IP Phone Security and CTL (Certificate Trust List)

- Purpose on page 1
- Phone Security and CTL Overview on page 1
  Configuration on page 3
    - 1. Obtain USB eTokens on page 3
    - 2. Activate CTL Provider and CAPF Services on page 4
    - 3. Download and Install the CTL Client on page 5
    - 4. Run the CTL Client using eTokens on page 6
    - 5. Restart Required Servers on page 12
    - 6. Install LSCs on Phones on page 13
    - 7. Create and Apply Phone Security Profiles on page 15

  Troubleshooting on page 17
    Verification and Repair Checklist on page 17
    - 1. Verify all certificates on all servers. on page 17
    - 2. Verify CTL contents match current certificates. on page 19
    - 3. Verify CM serves TFTP CTL file on page 22
    - 4. Verify phone properly validates and accepts the CTL file. on page 25
    - 5. Verify CTL contents on phone on page 27
    - 6. Verify SSL connection between phone and CAPF service if using LSCs on page 28
    - 7. Verify SSL connection between phone and CM server for registration on page 32
    - 8. Verify Encrypted Calls on page 35

Purpose

The purpose of this document is to act as a supplement to the official Communications Manager Security Guide by providing examples, explanation, and diagrams for Phone Security using Certificate Trust Lists.

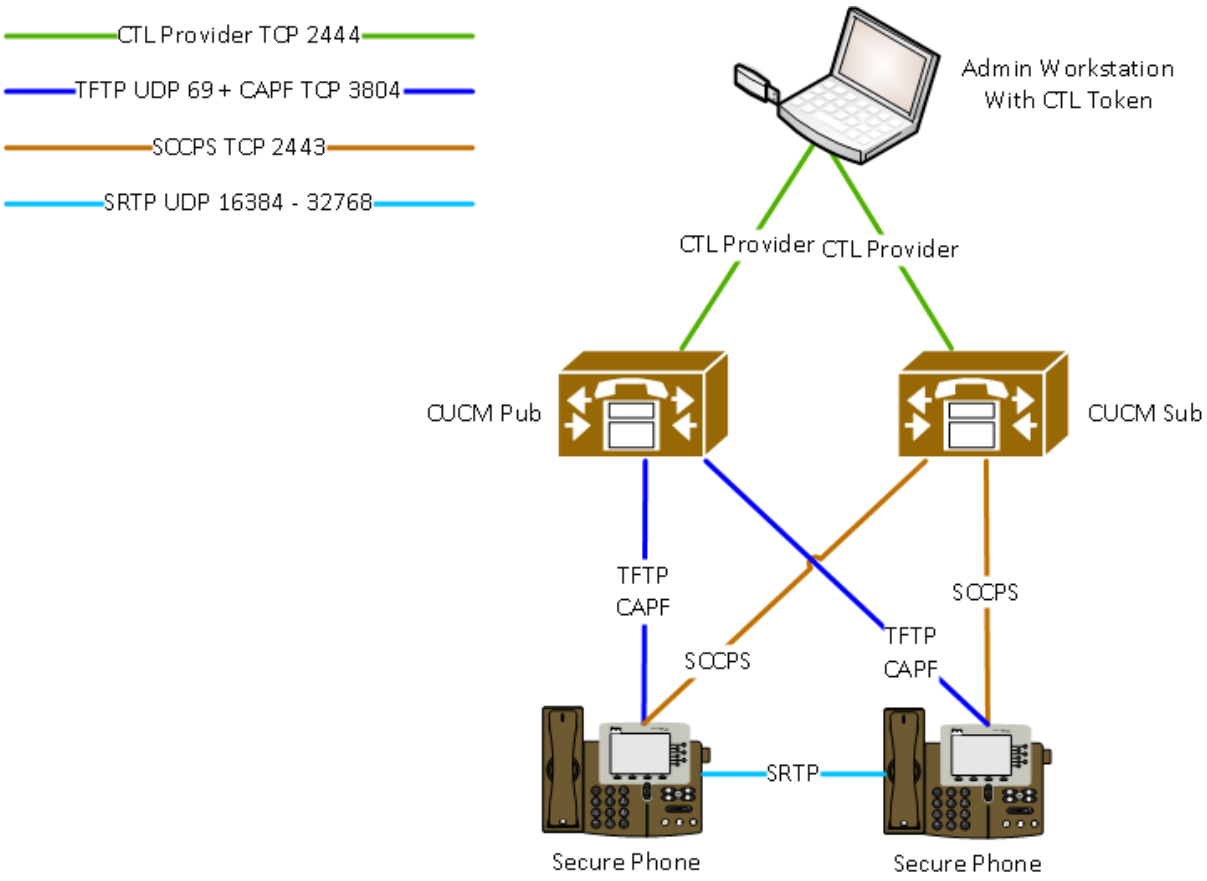
Phone Security and CTL Overview

Phone Security with CTL provides the following functions:

1. Authentication of TFTP downloaded files (configuration, locale, ringlist, etc) using a signing key.

2. Encryption of TFTP configuration files using a signing key.
3. Encrypted call signaling for IP Phones.
4. Encrypted call audio (media) for IP Phones.

Note that the first two functions can also be provided by Security By Default using ITL. The second functions of encrypted signaling and media can only be provided by using CTL files. Refer to the Security By Default document for more information on Authenticated and Encrypted configuration files.



Configuration

1. Obtain USB eTokens

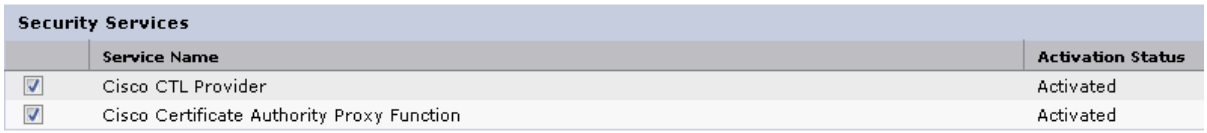
At least two USB eTokens are required for turning on Phone Security. These tokens are the key to signing the CTL file, and must not be lost. Multiple tokens can be used in a CTL file for redundancy since they are so important. They should be stored in secure, separate locations with their current passwords also stored safely.

In case a single token is lost or destroyed, the other tokens used at the initial signing of the CTL file can be used instead.

A token will self destruct after 15 failed password attempts, so remembering the token password and having backup tokens is extremely important.

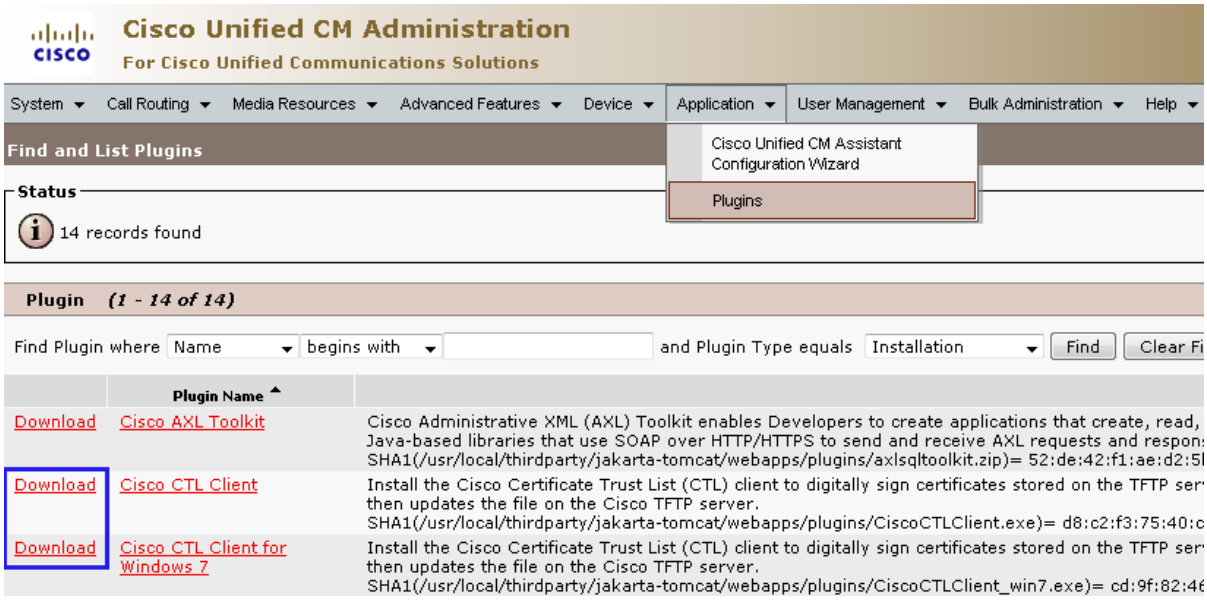
2. Activate CTL Provider and CAPF Services

CTL Provider accepts connections from the CTL Client to generate the CTL File and collect certificates from all nodes. The CAPF (Certificate Authority Proxy Function) service is responsible for signing and storing LSCs (Locally Significant Certificates) from phones.

