sie den Vertrauenspunkt. Der Trustpoint muss das Zertifikat der Zertifizierungsstelle vor dem CUBE-Zertifikat besitzen. Der Befehl, mit dem das Zertifikat importiert werden kann, lautet:

```
Crypto pki authenticate CUBE_CA_CERT
```

Schritt 7: Sobald Sie unser Zertifikat installiert haben, müssen Sie diesen Befehl ausführen, um unser CUBE-Zertifikat zu importieren

```
Crypto pki import CUBE_CA_CERT cert
```

Schritt 8: Konfigurieren Sie SIP-UA zur Verwendung des von Ihnen erstellten Trustpoints.

```
sip-ua
crypto signaling default trustpoint CUBE_CA_CERT
```

Schritt 9. Konfigurieren Sie DFÜ-Peers wie unten gezeigt:

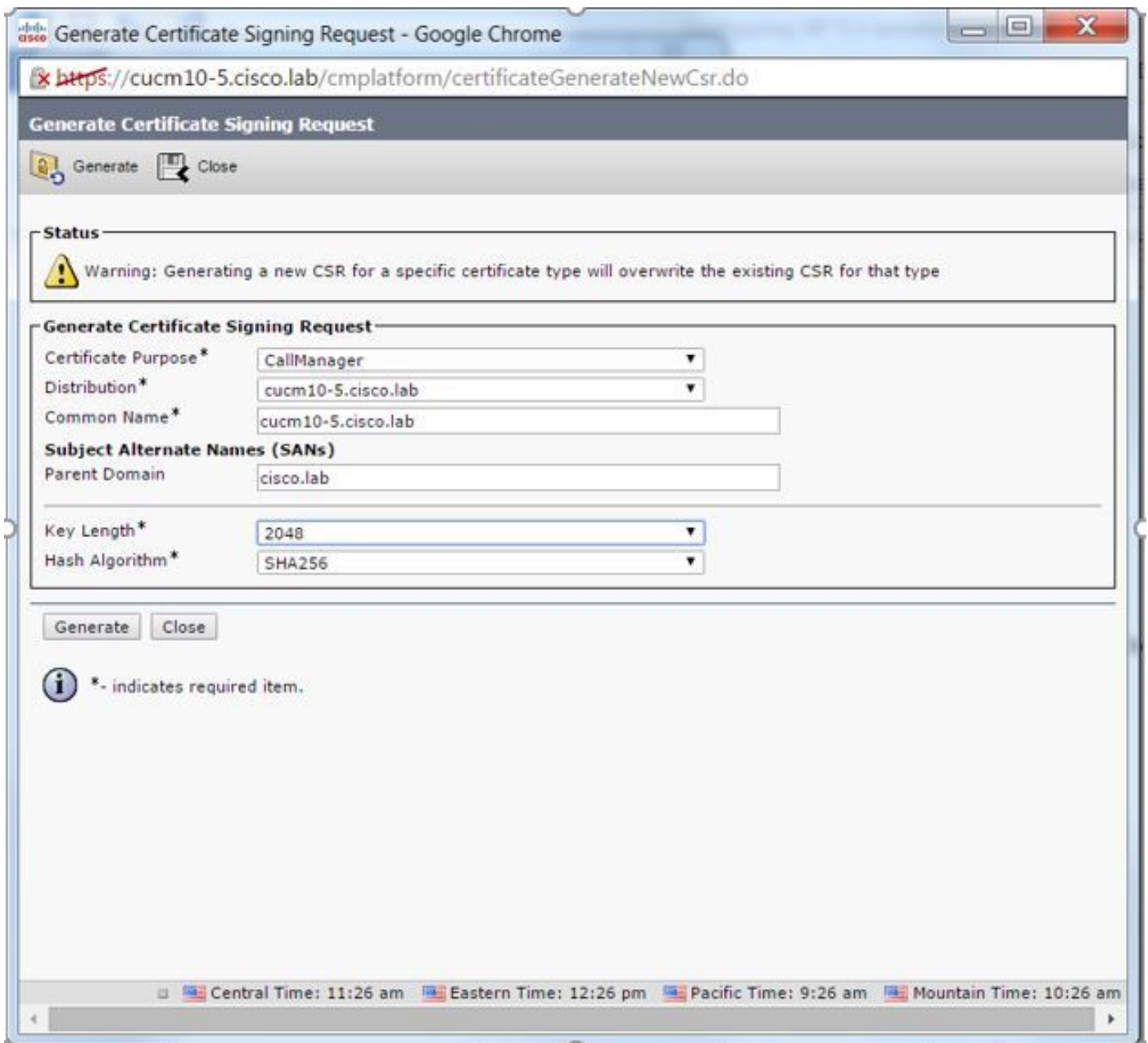
```
dial-peer voice 9999 voip
answer-address 35..
destination-pattern 9999
session protocol sipv2
session target dns:cucm10-5
session transport tcp tls
voice-class sip options-keepalive
srtp
```

