A MITEL
PRODUCT
GUIDE

OpenScape Solution Set V11

Certificate Management and Transport Layer Security (TLS)

Administrator Documentation
07/2025

A31003-S11A0-M100-01-76A9

Mitel

Notices

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Europe Limited. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes. No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

Trademarks

The trademarks, service marks, logos, and graphics (collectively “Trademarks”) appearing on Mitel’s Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively “Mitel), Unify Software and Solutions GmbH & Co. KG or its affiliates (collectively “Unify”) or others. Use of the Trademarks is prohibited without the express consent from Mitel and/or Unify. Please contact our legal department at iplegal@mitel.com for additional information. For a list of the worldwide Mitel and Unify registered trademarks, please refer to the website: http://www.mitel.com/trademarks.

Contents

Contents

History of Changes

Date	Changes	Reason
13-12-2019	Initialization V10	
15-01-2020	Updated Chapter 3.3.9 Regeneration of UC certificates in default keystore	UCBE-21926
22-01-2020	Updated Chapter 3.19 OpenScape Contact Center	OSCCSU-2860
07-02-2020	Updated Chapter 3.3.9 Regeneration of UC certificates in default keystore	UCBE-20265
24-03-2020	Updated Chapter 3.4.5 Replacing the default certificate in the MS Tomcat keystore	OSV-16867

1 Introduction

This guide describes how to properly install and maintain certificates within an OpenScape solution. This includes the following products:

- OpenScape Voice
- OpenScape UC Applications
- OpenScape Branch
- OpenScape SBC
- OpenScape Deployment Service
- OpenStage, Optipoint, and Soft-Client Phones
- OpenScape Xpressions
- OpenScape Trace Manager
- OpenScape Web Collaboration
- OpenScape Mobile
- OpenScape 4000/HiPath 4000/RG83xx
- RG8700
- Mediatrix
- Attendant Supervisor Console

Each of these products contains its own documentation for certificate management. However, each product does not provide a solution overview. This document intends to provide a single comprehensive overview of how certificates work in Unify solutions. It will detail why our products need certificates, how they are related to security, and what is required to replace them with customer specific versions.

NOTICE: This guide may be applied to version 4, 5, 6, 7, 8 and 9 OpenScape solutions.

1.1 Acronyms and Glossary of Terms

Acronyms and Terms	Definition
Asymmetric encryption	A type of encryption comprised of two mathematically linked keys, a public and private key – a key pair. The public key can be used to encrypt data which only the private key can decrypt. Also known as public key cryptography. In the scope of this document the public key is part of the certificate.
CA	Certificate Authority – “An authority trusted by one or more users to create and assign public-key certificates. Optionally the certification authority may create the users' keys.”[1] See also definitions for root and subordinate CA.
Certificate (X.509)	“[D]ata structures that bind public key values to subjects. The binding is asserted by having a trusted CA digitally sign each certificate. The CA may base this assertion upon technical means (a.k.a., proof of possession through a challenge-response protocol), presentation of the private key, or on an assertion by the subject.”[2] Subject information typically contains identifying information such as country, state, locality, email address, and etc. Also referred to as X.509 certificates.
Certificate Validation	“The process of ensuring that a certificate was valid at a given time, including possibly the construction and processing of a certification path, and ensuring that all certificates in that path were valid (i.e., were not expired or revoked) at that given time.” [1] Also referred to as certificate chaining.
Common Name (CN)	A field within the subject and issuer name sections of a certificate. This field typically contains basic identification information, such as the name of the issuing CA, a person’s name, company name, an IP address or DNS name. Commonly a FQDN or DNS name is placed in this field, but is not required.
CRL	Certificate Revocation List – “A signed list indicating a set of certificates that are no longer considered valid by the certificate issuer.” [1] This list is produced by the certificate authority responsible for the certificates.
DER	Distinguished Encoding Rules – A binary format for storing a public key certificate and its associated attributes and values. A single DER file can only contain one certificate.
Distinguished Name	A unique identifying string used to define some entity. Distinguished names are defined by the X.500 standard (not discussed here). Examples include the subject and issuer fields in an X.509 certificate.

Acronyms and Terms	Definition
DNS	Domain Name System – A system that resolves domain names – such as www.example.com – to IP addresses.
FQDN	Fully Qualified Domain Name – E.g. www.example.com
Issuer Name	“The issuer field [within a certificate] identifies the entity that has signed and issued the certificate. The issuer field MUST contain a non-empty distinguished name (DN).” [2] Issuer information typically contains identifying information such as country, state, locality, email address, and etc.
JKS	Java Key Store – A Java standards data type than can hold and manage private keys and their associated X.509 certificates as well as trusted CA certificates.
Key pair	Refers to the associated public and private key defined by asymmetric encryption and public key cryptography.
Keystore	A generic term for data structures that can hold public and private keys. Examples include the Java key store and PKCS#12 format.
MTLS	Mutual Transport Layer Security – A form of TLS where the transmitting party requests the client’s credentials. If both parties can establish trust in the other then the connection is called mutually authenticated – also referred to as client authenticated. This adds another layer of security by also authenticating the client party.
PEM	Privacy-Enhanced Mail - A character based format for storing a public key certificate and its associated attributes and values. A single PEM file may contain one or more certificates.
PFX	Another name for the PKCS#12 format. See PKCS#12. The PFX file extension is used in Microsoft operating systems. [3]
PKCS#12	Personal Information Exchange Syntax – A standard defined by RSA Laboratories for storing private keys and their associated certificates. [4] Often referred to as a keystore due to its ability to store more than one key and certificate pair. May also contain CA certificates.
PKI	Public Key Infrastructure – “The infrastructure able to support the management of public keys[,] […] authentication, encryption, integrity or non-repudiation services.” [1] Includes “hardware, software, people, policies, and procedures needed to create, manage, distribute, use, store, and revoke digital certificates.” [5]

Acronyms and Terms	Definition
Private Key	The secret key used in asymmetric encryption and public key cryptography. This key is used to decrypt information encrypted with its associated public key. The secret key should never be shared.
Public Key	The publicly known key used in asymmetric encryption and public key cryptography. This key is used to encrypt information. Only its associated private key can decrypt this information.
Root CA	The top level CA in the CA hierarchy. The certificate of a root CA is self-issued and self-signed. Meaning, it has the same issuer and subject name. Also known as a trust anchor.
Self-signed/issued Certificate	A certificate signed using its own private key. In practice this is reserved for root CA certificates at the top of the certificate hierarchy. Root CA certificates are self-issued, meaning they have the same issuer and subject name. Note: Many products ship with self-signed self-issued certificates. These are not root CA certificates. They simply allow a product to work out-of-the-box. These certificates should not be trusted.
SSL	Secure Sockets Layer – Predecessor to TLS. See TLS.
Subject Alternative Name (SAN)	“The subject alternative name extension allows identities to be bound to the subject of the certificate. These identities may be included in addition to or in place of the identity in the subject [name] field of the certificate.” [2] Typical identities include DNS, URI names, or IP addresses. Example, a URI of https://www.example.com or a DNS of www.example.com. There is no defined limit on the number of entries in the subject alternative name field.
Subject Name	“The subject field identifies the entity associated with the public key stored in the subject public key field. The subject name MAY be carried in the subject field and/or the subjectAltName extension.” [2] Subject information typically contains identifying information such as country, state, locality, email address, and etc. The subject name field must be unique for certificates signed by the same CA.
Subordinate CA	Also known as an intermediate CA. A certificate authority granted certificate authority privileges by a root or other subordinate CA. A subordinate CA can perform the same functions as any other CA. It does not have any authority over CAs or certificates above its hierarchy level.
Symmetric Encryption	A type of encryption comprised of a single key, typically just a password. Any user who knows the password can decrypt the data.

Acronyms and Terms	Definition
TLS	Transport Layer Security – A protocol “to provide privacy and data integrity between two communicating applications.” [6] Makes use of X.509 certificates to identify the transmitting party and negotiate a common shared secret for connection encryption. Supersedes SSL.
Trust Anchor	The final authoritative entity in the certificate trust hierarchy. Also know as a root CA.
Truststore	A type of keystore that only contains CA or other trusted certificates. [7] A program may reference a truststore to find a CA certificate to verify an end entity certificate. File format is typically PKCS#12 or JKS.
X.509	A standards based framework initially put forth by the ITU-T that “defines a framework for public-key certificates and attribute certificates.” [1] This framework is further built on by the IETF standards body. [8]

2 Overview

This section explains the relationship between transport layer security (TLS) and certificates. It will explain why new certificates and proper certificate management are required to guarantee security within the OpenScape solution.

2.1 Transport Layer Security (TLS) Overview

TLS is an industry standard transport layer security protocol “designed to prevent eavesdropping, tampering, or message forgery”.[6] It replaces SSLv3 and SSlv2 which are now considered deprecated. There are currently three versions of TLS, TLSv1.0, TLSv1.1 and TLSv1.2. Most products today support TLSv1.0. TLSv1.1 and TLSv1.2 are still not widely supported. Changes to standards for TLSV1.2 have been made as late as 2011. TLS runs on top of transport layer protocols, such as TCP. It is best suited to protect client/server application level protocols, such as HTTPS and SIP.

All OpenScape products support TLSv1.0. However, not all protocols used by OpenScape products support TLS. Examples include CSTA and MGCP. At this time these protocols should be secured using IPSec. Future versions of our products will support newer TLS versions and may include TLS support for these other protocols.

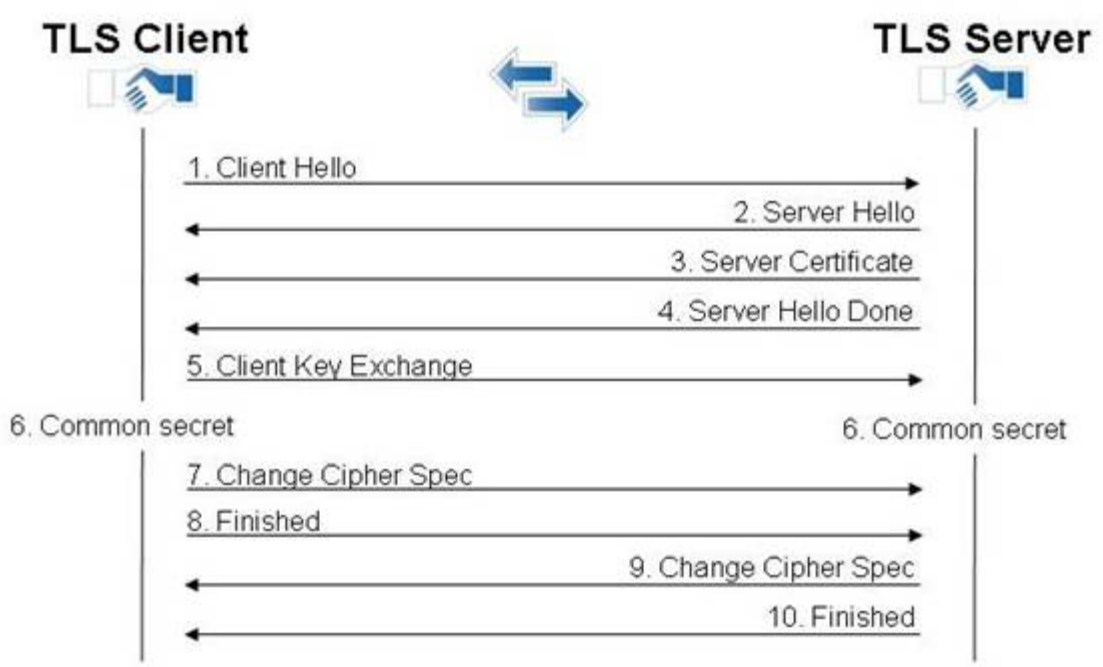
To provide encryption TLS depends on public key certificates (certificates will be discussed in the next section) to negotiate and exchange an encryption key between the client and server. As part of this exchange the client can verify the received certificate from the server using the CA certificate(s) that signed the server certificate. If the client cannot verify the received certificate it may decide to not allow the connection to proceed. The best example of a client failing to verify a server certificate is a user accessing a web site where the web server’s certificate information is not correct, expired, or no CA certificate(s) exists on the client side for verification. The client’s web browser would then prompt the user if they would like to proceed.

2.1.1 Transport Layer Security – Server Authentication (TLS)

Below is a general overview of a TLS handshake.

- The client sends a **CLIENT HELLO** message to the server.
- The server responds with a **SERVER HELLO**, its **CERTIFICATE**, and a **SERVER HELLO DONE**.
- The client generates a pre-master secret key that will later be used to encrypt the data flow. The client encrypts this using the server’s certificate. The client then returns this to the server in a **CLIENT KEY EXCHANGE** message.
- Both the client and server use other exchanged data in the previous messages to generate a common secret key that will be used to encrypt the data stream.
- Both sides then send a **CHANGE CIPHER SPEC** and **FINISH** message letting each other know they are ready to proceed.

Figure: 1: Simple TLS handshake.The client authenticates the server.



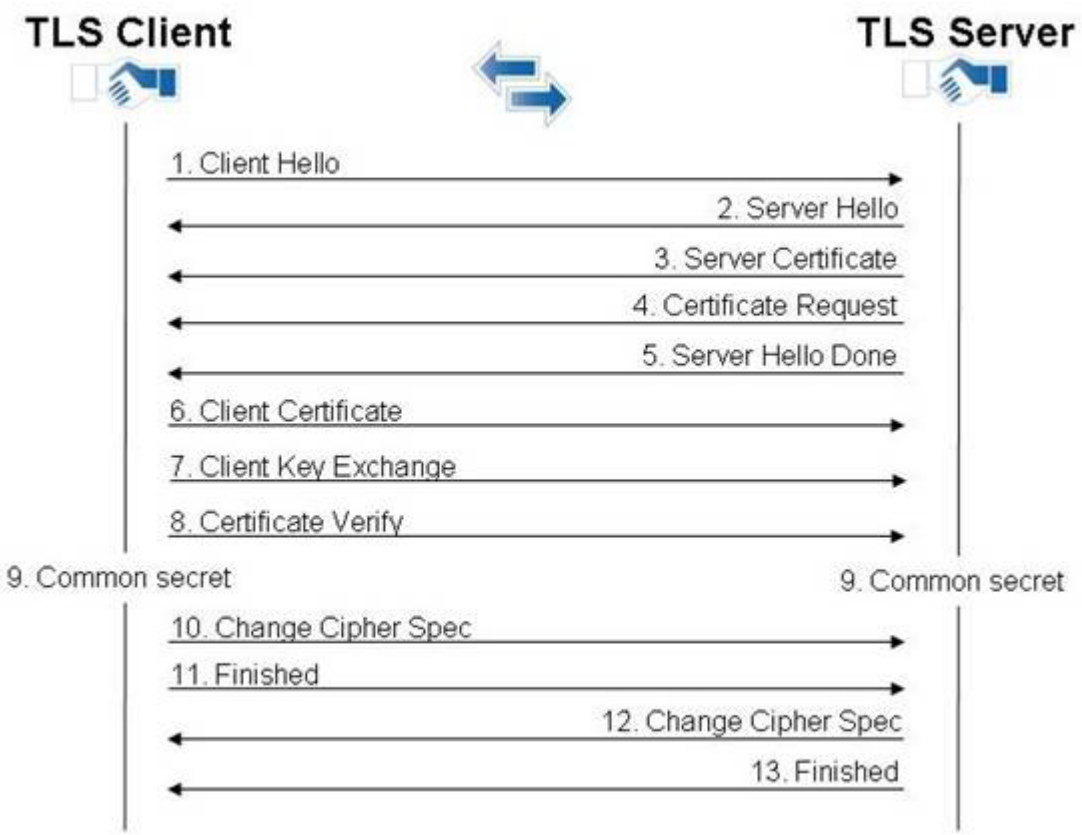
2.1.2 Transport Layer Security – Server and Client Authentication (MTLS)

Mutual TLS is the same as regular TLS, except in the handshake between the server and the client the server also requests the client send its certificate back to the server so the server can verify who it is communicating with.This is also referred to as client-authenticated TLS. If either end of the connection is unable to verify the others certificate then the connection will not be established. While MTLS can be used for regular user clients, it is typically reserved for server-to-server communication.

Below is a general overview of a MTLS handshake. Differences from a TLS handshake are highlighted.

- The client sends a **CLIENT HELLO** message to the server.
- The server responds with a **SERVER HELLO**, its **CERTIFICATE**, a **CERTIFICATE REQUEST**, and a **SERVER HELLO DONE**.
- The client generates a pre-master secret key that will later be used to encrypt the data flow. The client encrypts this using the server's certificate. The client then returns this to the server in a **CLIENT KEY EXCHANGE** message along with its **CERTIFICATE** and a **CERTIFICATE VERIFY**.
- Both the client and server use other exchanged data in the previous messages to generate a common secret key that will be used to encrypt the data stream.
- Both sides then send a **CHANGE CIPHER SPEC** and **FINISH** message letting each other know they are ready to proceed.