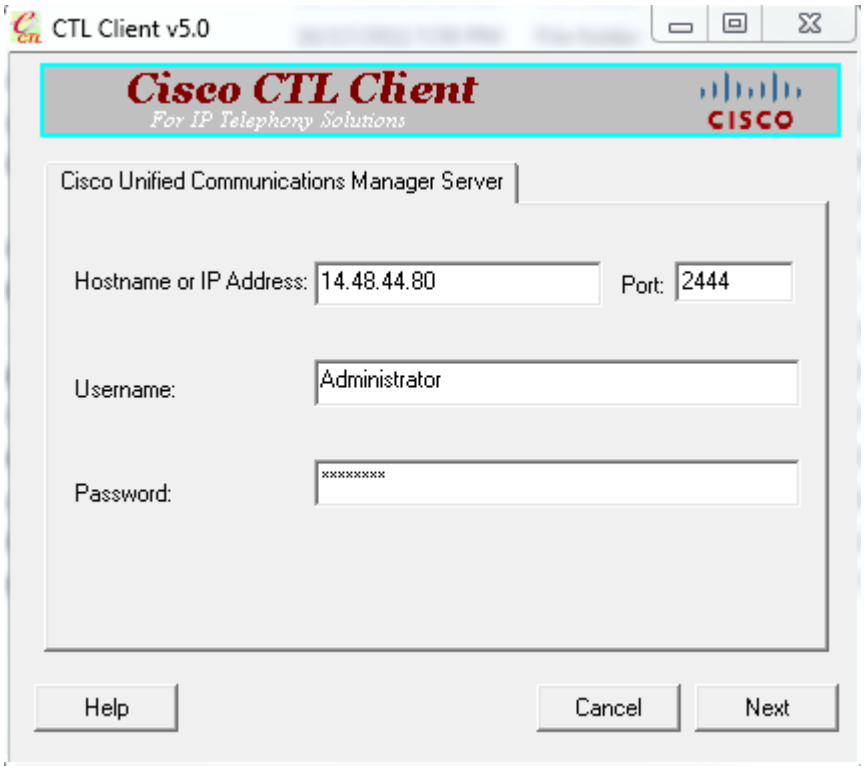


3. Download and Install the CTL Client

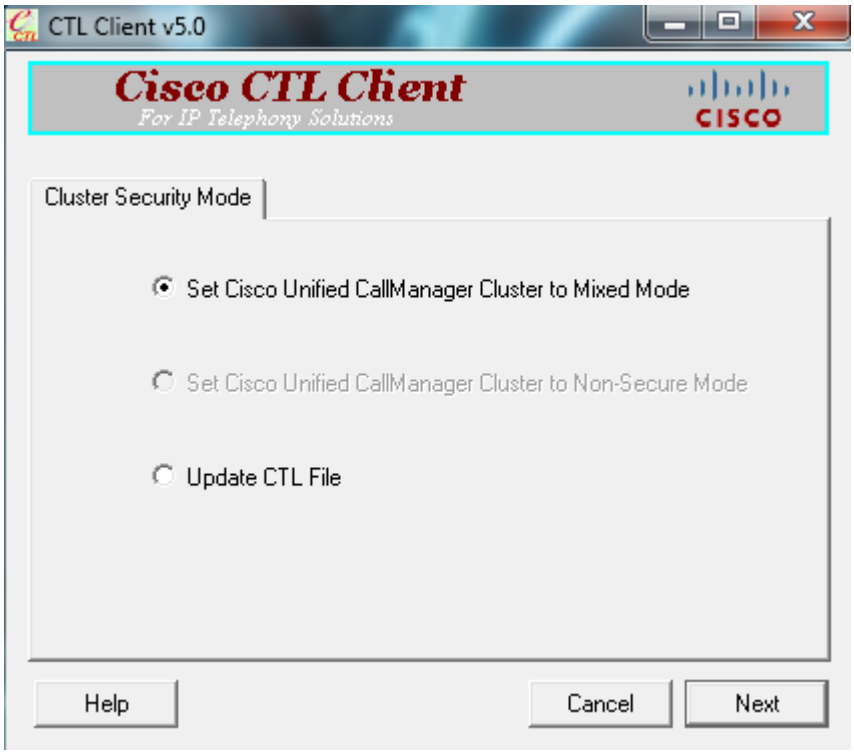
Starting in CUCM 8.6 Windows 7 is finally supported with the CTL Client. Make sure to download the correct CTL Client for the OS in use on the client PC.



4. Run the CTL Client using eTokens



The CTL Client will present the following options for a brand new install:



Set Cisco Unified CallManager Cluster to Mixed Mode:

This turns off auto registration and creates a CTL file.

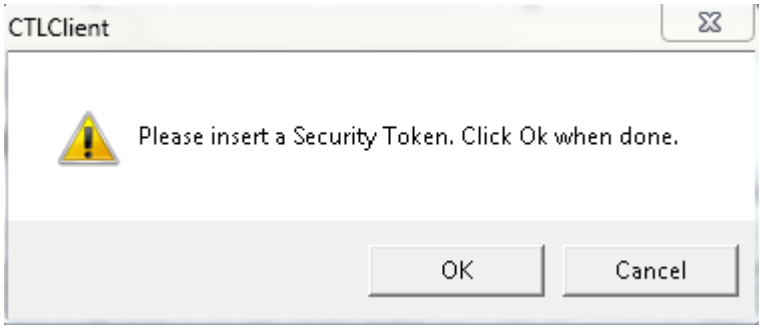
Set Cisco Unified CallManager Cluster to Non-Secure Mode:

This allows auto registration to be enabled and leaves any existing CTL file in place. This is the default mode so cannot be selected unless the cluster is already in Mixed Mode.

Update CTL File:

This allows any new certificates or servers to be added to the CTL file.

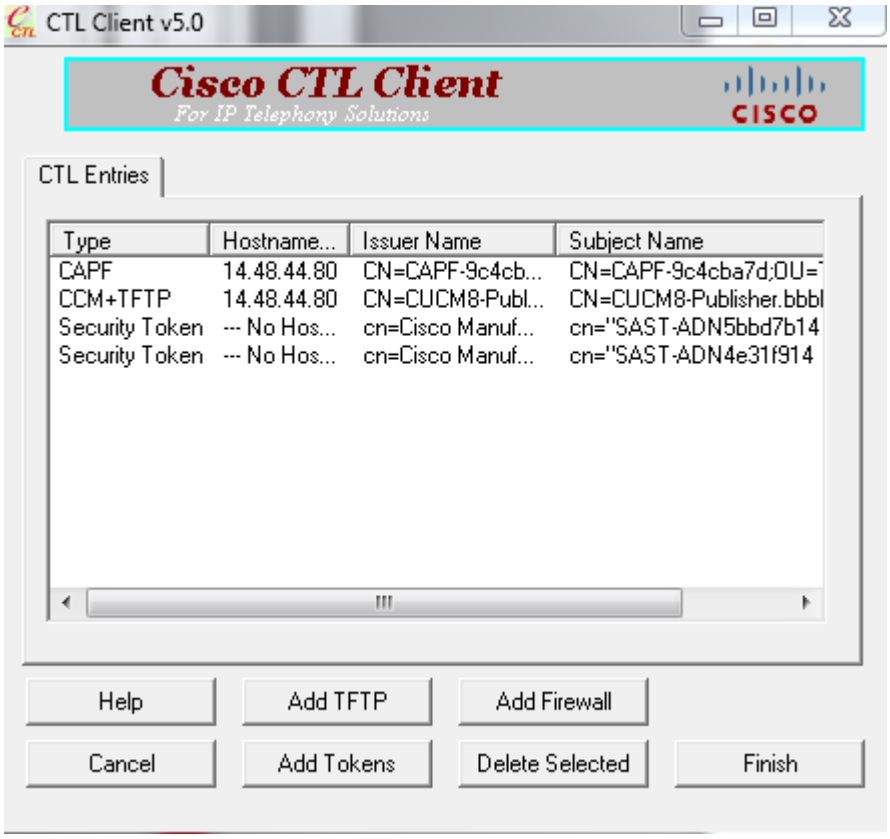
Choosing any one of these options will require a USB eToken to be inserted in the client PC:



Once inserted, information about the token is displayed:

At this point the CTL Client performs connections to all CUCM servers in the cluster on TCP port 2444 to retrieve existing CallManager and CAPF certificates. This requires proper name resolution if using host names under "CCMAdministration > System > Server".