Damit ist die CUBE-Konfiguration abgeschlossen.

Schritt 10: Jetzt erstellen Sie unseren CUCM CSR. Befolgen Sie die folgenden Anweisungen:

- Melden Sie sich beim CUCM-Betriebssystemadministrator an.
- Klicken Sie auf Sicherheit.
- Klicken Sie auf Zertifikatsverwaltung.
- Klicken Sie auf CSR erstellen.

Die CSR-Anfrage muss wie folgt aussehen:



Schritt 11: Laden Sie die CSR herunter und senden Sie sie an die Zertifizierungsstelle.

Schritt 11: Laden Sie die Zertifizierungsstellen-signierte Zertifikatkette in den CUCM hoch. Die Schritte sind wie folgt:

- Klicken Sie auf Sicherheit und dann auf Zertifikatsverwaltung.
- Klicken Sie auf Upload Certificate/Certificate Chain.
- Wählen Sie im Dropdown-Menü Zertifikatzweck die Option Call Manager aus.
- Navigieren Sie zu Ihrer Datei.
- Klicken Sie auf Upload.

Schritt 13: Melden Sie sich bei der CUCM-CLI an, und führen Sie diesen Befehl aus.

```
utils ctl update CTLFile
```

Schritt 14: Konfigurieren eines CUCM-SIP-Trunk-Sicherheitsprofils

- Klicken Sie auf das System und dann auf das Sicherheitsprofil, um die Trunk-Sicherheit zu aktivieren.


- Konfigurieren Sie das Profil wie im Bild gezeigt.

### SIP Trunk Security Profile Configuration

Save Delete Copy Reset Apply Config Add New

---

**Status**

 Status: Ready


---

**SIP Trunk Security Profile Information**


Name*	CUBE_CA Secure SIP Trunk Profile
Description	Secure SIP Trunk Profile authenticated by null String
Device Security Mode	Encrypted ▼
Incoming Transport Type*	TLS ▼
Outgoing Transport Type	TLS ▼
<input type="checkbox"/> Enable Digest Authentication	
Nonce Validity Time (mins)*	600
X.509 Subject Name	cucm10-5.cisco.lab
Incoming Port*	5061
<input type="checkbox"/> Enable Application level authorization	
<input checked="" type="checkbox"/> Accept presence subscription	
<input checked="" type="checkbox"/> Accept out-of-dialog refer**	
<input checked="" type="checkbox"/> Accept unsolicited notification	
<input checked="" type="checkbox"/> Accept replaces header	
<input checked="" type="checkbox"/> Transmit security status	
<input type="checkbox"/> Allow charging header	
SIP V.150 Outbound SDP Offer Filtering*	Use Default Filter ▼

**Hinweis:** In diesem Fall muss der Betreffname X.509 mit dem Betreffnamen des CUCM-Zertifikats übereinstimmen, wie im hervorgehobenen Teil des Bildes gezeigt.