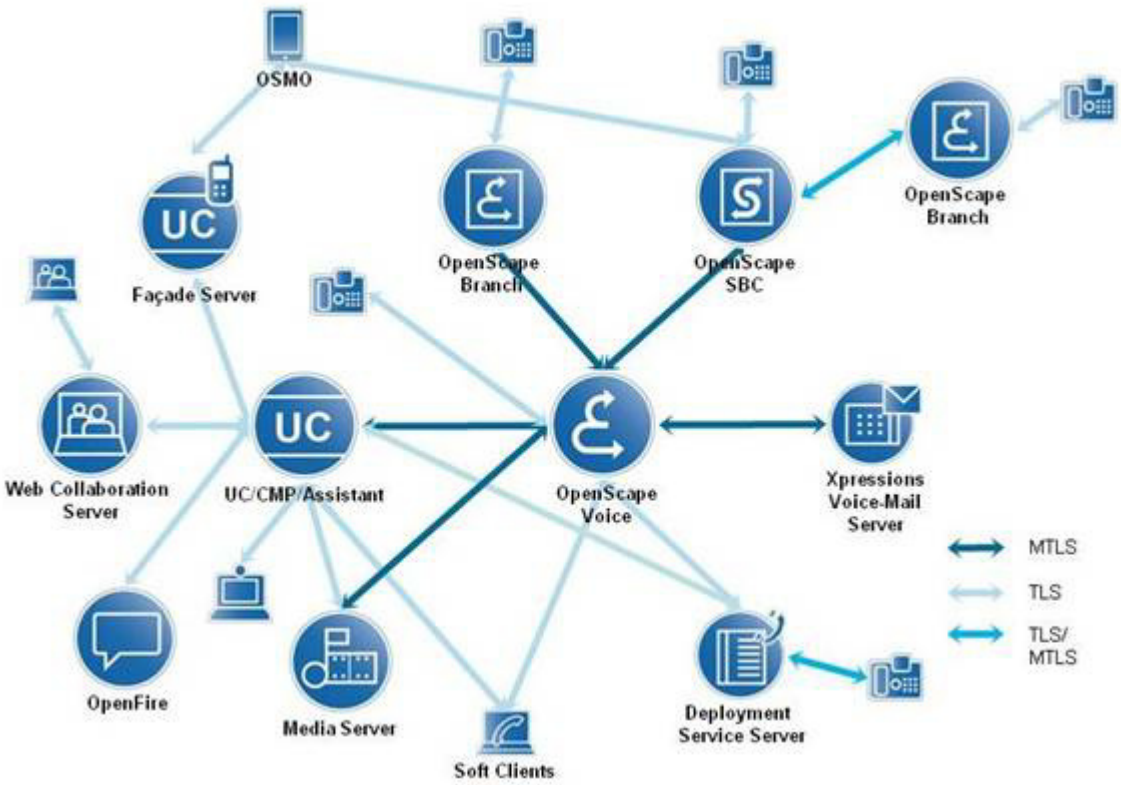
Figure: 2: Mutual TLS handshake. The server also requests the client's certificate. The client and server both authenticate the other’s identity.



2.1.3 OpenScape TLS/MTLS Communication Overview

Most protocols in the OpenScape product portfolio employ TLS to provide security. Since our products are heavily interconnected it can be difficult to visualize just how important TLS really is for our product security.Below is a graphic detailing TLS connections amongst our products. The graphic attempts to show a detailed example but does not contain all possible TLS connections within our products.

Figure: 3: Example set of TLS connections in OpenScape products.



2.2 Certificates and PKI

A certificate is a way to certify the identity of an end entity. A certificate contains a public key and organizational information bound to this public key. This allows for security while at the same time identifying the end party with whom the connection is being established. This certificate is signed by some known trusted entity. These trusted entities are called certificate authorities and the process of them signing the certificate tells the end user verifying the certificate that some mutually trusted third party – the certificate authority – has taken the time to ensure the information is true.

A certificate has an associated private key. Together, the certificate and private key are referred to as a key pair. The certificate is considered public information.It can be freely distributed. The key must be kept secret. To communicate securely information can be encrypted using the certificate and then transmitted to the end entity. The end entity then uses their private key to decrypt the message. Only the private key can be used to decrypt the message. This type of encryption, where a public and private key are required, is known as asymmetric encryption, or public-key cryptography.

Management of certificates is a difficult task. Many issues need to be addressed, such as distribution of certificates, signing certificates, certificate usage policies, revocation status, validity checking, and etc. Any infrastructure put in place to support this kind of cryptography and provide support for its management

functions is called a Public Key Infrastructure (PKI). Well known examples of PKIs are trusted third party certificate authorities, such as Verisign.Organizations may also have their own internal PKI.

2.2.1 Certificates and TLS

How do certificates apply to TLS? Certificates allow TLS to do three things at once, encrypt the data stream, verify the end users, and sign the data to insure that it has not been tampered with. However, there are some problems with using a certificate based system, such as performance and no guarantee everyone has a certificate.

Asymmetric cryptography has a higher performance overhead than regular symmetric encryption (encryption where a single key can be used by both sides to encrypt and decrypt). To overcome this, certificates are used to securely communicate a symmetric encryption key. Once this key has been negotiated symmetric encryption is then used. This can be seen in Fig. 1 and Fig. 2.The negotiated common secret was securely communicated to each party using asymmetric encryption with the use of certificates. Each party will now use this new securely transmitted symmetric key and thus the performance problem is avoided.

All parties do not require a certificate in order to achieve secure communication. Public key cryptography was designed to have a public component which anyone could use to initiate a secure channel to the receiving party. This type of setup allows the few devices which need to provide the secure communication to require a certificate, such as web servers. However, these few devices often communicate with thousands of different clients. If these devices also wish to authenticate their clients then each client would also need their own unique certificate. Correctly distributing thousands of certificates can be difficult. For this reason mutual authentication is often left to niche situations or to organizations with their own PKI.

2.2.2 General Certificate Hierarchy Overview

Fig. 4 shows an example certificate hierarchy structure.In any certificate hierarchy there is always a single trusted root certificate authority (CA). From this, the root CA can sign end entity certificates or other subordinate CA certificates. In environments where many certificates must be created and/or there are more complicated organizational issues, it’s best practice to create subordinate CAs (also called an intermediate CA).Subordinate CAs can be used to delegate responsibility or to sign certificates of a certain type, such as email certificates.

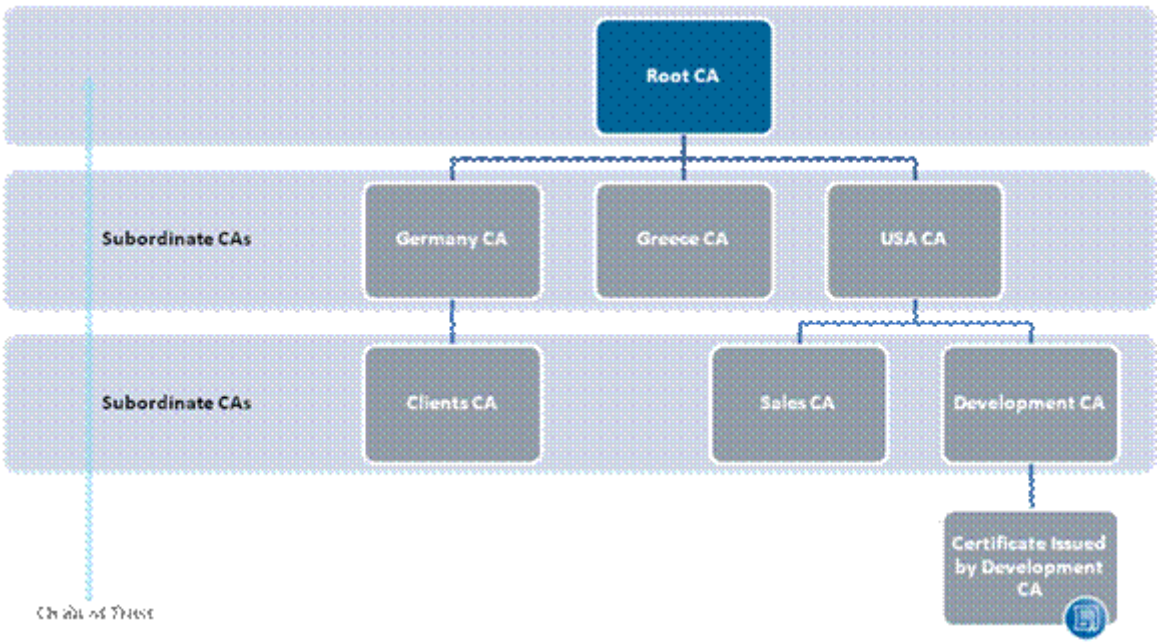
As we can see in Fig. 4 there is one root CA, three CAs subordinate to the root CA, and three CAs subordinate to the USA CA. Under the engineering CA a certificate was issued. Each CA has the power to create certificates or more certificate authorities (if allowed). Each subordinate certificate or CA is signed by the CA above them in the hierarchy. For example, the engineering certificate is signed by the engineering CA. The engineering CA is signed by USA CA, the USA CA is signed by the root CA, and finally the root CA is signed by itself. This is called the

chain of trust. When verifying certificates each level in the chain is verified until a final self-signed certificate is reached, the root CA. If each step is successful then the certificate is trusted.If a single step fails then the certificate cannot be trusted.

Public certificate authorities such as VeriSign will never sign a subordinate CA for end user use. If they did then users could use their subordinate CA to sign any certificate they wanted. For example, they could use their CA to sign a duplicate certificate for a secure website at www.unify.com. This would allow a nefarious user to possibly impersonate the website. This would completely break the chain of trust model public signing authorities are dependent on. For this reason public signing authorities strongly protect their own root and subordinate CAs.

To get around this, larger companies will have their own in-house public key infrastructure to create and distribute their own certificates. There is nothing wrong with this. It simply allows them to create subordinate CAs to better manage their organization and avoid the cost public signing authorities charge for certificate creation.

Figure: 4: An example certificate hierarchy.



Multiple subordinate CAs and levels are possible. An end entity verifying a certificate must verify each signing CA. The chain of trust flows up the tree to the root CA.

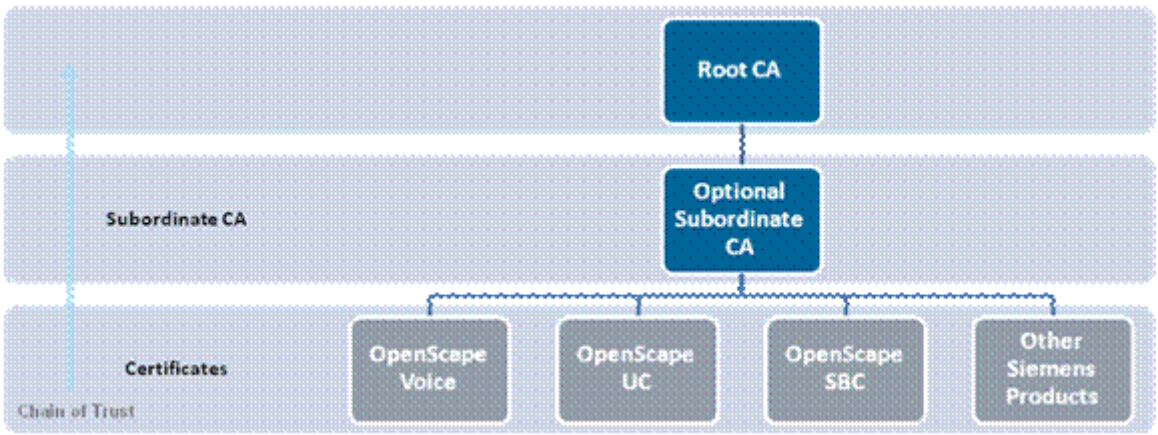
2.2.3 OpenScape Products Certificate Hierarchy

It is suggested all products in the OpenScape solution have their certificates created from a single source. Fig. 5 shows an example of this. There are some exceptions. However, for guaranteed operation it's best to use a single signing CA.

Three different CA scenarios exist for the OpenScape solution:

- A public certificate authority. This solution can be cost prohibitive. As we can see in Fig. 5 there are many products in a solution. The cost would add up.
- An in-house certificate authority and PKI environment provided by the customer. No cost to create certificates.Certificates would conform to the company's certificate policy.
- A small self-created certificate authority for the purpose of just creating certificates for the OpenScape solution. This is very similar to the previous option. However, this scenario assumes there is not a PKI environment in the customer's network and that certificates will be created and managed manually.

Figure: 5: OpenScape solution certificate hierarchy.



Any number of subordinate CAs can be used. However, all certificates must originate from the same CA. No additional CAs should be used to create certificates for the solution.

3 Replacing Certificates in an OpenScape Solution

Certificates can be created using a number of different methods. Examples are a well known trusted third party certificate authority, such as VeriSign, an internal organizational certificate authority, or certificate authorities and certificates can be created manually using simple third party tools. This section assumes certificates have already been created for all products and are available to the installer. The installer will need the key pair (certificate and the private key) as well as the certificate authority certificates used to sign the key pair. This includes any subordinate CAs.

What certificates should be replaced? All of them. All OpenScape products come with default certificates. Some are auto-generated and others are the same across each product install. Therefore, for security reasons, they should all be replaced. If only some are replaced then certain functions may no longer work. An example of this is the MTLS connection between the OSV and UC (Fig. 3). There exists a current restriction where each side must be signed by the same certificate authority. If not then the connection will fail.

Note:

- Certificates can come in different formats. The basic format is PEM formatted certificates and keys. The key portion can also be encrypted with a password to protect it in case of loss or theft. The OpenScape solution certificate requirements vary from product to product. Therefore each product may require the certificate and key be in a specific format before it can be correctly installed. Examples include PEM format, PKCS#12 keystore, JKS keystore, and etc. If you find you need to convert from one format to another then please consult Section 5, "Formatting Certificates", for conversion procedures. Each product below does define what format is required.
- We assume one certificate for each physical device. For physical devices that operate as one logical unit then only one certificate is required. This one certificate will be installed on each physical node in the logical configuration. These are typically redundant devices that provide failover in case one device fails. Examples include the OpenScape Voice, OpenScape SBC and OpenScape Branches operating in failover configurations.
- An attempt should be made to use certificates created by the customer. Many customers will have their own internal certificate authority and will want/need to integrate OpenScape products into their certificate strategy. Please consult with the customer first before manually creating certificates. If you find you need to create your own certificate authority and certificates then please consult Section 6, "Creating Your Own Root CA", for instructions.
- Be aware that all commands in this section are on a single line. Due to the length of some commands they may wrap around to another line in this documentation.
- Important: Products in the OpenScape solution have differing filename requirements. Some require specific file names while others may have any name. In the sections below areas where filenames are not specific will be enclosed with <> brackets, <example_file_name>.

3.1 Certificate Restrictions and Requirements

Certain products in the Unify solution contain certain restrictions or requirements that will directly affect certificate creation and usage for other products. These restrictions or requirements must be considered when deploying certificates in your solution.

3.1.1 What Certificate Authorities Should be in a Certificate

According to RFC 5280, each certificate should contain a copy of each issuing CA in the chain of trust up to, but not including, the root CA. For example, looking at Figure 4, "An example certificate hierarchy," the certificate signed/issued by the development CA, which in turn was signed/issued by the USA CA, which in turn was signed/issued by the root CA, should also include the development CA and USA CA certificate, but not their private keys. The only private key to be included is for the server certificate.

3.1.2 OpenScape Branch and SBC

The V8 OpenScape Branch and V8 OpenScape SBC cannot interpret certificates with any extensions marked as critical other than the basic constraints field. If a certificate is received by the OpenScape Branch or OpenScape SBC with any other field marked as critical, the TLS connection will fail. For example, if the OpenScape Voice certificate contains any field marked as critical, other than the basic constraints field, then the communication between the OpenScape Voice and OpenScape Branch or OpenScape SBC will fail.

3.1.3 Mediatrix

Mediatrix devices require that their certificates contain the extended key usage fields TLS Web Client Authentication and TLS Web Server Authentication. If these fields are not present in the Mediatrix certificate, then the Mediatrix will fail to communicate via TLS.

3.1.4 SHA1 and SHA2

Starting with OpenScape Solution V8 (incl. UC V7) the use of SHA-2 (SHA-256) signatures in certificates is recommended since SHA-1 is no longer considered secure.