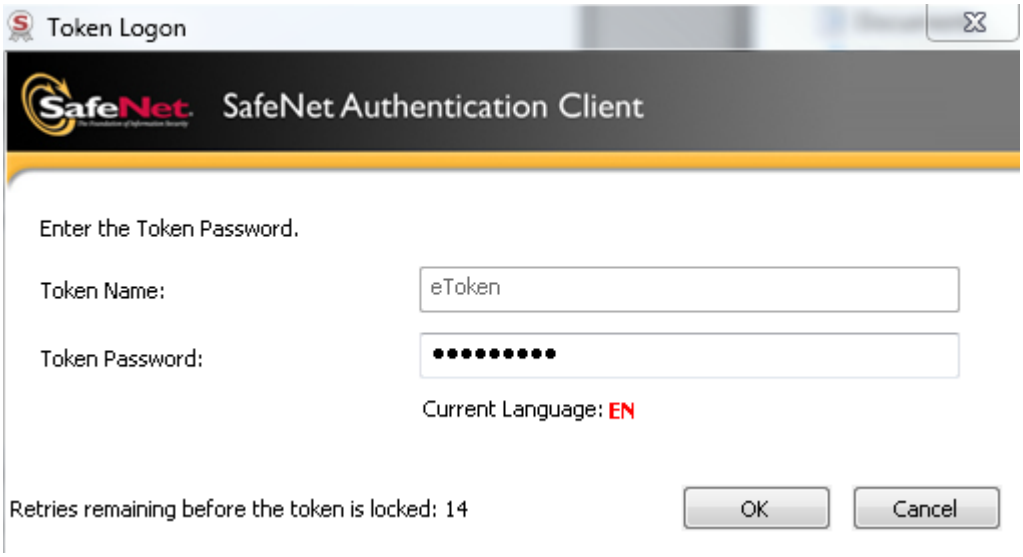
The list of all servers and certificates is displayed, along with all tokes in the existing CTL file.



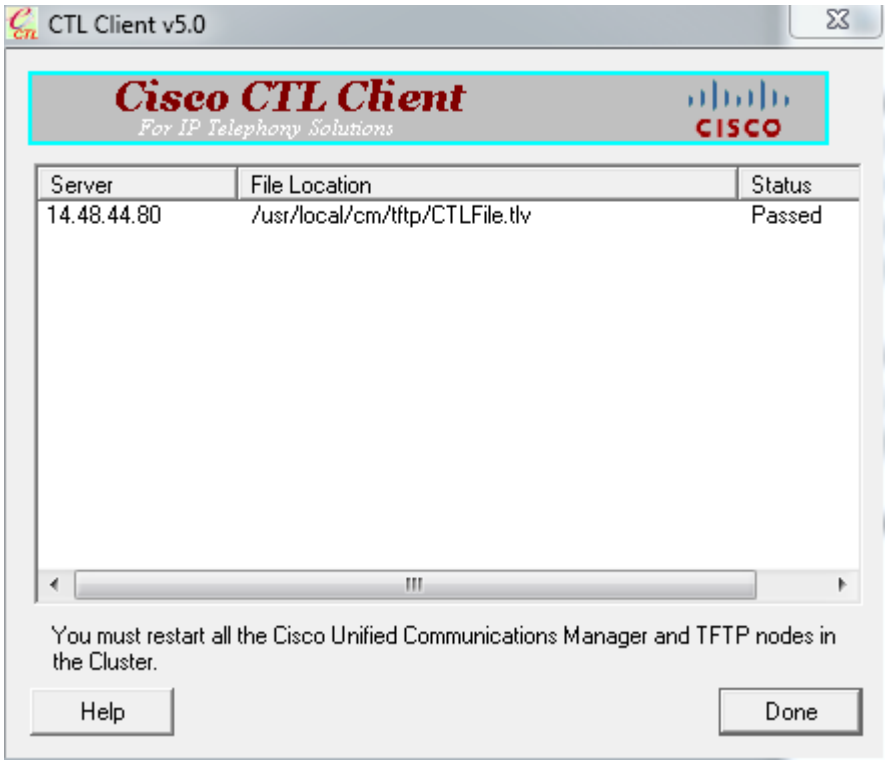
If only one eToken is displayed, the "Add Tokens" option must be used to add another token before the cluster can be set to Mixed Mode.

Once Finish is selected, the CTL Client will ask for the private key password of the USB eToken. This allows the eToken to be used to sign the newly created CTL file, which contains all of the certs and tokens displayed above.

Note here that the password has been incorrectly entered once. The eToken software warns that only 14 more attempts are allowed before the token is permanently locked (destroyed). A successful password entry resets this counter back to 15.



If the correct password is entered, the CTL Client unlocks the private key from the eToken and uses it to sign the CTL File. This newly signed CTL File then gets written to every server on the cluster using another connection to the CTL Provider on TCP 2444. Again this requires network connectivity and name resolution from the CTL Client PC to each server in the cluster.



5. Restart Required Servers

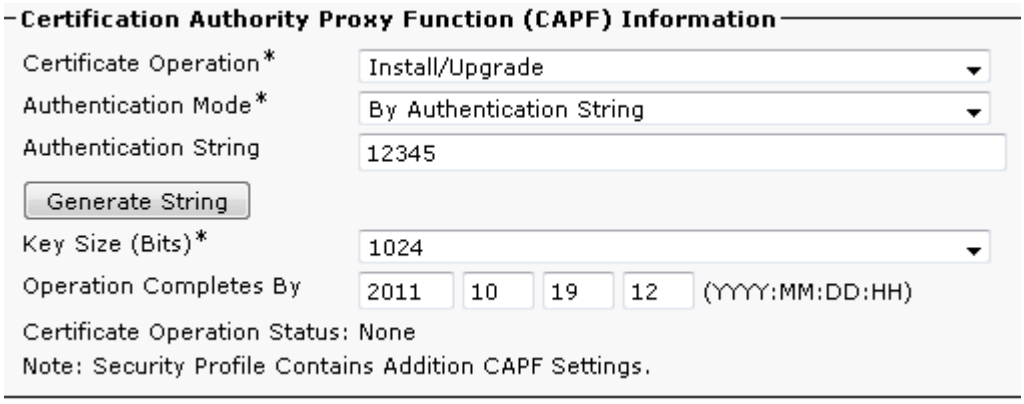
The recommended procedure is to restart all TFTP servers, followed by all servers running the CallManager process. Restarting the TFTP servers allows the TFTP process to load in the newly generated CTLFile.tlv. Restarting the nodes running CallManager causes the phones to reset and download the new CTL file from the configured TFTP server.

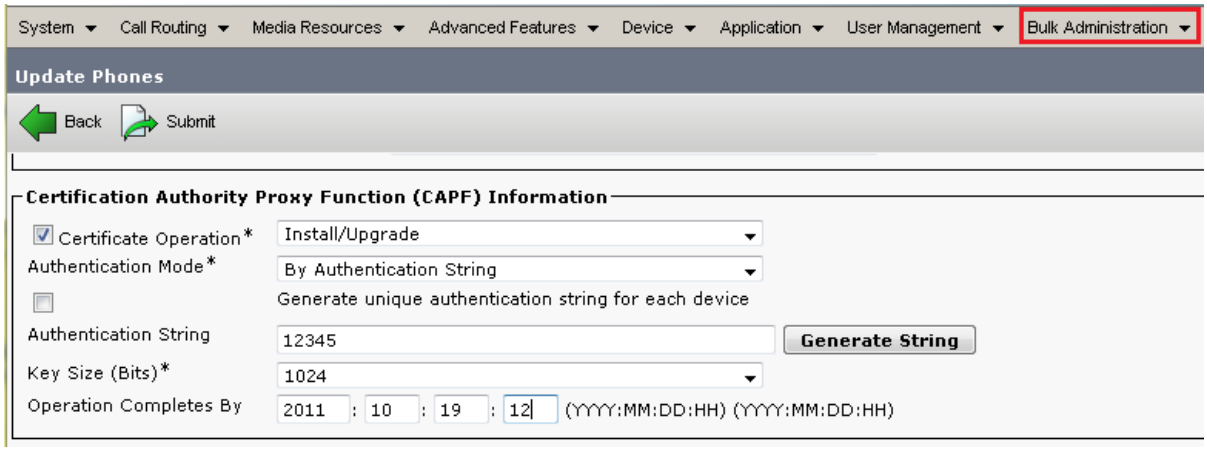
```
admin:utils system restart

Do you really want to restart ?

Enter (yes/no)?
```

6. Install LSCs on Phones

After the CAPF service is activated and the phones obtain the CAPF certificate by downloading the CTL File, phones can connect to CAPF to obtain LSC files.

Set the phone CAPF Certificate Operation to Install Upgrade using Device > Phone, or Bulk Administration Tool > Phones > Update Phones > Query.



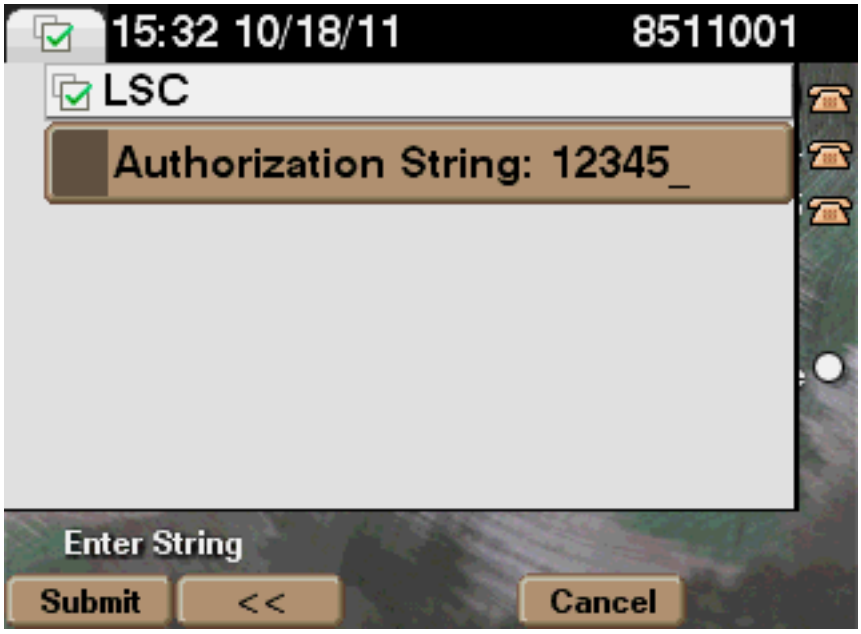


After setting the Certificate Operation, reset the phones.