## Certificate Details for cucm10-5.cisco.lab, CallManager

 Regenerate  Generate CSR  Download .PEM File  Download .DER File

### Status

 Status: Ready

### Certificate Settings

Locally Uploaded	10/02/16
File Name	CallManager.pem
Certificate Purpose	CallManager
Certificate Type	certs
Certificate Group	product-cm
Description(friendly name)	Certificate Signed by AD-CONTROLLER-CA

### Certificate File Data

```
[
Version: V3
Serial Number: 1D255E0000000000000007
SignatureAlgorithm: SHA256withRSA (1.2.840.113549.1.1.11)
Issuer Name: CN=AD-CONTROLLER-CA, DC=cisco, DC=lab
Validity From: Wed Feb 10 10:45:23 CST 2016
           To: Fri Feb 10 10:55:23 CST 2017
Subject Name: CN=cucm10-5.cisco.lab, OU=TAC, O=CISCO, L=RICHARSON, ST=TEXAS, C=US
Key: RSA (1.2.840.113549.1.1.1)
Key value:
3082010a0282010100ae8db062881c35163f1b6ee4be4951158fdb3495d3c8032170c9fb8bafb385a2
27b00ec1024807f0adc49df875189779c7de1ae1e7e64b45e6f9917fa6ca5687d9aeaf20d70018e8d5
58a832360b82702249fc98855012c7d2cc29eea0f92fad9e739d73b0fa24d7dd4bd9fc96be775fda997
f03a440645ad64fa9f083ed95445e200187dd8775aa543b2bab11a5e223e23ef03bb86bb9fd969b3d9
3ba2550c35ea06ed5149aef2253c2455a622122e0aa3b649a090911995069a2cfd4ab4ab1fe15b242
```

Regenerate

Generate CSR

Download .PEM File

Download .DER File

Schritt 15: Konfigurieren Sie einen SIP-Trunk wie gewohnt auf dem CUCM.

- Stellen Sie sicher, dass das Kontrollkästchen SRTP Allowed (SRTP zugelassen) aktiviert ist.
- Konfigurieren Sie die richtige Zieladresse, und stellen Sie sicher, dass Port 5060 durch Port 5061 ersetzt wird.
- Stellen Sie sicher, dass Sie im SIP-Trunk-Sicherheitsprofil den in Schritt 14 erstellten SIP-Profilnamen auswählen.



**SIP Information**

**Destination**

Destination Address is an SRV

Destination Address: 1\* [redacted] Destination Address IPv6: [empty] Destination Port: 5061

MTP Preferred Originating Codec\*: 711ulaw

BLF Presence Group\*: Standard Presence group

SIP Trunk Security Profile\*: ISR4451-B Secure SIP Trunk Profile

Rerouting Calling Search Space: < None >

Out-Of-Dialog Refer Calling Search Space: < None >

SUBSCRIBE Calling Search Space: < None >

SIP Profile\*: Standard SIP Profile-options [View Details](#)

DTMF Signaling Method\*: No Preference

## Überprüfen

Wenn zu diesem Zeitpunkt alle Konfigurationen in Ordnung sind,

Auf dem CUCM zeigt der SIP-Trunk-Status Full Service an (Vollständiger Service), wie im Bild gezeigt.

Name	Description	Calling Search Space	Device Pool	Route Pattern	Partition	Route Group	Priority	Trunk Type	SIP Trunk Status	SIP Trunk Duration
ISR4451-B			0711-Secure					SIP Trunk	Full Service	Time In Full Service: 0 day 0 hour 0 minute

Auf CUBE zeigt der DFÜ-Peer diesen Status an:

```

TAG      TYPE  MIN  OPER PREFIX      DEST-PATTERN      FER THRU SESS-TARGET  STAT PORT
KEEPALIVE

9999    voip  up   up           9999              0 syst dns:cucm10-5   active

```

Dieser Prozess gilt für andere Router. Der einzige Unterschied besteht darin, dass Sie das von einem Drittanbieter bereitgestellte Zertifikat hochladen, anstatt den CUCM-Zertifikat hochzuladen.

## Fehlerbehebung

Aktivieren Sie diese Debug-Optionen auf CUBE

```

debug crypto pki api
debug crypto pki callbacks
debug crypto pki messages
debug crypto pki transactions
debug ssl openssl errors
debug ssl openssl msg
debug ssl openssl states
debug ip tcp transactions

```