Version 7 solutions commonly interact with or contain version 4, 5, or 6 components, and SHA-2 signatures have not been widely tested in version 7. Certain products or services may fail when using SHA-2.

Versions 4, 5, and 6 do not support SHA-2 signatures in their certificates. SHA-1 is the only recommended signature.

3.1.5 Year 2038 Problem

Many computer systems store time in a 32-bit clock. This clock is set to expire in early 2038. Certificates should not be created which have a date extending into or past 2038. Many systems will not accept certificates with these dates.

3.2 OpenScape Voice

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between the OpenScape Voice and all products.

INFO: The "NetscapeCertType" certificate extension is **not** supported by OpenScape Voice.

Step by Step

1) Log in to **CMP**.

Navigate to **Configuration > OpenScape Voice** and from the dropdown list select your switch.

2) Click on the **Administration** button and under **Certificate Management** select **Trusted CA Certificate Stores**.

3) On the pop-up window click on the checkbox next to **.../root.pem** and then click **Edit**.

4) On the next pop-up window click **Browse...** navigate to the location where **root.pem** is stored and then click **Upload...**

5) After closing all pop-up windows navigate to **Certificate Management** and select **Certificate Key Stores**.

6) On the pop-up window click **Browse...** navigate to the location where **server.pem** and **client.pem** are stored and then click **Upload...**

If the certificates were replaced properly after navigating to **Certificate Management > Certificate Monitoring** the old certificates will show under Status **CertRemoved**. The new certificates will show under Status **CertInstalled** and under NodeID **Both**.

INFO: Please wait a few minutes until the list is updated.

3.2.1 How to add certificates issued by different Certification Authorities

It is suggested that all of the certificates that are used for the products of an OpenScape Solution are issued from a single Certification Authority. However it

is possible to use certificates from different sources by performing the following steps.

Prerequisites

- The OSV certificates have been uploaded following the procedure described in Section 3.2 "OpenScape Voice".

Step by Step

1) Log in to **CMP**.

Navigate to **Configuration > OpenScape Voice** and from the dropdown list select your switch.

2) Click on the **Administration** button and under **Certificate Management** select **Trusted CA Certificate Stores**.

3) On the pop-up window click on the checkbox next to **.../root.pem** and then click **Edit**.

4) On the next pop-up window click **Browse...** navigate to the location where the ***.pem file with the CA** is stored and then click **Upload...**

If the certificates were uploaded successfully after navigating to **Certificate Management > Certificate Monitoring** the new certificates will show under Status **CertInstalled** and under NodeID **Both**.

INFO: Please wait a few minutes until the list is updated.

3.3 OpenScape UC (All Deployments)

The OpenScape UC server contains numerous locations where the same certificate for each individual service must be placed. You need a separate certificate for each device but not for each service on a single device. For example, with an OpenScape UC small deployment, the same certificate can be used for each service since all services exist on the same physical machine. For an OpenScape UC large deployment, individual certificates would be needed for the backend, frontend, and media server. The exception is frontend servers in large deployments. All frontend servers are managed by a single certificate placed on the backend server. See the WebClient large deployment section for more information.

Replacement of a certificate on a UC system requires that symphonia be restarted before it becomes active. To avoid multiple restarts of the OpenScape UC system, replace the certificate in all locations and then restart symphonia once. All actions are performed as the root user.

3.3.1 Assistant

Certificate format: PKCS#12 formatted keystore.

Purpose: Secure SOAP communication between the Assistant and OSV.

OpenScape Voice Assistant reads the configuration from **symphonia-client-config.properties** in order to communicate with the OSV via TLS.

> ***NOTICE:*** The **generateKeyStore.sh** script is obsolete beginning from V10.

Step by Step

1) Backup the existing keystore:

```
cp /opt/siemens/common/conf/axis_soap_tls/
osvkeystore.p12 /opt/siemens/common/conf/axis_soap_tls/
osvkeystore.p12.bak
```

2) Copy in the new keystore:

```
cp <path_to_new>/osvkeystore.p12 /opt/siemens/common/
conf/axis_soap_tls/osvkeystore.p12
```

3) Encode the keystore password.

The Assistant needs the keystore password in order to access the keystore and load the certificates. Run the following command and provide the password you used for the keystore. The output will be a long string that will be used in step 4.

```
/opt/siemens/servicetools/cmp/encodePassphrase.sh
Provide passphrase to protect:
Repeat passphrase:
```

4) Edit the **symphonia-client-config.properties** file and replace the value of **keystorePassword** with the string that was generated in step 3:

```
vi /opt/siemens/assistant_ssl/symphonia-client-
config.properties
```

> ***INFO:*** The default **keystorePassword** and **trustorePassword** is **importkey**.

5) Upload the OSV CA certficate (in PEM format) to the system and then execute the following command:

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
importcert -file [RootCACertificateLocation] -keystore /
opt/siemens/common/conf/axis_soap_tls/osvtrustore.jks
```

> ***NOTICE:*** The last argument is the path of the **truststore** file and can be found in the **symphonia-client-config.properties** file (e.g. truststore=/opt/siemens/common/conf/axis_soap_tls/osvtrustore.jks).

When prompted type the trustorePassword (default is **importkey**).

6) Test if the import was successful using the following command:

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
list -keystore /opt/siemens/common/conf/axis_soap_tls/
osvtrustore.jks
Enter keystore password:
```

You can list more details by adding the option **-v**.

7) Change ownership and permissions of the certificate files:

```
chown sym:sym /opt/siemens/common/conf/axis_soap_tls/
osvkeystore.p12
chmod 0644 /opt/siemens/common/conf/axis_soap_tls/
osvkeystore.p12
chown sym:sym /opt/siemens/common/conf/axis_soap_tls/
osvtrustore.jks
chmod 0644 /opt/siemens/common/conf/axis_soap_tls/
osvtrustore.jks
```

8) Restart **symphonia daemon**.

3.3.2 Bcom

Certificate format: PEM – Key not encrypted.

Important: Only PKCS#1 format is supported. For details on how to convert a PKCS#8 certificate to PKCS#1, refer to Chapter 5 "Formatting Certificates".

Purpose: Secure SOAP communication between the OpenScape UC and OpenScape Voice

Note: At this time, this only secures SOAP communication between OpenScape UC and OpenScape Voice for OpenScape Mobile related functions.

Note: This certificate MUST have the same certificate authority as the OpenScape Voice certificate.

Note: The private key section should begin with **----BEGIN RSA PRIVATE KEY----** and end with **----END RSA PRIVATE KEY----**.

Step by Step

1) On the UC server, concatenate the certificate and key into one temporary file and then run the following script. **Note:** Once the script is run, the concatenated file will be automatically deleted.

```
cat <new_uc_cert> <new_uc_key> > <temp_file>
/opt/siemens/servicetools/bcom/bcomImportCert.sh
<temp_file>
```

2) The TLS connection between Bcom and OSV will be automatic after symphonia is restarted.

3.3.3 CMP

Certificate format: PKCS#12 formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between the CMP and web browser.

Step by Step

1) Create a new PKCS#12 formatted keystore named ‘tomcat-keystore.p12’. Refer to Section 5, "Formatting Certificates", to convert your certificate and key to a PKCS#12 keystore.

2) Backup the existing keystore.

```
cp /opt/siemens/common/conf/tomcat-keystore.p12
/opt/siemens/common/conf/tomcat-keystore.p12-backup
```

3) Copy in the new keystore.

```
cp <path_to_new>/tomcat-keystore.p12 /opt/siemens/
common/conf/tomcat-keystore.p12
```

4) Encode the keystore password. The CMP needs to access the keystore in order to load the certificates. To do this, it must know the keystore password. The CMP stores the keystore password in an encrypted format. We use a tool to encrypt the password and then we store it in a file. Run the following command and provide the password you used for the keystore. The output will be a very long string. Note: This step does not apply to versions 4 and 5.

```
/opt/siemens/servicetools/cmp/encodePassphrase.sh
Provide passphrase to protect:
Repeat passphrase:
```

5) Store the output from **step 4**. Edit the **server.xml** file and replace the existing keystore password. This password is stored with the variable name keystorePass=””. Remove the existing password and store the password from step 4) between the “”. **Note:** This file contains settings for more than one service. Take care to edit the connector with name CMP_https. **Note:** If using versions 4 or 5, then use the password of the keystore file.

```
/opt/siemens/share/tomcat/conf/server.xml
<Connector …… keystoreFile="/opt/siemens/common/conf/
tomcat-keystore.p12"
keystorePass="place new password here"
keystoreType="PKCS12"
name="CMP_https" port="443"
……</Connector>
```

6) Edit the **trust-store.properties** file and replace the existing keystore password. This password is stored with the variable name keystorePass="". Remove the existing password and store the password from **step 4** between the **" "**.

```
/opt/siemens/common/conf/application-external/trust-store.properties
```

7) Set correct ownership and permissions.

```
chown sym:sym /opt/siemens/common/conf/tomcat-keystore.p12
```

```
chmod 0644 /opt/siemens/common/conf/tomcat-keystore.p12
```

8) Restart the application computer.

```
/etc/init.d/symphoniad restart
```

3.3.4 SPML Responder

Certificate format: PKCS#12 formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between SPML and SPML enabled devices.

Step by Step

› The steps in this section are exactly the same as those in section 'CMP'. The only difference is the keystore is named 'spmlresponder-keystore.p12'. Simply make a copy of the tomcat-keystore.p12 keystore. Edit the same xml file and use the same password. **Note:** This file contains settings for more than one service. Take care to edit the connector with name spmlresponder_https_connector.

```
/opt/siemens/share/tomcat/conf/server.xml
```

```
<Connector …… keystoreFile=”/opt/siemens/common/conf/spmlresponder-keystore.p12”
```

```
keystorePass=”place new password here” keystoreType=”PKCS12”
```

```
name="spmlresponder_https_connector" port="9943"
```

```
……</Connector>
```

3.3.5 UCMA

Certificate format: X.509.

Purpose: Secure communication between the OpenScape UC Application and the UCMA Proxy Web Service.

For details please refer to Section "Importing the Certificate to the OpenScape Application Computer" in **OpenScape UC Application, Configuration and Administration, Administrator Documentation** guide.

3.3.6 WebClient - Simplex and Small Deployments

Certificate format: PKCS#12 or JKS formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between the WebClient, web browsers, OpenScape Desktop Enterprise Edition (WE) client, and OpenScape Fusion client.

Step by Step

1) By default, the WebClient uses a JKS keystore. You can continue to use a JKS keystore. However, to do so requires you first create a PKCS#12 keystore and then convert it to JKS. These instructions will only use a PKCS#12 formatted keystore.

Backup the existing keystore and supporting files to a safe location:

```
/opt/siemens/HiPathCA/distribution/.keystore
```

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
CAMgmt/ssl.cfg
```

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
Portal/ssl.cfg
```

2) Rename your certificate file to .keystore and replace the existing one with the new by copying the latter to /opt/siemens/HiPathCA/distribution

```
chown sym:sym /opt/siemens/HiPathCA/distribution/
.keystore
```

```
chmod 0644 /opt/siemens/HiPathCA/distribution/.keystore
```

3) Edit the CAMgmt ssl.cfg file to set the keystore type and password.

/opt/siemens/HiPathCA/config/services/default/Httpd/CAMgmt/ssl.cfg

```
keystoreFile=.keystore
```

```
com.siemens.ca.server.SSLport.keystoreType=PKCS12
```

```
com.siemens.ca.server.SSLport.keystorePass=your_password
```

> ***INFO:*** The password needed here is the one provided by the certificate issuer, which was used when creating the new certificate.

4) Repeat step 2) and edit the portal ssl.cfg file to set the keystore type and password.Use the same type and password. **Note:** This file also contains a section for APR SSL.It is not used.

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
Portal/ssl.cfg

keystoreFile=.keystore

com.siemens.ca.server.SSLport.keystoreType=PKCS12

com.siemens.ca.server.SSLport.keystorePass=your_password
```

> ***INFO:*** The password needed here is the one provided by the certificate issuer, which was used when creating the new certificate.

5) Set correct ownership and permissions for the CAMgmt and Portal ssl.cfg files.

```
chown sym:sym <path to files here>

chmod 0644 <path to files here>
```

3.3.7 WebClient - Large and Very Large Deployments

Certificate format: PKCS#12 or JKS formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between the WebClient, web browsers, OpenScape Desktop Enterprise Edition (WE) client, and OpenScape Fusion client.

Large and very large deployments utilize frontend servers to handle all WebClient connections. The backend server also has a WebClient process, but is not accessed directly. The backend server is responsible for managing the certificates on all frontend servers. When a frontend server starts up it consults the backend server for its certificate (along with other configuration information). The backend server does not maintain separate certificates for each frontend server. All frontend servers receive the same certificate. Therefore, it is important to include all FQDNs for all frontend servers in one certificate. If another frontend sever is added then a new certificate will need to be created to include this server’s FQDN.

There is one keystore used for both backend and frontend servers. It is located under the **/opt/siemens/HiPathCA/distribution** directory and should be named ".keystore". **CAMgmt ssl.cfg** and **Portal ssl.cfg** should be modified in order to adjust to the keystore that was produced.

Step by Step

1) Backup the default keystore and supporting files to a safe location:

```
/opt/siemens/HiPathCA/distribution/.keystore
```

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
CAMgmt/ssl.cfg
```

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
Portal/ssl.cfg
```

2) Edit the CAMgmt ssl.cfg file to set the keystore name, type, and password for the backend.

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
CAMgmt/ssl.cfg
```

```
keystoreFile=.keystore
```

```
com.siemens.ca.server.SSLport.keystoreType=PKCS12
```

```
com.siemens.ca.server.SSLport.keystorePass=your_password
```

> ***INFO:*** The password needed here is the one provided by the certificate issuer, which was used when creating the new certificate.

3) Edit the Portal ssl.cfg file to set the keystore name, type, and password for the frontend. **Note:** This file also contains a section for APR SSL. It is not used.

```
/opt/siemens/HiPathCA/config/services/default/Httpd/
Portal/ssl.cfg
```

```
keystoreFile=.keystore
```

```
com.siemens.ca.server.SSLport.keystoreType=PKCS12
```

```
com.siemens.ca.server.SSLport.keystorePass=your_password
```

> ***INFO:*** The password needed here is the one provided by the certificate issuer, which was used when creating the new certificate.

4) Set correct ownership and permissions for the CAMgmt ssl.cfg, Portal ssl.cfg, backend.keystore, and frontend.keystore files.

```
chown sym:sym <path to files here>
```

```
chmod 0644 <path to files here>
```

5) Restart WebClient services on the backend first, and then the frontend servers.

```
/etc/init.d/symphoniad restart WebClient_BE
```

```
/etc/init.d/symphoniad restart WebClient_FE
```

3.3.8 Apache

Certificate format: PEM – Key Encrypted.

Purpose: Secure communication for phone firmware delivery between the UC and DLS, UC and phones, UC and OpenScape Branch, and UI patching on large and very large deployments.

Note: Applies only to versions 6 and 7.

Note: The Apache server is used for secure communication between nodes in a large or very large deployment when the UI patching feature is used. In this case the certificate on the backend server must contain the FQDN returned by the command ‘hostname –f’ executed on the backend server. To allow for flexibility, this FQDN can be placed in the subject alternative name field of the certificate along with any other FQDNs used by the Apache/Backend server.

Note: If the UI patching feature is used then prepareupdate.sh must be run again. Follow the instructions in the section titled “Updating using the CMP (UI Patching) instead of osc-setup” in the OpenScape UC Installation and Upgrade guide.

Step by Step

1) Backup the default certificate, key, and CA certificates file.

```
/etc/apache2/ssl.crt/sen_cert.pem
/etc/apache2/ssl.crt/sen_cacert.pem
/etc/apache2/ssl.key/sen_key.pem
```

2) Concatenate your CA certificate into the existing CA certificate file. **Note:** The double arrows “>>” in the below command are important.

```
cat <CA_cert> >> /etc/apache2/ssl.crt/sen_cacert.pem
```

3) Rename and copy your certificate and key into the locations described in step 1).Change ownership and permissions. **Note:** The key needs to be password protected. See Section 5, "Formatting Certificates", to convert your key to an encrypted format.

```
chown root:root sen cert.pem, sen key.pem, and sen
ca_cert.pem
chmod 0644 sen cert.pem, sen key.pem, and sen ca_cert.pem
```

4) Encode the key password. Apache needs to access the key in order to load the certificates. To do this it must know the key’s password. Apache stores the key password in an encrypted format. We use a tool to encrypt the password and then we store it in a file. Run the following command and provide the password you used for the key. The output will be a very long string.

```
/opt/siemens/servicetools/cmp/encodePassphrase.sh
Provide passphrase to protect:
Repeat passphrase:
```

5) Store the password from step 4) in the following file.Remove the existing password from the file first.

```
/etc/apache2/ssl.key/apache_httpd_pass
```

6) Issue the following command to restart the Apache service. **Note:** Symphonia does not need to be restarted.

```
rcapache2 restart
```

3.3.9 Regeneration of UC certificates in default keystore

The following procedure is used for regenerating the default certificates included in the default container-keystore_5da1e680-f1e0-11d9-bbed-0002a5d5c51b-localhost.p12 (or container-keystore_xxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.p12).