If Null Sting or Existing Certificate have been chosen as the authentication mode no further action will be required.

If a string was chosen for the Authentication Mode then this will need to be entered manually into the phone console.

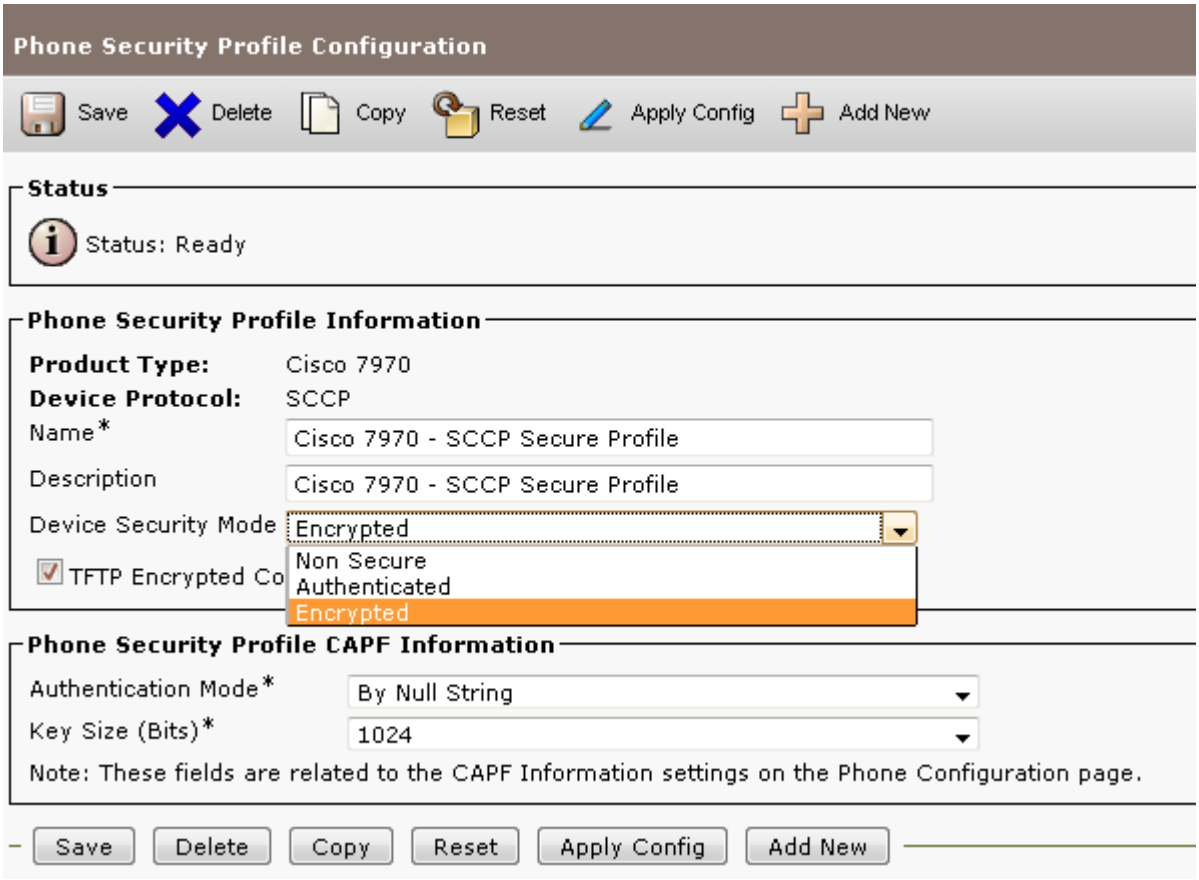
Settings > Security Configuration > **# > LSC > Update



7. Create and Apply Phone Security Profiles

Now that all of the underlying pieces are in place, phones can have security enabled via the Phone Security Profile. These profiles are specific to the model of phone being configured. A profile will need to be created for each model of phone in use.

CCMAdministration > System > Security > Phone Security Profile



Device Security Mode controls the primary phone security settings with the following options:

Non Secure - unencrypted signaling and unencrypted media (voice / RTP / Real Time Protocol)

Authenticated - encrypted signaling and unencrypted media

Encrypted - encrypted signaling and encrypted media

The separate checkbox for TFTP Encrypted Config controls whether or not the CUCM server sends an encrypted TFTP configuration file to the phone. The encryption of the TFTP file is independent of the Device Security Mode settings, but an encrypted config file is recommended on phones that support it.

The Security Profile needs to be applied at the Device level, so the Bulk Administration Tool is the most appropriate method to apply this profile to a larger number of phones.

Troubleshooting

Adding Phone Security to a CM cluster brings an additional layer that must be considered when planning and performing administrative tasks. Items such a certificates and certificate expiration dates should be taken into consideration. Certain administrative operations like changing host names may require regenerating certificates and CTL files.

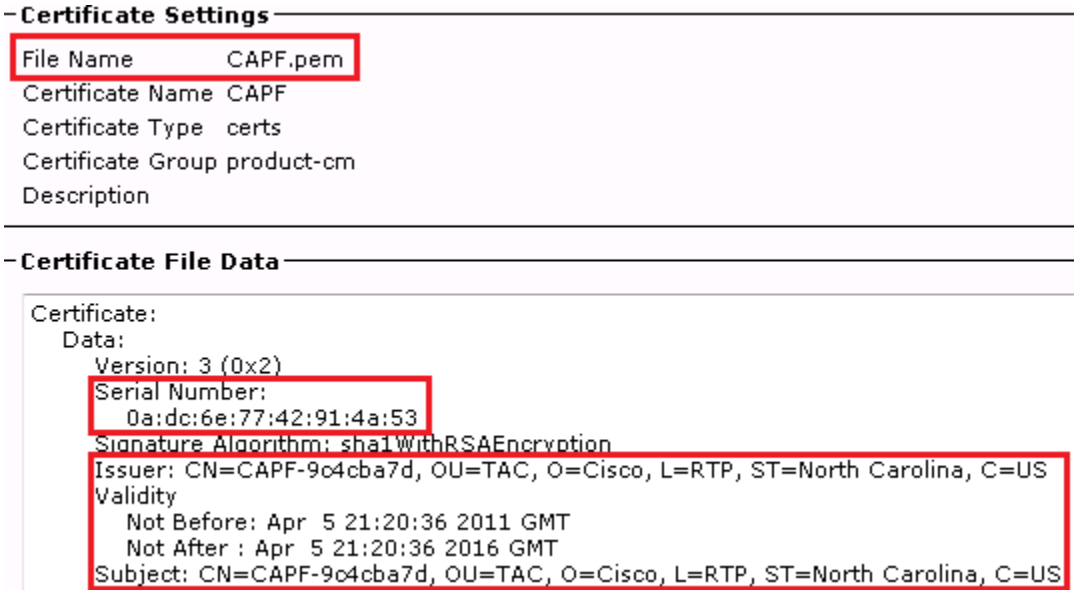
The troubleshooting section here supplements the official Troubleshooting Guide and will provide steps to identify the current state of a cluster and recommend any corrective action necessary.

Verification and Repair Checklist

1. Verify all certificates on all servers.

Collect serial numbers, Common Names, and expiration dates of current CAPF.pem and CallManager.pem certificates on all servers. The certificates loaded onto the CM servers are extremely important. Any mismatch in certificates on the servers could cause phone LSC download failures, configuration file authentication failures, or phone registration failures.

Here is the CAPF.pem certificate. Note the easily identifiable random string in the Common Name. This comes in handy as a quick verification tool. The CAPF.pem is used to sign LSCs (Locally Significant Certificates) and for the SSL handshake between the phone and the CAPF process.



This CAPF.pem expires in 2016, and was generated on April 5, 2011. These pieces of information tell us what dates to watch for in the future as well as what operations happened in the past.

This becomes more obvious when the CallManager.pem certificate is shown. Note that the CallManager.pem certificate also expires in 2016, but was generated on August 23rd, 2011. Some certificate regeneration operation must have been performed on the cluster on Aug 23rd. Remember that this certificate is used in the SSL handshake between phones and the CallManager, as well as by the TFTP process to sign files.

2. Verify CTL contents match current certificates.

After checking the certificate contents, the next item to view is the CTL file on all TFTP servers. The OS Administration SSH CLI provides a simple command called "show ctl". The header of the CTL file contains the date the CTL file was last generated, the CN (Common Name) of the USB eToken used to sign the CTL, and the eToken serial number.