The regeneration will result in certificates with:

Subject: "Generated by Unify Software and Solutions GmbH & Co. KG."

Signature Algorithm: SHA-256 signature with RSA 2048bit encryption

Validity: 5 Years

NOTICE: It is not supported to replace the certificates contained in the default keystore with custom certificates generated and signed by customer infrastructure.

The following has to be executed on a BE node:

- Stop Symphonia:
  `/etc/init.d/symphoniad stop`
- Locate the file:
  `/opt/siemens/servicetools/install/installlib/antext.xml`
- Make a backup copy of the file (make sure to maintain its access rights)
  `cp /opt/siemens/servicetools/install/installlib/antext.xml /opt/siemens/servicetools/install/installlib/antext.xml.backup`
- Edit `/opt/siemens/servicetools/install/installlib/antext.xml`
  locate the following line:
  <target name = "-securityInstaller.reconf"
  Modify this entry by removing the "-", i.e. the line should look like this:
  <target name = "securityInstaller.reconf"
  save the file.
- Execute:
  `/opt/siemens/servicetools/install/bin/installlib.sh securityInstaller.reconf`

- Revert to the original file
  `cp /opt/siemens/servicetools/install/installlib/antext.xml.backup /opt/siemens/servicetools/install/installlib/antext.xml`

  After the above steps have been executed the following certificates will be renewed:

  `default-truststore.jks`
  `default-ca-keystore.p12`
  `container-keystore_1d499ad3-bd53-4174-886c-ddf5f1d9485e.p12`

  In case the rest of the certificates need to be re-generated e.g. Tomcat, CMP, spmlresponder-keystore.p12 , proceed to the following steps.

  NOTICE: This procedure will also re-generate and replace any custom certificates introduced by the customer

  In case it is required to regenerate to default values the Tomcat and SPML responder keystores, execute the following:

  – **Tomcat**:
    `sh /opt/siemens/servicetools/install/bin/installlib.sh installplugins.execute -Dinstall-plugin.name=mgmtportalinstall -Dinstall-plugin.action=reconfigure`
  – **SPML**:
    `sh /opt/siemens/servicetools/install/bin/installlib.sh installplugins.execute -Dinstall-plugin.name=spmlresponderinstallplugin -Dinstall-plugin.action=reconfigure`
- Start Symphonia
  `/etc/init.d/symphoniad start`

The following needs to be executed on the remote nodes:

- Stop Symphonia:
  `/etc/init.d/symphoniad stop`
- Copy the security configuration files produced in the BE `/opt/siemens/common/conf` into the remote node's `/opt/siemens/common/conf` folder.
  Files:
  – `authenticationauthority_config.xml`
  – `AuthenticationAuthorityKeyStore.sks`
  – `container-keystore_xxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.p12`
    (last container keystore generated in the BE, not the one ending with localhost! ,easily list-able by executing "`ls -ltr`" in the `/opt/siemens/common/conf` folder)

  - `symom.config.xml`
  - `VerifierKeyStore.sks`
- Execute `/opt/siemens/servicetools/install/bin/installlib.sh securityInstaller.updateServiceStatements`
- Start Symphonia
  `/etc/init.d/symphoniad start`

3.4 OpenScape Media Server

Certificate format: PKCS#12 formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between the OSV and Media server.

INFO: You only need to change the OpenScape Media Server certificate if the communication between the OpenScape Voice and OpenScape Media Server is SIP and TLS. The MGCP and CSTA protocols are not secured by TLS.

INFO: The integrated media server in simplex deployments does not require a certificate. SIP communication between the OpenScape Voice and OpenScape Media Server is internal. However, if TLS is still desired change the path entries in the below steps from /opt/siemens/ to /enterprise/.

INFO: These steps are for OpenScape Media Server V6 or later. Version 5 and earlier systems will need to generate a certificate signing request (CSR) using the keytool command and have this signed by a CA. These steps are not defined here. Contact next level of support. Another option is to perform the below commands on a different system using JRE1.6 or greater and then copying the resulting keystore to the OpenScape Media Server.

By default the OpenScape Media Server keystore format is JKS. This format is similar to PKCS#12. If replacing the OpenScape Media Server's certificate a PKCS#12 keystore will be imported into the OpenScape Media Server's JKS keystore.

3.4.1 Adding/Delete a CA from the Keystore

The keystore can utilize more than one CA. This allows the OpenScape Media Server to trust certificates from differing CAs.

3.4.2 Adding a CA

This will add a new CA to the keystore. Simply provide a generic name for the alias field and the filename the CA is stored in. The CA will be imported in to the keystore.

Step by Step

1) Change to the following directory.

```
/opt/siemens/mediaserver/application_host/providers/sip-connectivity
```

2) Add the new CA.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -import -trustcacerts -alias <CAName> -file <CAfile> -keystore tls-keystore.jks
```

3) Verify the CA was added.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -list -v -keystore tls-keystore.jks –storepass password
```

4) Restart the Media Server. Note: If this is a simplex or small deployment there will be a complete UC outage.

```
/etc/init.d/symphoniad restart
```

3.4.3 Deleting a CA

It may be necessary to delete a CA from the keystore if trust has been lost in this CA or if the CA has expired. The CA's alias name must be provided. The keystore may need to be examined first to determine the correct alias name. Note: If a Polycom CA certificate is found it may be safely removed.

Step by Step

1) Change to the following directory.

```
/opt/siemens/mediaserver/application_host/providers/sip-connectivity
```

2) Locate the CA to be removed by alias name.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -list -v -keystore tls-keystore.jks –storepass password
```

3) Remove the CA.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -delete -alias <CAName> -keystore tls-keystore.jks –storepass password
```

4) Restart the Media Server. Note: If this is a simplex or small deployment there will be a complete UC outage.

```
/etc/init.d/symphoniad restart
```

3.4.4 Adding a New Certificate to the Keystore

The OpenScape Media Server's server certificate can be replaced at any time. However, only one certificate should exist in the keystore at a time. The old certificate must be removed before the new certificate is added. The keystore may need to be examined first to determine the correct alias name of the old certificate.

Step by Step

1) Change to the following directory.

```
/opt/siemens/mediaserver/application_host/providers/sip-connectivity
```

2) Locate the old certificate by alias name.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -list -v -keystore tls-keystore.jks –storepass password
```

3) Remove the old certificate.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -delete -alias <CertName> -keystore tls-keystore.jks -storepass password
```

4) Add the new certificate. This step requires the certificate is already stored in a PKCS#12 keystore. This PKCS#12 keystore will be imported into the media server's JKS formatted keystore.

a) Follow this step only if the certificate to be imported is not in a PKCS#12 format. This step can also be found in the Section 5, "Formatting Certificates".

```
openssl pkcs12 -export -in mediaserver.crt -inkey mediaserverkey.pem -name mediaserver -out media-temp.p12
```

b) Import the PKCS#12 keystore into the OpenScape Media Server's keystore. Note: srcstorepass is the password of the PKCS#12 keystore.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -importkeystore -deststorepass password -destkeystore tls-keystore.jks -srckeystore <pkcsKeystore>.p12 -srcstorepass password -srcstoretype PKCS12
```

c) Verify that the PKCS#12 keystore was imported.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -list -v -keystore tls-keystore.jks –storepass password
```

d) Restart the OpenScape Media Server. Note: If this is a simplex or small deployment there will be a complete UC outage.

```
/etc/init.d/symphoniad restart
```

3.4.5 Replacing the default certificate in the MS Tomcat keystore

The OpenScape Media Server's Tomcat (port 7443) server certificate can be replaced at any time.

However, only one certificate should exist in the keystore at a time. The old certificate must be removed before the new certificate is added. The keystore may need to be examined first to determine the correct alias name of the old certificate.

Step by Step

1) Change to the following directory.

```
/opt/siemens/mediaserver/application_host/providers/
tomcat/tomcat-home/conf/
```

2) Locate the old certificate by alias name (default is "ms-tomcat").

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
list -v - keystore keystore.jks -storepass password
```

3) Remove the old certificate.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
delete -alias <CertName> -keystore keystore.jks -
storepass password
```

4) Add the new certificate. This step requires the certificate is already stored in a PKCS#12 keystore. This PKCS#12 keystore will be imported into the media server's JKS formatted keystore.

a) Follow this step only if the certificate to be imported is not in a PKCS#12 format. This step can also be found in the Section 5, "Formatting Certificates".

```
openssl pkcs12 -export -in mediaserver.crt -inkey
mediaserverkey.pem -name mediaserver -out media-
temp.p12
```

b) Import the PKCS#12 keystore into the OpenScape Media Server's Tomcat keystore. Note: srcstorepass is the password of the PKCS#12 keystore.

```
opt/siemens/share/ibm-java-<version>/jre/bin/keytool
-import -keystore -deststorepass password -destkey-
store keystore.jks -srckeystore <pkcsKeystore>.p12 -
srcstorepass password -srcstoretype PKCS12
```

c) Verify that the PKCS#12 keystore was imported.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool
-list -v -keystore keystore.jks -storepass password
```

d) Restart the OpenScape Media Server. Note: If this is a simplex or small deployment there will be a complete UC outage.

```
/etc/init.d/symphoniad restart
```

3.5 OpenScape UC Façade Server

Certificate format: PKCS#12 formatted keystore also containing the certificate authority certificate(s).

Purpose: Secure communication between the Façade server and mobile devices, such as Apple and Android based phones.

The Façade server utilizes two different TLS connections. The first is the private facing TLS connection between the Façade and UC server. The second is the public facing TLS connection between the Façade server and mobile users.

3.5.1 Façade to UC

These steps retrieve the internal CA certificate from the UC backend server and store it in the Façade's truststore. Note: Commands are on a single line.

INFO: These steps may need to be executed again after upgrading the UC backend servers.

INFO: These steps may need to be executed again after upgrading or updating the Façade server.

INFO: If step five is executed again then a new alias name should be used.

Step by Step

1) On the Façade server execute:

```
echo "" | openssl s_client -showcerts -prexit -connect
<backendipaddress>:4709 2>/dev/null | sed -n
echo '/BEGIN\CERTIFICATE/,/END\ CERTIFICATE/p' | tac |
sed -n -e
'1,/BEGIN/!d;p' | tac > uccacert.pem
```

2) On the Façade server execute:

```
openssl x509 -modulus -noout -in uccacert.pem | openssl
md5
```

This will produce a value needed in step 4).

3) On the UC backend server execute:

```
echo "" | openssl s_client -showcerts -prexit -connect
<backendipaddress>:4709 2>/dev/null | sed -n
```

```
echo '/BEGIN\CERTIFICATE/,/END\ CERTIFICATE/p' | tac |
sed -n -e
'1,/BEGIN/!d;p' | tac > uccacert.pem
```

4) On the UC backend server execute:

```
openssl x509 -modulus -noout -in uccacert.pem | openssl
md5
```

Compare this value with that of step 2).

If they are the same then the UC CA certificate can be safely installed on the Façade server in the next step.

5) On the Façade server execute:

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
import -trustcacerts -alias ucCAcert -file uccacert.pem -
keystore /opt/siemens/share/ibm-java-<version>/jre/lib/
security/cacerts -storepass changeit
```

INFO: This command will ask you to confirm the action. Answer yes. The UC CA certificate has been installed on the Façade server.

6) Restart the Façade server.

```
/etc/init.d/façaded restart
```

3.5.2 Façade to Mobile Users

Note: Commands are on a single line.

Step by Step

1) Backup the existing keystore.

```
/user/local/tomcat6/conf/OSC-Facade-Server-Default-
Cert.p12
```

2) Create a new PKCS#12 formatted keystore using the certificate, key, and certificate authority certificates. When prompted for a password use the default Façade server password, "qe13.eq31". Reference Appendix, Formatting Certificates, for instructions.

INFO: If you wish to use a different keystore password then you can edit the server.xml file in the same folder as the keystore. Replace the 'keystorePass' variable with a password of your choosing.

3) Copy the keystore created in step 2) to the keystore folder. Use the same name as the default keystore. Check permissions and ownership.

```
cp New-Facade-Server.p12 /usr/local/tomcat6/conf/OSC-Facade-Server-Default-Cert.p12
chown sym:sym OSC-Facade-Server-Default-Cert.p12
chmod 0644 OSC-Facade-Server-Default-Cert.p12
```

4) Restart the Façade server.

```
/etc/init.d/facaded restart
```

3.5.3 Creating certificates without password protection for HAProxy unattended startup

This is for cases where HAProxy is used within the same system as the Facade server and unattended startup of HAProxy is required e.g. after Linux system boot. Then it is necessary to use a certificate without password protection and store it in a secure location.Please refer to chapter **Creating certificates without password protection for HAProxy unattended startup** in document *OpenScape UC Application, Configuration and Administration*.

3.6 OpenScape UC Openfire Server

Certificate format: PEM – For the CA certificate, Openfire certificate and Openfire key.

Purpose: Secure communications between OpenScape UC and Openfire, between Openfire servers operating in a federated environment, and the Openfire web interface and web browsers.

Note: Openfire servers use a sub-domain named ‘conference’ when they are participating in a federated environment (Openfire to Openfire communication). Therefore the certificate for the Openfire server should also contain this sub-domain. For example, if the domain name is openfire1.example.com then the sub-domain name would be conference.openfire1.example.com. Certificates for Openfire servers should contain both domain names.

Step by Step

1) Stop the Openfire server.

```
/etc/init.d/openfire stop
```

2) Import the CA certificate into the Openfire truststore. The alias is just a reference. Supply an alias name similar to the common name of the CA certificate.

NOTE: All CA certificates in the certificate chain must be imported.

3) Navigate to /opt/openfire/resources/security and then execute the following command.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
import -alias <CA_name> -file <path_to_CA_cert> -keystore
truststore –storepass changeit
```

4) Start the Openfire server.

```
/etc/init.d/openfire start
```

5) Login to the Openfire administration console. Navigate to **Server** > **Server Settings** > **Server Certificates**.

Example: https://<IP_of_Openfire_Server>:9091

6) Click on **Import**. The Import Signed Certificates screen opens. Paste the contents of the private key and certificate into their respective fields. Click **Save**.

NOTE 1: Contents of the certificate should also include the contents of the rootCA. First copy the contents of the certificate and then the contents of the rootCA.

NOTE 2: The common name of the certificate should be the same as the hostname of the openfire server.

NOTE 3: Two certificates should be added: one with RSA algorithm and one with DSA algorithm.

7) Once saved you are returned to the Server Certificates window. The new certificate should now show in the list. **Delete** the original certificates, both the RSA and DSA.

NOTE: If the new certificate is not showing in the list then the entire certificate's CA chain was not imported or the Openfire server was not restarted after importing the CA certificates. Openfire may also present a message stating it "failed to establish chain from reply". Ensure that all CAs in the certificate chain were imported.

8) Once the old certificates are deleted **restart** the HTTP server by clicking on **here** in the restart prompt dialogue in the same screen. Once the server restarts the new certificate will be active.

3.7 OpenScape Mobile (Apple and Android)

Certificate format: PEM – For the CA certificate.A PKCS#12 formatted keystore also containing the certificate authority certificate(s) for client side certificates.

Purpose: Secure communication between the OpenScape Mobile devices and OpenScape Façade, OpenScape Session Border Controller, and OpenScape Voice servers.

By default OpenScape Mobile does not require any certificates to be installed. The OpenScape Mobile device will still communicate over TLS to the OpenScape Façade, Session Border Controller, or Voice servers. However, it is strongly recommended to perform certificate validation to further protect the TLS connection. To perform certificate validation the CA certificate used to sign the

OpenScape Façade server must be installed on the OSMO device. This will allow the OpenScape Mobile device to verify the HTTPS connection to the OpenScape Façade server only.

Note: Future versions of OpenScape Mobile will allow for server certificate validation against the OpenScape Session Border Controller and Voice servers.

For the iOS (Apple) there is:

- OSMO

  NOTICE: OSMO is no longer supported for newer versions.

- OSMO Pro

  NOTICE: OSMO Pro will replace OSMO.

- MDM

3.7.1 Installing and Using the OpenScape Mobile CA Certificate on iPhone

Certificate format: PEM – For the CA certificate.A PKCS#12 formatted keystore also containing the certificate authority certificate(s) for client side certificates.

Purpose: Secure communication between the OpenScape Mobile / OpenScape Mobile Pro devices and OpenScape Façade, OpenScape Session Border Controller, and OpenScape Voice servers.

Note: The CA certificate must use the '.crt' filename extension in order for the installation/import to work.

Step by Step

1) Open an email containing the attached CA certificate.Open the attached CA certificate. **Note:** Take care to verify the email sender is trusted. Installing a CA certificate from an unknown source can result in compromised communications.

2) The **Install Profile** screen will appear and will contain the name of the CA certificate. Click on **Install**. The Install Profile dialogue will open to confirm the installation. Click on **Install Now**.

3) The certificate will be installed and should now show as **trusted**. Click on **Done** to return to your email.

4) Once the certificate has been installed navigate to the advanced settings tab in OpenScape Mobile. Set the **Allow Invalid Certificates** option to **Off**.

5) Certificates can be viewed and removed from the iPhone by navigating to **Settings** > **General** > **Profiles**.

3.7.2 Installing and Using the OpenScape Mobile CA Certificate for Android

Certificate format: PEM – For the CA certificate.A PKCS#12 formatted keystore also containing the certificate authority certificate(s) for client side certificates.

Purpose: Secure communication between the OpenScape Mobile devices and OpenScape Façade, OpenScape Session Border Controller, and OpenScape Voice servers.

Note: The CA certificate must use the '.crt' filename extension in order for the installation/import to work.

Step by Step

1) Open an email containing the attached CA certificate. Open the attached CA certificate. **Note:** Take care to verify the email sender is trusted. Installing a CA certificate from an unknown source can result in compromised communications.

2) Browse to the Android download folder. Click on the downloaded certificate. The **Name the certificate** dialogue will open. Enter a name and press **OK**. The certificate is now installed.

3) Once the certificate has been installed navigate to the advanced settings tab in OpenScape Mobile.**Uncheck** the **Allow Invalid Certificates** option.

4) Certificates can be viewed and removed from the Android phone by navigating to **Settings > Personal > Security > Trusted Credentials > User tab.**

3.7.3 Installing and Using the OpenScape Mobile Client Certificate

OpenScape Mobile has the ability to import and use client certificates. This allows the server side to also verify the OpenScape Mobile user (a mutually authenticated connection). This feature is only released for certain deployments. Customers should first contact technical support.

Step by Step

1) On the OpenScape Mobile device, open an email containing the client certificate. The certificate will have an extension of type <filename>.osmc.

2) OpenScape Mobile will prompt you to install the certificate. This is simply a password protected PKCS#12 keystore containing the client's certificate, key, and certificate authority certificates. The password should be delivered via a new email or other method.

3.8 OpenScape Mobile Client for Windows Phone 8

Certificate format: CER, P7B, PEM or PFX - For the CA certificate. Only server side authentication is supported for the OpenScape Mobile Client for Windows Phone 8 application.

Purpose: Secure communication between the OpenScape Mobile Client for Windows 8 devices and OpenScape Façade server.

Note: The CA certificate must use the '.cer, .p7b, .pem, or .pfx' filename extension in order for the installation/import to work.

OpenScape Mobile Client for Windows Phone 8 communicates over a secure connection, i.e. https. In order to use this secure connection, the CA certificate used to sign the Façade server certificate must first be installed on the Windows phone. These instructions are detailed below.

OpenScape Mobile Client for Windows Phone 8 does not have the ability to import and use client certificates in order to establish a mutually authenticated connection to the OpenScape Façade server. Therefore, the OpenScape Façade server must not be configured to allow client side certificates in deployments that support the operation Windows Mobile devices.

In general, all required certificates (e.g., Root CA and Intermediate CA Certificates) can be installed on the Windows phone using either of the following two methods:

- Installing certificates via email.

- Installing certificates via Internet Explorer.

3.8.1 How to Install a CA Certificate via email

Step by Step

1) Using the default email client, open an email from your system administrator that contains the root CA certificate (and intermediate CA certificates) as an attachment.

2) Tap on the email attachment to install the certificate. You can then review and install the certificate. Depending on the format of the certificate - '.cer, .p7b, .pem, or .pfx' - a password may be required to open it. The root CA and intermediate CA certificates have been installed on your phone.

3.8.2 How to Install a CA Certificate via Internet Explorer

Step by Step

1) Using Internet Explorer installed on your phone access the URL from which you can download the root CA (and intermediate CA) certificate.

2) When you access the page and tap the certificate it will be opened on the device. You can inspect the certificate and choose to continue. If you continue the certificate will be installed on the device. The root CA and intermediate CA certificates have been installed on your phone.

3.9 OpenScape Xpressions

In an OpenScape solution OpenScape Xpressions is installed as a voice-only configuration. This means all of the UC components which are already included in separate OpenScape products are not installed. This is the intended configuration method and therefore this section will only cover components which are part of voice-only solution. At this time this document covers the IPApl, SMTPApl, and WebApl. Please reference Appendix C of the OpenScape Xpressions Server Installation guide for instructions on installing OpenScape Xpressions as a voice-only configuration.

The IPApl can have its certificate information placed in any directory on the OpenScape Xpressions server. However, the SMTPApl and WebApl certificate location is coded in a Windows registry entry. To avoid copying the same certificate to different locations we will point the IPApl to this location to allow for easy certificate management. This location is <XPR Install>\res\certs, which is by default located at C:\Siemens\xpr\res\certs.

To change the certificate location of the SMTPApl and WebApl reference Appendix E.1.2 – Global Entries, in the OpenScape Xpressions Server Administration guide.

3.9.1 IPApl

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between OpenScape Xpressions and OpenScape Voice, OpenScape Xpressions and third party email servers, and OpenScape Xpressions and web browsers.

Step by Step

1) Place the certificate, key, and any CA certificates in the default certificate location of **C:\Siemens\xpr\res\certs** or **C:\OpenScape\xpr\res\certs**. The names of the certificate, private key and CA certificate should be the following:

- certificate: server.crt
- private key: server.key

- CA certificate: root.crt

INFO: Xpressions can trust multiple CAs. Place each CA to be trusted in a separate file. All CA certificates should be in one file.

2) Log into the Xpressions monitor and navigate to **IPApl > Advance Settings > Device > SIP Protocol Stack > Edit > Security**.

3) In the **Certificate Directory** field, enter the folder location defined in step 1).

4) In the **Own Certificate** field, enter the file name of the Xpressions certificate.

5) In the **Own Private key** field, enter the file name of the Xpressions key.

6) In the **Trusted Certificate Authorities** field, click on **Add** and enter the file name of the <CA certificate file>. These are the CA file names defined in step 1) above. Note: If there is more than one CA to be trusted repeat this step for each CA file.

7) Check the **Peer Authentication** checkbox.Remember, secure endpoints on the OpenScape Voice are MTLS connections. Also known as peer authenticated connections.

8) Navigate to the **SIP** tab and check the **TLS** radio button.

9) Click **OK** close the IPApl settings window. The IPApl will automatically restart. Monitor the Default Logging window for this restart and any errors related to enabling of TLS and certificates.

3.9.2 SMTPApl

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between OpenScape Xpressions and OpenScape Voice, OpenScape Xpressions and third party email servers, and OpenScape Xpressions and web browsers.

Step by Step

1) Place the <certificate>, <key>, and any <CA certificates> in the default certificate location of **C:\Siemens\xpr\res\certs**. All CA certificates should be in one file.

2) Log into the Xpressions monitor and navigate to **SMTPApl >Edit Settings à Secure Sockets**.

3) In the **Certificate file** field, enter the file name of the Xpressions <certificate>.

4) In the **Private Key** file field, enter the file name of the Xpressions <key>.

5) In the **Trusted Certificates List** field, enter the file name of the <CA certificates> file.

6) Clear the **Passphrase for Private Key** field. **Remember**, in these instructions the IPApl is using the same certificate as the SMTPApl and WebApl and the IPApl does not support encrypted private keys.

7) Check the Enable Secure Sockets checkbox.

8) The **Verify Client** and **Server Certificates** checkboxes do not need to be checked.

9) Click **OK** to close the SMTPApl settings window. The SMTPApl will automatically restart.

3.9.3 WebApl

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between OpenScape Xpressions and OpenScape Voice, OpenScape Xpressions and third party email servers, and OpenScape Xpressions and web browsers.

Step by Step

1) Place the <certificate>, <key>, and any <CA certificates> in the default certificate location of **C:\Siemens\xpr\res\certs**. All CA certificates should be in one file.

2) Log into the Xpressions monitor and navigate to **WebApl** > **Edit Settings** > **SSL**.

3) In the **Certificate file** field, enter the file name of the Xpressions <certificate>.

4) In the **Private Key fil**e field, enter the file name of the Xpressions <key>.

5) In the **Trusted Certificates List** field, enter the file name of the <CA certificates> file.

6) Clear the **Passphrase for Private Key** field. **Remember**, in these instructions the IPApl is using the same certificate as the SMTPApl and WebApl and the IPApl does not support encrypted private keys.

7) **OK** to close the WebApl settings window.The WebApl will automatically restart.

3.10 OpenScape Trace Manager

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between the OpenScape Voice Trace Manager and web browsers.

Step by Step

1) Rename the new certificate file to “**server.crt**”.

2) Rename the new key file to “**server.key**”.

3) Make backups of the existing certificate and key in the folder C:\Program Files (x86)\Apache Software Foundation\Apache2.2\conf\

4) Copy the new certificate and key into the folder defined in step 3).

5) Restart the Apache web server, Option 1: **Start > All Programs > Apache HTTP Server > Control Apache Server >Restart**

6) Restart the Apache web server, Option 2: **Start > Administrative Tools > Services**

   Right-click on the **Apache 2.2** service and select **Restart**

3.11 OpenScape Web Collaboration

By default the Web Collaboration server employs a small executable that allows clients to securely participate in Web Collaboration sessions. It is also possible for clients to participate in Web Collaboration sessions using their web browsers. For browser-based sessions to work a separate installation of the Web Collaboration Web Client server must be performed. This Web Client server is an instance of the Windows based IIS server. By default, this Web Client server installation operates over HTTP and not HTTPS and is therefore not secure.

Note: If you are not using browser based sessions, you can still install certificates on the base Web Collaboration servers to allow for secure download of the executable client.

Before proceeding with these steps it's expected the Web Collaboration Web Client server has already been installed and proper function verified.

Outline:

- Install the CA certificate(s). This applies to all web collaboration servers in the setup. **Note**: It's possible this step may be skipped if your server certificates are signed by trusted third party signatories, such as VeriSign or Thawte. Microsoft's internal certificate store already contains many trusted third party certificates.
- Install the server certificate for the base Web Collaboration server.
- Install the server certificate for the Web Client IIS web server.
- Enable HTTPS support in the Web Collaboration server.
- Enable HTTPS support in IIS and choose a new binding port (default 443).
- Configure the settings.ini file for the web collaboration server (not the Web Client).

3.11.1 Install the CA Certificate(s)

Certificate format: PEM – For CA certificates.

Purpose: Secure communication between the Web Collaboration Web Client server and the base Web Collaboration servers. Secure communication between the Web Collaboration Web Client server and web browsers. Secure download of the executable client from the Web Collaboration servers.

Step by Step

1) Click **Start**, click **Run**..., type **mmc**, and then press **ENTER**.
2) On the **File** menu, click **Add/Remove Snap-in**.
3) Under **Available snap-ins**, click **Certificates**, and then click **Add**.
4) Under This **snap-in will always manage certificates for**, click **Computer account**, and then click **Next**.
5) Click **Local computer**, click **Finish**, and finally click **OK**.
6) In the left hand pane navigate to **Certificates > Trusted Root Certification Authorities.**
7) Right-click the **Trusted Root Certification Authorities** and navigate to A**ll Tasks > Import….**
8) Follow the steps in the Certificate Import Wizard to import the <CA certificate>.

3.11.2 Install the Web Collaboration Server Certificate

Certificate format: PKCS#12 formatted keystore also containing the certificate authority certificates (if installing the Web Collaboration Web Client server).

Purpose: Secure communication between the Web Collaboration Web Client server and the base Web Collaboration servers. Secure communication between the Web Collaboration Web Client server and web browsers. Secure download of the executable client from the Web Collaboration servers.

Step by Step

1) Click **Start**, click **Run**..., type **mmc**, and then press **ENTER**.
2) On the File menu, click **Add/Remove Snap-in**.
3) Under **Available snap-ins**, click **Certificates**, and then click **Add**.
4) Under **This snap-in will always manage certificates for**, click **Computer account**, and then click **Next**.
5) Click **Local computer**, click Finish, and finally click **OK**.
6) In the left hand pane navigate to **Certificates > Personal**.
7) Right-click Personal and navigate to **All Tasks** > **Import…**.
8) Follow the steps in the Certificate Import Wizard to import the <server certificate>.

3.11.3 Install the Web Client Server Certificate

Certificate format: PEM – For CA certificates. PKCS#12 formatted keystore also containing the certificate authority certificates (if installing the Web Collaboration Web Client server).

Purpose: Secure communication between the Web Collaboration Web Client server and the base Web Collaboration servers. Secure communication between the Web Collaboration Web Client server and web browsers. Secure download of the executable client from the Web Collaboration servers.

Step by Step

› Repeat the steps in the previous section (and for any other Web Collaboration server, if any).

3.11.4 Enable HTTPS in the Web Collaboration Server

Step by Step

1) Click **Start**, **All Programs**, **Web Conference Server**, and then **Certificate Installer**. This will open a window for installing the Web Collaboration server certificate.

2) Click **Select Cert**. A window will open asking you to choose the certificate you previously installed. Select the certificate for the Web Collaboration server and click **OK**.

3) Under **IP Address**, enter the IP address of the Web Collaboration server. You can also use the drop down box.

4) Click **Install Cert**. The certificate for the Web Collaboration server has now been installed.

5) **Restart** the Web Collaboration server.

3.11.5 Enable HTTPS in the Web Client Server

Step by Step

1) Click **Start**, click **Administrative Tools**, and then click **Server Manager**.

2) Under Roles, select **Web Server (IIS)**, and then **Internet Information Services (IIS) Manager**.

3) Under the **Connections** pane, expand your computer name, expand **Sites**, right-click on **webclient** and select **Edit Bindings...**.

4) Select **Add**. Under **Type** select **https**.

Important: Under **IP Address**, choose the IP address assigned to the Web Client, **not** the web collaboration server. Under **Port**, set to port 443.

INFO: You can use a port of your choosing. This port must be the same as defined in the Web Collaboration settings.ini file variable **WebclientURLBase**.

---

5) Under **SSL Certificate**, select the certificate you previously installed. **Click OK** and then **Close**.

6) Under the **Connections** pane, click on your computer name, and then under the **Actions** pane under **Manage Server,** click **Restart**..

IIS setup is now complete.

3.11.6 Configure the settings.ini File

Step by Step

1) Open the settings.ini file for editing.

This is located under **C:\Program Files (x86)\WebConference-Server\settings.ini**.

2) At the end of the file, modify the following line, (changing from http to https).

INFO: If you are using a port other than 443, then this port number must also be defined on this line. This port number will be the same as defined in step 4) in the previous section. Both possibilities are shown below.