Note that the CTL was generated AFTER the CallManager.pem certificate generation date above. This is good because the CTL file should contain the latest version of the CallManager.pem. If the CTL file had a date that was BEFORE the CallManager.pem or CAPF.pem file generation dates, the CTL Client would need to be run again to get the latest certificates.

```
admin:show ctl
Length of CTL file: 4712
The CTL File was last modified on Wed Aug 31 13:28:03 EDT 2011

      Parse CTL File
      ----------------
```

```
Version:       1.2
HeaderLength:  304 (BYTES)

BYTEPOS TAG           LENGTH  VALUE
------- ---           ------  -----
3       SIGNERID      2       117
4       SIGNERNAME    56      cn="SAST-ADN4e31f914       ";ou=IPCBU;o="Cisco Systems
5       SERIALNUMBER  10      BD:A3:02:00:00:00:D8:88:64:1F
6       CANAME        42      cn=Cisco Manufacturing CA;o=Cisco Systems
```

The first entry inside the CTL is the full certificate of the eToken. This eToken with a serial number of "ADN4f31f914" was the eToken used to sign the CTL file. The serial number is printed on the token packaging and on the token itself, so the serial number in the Subject CN (Common Name) can be helpful to match the tokens used during signing.

```
      CTL Record #:1
              ----
BYTEPOS TAG           LENGTH  VALUE
------- ---           ------  -----
1       RECORDLENGTH  2       1178
2       DNSNAME       1
3       SUBJECTNAME   56      cn="SAST-ADN4e31f914       ";ou=IPCBU;o="Cisco Systems
4       FUNCTION      2       System Administrator Security Token
5       ISSUERNAME    42      cn=Cisco Manufacturing CA;o=Cisco Systems
6       SERIALNUMBER  10      BD:A3:02:00:00:00:D8:88:64:1F
7       PUBLICKEY     140
9       CERTIFICATE   894     67 EB 23 F8 F5 16 55 A9 8E C8 CB 8A 4F 9E A2 0A AB 45 B6 E6 (SHA1 Hash HEX)
10      IPADDRESS     4
This etoken was used to sign the CTL file.
```

Multiple tokens can also be included inside the CTL file. At least 2 eTokens must be present in a CTL file. One token will be used for signing, and the other token is simply present as a backup trust point. The phone will trust any CTL file signed by other of these two tokens.

This output shows that the following eToken wasn't used to sign the CTL file, it's just the backup eToken. To update the CTL file at least one of the tokens inside the current CTL file must be found.

```
      CTL Record #:2
              ----
```

```
BYTEPOS TAG            LENGTH  VALUE
------- ---            ------  -----
1       RECORDLENGTH    2       1180
2       DNSNAME         1
3       SUBJECTNAME     56      cn="SAST-ADN5bbd7b14        ";ou=IPCBU;o="Cisco Systems
4       FUNCTION        2       System Administrator Security Token
5       ISSUERNAME      42      cn=Cisco Manufacturing CA;o=Cisco Systems
6       SERIALNUMBER    10      AA:C9:20:00:00:00:78:C4:2E:22
7       PUBLICKEY       141
9       CERTIFICATE     895     A4 A3 8D 11 57 5A B8 E2 60 6E AF 4A 54 0A 20 B8 CA 0B D3 40 (SHA1 Hash HEX)
10      IPADDRESS       4
This etoken was not used to sign the CTL file.
```

The next record after the eTokens is the CallManager.pem certificate (denoted by function CCM+TFTP). This certificate is used by CM to sign configuration files and establish SSL connections between phones and the CM server if a Secure Profile is used on the phone.

Note that the serial number here matches the serial number in the CallManager.pem in the OS Admin page above. If this serial number differed between the two places, the CTL Client would need to be run to bring the CTL file in sync with what CM is actually using for a certificate.

```
       CTL Record #:3
              ----
BYTEPOS TAG            LENGTH  VALUE
------- ---            ------  -----
1       RECORDLENGTH    2       1059
2       DNSNAME         12      14.48.44.80
3       SUBJECTNAME     63      CN=CUCM8-Publisher.bbbburns.lab;OU=AS;O=Cisco;L=RTP;ST=NC;C=US
4       FUNCTION        2       CCM+TFTP
5       ISSUERNAME      63      CN=CUCM8-Publisher.bbbburns.lab;OU=AS;O=Cisco;L=RTP;ST=NC;C=US
6       SERIALNUMBER    8       74:AE:9E:6B:DB:C2:A5:1D
7       PUBLICKEY       140
9       CERTIFICATE     738     00 7A DE F4 25 26 7A FC 5E 02 B4 D2 BB A4 14 42 2B A5 A0 9C (SHA1 Hash HEX)
10      IPADDRESS       4
```

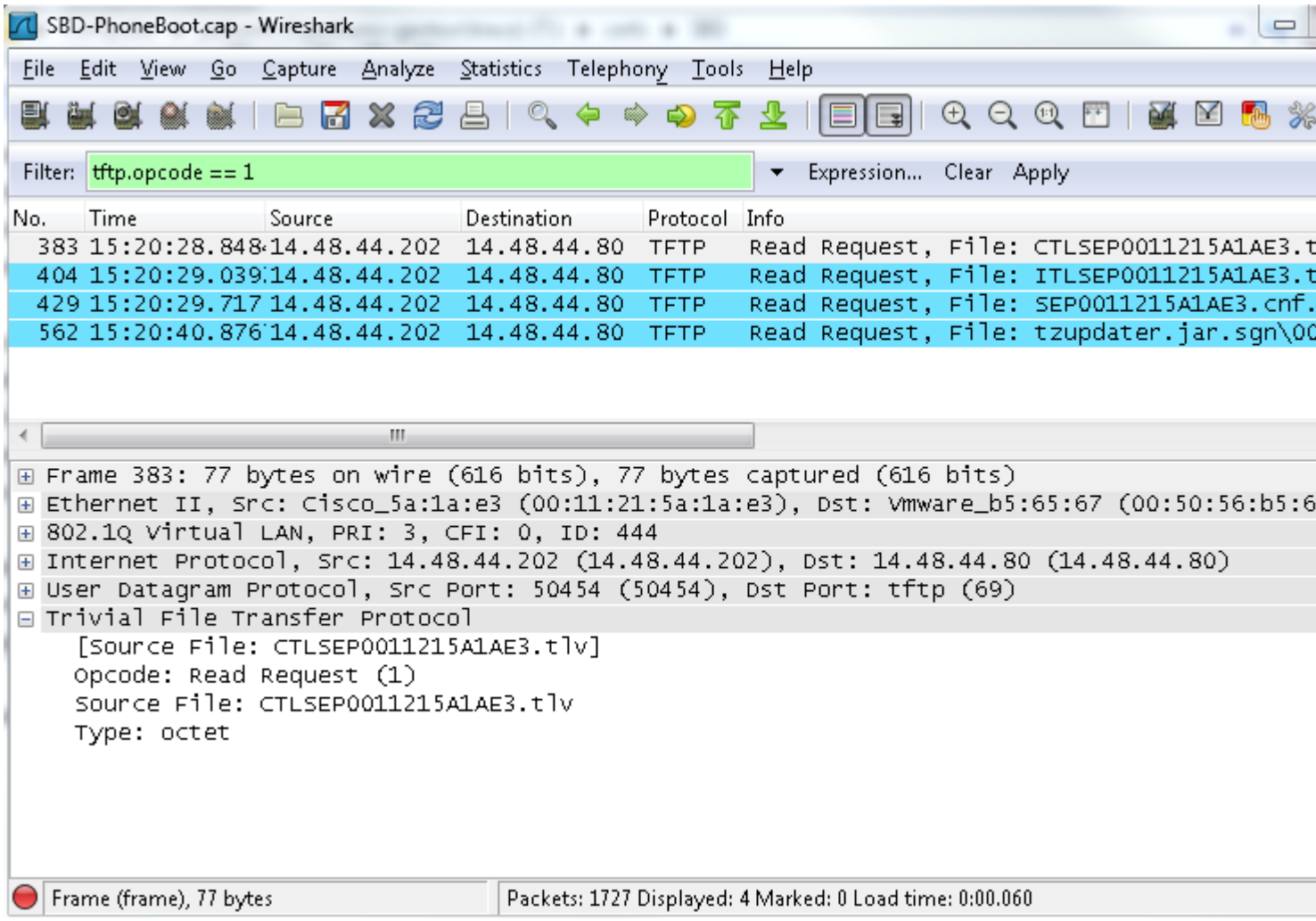
The final entry in the CTL file is the CAPF certificate. The serial number here also must match the OS Admin CAPF.pem, so phones are allowed to connect to the CAPF service. If there is a mismatch the same step of re-running the CTL Client must be performed.

```
      CTL Record #:4
                ----
BYTEPOS TAG            LENGTH  VALUE
------- ---            ------  -----
1       RECORDLENGTH    2       991
2       DNSNAME         12      14.48.44.80
3       SUBJECTNAME     61      CN=CAPF-9c4cba7d;OU=TAC;O=Cisco;L=RTP;ST=North Carolina;C=US
4       FUNCTION        2       CAPF
5       ISSUERNAME      61      CN=CAPF-9c4cba7d;OU=TAC;O=Cisco;L=RTP;ST=North Carolina;C=US
6       SERIALNUMBER    8       0A:DC:6E:77:42:91:4A:53
7       PUBLICKEY       140
9       CERTIFICATE     674     C7 3D EA 77 94 5E 06 14 D2 90 B1 A1 43 7B 69 84 1D 2D 85 2E (SHA1 Hash HEX)
10      IPADDRESS       4

The CTL file was verified successfully.
```

At the completion of this step, the CTL file will be in sync with the certificates loaded onto the CM servers.