```
WebclientURLBase=https://<ip_address_of_web_client>/
joinclient.aspx?inv=%1
```

or

```
WebclientURLBase=https://
<ip_address_of_web_client>:<port_number>/
joinclient.aspx?inv=%1
```

3) Restart the Web Collaboration server.

Click on **Start**, **Administrative Tools**, then **Server Manager**. Expand **Configuration** and then click on **Services**. Scroll down, right click on **Web Conference Server** and select **Restart**.

At this point your web conferencing sessions are now secure.

3.11.7 OpenScape Web Collaboration Mobile App

Mobile users can participate in and create Web Collaboration sessions via the OpenScape Web Collaboration app available in the Apple and Android App stores. This web app connects to the OpenScape Web Collaboration Web Client server. If the OpenScape Web Collaboration Web Client server is accessed via a secure connection (HTTPS), then the Apple or Android phone must have the CA certificate used to sign Web Client server certificate installed. If this CA certificate is not installed then the OpenScape Web Collaboration mobile app will be unable to connect.

Step by Step

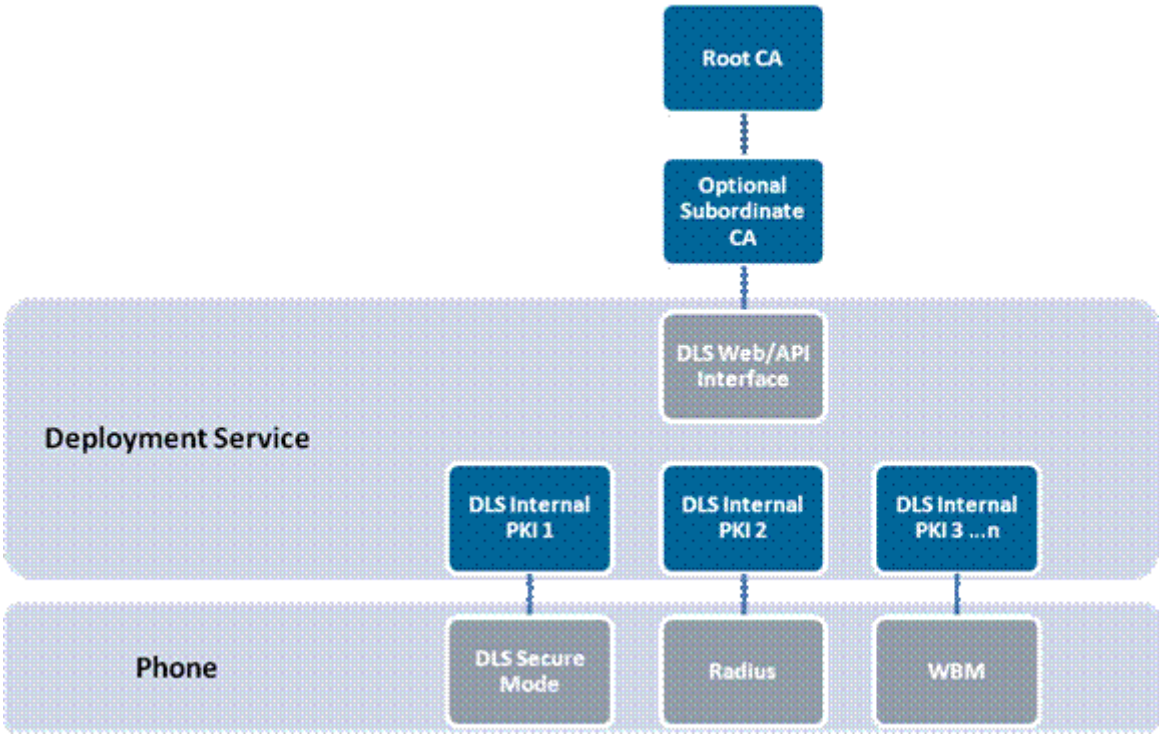
› Please refer to the section ’OpenScape Mobile (Apple and Android)’ in this document for instructions on installing the CA certificate.

3.12 OpenScape Deployment Service (DLS)

The OpenScape Deployment Service certificate functions operate differently than other Unify products. In order to support distribution of mass certificates to phones and clients the OpenScape Deployment Service supports its own internal public key infrastructure (PKI). This internal PKI operates like most other PKIs. It can create its own CA, create certificates signed by this CA, and even automatically distribute these certificates. On top of this, it can also maintain more than one internal PKI to allow for certificate management of different functions within the phone. Figures 6 and 7 show example certificate and PKI hierarchies.

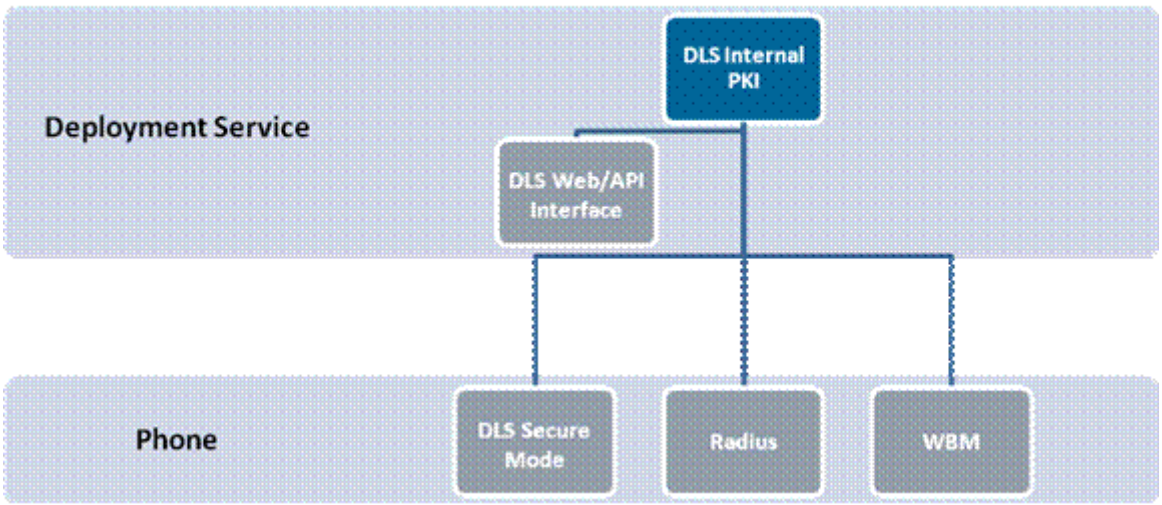
The OpenScape Deployment Service also supports integration with an external Microsoft certificate authority. In this case, the Deployment Service PKI depends on the Microsoft CA to manage and create certificates while the DLS handles the distribution aspect. Use of the Microsoft CA is not discussed in this document. Instead please refer to the OpenScape Deployment Service PKI Basic Configuration guide. Fig. 7 shows an example certificate hierarchy using a Microsoft certificate authority.

Figure: 6: Example Deployment Service certificate hierarchy (1)



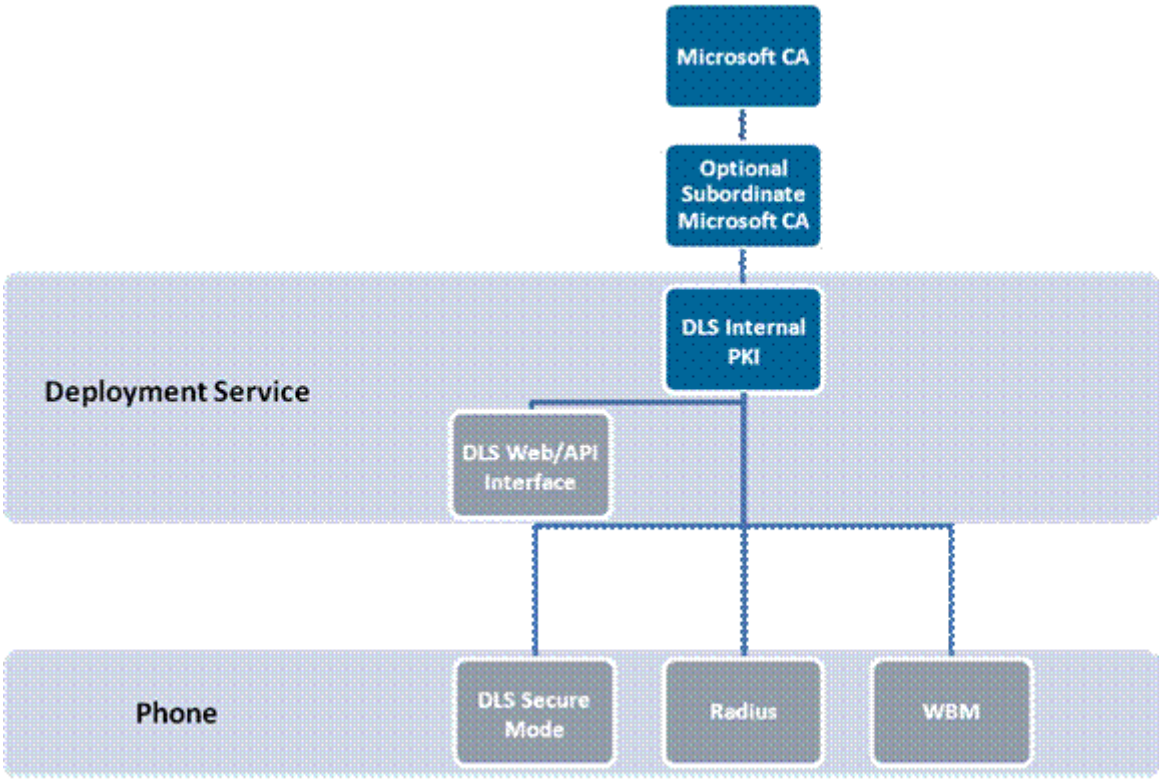
In the above case, the Deployment Service's public facing interfaces employ a certificate from the customers CA while the phones' certificates are managed by the Deployment Service's internal PKI. The Deployment Service can maintain more than one PKI to manage different phones functions requiring certificates. A single internal PKI can also be used.

Figure: 7: Example Deployment Service certificate hierarchy (2)



In the above case the Deployment Service uses a single internal PKI to manage both public facing Deployment Service interfaces and all phone functions requiring certificates.

Figure: 8: Example Deployment Service certificate hierarchy (3)



In the above case, the Deployment Service's internal PKI uses an external Microsoft certificate authority to provide all PKI functions. Usage of a Microsoft certificate authority is not discussed in this document.

3.12.1 Creating a New PKI

Note that it is possible to create more than one PKI. Different PKIs can be used for different purposes. For example, one PKI could manage all WBM certificates while another could manage RADIUS server certificate distribution. Or, one PKI could manage all certificates.

Creation of a new PKI follows three basic steps. Together they create a new PKI.

- Create a new Internal CA
- Create a new Plug-In Configuration
- Create a new Connector Configuration

3.12.1.1 Internal CA

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

Step by Step

1) Log into the Deployment Service and navigate to **Administration** > **PKI** > **Internal CA**.

2) Click on **New** and enter a **CA Name** and **CA Description**. Click on **Save**.

3) Click **Create CA**. Enter **Subject/Issuer DN** information.

This step defines the identity information of the CA and will also be seen in any certificate created by this CA. Certificates should at least include country, state or province, organization, organizational unit, and common name information.

Example:C=US,ST=FL,L=Boca Raton,O=Unify,OU=Sales,CN=MyNewCA

4) Alternate to step 3). It is also possible to create a new internal CA using a predefined CA. This CA certificate can be imported into the Internal CA.

Click on **Import CA**. **Browse** for the CA keystore. The keystore must be in a PKCS#12 format. Enter the keystore **Passphrase** and click on **OK**.

5) Set **Key Algorithm** to RSA. Set **Key Size** to 2048.

INFO: 1024 and 4096 are also possible.

6) Set **Valid from** and **Valid to** dates.

Default lifetime is 1 year time. Set this to a more acceptable lifetime for a CA, such as 10 years.

7) Click **OK**.

The new CA is created and this information can now be seen.

8) Finally, check **Enable Internal CA** and then **Save**.

3.12.1.2 Plug-In Configuration

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

Step by Step

1) Create a new Internal CA as done in the prior section, "Internal CA".

2) Navigate to **Administration** > **PKI** > **Plug-In Configuration**.

3) Click on **New** and provide a name and description in the **PKI Connector Plug-In** and **Description** fields. Click on **Save**.

4) Navigate to the **Plug-In Properties** tab and select the **Table** view radio button.

   There are two default certificate values that should be changed, **internal.default.validity.days** and **internal.x509name.template**.

5) Select **internal.default.validity.days**.

   This field defines the number of days certificates created by the Deployment Service are valid for. A typical length is 3 years. Click on **Save**.

6) Select **internal.x509name.template**.

   This field defines the subject information for certificates the Deployment Service will create. This field acts as a template. Most information is filled, except for the common name (CN). When issuing a certificate the Deployment Service will automatically input the CN based on the end points IP address or FQDN. This allows for easy point and click or automated certificate deployment. This template should match the issuer name information supplied for the root CA in step 1, except for the CN field. The CN field MUST contain a '?'. Click on **Save**.

   Example:C=US,ST=FL,L=Boca Raton,O=Unify,OU=Sales,CN=?

7) Click on the **Issuing CAs** tab. Click **Synchronize**.

   This will pull in all of the enabled internal CAs. We will only used the internal CA we created in step 1). This will be seen in the next steps.

8) Finally, check **Enable Plug-In** and Click on **Save**.

3.12.1.3 Connector Configuration

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

Step by Step

1) Create a new Internal CA and Plug-In Configuration as done above in sections 'Internal CA' and 'Plug-In Configuration'.

2) Navigate to **Administration** > **PKI** > **Connector Configuration.**

3) Click on **New** and provide a name and description in the **Configuration Name** and **Description fields**. Click **Save**.

4) Select **Plug-In Configuration**. Select the name of the Plug-In Configuration created in section 'Plug-In Configuration' and click on **OK** and then **Save**.

5) Select **Issuing CA Name**. Select the name of Issuing CA created in section 'Internal CA' and click on **OK** and then **Save**.

6) Select the **Trust-Anchor** tab, click on **Import Certificate**. Select the **PKI** radio button and under **Import from Connector** select the name of the issuing CA created in section ’Internal CA’. Click **OK** and **OK** again. The CA defined in section ’Internal CA’ is now the trust anchor (root CA) for this PKI. The trust anchor fields should now show this.

7) Navigate to the **Request Parameter** tab. The default settings should be acceptable. Click on **Test**.The Deployment Service will internally create and sign a new certificate to ensure proper operation. A success message should be returned.

8) Finally, check **Enable Connector** and then **Save**. At this point a new internal PKI has been created for the DLS. This new PKI can be used to create and deploy certificates to phones and clients.

3.12.2 Deploying the Signaling and Payload Encryption (SPE) Certificate

The SPE CA certificate is simply the CA certificate (or chain of CA certificates) used to sign the OpenScape Voice certificate. When this CA certificate is placed on the phones and soft clients, it allows the phones and soft clients to identify and verify the OpenScape Voice based on the certificate received from the OpenScape Voice. By default, phones and soft clients perform no certificate verification. Phones and soft clients can still connect to the OpenScape Voice via TLS and place secure calls.

In case phones and soft clients connect to the OpenScape Voice via MTLS an additional SPE certificate and a key are required in order to perform certificate verification.

Before enabling certificate verification for all phones and soft clients it is advised to manually place the certificate on a couple of devices and enable certificate verification. If the test fails, the device will fail to register against the OpenScape Voice. If the test succeeds, then automatic or manual certificate deployment can be used to distribute the certificate to all phones and soft clients.

INFO: It is suggested that all of the certificates that are used for the products of an OpenScape Solution are issued from a single Certification Authority. However it is possible to use certificates from different sources, if the CA certificate from the Certification Authority that issued the private certificate for the devices, is added to the Trusted CA Certificate Stores in CMP for OpenScape Voice. The same way phones should have a client certificate and a CA certificate from the authority that signed the OpenScape Voice certificate to validate the server certificate.

For further information please refer to Section "How to add CA certificates issued by different Certification Authorities" in this document.

3.12.2.1 Manual Deployment (TLS)

Certificate format: X.509 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/ clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

A PKI does not need to be created in order to execute this section.

Step by Step

1) Navigate to **IP Devices** > **IP Phone Configuration** > **Signaling and Payload Encryption** > **SPE CA Certificates** tab.
2) Click on **Import Certificate**.
3) Check **Import certificates to DLS and activate on device (1-step)**. Select **Import using:File**. Click on **Browse** and locate the file containing the CA certificates.
4) Finally, click **OK**.

   The certificate will be imported and a new entry will appear containing the CA certificate information.
5) You can now enable certificate verification. See section 'How to Enable SPE Certificate Verification'.

3.12.2.2 Manual Deployment (MTLS)

Certificate format: PKCS#12 formatted keystore. X.509 for the CA certificate.

Purpose: Secure communication between the OSV and phone(s).

Step by Step

1) Navigate to **IP Devices** > **IP Phone Configuration** > **Signaling and Payload Encryption** > **SPE Certificates** tab.
2) Click on **Import Certificate**.
3) Check **Import certificates to DLS and activate on device (1-step)**. Select **Import using:File**. Click on **Browse** and locate the file containing the certificates.
4) Click **OK**.
5) Select the **SPE CA Certificates** tab.
6) Click on **Import Certificate**.
7) Check **Import certificates to DLS and activate on device (1-step)**. Select **Import using: File**. Click on **Browse** and locate the file containing the CA certificates.

8) Click **OK**.

9) You can now enable certificate verification. See section ’How to Enable SPE Certificate Verification’.

3.12.2.3 How to Enable SPE Certificate Verification

The phones support three levels of certificate verification: none, full, and trusted. Each level performs more stringent checks. If a check fails on either the trusted or full levels, the phone will not be able to register against the OpenScape Voice or other SIP servers.

None: The default option. No checking of the received certificate is performed. The received certificate is only used to provide an encrypted connection. A CA certificate is not required to be loaded on the phone.

Trusted: The phone checks the chain of trust for the received certificate. A CA certificate is required to be loaded on the phone. The expiration date and revocation status are also check. Revocation is only check if an OCSP server has been defined.. No other fields are checked.

Full: Everything performed in “trusted” plus correct identity and correct use of all extensions marked as critical. Identity checks compare the FQDN, IPv4, or IPv6 address in the Common Name or Subject Alternative fields of the certificate.

To enable verification perform the following steps. **Note**: OpenStage phones running firmware version V3 or later will perform step 1).Older firmware versions and OptiPoint phones will perform step 2).

Step by Step

1) Navigate to I**P Devices > IP Phone Configuration > Security Settings > Certificate Policy** tab. Under **SIP Server Authentication Policy** choose a verification level. Click **Save**.

2) Navigate to **IP Devices > IP Phone Configuration > Signaling and Payload Encryption > SIP Setting**s tab. Check **TLS server validation.**

3.12.3 Deploying New Web Based Management (WBM) Certificates to Phones

Unify phones are administrable via a secure web based management interface and all phones ship with a default certificate. Certificates for the WBM interface can be deployed manually or automatically by the Deployment Service.

3.12.3.1 WBM Certificates to Phones via Manual Deployment

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

Step by Step

1) Navigate to **IP Devices > IP Phone Configuration > Security Settings > WBM Server Certificate** tab.

2) Click **Import Certificate**. For Certificate Type select **WBM Server Certificat**e. Check **Import Certificate to DLS and activate on device (1-step)**. Check **Import Using: PKI**. Under **Import from PKI** select the PKI you wish to use. Finally, click OK.

3) The certificate will be generated by the DLS and deployed to the phone automatically. Wait a few seconds then click on Refresh. You should now see the Imported and Active Certificate fields contain new certificate information. If only the Imported Certificate fields contains information then its possible the activate on device option was not selected in step 2). In this case check **Activate Certificate** and click on **Save**. This will activate the certificate on the phone.

3.12.3.2 WBM Certificates to Phones via Automatic Deployment

Certificate format:PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

A certificate deployed using automatic methods will only be visible under the active certificate field and not the imported certificate field. The imported certificate field is used for manual deployments. If you have deployed certificates in the past using the manual method then an older certificate may still be present. This does not impact the currently active certificate.

Step by Step

1) Navigate to **Administration > Automatic Certificate Deployment**.

2) Click on **New**. Under **Location** select a location. Under **Certificate Type** select **WBM Server Certificate (IP Phone)**. Click **Save**. Note: All phones defined by this location will receive new WBM certificates. Locations can be defined under **Administration** > **Server Configuration** --> **Location**.

3) Click **Import Certificate**. Check I**mport using: PKI.** Under **Import from PKI** select your PKI. Click **OK**.

4) Under **Deploy Date** select a date and time to deploy the certificates. Click **Save**.

5) Check the **Activate Certificate / PKI Configuration** checkbox. Click **Save**.

Once you click save the Deployment Service will automatically create jobs to deploy new WBM certificates to all phones defined by this location. Job completion can be monitored under **Job Coordination > Job Control**. Once the job is complete new certificate information can be viewed by navigating to **IP Devices > IP Phone Configuration > Security Settings > WBM Server Certificate** tab.

3.12.4 Phone Secure Mode Operation

By default, phones communicate with the Deployment Service using a standard TLS connection. However, if greater security is desired, phones can be set to communicate with the Deployment Service in secure mode. In secure mode communication takes place via mutual TLS (MTLS). MTLS offers the benefit of a mutually authenticated connection, meaning the Deployment Service verifies the phone and the phone verifies the Deployment Service. If either side cannot verify the other then the connection fails. This method provides an extra layer of protection for the phone as it will not allow another Deployment Service to connect it and manage the phone.

3.12.4.1 Set the Workpoint Interface Configuration PKI

Certificate format:PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers. This section changes the PKI used for secure mode operation of phones.

Step by Step

1) Create a new PKI or use an existing one. See section 'Creating a New PKI'.

2) Navigate to **Administration** > **Workpoint Interface Configuration**.

3) In the **Secure Mode** tab, under **Server Credentials**, select **PKI Configuration**. Select your PKI and click **OK** and then **Save**.

4) Next click **Create**. This creates a new server credential for the Deployment Service based on the PKI chosen in step 1. A new entry will appear in the table. It should be active by default. If it is not click on it and click **Activate**.

5) Repeat steps 2 and 3 for **Client Credentials**

6) After the new PKI has been activated, you can safely delete the default credentials.

3.12.4.2 Set the Phones to Secure Mode

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/ clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

Step by Step

1) Set the PKI in the Workpoint Interface Configuration. See previous section 'Set the Workpoint Interface Configuration PKI'.

2) To place phones in secure mode, navigate to **IP Devices > IP Device Management > IP Device Configuration > DLS Connectivity** tab.

3) Under **Security Settings** check **Secure mode required**. For **PIN Mode** you can choose between **No PIN**, **Default PIN**, or **Individual PIN**.

A PIN encrypts the phones certificate information as it is being transferred to the phone. The default option is Default PIN. This uses the PIN generated by the DLS. This can be found under the Workpoint Interface Configuration screen. If a PIN is used, then the PIN must be entered on the phone before secure mode operation is complete.

4) Click **Save**.

The Deployment Service will communicate new certificate information to the phone. If a PIN was used, then the PIN must be manually entered on the phone to complete the secure mode setup.

On an OpenStage phone login as admin and navigate to **Admin** > **Network** > **Update Service (DLS)**.

The **Security Status** field will say Awaiting PIN. Enter the PIN in the **Security PIN** field. Once the PIN is entered **Save & Exit**.Secure mode setup is now complete.

5) Transition from insecure to secure mode can be monitored under the **Security State Protocol** tab. A phone can be removed from secure mode by un-checking the **Secure mode required** checkbox.

INFO: If a phone is in secure mode and is unable to communicate with a DLS, it is possible to manually take the phone out of secure mode.

On an OpenStage phone, login as **admin and navigate to Admin > Network** > **Update Service (DLS) > Options** and select **Default Security**.

This will reset the phone it its default security mode. The phone will now be able to communicate with any Deployment Service.

If the phone is to communicate again with the existing Deployment Service, then the Deployment Service must also take the phone out of secure mode.

3.12.5 Replacing the Deployment Service's Web Interface and API Certificates

Certificate format: PKCS#12 formatted keystore or use of the Deployment Service's internal PKI infrastructure.

Purpose: Secure communication between the Deployment Service and phones/clients, Deployment Service and OpenScape Voice Assistant, and the Deployment Service and web browsers.

The Deployment Service contains a web and API interface. These interfaces are used to communicate with a web browser or the OpenScape Voice Assistant. For this reason, it may be desirable to have these interfaces participate in the customer's PKI rather than the Deployment Service's internal PKI.

Step by Step

1) Navigate to **Administration > Server Configuration > TLS Connector Configuration**.
2) Click **Import and Activate Certificate**. Select **DLS Client GUI.** Select **Import using: File** and then **Browse** for the PKCS#12 formatted keystore. Click **OK**. The certificate is imported and activated.
3) Repeat step 2) for the DLS API. Use the same PKCS#12 keystore.

3.13 OpenScape Branch and OpenScape SBC

This section covers replacing the certificates for the web interface of the OpenScape Branch and OpenScape SBC. It does not yet cover certificate management for the VoIP interfaces in these devices. Please refer to the OpenScape Branch or OpenScape SBC administrator documentation for VoIP interface certificate management.

3.13.1 Replacing the Web Interface Certificate

3.13.1.1 Creating a HTTPS Profile

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between the OpenScape Branch or OpenScape SBC and web browsers.

Step by Step

1) Login to the OpenScape Branch Management portal and navigate to **Security** > **Certificate Management**.

2) Certificates must be uploaded before creating the profile.

Under Certificates Upload you must upload the <CA certificate>, the <HTTPS certificate>, and the <HTTPS key>. This is performed under the sub-options **CA Certficates**, **X.509 Certificates**, and **Key Files**, respectively.

3) Once the file have been uploaded, click on **Add** under **Certificate Profiles**.

The certificate profile window will open.

4) For the **Certificate Profile Name**, enter a desired name.

5) For the **Local Server Certificate File**, select the <HTTPS server certificate> uploaded in step 2).

6) For the **Local CA File**, select the <CA certificate> uploaded in step 2).

7) For the **Local Key File**, select the <server key> uploaded in step 2).

8) Click on **Save**. The certificate profile window will close. The new profile will now appear under certificate profiles.

9) Click on **Save** again.

The certificate management window will close and the main window is displayed.

10) Click on **Apply Changes** to save the new certificate profile. If this is a redundant system all changes will be automatically copied to the redundant node. Allow up to 10 minutes for this to occur.

3.13.1.2 Applying the New HTTPS Profile - OpenScape Branch

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between the OpenScape Branch and web browsers.

Step by Step

1) Navigate to **Security**. Under **PKI Configuration**, check the **Enable PKI Configuration** box. This will activate the **PKI Configuration** button. Click on the **PKI Configuration** button. The PKI Configuration window will appear.

2) Under **HTTPS PKI Configuration**, select the desired **HTTPS Profile**.

3) Click on **Save**. The PKI Configuration window will close and the main window is displayed. Click on **Apply Changes** to activate the HTTPS profile.

If this is a redundant system all changes will be automatically copied to the redundant node. Allow up to 10 minutes for this to occur.

INFO: The web management portal will restart automatically. Access to the web management portal will be lost as the new certificate is applied. You may need to refresh your browser. Call processing is not affected. Allow up to 10 minutes for this to occur.

3.13.1.3 Applying the New HTTPS Profile - Session Boarder Controller

Certificate format: PEM – Key not encrypted.

Purpose: Secure communication between the OpenScape Session Boarder Controller (SBC) and web browsers.

Step by Step

1) Navigate to **Backup/Restore**. Click **Export** to export the latest configuration file.

2) Open the exported file into a text editor. Search for the string **httpscert-profile.** The result may appear in the following two formats:

<httpsCertProfile />

<httpsCertProfile>profile name</httpsCertProfile>

3) Using the name assigned to the newly created HTTPS profile, modify this string to include this name.

For example, if the name of the HTTPS profile is **'HTTPS-New'**, the new string would look as follows. **Note:** The entry is case sensitive.

<httpsCertProfile>**HTTPS-New**</httpsCertProfile>

4) Save the modified configuration file.

5) Navigate to **Backup/Restore** and click on **Import**.

The Import XML window will open.

6) Click on **Apply Changes** to activate the HTTPS profile.

If this is a redundant system, all changes will be automatically copied to the redundant node. Allow up to 10 minutes for this to occur.

INFO: The web management portal will restart automatically. Access to the web management portal will be lost as the new certificate is applied. You may need to refresh your browser. Call processing is not affected. Allow up to 10 minutes for this to occur.

3.14 OpenScape 4000/HiPath 4000/RG83xx

Certificate format: PEM - CA certificate and server certificate and key are PEM format.

Purpose: Secure communication between the OpenScape 4000/HiPath 4000/ RG8300 and OpenScape Voice.

The instructions in this section provide steps for adding or replacing signaling and payload encryption (SPE) certificates for SIP trunking in OpenScape 4000// RG83xx versions, V7 and V8 and V10. It does not provide instructions for any other certificate areas. Refer to the individual product manuals if certificates are needed in these other areas.

3.14.1 Adding a CA Certificate

The OpenScape 4000/HiPath 4000/RG83xx can utilize more than one CA certificate at a time. This allows the OpenScape 4000/HiPath 4000/RG83xx devices to interact with multiple endpoints signed by different certificate authorities.

Perform the following steps to add a CA certificate.

Step by Step

1) Login into the desired board and navigate to **Explorers** > **Security** > **Signaling and Payload Encryption (SPE)** > **SPE CA Certificate(s)**.

2) Right-click **SPE CA Certificate(s)** and select **Import trusted CA Certificate**.

The **Load a SPE CA Certificate via HTTP** window will appear.

3) Click on **Browse** and locate the <CA certificate> to be imported.

4) Click **View Fingerprint of Certificate**.

This will display the contents of the certificate.

5) Click **OK** to close this window.

6) Click **Import Certificate from File**.

The <CA certificate> will be imported. A new entry will be seen under the **SPE CA Certificates** menu.

7) Click the **Diskette Icon** on the bottom of the screen to save changes.

8) Click the **Reset Icon** on the bottom of the screen to reboot and activate changes.

INFO: If also adding a new server certificate then add and save that certificate before restarting.

3.14.2 Adding or Replacing the Server Certificate

The OpenScape 4000/HiPath 4000/RG83xx can only have one server certificate. Therefore, this certificate can only be added if a certificate was not previously installed, or it can be replaced. After this certificate is installed the system will require a restart.

Perform the following steps to add or replace the server certificate.

Step by Step

1) Login into the desired board and navigate to **Explorers** > **Security** > **Signaling and Payload Encryption (SPE)** > **SPE Certificate**.
2) Right-click **SPE Certificate** and select **Import SPE certificate plus private key**.

   The **Load a SPE Key Certificate via HTTP** window will appear.
3) Click on **Browse** and locate the <server certificate> file to be imported. Both the certificate and key must be in the same file.
4) Click **View Fingerprint of Certificate**.

   This will display the contents of the certificate.
5) Click **OK** to close this window.
6) Click **Import Certificates from File**.

   The <server certificate> will be imported. The entry under the **SPE Certificates** menu will be updated with the new certificate.
7) Click the **Diskette Icon** on the bottom of the screen to save changes.
8) Click the **Reset Icon** on the bottom of the screen to reboot and activate changes.

3.15 RG8700

Certificate format: PEM - CA certificate and server certificate and key are PEM format.

Purpose: Secure communication between the RG8700 and OpenScape Voice.

The following steps will replace the CA certificate, server.pem, and client.pem files on the RG8700 series.

> ***INFO:*** We are not updating the dh1024.pem file. This file can be left as is.

Step by Step

1) Obtain the proper CA certificate for your solution and rename it to **CAcert.pem**.
2) Concatenate your server certificate and key into a single file and name it **server.pem**.

   ```
   cat rg8700key.pem rg8700cert.pem > server.pem
   ```
3) Copy server.pem to **client.pem**.

INFO: The client.pem file is designed for users who distinguish the usage of their certificates between server and client only connections. This requires the server.pem certificate contain an extended key usage setting for "TLS Web server authentication" and for client.pem to contain an extended key usage setting for "TLS Web client authentication". In practice, most people do not set these values, or they contain both values. This allows the certificate to be used for both purposes. If you did not set these values or do not know then simply copy server.pem to client.pem.

4) Open a SFTP session to the RG8700 and upload all three files (**CAcert.pem**, **server.pem**, and c**lient.pem**) to the folder **/ad0/data/ssl/certs**.

5) Restart the RG8700 for the new certificates to take effect.

3.16 Mediatrix

Certificate format: PEM - CA certificate and server certificate and key are PEM format.

Purpose: Secure communication between the Mediatrix and OpenScape Voice.

Mediatrix devices can support multiple CA certificates and server certificates at the same time. However, once a second server certificate is loaded, the Host Certificate Associations should not overlap. Meaning, do not use more than one server certificate for a single service.

3.16.1 Adding a CA Certificate

Step by Step

1) Login to the Mediatrix and navigate to **Management** > **Certificates**.

2) Under **Certificate Upload Through Web Browser**, select **Type of Other** and then click **Browse** and select the <CA certificate> to be uploaded.

3) Once the certificate has been selected, click **Upload Now**.

The new CA certificate will appear on the same screen under **Other Certificates and Certificate Authorities**.

4) Activate the new CA certificate by navigating to **System** > **Services**. Under **User Services,** services which need to be restarted will be highlighted in bold and also have the text **Restart needed**. Restart these services.

An outage will occur. After services have restarted the newly uploaded CA will be active.

3.16.2 Adding a Server (Host) Certificate

INFO: Please reference the Mediatrix requirement at the beginning of this chapter.

The <file> to be uploaded must have the certificate and key in a single file. The certificate should be first, followed by the key, and then followed by any CA certificates making up the trust chain.

Step by Step

1) Login to the Mediatrix and navigate to **Management** > **Certificates**.

2) Under **Certificate Upload Through Web Browser**, select **Type of Host** and then click **Browse** and select the <server (host) certificate> to be uploaded.

3) Once the certificate has been selected, click **Upload Now**.

The new server (host) certificate will appear on the same screen under **Host Certificates**.

4) Proceed to the next section before restarting services.

3.16.3 Setting the Host Certificate Associations

Once any number of server (host) certificates have been uploaded it is possible to assign a certificate to individual services. However, multiple certificates should not be assigned to the same service..

Step by Step

1) Login to the Mediatrix and navigate to **Management** > **Certificates**.

2) Under **Host Certificate Associations**, check which service will use a certificate. If only a single certificate is used, then check all services.

3) Click the **Submit** button to save changes.

4) Restart services. Navigate to **System** > **Services**. Under **User Services,** services which need to be restarted will be highlighted in bold and also have the text **Restart needed**. Restart these services.

An outage will occur.

3.17 Attendant Supervisor Console (ASC)

Certificate format: PEM - CA certificate and server certificate and key are PEM format.

Purpose: Secure communication between the ASC and OpenScape Voice.

INFO: This certificate MUST have the same certificate authority as the OSV certificate.

3.17.1 Import the Certificate of OpenScape Voice to the Attendant Supervisor Console (ASC)

Note: The installation requires administrator rights to be carried out.

Step by Step

1) Copy the certificate file **root.pem** of OpenScape Voice (/usr/local/ssl/certs/) to the recording server via WinSCP.

2) Navigate to the **C:\Program Files (x86)\ASC\ASC Product Suite\scripts** directory and execute **certimporter.exe**.

3) In the displayed window, select **PBX Trust** on the side menu.

4) Click on the **...** button next to the **Certificate X.509** field and point to **the root.pem** file that you have copied from the PBX. Then click Open.

5) Click **Open**.

6) In the pop-up window specify the alias for the certificate and click **OK**.

7) A second pop-up window is displayed after **root.pem** is successfully imported. Click **OK** to proceed.

8) Click **Exit** in order to close the program.

3.17.2 Import the Certificate of the Attendant Supervisor Console (ASC) to OpenScape Voice via CMP

Step by Step

1) Backup the OSV **root.pem** from both OSV nodes.

2) Navigate to **Configuration** > **OpenScape Voice** > **Certificate Management** > **Trusted CA Certificate Store**.

3) Select the **root.pem** and click **Edit**.

4) Click **Browse**, select the ASC certificate and click **Upload**.

5) Click **Save**.

6) Verify that **root.pem** was updated for both OSV nodes.

3.18 OpenStage, OptiPoint, and OpenScape Soft-Clients

OpenStage, OptiPoint, and the OpenScape soft-clients - such as OpenScape PE, OpenScape WE, and OpenScape Fusion - register with SIP endpoints, such as OpenScape Voice, OpenScape SBC, or OpenScape OSB.

OpenStage and OptiPoint devices will require a SPE CA certificate if they are connecting via TLS or a SPE and a SPE CA certificate if they are connecting via MTLS.

> ***INFO:*** OpenScape soft-clients do not perform certificate validation of SIP endpoints at this time.

To deploy CA certificates for OpenStage and OptiPoint devices please refer to Section "Deploying the Signaling and Payload Encryption (SPE) Certificate" in this document.

OpenScape soft-clients - such as OpenScape WE and OpenScape Fusion clients - always connect via HTTPS to the OpenScape UC WebClient. If the CA certificate which signed the OpenScape UC WebClient server certificate is not installed on the client's PC, then the soft-clients will report certificate errors. These errors cannot be suppressed. A valid CA certificate must be installed. Please reference the section titled "Importing CA Certificates into Internet Explorer" in this document.

3.19 OpenScape Contact Center

3.19.1 OpenScape Contact Center - Application Server

The Application Server is responsible for providing access the OSCC web services. For the Agent Portal Web, it connects to the Corporate E-mail Server to download the e-mail content to the agent and it also connects the LDAP Server for Directory Search.

3.19.1.1 Web server

A certificate set for the web server, including the CA Certificate, Server Certificate and Private Key is required

Certificate format: PEM - For the CA certificate, server certificate and private key (non encrypted).

Purpose: This certificate shall be used by the Application Server as web server. It shall secure the following connections to the Application Server:

- Agent Portal Web
- Mobile Supervisor
- Web Supervisor

- REST SDK
- OpenMedia Connector
- Web Manager

The OpenScape Contact Center Application Server uses a keystore in format JKS or PKCS12. In order to replace the default certificate by a certificate generated by a Certificate Authority in the OpenScape Contact Center Application Server, the following process should be followed:

- Obtain Certificate from the Authority.
- Identify the Java instance which is being used by Tomcat in the Application Server. Verify environment variable JRE_HOME
- Move to the JRE_HOME folder, as for example:
  `cd "\Program Files (x86)\Java\jre1.8.0_161"`

  NOTICE: In OpenScape Contact Center V10, IBM Java is used and JRE_HOME folder by default is:`\Program Files\OpenScape\Contact Center\Java\IBM\jre`.

- Import the Chain Certificate (CA Certificate) into the keystore (only if not already there):`bin/keytool -import -alias root -keystore %JRE_HOME%\lib\security\cacerts -trustcacerts -file <file_name_of_the_chain_certificate>`

  The `cacerts` file is located in the following path:
  `%JRE_HOME%\lib\security`
- "If the certificate is not in a pkcs12 keystore, create the pkcs12 keystore from certificate and private key .pem files:`openssl pkcs12 -export -in .\<server certificate> -inkey .\<server private key> -out mystore.p12 -name tomcat`
  By executing this command a password will be requested for the pkcs12 keystore (mystore.p12)
-
- Tomcat can access the Certificate and Private key in two ways: as a pkcs12 keystore or as a Java keystore:
  - If the decision is to use the pkcs12 keystore (mystore.p12), just configure the server.xml file with the proper keystore path plus password, as for example:`<Connector port="443" protocol="org.apache.coyote.http11.Http11NioProtocol" maxThreads="150" SSLEnabled="true" scheme="https" secure="true" clientAuth="false" sslProtocol="TLS"> keystoreFile="${catalina.base}\conf\mystore.p12" keysto-rePass="<PKCS#12 keystore password>" keystoreType="PKCS12"/>`
  - If the decision is to use a Java keystore, then the pkcs12 keystore must be imported into a Java keystore. The following command shall be used:
    `.\bin\keytool.exe -importkeystore -srckeystore <path>\mystore.p12 -srcstoretype PKCS12 -destkeystore "${catalina.base}\conf\keystore.jks" -deststorepass`