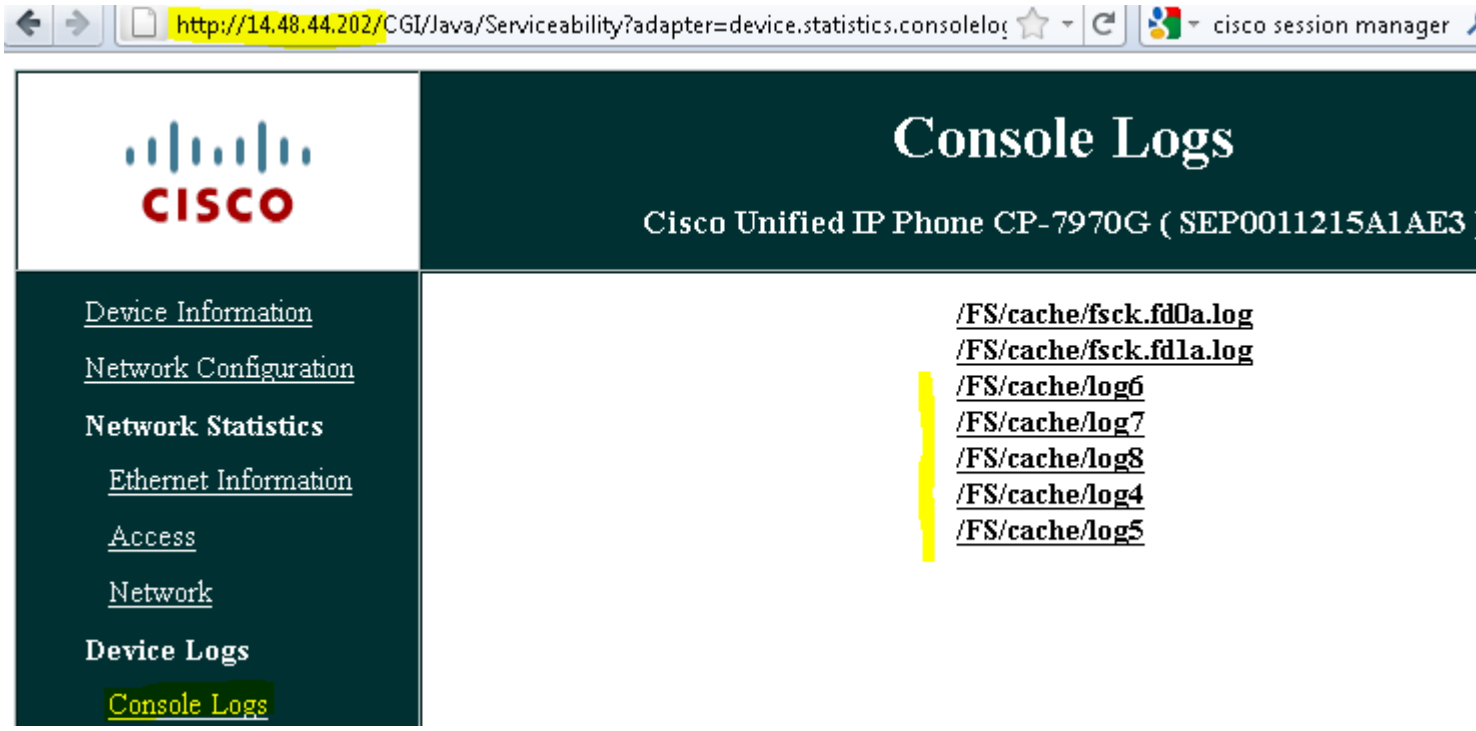
3. Verify CM serves TFTP CTL file

The next item to check in the troubleshooting process is whether or not the CM server is providing a CTL file via TFTP. A quick way is to take a packet capture at the IP Phone or the CM TFTP server. Here the phone requests the CTL file as the first file it downloads at boot.



The phone requested a CTL file, and if the filter on the previous capture is removed the transfer of that file can be viewed in detail.

Another method to verify the CTL file is downloaded is to look at the Phone Console logs under the web page of the phone. This requires the setting "Web Access" under "CCMAdmin > Device > Phone > Product Specific Configuration" to be "Enabled".



Here the console logs show the CTL file was downloaded:

```
837: NOT 09:13:17.561856 SECD: tlRequestFile: Request CTLSEP0011215A1AE3.tlv
846: NOT 09:13:17.670439 TFTP: [27]:Requesting CTLSEP0011215A1AE3.tlv from 14.48.44.80
847: NOT 09:13:17.685264 TFTP: [27]:Finished --> rcvd 4762 bytes
```

In addition to packet captures and phone console logs, the TFTP traces also show TFTP file transfers. Here is a shortcut to view the current TFTP trace in real time as a phone resets.

```
admin:file tail activelog cm/trace/ctftp/sdi recent

11:08:20.766 |TFTPEngine::isReadRequest[0x9950830~188~14.48.44.202~49530], [CTLSEP0011215A1AE3.tlv] opcode(1), Mode
11:08:20.779 |TID[a5a4fba0] TFTPEngine::processMessage[0x9950830~188~14.48.44.202~49530], Transferred[CTLFile.tlv] Socke
```

4. Verify phone properly validates and accepts the CTL file.

After verifying the CUCM server is presenting a valid CTL file, the next step is for the phone to validate that CTL file.

Phone console logs also show that the CTL file signature (the eToken signer) was trusted:

```
877: NOT 09:13:17.925249 SECD: validate_file_envelope: File sign verify SUCCESS; header length <296>
```

Status Messages displayed on the phone can also be helpful to verify a CTL file was downloaded succes

Either in the Status Messages web page of a phone, or under the phone itself "Settings > Status > Status Messages", the following line means a CTL file (or ITL file) has been successfully downloaded and verified:

```
16:01:16 Trust List Updated
```

If the phone could not validate the new CTL file the error message would be

```
Trust List Update Failed
or
Trust List Verification Failed
```

If the phone fails to validate the CTL file, that means the phone's existing Certificate Trust List does not have the same eTokens inside of it that the newly downloaded CTL was signed with, or that the newly downloaded CTL was corrupt.

A corrupt CTL can be checked with "show ctl", looking for the output "The CTL File was verified successfully", or the error condition "Verification of the CTL File Failed". Generally a corrupt CTL file can be repaired by running the CTL Client.

If the phone's old CTL file contains only eTokens that are no longer available, the CTL File will need to be deleted from the phone manually.

At this point the dilemma is "Did the phone download the latest CTL File?". The Status Messages and Phone Console logs can be used for verification, but other methods also exist.

The simplest method for verifying if a number of phones have the correct file is to compare the file sizes of the CTL file on the phone with the file size on the TFTP server.

First, create an SSH username and password for the IP Phone under CCMAdministration and enable SSH on the phone. Reset the phone.

Next SSH to the phone with the configured username and password. When prompted for the second login use "default / user". This example discusses 7961 and similar model phones. Use the following debug guide for 89XX and 99XX model phones.

From the phone

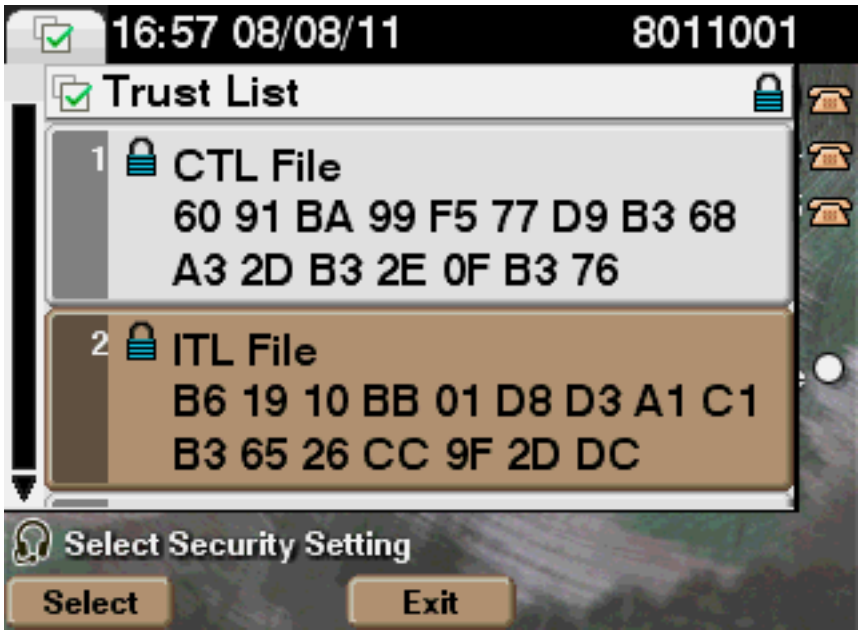
```
$ ls -l /tmp/*.tlv
-rw-r--r--  1 default  usr        4712 Oct 04 19:15 /tmp/CTLFile.tlv
-rw-r--r--  1 default  usr        3899 Oct 04 19:15 /tmp/ITLFile.tlv
```

From the TFTP Server

```
admin:file list tftp *.tlv detail
31 Aug,2011 13:28:03       4,712  CTLFile.tlv
16 Sep,2011 11:15:45       3,899  ITLFile.tlv
```

That method is a close approximation based on the size of the file. For an exact comparison of the contents, first look at the IP Phone to view the md5sum of the current CTL and ITL files

Settings > Security Settings > Trust List



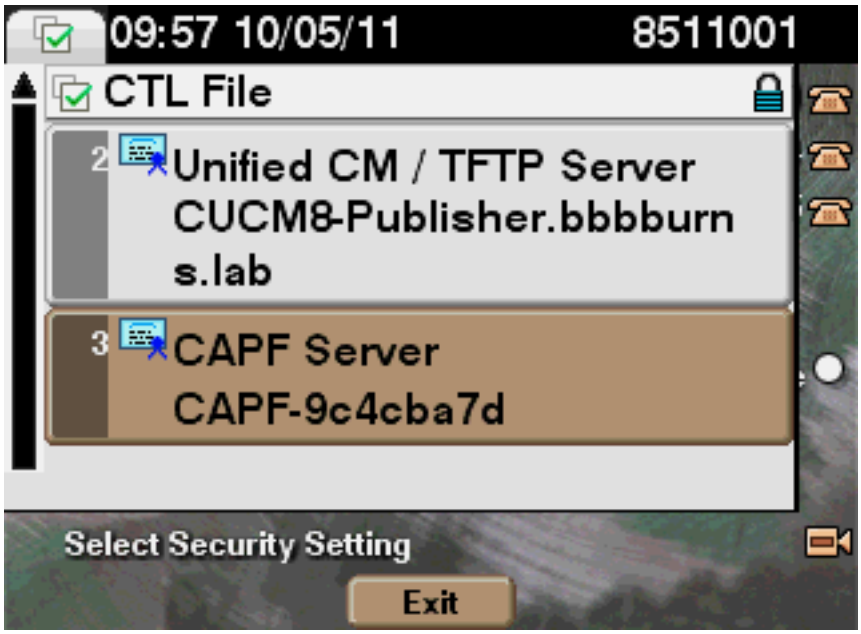
In 8.6(2) and later versions of Communications Manager this hash should be visible on the server with "show ctl" and "show itl". Prior to 8.6(2), use TFTP and md5sum to verify the hash of this file as it exists on the server. This example checks the hash of the ITL file. Just replace ITL with CTL and the example will work for both files.

```
jasburns@jasburns-gentoo /data/trace/jasburns/certs/SBD $ tftp 14.48.44.80
tftp> get ITLSEP0011215A1AE3.tlv
Received 5438 bytes in 0.0 seconds
tftp> quit
jasburns@jasburns-gentoo /data/trace/jasburns/certs/SBD $ md5sum ITLSEP0011215A1AE3.tlv
b61910bb01d8d3a1c1b36526cc9f2ddc  ITLSEP0011215A1AE3.tlv
```

5. Verify CTL contents on phone

A shortcut to verifying that the CTL file on the phone matches exactly byte for byte with the file on the server is just to quickly look at the phone's Trust List.

Settings > Security Settings > Trust List > CTL File



The name of the CM / TFTP server does match with the name of this server's CallManager.pem file.

The CAPF-<random string> also matches the CAPF.pem certificate that is currently in use.

6. Verify SSL connection between phone and CAPF service if using LSCs

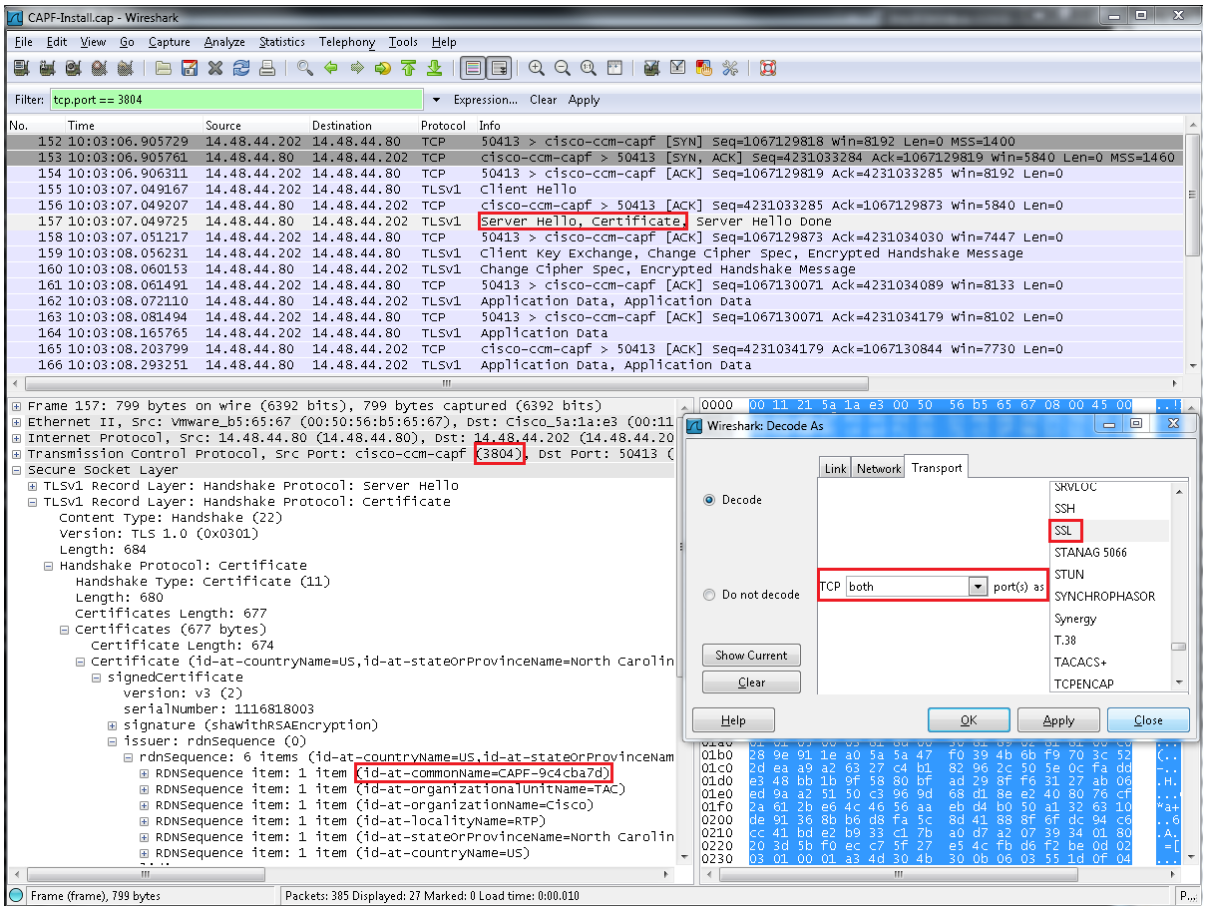
Now that the phone has a CAPF (Certificate Authority Proxy Function) certificate via the CTL, the phone can connect to CAPF to download a certificate.

A packet capture on the CM server can be used to verify the CAPF SSL handshake completed. Here the filter captures all traffic from the IP of the phone. Then the file is uploaded to another server using SFTP and the "file get" command.

```
admin::utils network capture host ip 14.48.44.202 size ALL count 10000 file CAPF-Install

admin:file get activelog platform/cli/CAPF-Install.cap
```

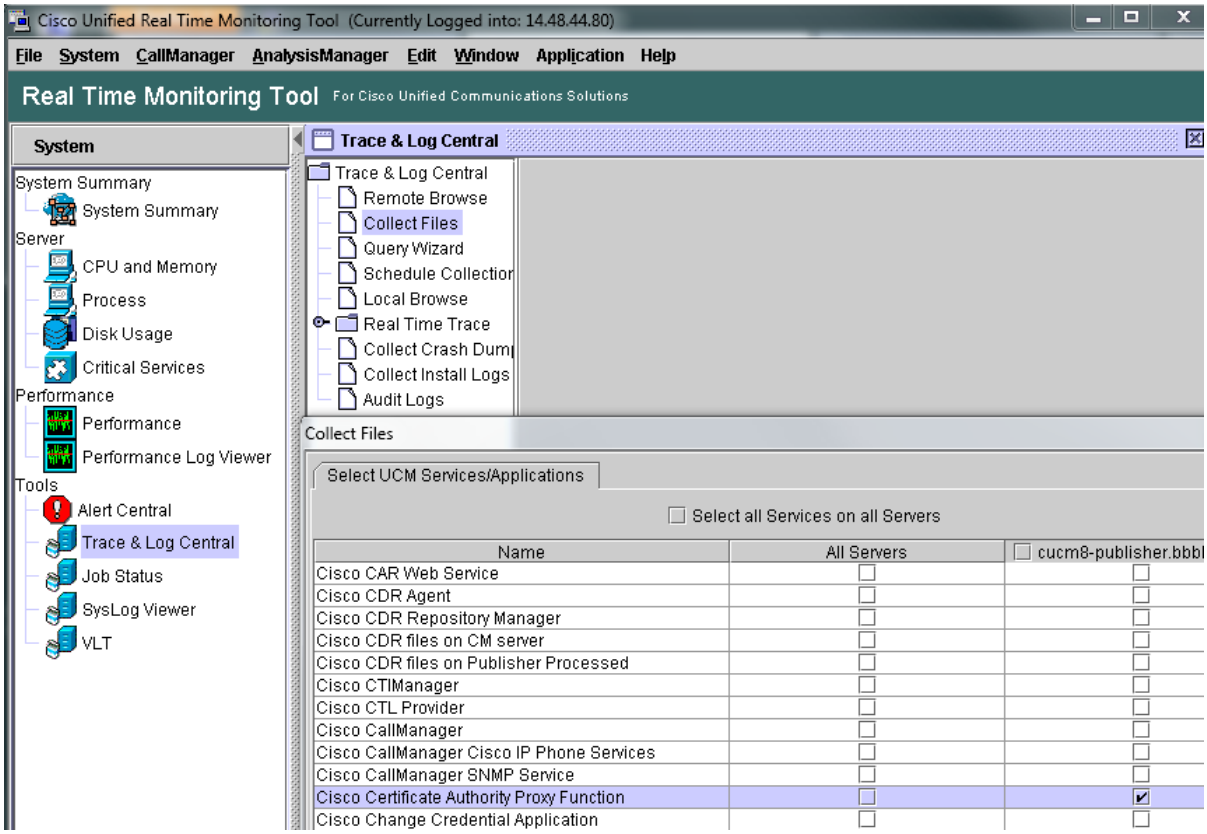
The packet capture shows the SSL handshake that happens on the CAPF port, TCP 3804. To view this exchange, right click on any packet in the TCP port 3804 stream and go to "Decode As".

Here the certificate presented by the CAPF server matches the certificate in the CTL, and the certificate the phone displayed earlier. This SSL handshake succeeded because it started sending "Application Data", which would be the CAPF exchange.



Subsequently, the phone has an LSC installed:

If that process had failed despite the SSL handshake success, the next spot to examine would be the CAPF traces. If the SSL handshake failed, it would be time to check the CAPF certificate and update the CTL file again with the CTL Client.



The CAPF traces show that the phone connects, generates a key (which takes some time as seen by the gap in traces), then the CAPF server generates a certificate for the phone.

```
10:03:06.983 |   debug 3:UNKNOWN:Got a new ph conn 14.48.44.202 on 15
10:03:08.060 |   debug TLS HS Done for ph_conn .
10:03:08.065 |   debug     MsgType             : CAPF_MSG_AUTH_REQ
10:03:08.341 |   debug     MsgType             : CAPF_MSG_REQ_IN_PROGRESS
10:03:08.341 |   debug 3:SEP0011215A1AE3:CAPF CORE: Rcvd Event: CAPF_EV_REQUEST_IN_PROGRESS in State: CAPF_S
10:03:21.148 |   debug 3:SEP0011215A1AE3:Incoming Phone Msg:
10:03:21.158 |   debug     MsgType             : CAPF_MSG_KEY_GEN_RES
10:03:21.162 |   debug Generated the cert
10:03:21.724 |   debug 3:SEP0011215A1AE3:Certificate upgrade successful
```

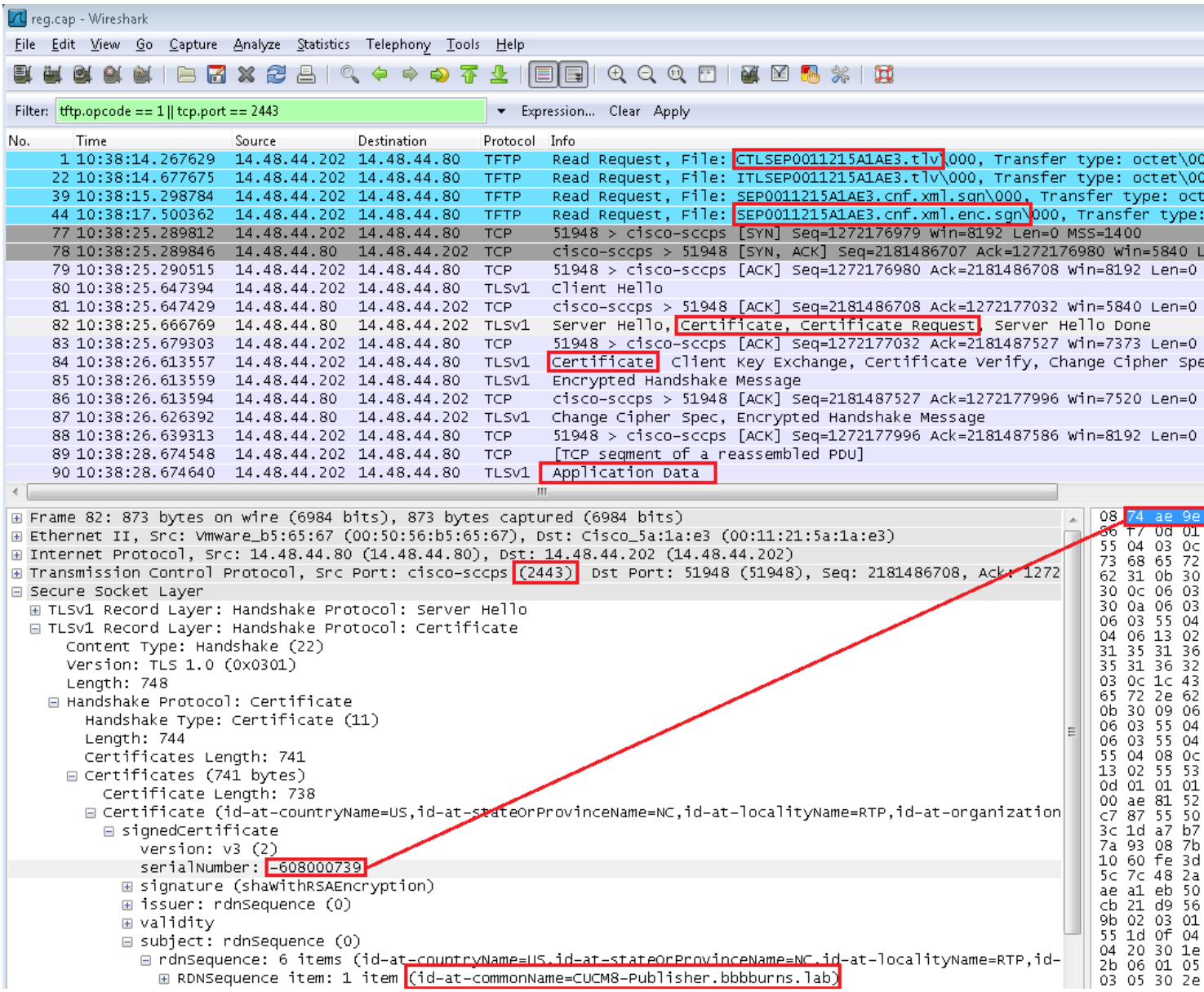
7. Verify SSL connection between phone and CM server for registration

Now that the phone has a CTL and LSC, the next step is secure phone registration. For SCCP phones this happens on TCP Port 2443.

Use the same steps as before to capture all packets from this specific phone.

The first thing that's different is the phone downloading SEP<MAC Address>.cnf.xml.enc.sgn. This signifies and encrypted TFTP configuration file as set under the Device Security Profile.

Here the phone connects on TCP port 2443, so this port must be Decoded As SSL in Wireshark. The CUCM presents the CallManager.pem certificate (verifiable by serial number and common name) and then asks for the certificate of the phone. As before the SSL handshake completed successfully since the Application Data phase is reached.



In addition to the packet capture, phone registration via Secure SCCP is also visible in the Cisco CallManager SDI traces:

```
10:38:26.621 |SdlSSLTCPListener::verify_cb pre-verified=1,cert verification errno=0,depth=0
10:38:26.626 |New connection accepted. DeviceName=, TCPPid = [1.100.13.7], IPAddr=14.48.44.202, Port=51948,
10:38:29.051 |StationD:    (0000048) ClusterSecurityMode = (1) DeviceSecurityMode = (3)
10:38:29.051 |StationD:    (0000048) TLS Connection Cipher - INFO:deviceName=SEP0011215A1AE3, Cipher=AES128-SHA, Secu
```

8. Verify Encrypted Calls

Calls that have both encrypted signaling and encrypted media will show the lock icon in the lower right corner of the call window. This corresponds to Device Security Mode: Encrypted.

Calls that use encrypted signaling between both ends, but that do not use encrypted media will show the shield icon. This corresponds to Device Security Mode: Authenticated.

The security status of the least secure party is used to determine which icon is displayed and what call security is used. Take a look at the following table for examples:

Phone A	Phone B	Icon Displayed
Encrypted	Encrypted	Lock - Encrypted Audio
Encrypted	Authenticated	Shield - Unencrypted Audio
Encrypted	None	None - Unencrypted Audio