```
<Java keystore password> -storepass <Java keystore
password>
```

NOTICE: By executing this command a password will be requested for the pkcs12 keystore (mystore.p12).

NOTICE: No alias parameter shall be entered.

The keystore name and the corresponding keystore password shall be configured in the file Server.xml which is also in the conf folder as in the example below: `<Connector port="443" protocol="org.apache.coyote.http11.Http11NioProtocol" maxThreads="150" SSLEnabled="true" scheme="https" secure="true" clientAuth="false" sslProtocol="TLS"> keystoreFile="${catalina.base}\conf\keystore.jks" keystorePass="<Java keystore password>" keyPass="<PKCS#12 keystore password"/>`

3.19.1.2 E-mail IMAP Client

The CA certificate which was used to sign the certificate of the corporate e-mail server must be installed in the Java trusted keystore.

Certificate format: DER (extension .cer) - For the CA certificate.

Purpose: This certificate shall be used by the Application Server to validate the certificate which is received from the Corporate E-mail Server.

In order to install the CA Certificate, follow this procedure:

- Obtain the CA Certificate from the Corporate E-mail Server.
- Identify the Java instance which is being used by Tomcat in the Application Server. Verify environment variable JRE_HOME
- Move to the JRE_HOME folder, as for example:
  `cd "\Program Files (x86)\Java\jre1.8.0_161"`
- Execute the following command to install the certificate in the trusted keystore of Java
  ```
  bin\keytool.exe -keystore lib\security\cacerts -
  importcert -alias <aliasname> -file <file.cer>alias -
  enter a name which identifies the certificate in the
  keystore
  ```
- Restart the service Agent Portal Java.

3.19.1.3 LDAPS Client

The CA certificate which was used to sign the certificate of the LDAP server must be installed in the Java trusted keystore.

Certificate format: DER (extension .cer) - For the CA certificate.

Purpose: This certificate shall be used by the Application Server to validate the certificate which is received from the LDAP Server.

In order to install the CA Certificate, follow this procedure:

- Obtain the CA Certificate from the LDAP Server
- Identify the Java instance which is being used by Tomcat in the Application Server. Verify environment variable JRE_HOME.
- Move to the JRE_HOME folder, as for example:
  `cd "\Program Files (x86)\Java\jre1.8.0_161"`

> ***NOTICE:*** In OpenScape Contact Center V10, IBM Java is used by default and the JRE_HOME folder is: `\Program Files\OpenScape\Contact Center\Java\IBM\jre`

- Execute the following command to install the certificate in the trusted keystore of Java:
  `bin\keytool.exe -keystore lib\security\cacerts -importcert -alias <aliasname> -file <file.cer>alias - enter a name which identifies the certificate in the keystore`
- Restart the service OpenScape Contact Center Application Service.

3.19.2 OpenScape Contact Center - OpenFire Server

Certificate format: PEM - For the server certificate and private key (non encrypted).

Purpose: Secure communications between the Agent Portal and the OpenFire server for Chat between agents or between agents and supervisors and for the access to the administration web interface.

> ***IMPORTANT:*** Two default self-signed certificates, being one for RSA (<hostname>_RSA), the another one for DSA (<hostname>_DSA) and private key are installed automatically on the OpenFire machine as part of the installation process. However, these files must be replaced with a trusted certificate obtained from a recognized certification authority. All other self-signed certificates will be rejected.

Installing a certificate for the OpenFire Server:

- Go to the configuration web site of the OpenFire server.
- Log in to the configuration site under the admin user account.
- Click on the tab **TLS/SSL Certificates**.
- Scroll down to **XMPP Client Stores**.
- On the **Identity Store** area, click on the link **Manage Store Contents**.
- Click on the link **imported here** to import the new certificate.

- Open the key file with a text editor and copy-paste the content to the area labeled as **Private Key**.
- Open the certificate file with a text editor and copy+paste the content as is to the area labeled as **Certificate**.
- Delete the default certificates.
- Restart the Openfire service.

3.19.3 OpenScape Contact Center - Web collaboration server

Certificate format: PEM - For the CA certificate

PKCS12 - For the server certificate and private key (password is optional).

Purpose: Secure communications between the Corporate Web Server and the OpenScape Contact Center Web Interaction Server.

To install a certificate for a secure Web connection:

- Ensure that SSL security is enabled on the corporate Web server. For details, see the *OpenScape Contact Center - System Management Guide*.
- In the Manager application, an SSL-enabled port must be selected for the Web connection. Default port is 8443.
- The common name of the certificate must be the host name of the OpenScape Contact Center main server machine. The same host name must be configured in the OpenScape Contact Center Web components on the corporate Web server.

  NOTICE: In a high availability (warm standby) environment, the common name of the certificate must be the cluster Server Name in the HPPC Group Resources (Client Access Point) and the certificate must be installed on both the primary and backup server machines.

- Log on to the OpenScape Contact Center main server machine under the **hppc** user account.
- Use the Microsoft Management Console to import the certificate and install the certificate in the My certificate store (the personal certificate store).

  NOTICE: It may be necessary to install the CA Certificate on the Corporate Web Server on which the OpenScape Contact Center Web Chat component is running.

  NOTICE: : If the policy **System cryptography: Force strong key protection for user keys stored on the computer** is set to **User must enter a password each time they use a key** and a new certificate must be installed for the Web Interaction Server, the policy must be changed to **User input is not required when new keys are stored and used** before installing the certificate.

After the certificate is installed the policy can be set back to **User must enter a password each time they use a key**.The Import option **Enable strong private key protection. You will be prompted every time the private key is used by an application if you enable this option** on the Certificate Import Wizard screen must be unchecked

This is the sequence of actions to install the certificates in the OpenScape Contact Center server via mmc:

- Navigate to **File > Add-Remove Snap-in...** and select **Certificates**.
- Click the **Add >** button and select to manage certificates for **My user account**.
- Click the **Add >** button again and select to manage certificates for the computer account.
- In the **Console Root > Personal > Certificates** and import the certificate (***.cer** file). into both the personal certificate store of the current user and the CA certificate into the trusted root CA store for the local computer.
- Follow the same procedure for your local computer by navigating to **Console Root > Certificates (Local Computer) > Trusted Root Certification Authorities > Certificates** and import the certificate (***.cer** file) into the trusted root CA store for the local computer.
- Make sure that the **Client Authentication** is not selected (check certificate properties).

3.19.4 OpenScape Contact Center - Network Communication

Certificate format: PEM - For the CA certificate, server certificate and private key (non-encrypted).

Purpose: Secure communications between the client applications, that means Agent Portal, Client Desktop, Manager and System Monitor and the OpenScape Contact Center Main Server.

To install a certificate for a network communication:

- Log on to the main server machine under the **hppc** user account.
- Stop the OpenScape Contact Center service on the main server machine. Wait for the service to completely shut down before proceeding.
- Replace the default self-signed certificate (`oscc_cert.pem`) and private key (`oscc_privkey.pem`) from Unify with your own certificate and private key.
- Start the OpenScape Contact Center service on the main server machine.

NOTICE: A known Certificate Authority shall already be included in the default trusted keystore of Java on the Agent Portal computers. If the CA Certificate is not included in the default trusted keystore, it is possible to add the CA Certificate to the trusted keystore.

The certificates for Network Communication can also be created from an Own Root CA for OpenScape Contact Center.

> ***INFO:*** This description was adapted from the chapter Creating Your Own Root CA of this document. This procedure uses OpenSSL commands. You can use other third party PKI tools to create a CA and manage certificates. They will be much easier to use and manage.

Simple Certificate Creation

- Create the CA configuration file. Copy the contents of the below text into a file named ca.cnf.

  ```
  ## CA configuration file. #HOME = . [ req ] default_bits
  = 2048 default_md = sha256 distinguished_name =
  req_distinguished_name x509_extensions = v3_ca [
  req_distinguished_name ] countryName = Country Name (2
  letter code) countryName_default = US countryName_min = 2
  countryName_max = 2 stateOrProvinceName = State or
  Province Name (full name) localityName = Locality Name
  (eg, city) organizationName = Organization Name (eg,
  company) organizationalUnitName = Organizational Unit
  Name (eg, section) commonName = Common Name (eg, CA name)
  commonName_max = 64 [ v3_ca ] keyUsage = keyCertSign,
  cRLSign subjectKeyIdentifier=hash authorityKeyIden-
  tifier=keyid:always,issuer:always basicConstraints =
  CA:true #EOF
  ```

- Create the server configuration file. Copy the contents of the below text into a file named server.cnf.

  ```
  ## Server configuration file. If you are creating certif-
  icates for # more then one server and are using subAltName
  then you should # create a copy of this file for each
  server. #[req] default_bits = 2048 default_md = sha256
  distinguished_name = req_distinguished_name
  req_extensions = v3_req [req_distinguished_name]
  countryName = Country Name (2 letter code)
  countryName_default = US countryName_min = 2
  countryName_max = 2 stateOrProvinceName = State or
  Province Name (full name) localityName = Locality Name
  (eg, city) organizationName = Organization Name (eg,
  company) organizationalUnitName = Organizational Unit
  Name (eg, section) commonName = Common Name (eg, FQDN)
  commonName_max = 64 [v3_req] keyUsage = keyEncipherment,
  digitalSignature, nonRepudiation subjectKeyIdentifier =
  hash basicConstraints = CA:false #EOF
  ```

- Create the root CA certificate. This single step will create a self-signed root CA that will be used to sign server certificates. You will be prompted to enter

organization information such as country, state, city and etc. Be aware that the file names used in this command will be used in following steps.
openssl req -new -x509 -keyout cakey.cer -out cacert.cer -nodes -days 3650 -config ca.cnf

- The file server.cnf is a generic configuration file that can be used to create many server certificates. You will be prompted to enter organization information such as country, state, city and etc. The final prompt will ask for the common name.

  NOTICE: The organization information which is provided for the generation of the root certificate must be different from the organization information which is provided for the generation of the server certificate.

  – Create the key and signing request. openssl req -new -keyout server1key.pem -out server1req.pem -nodes -config server.cnf
  – Sign the request using the CA certificate and key created in step 3. openssl x509 -req -sha256 -CA cacert.cer -CAkey cakey.cer -CAcreateserial -extfile server.cnf -extensions v3_req -in server1req.pem -out server1cert.pem -days 3650
  – View your newly created certificate using the following command: openssl x509 -text -noout -in server1cert.pem
- Follow the procedure described above in this section to replace the certificate in the OpenScape Contact Center main server.
- Add the root certificate file (cacert.cer) to the Windows Trusted Root Certification Authorities on the server which is running the OpenScape Contact Center server **Control Panel > Network and Sharing Center > Internet Options > Content > Certificates**
- Add the root certificate to the keystore of the JDK in the clients on which Agent Portal is running. Go to `jdk\jre\lib\security` folder and import the certificate via the following command:
  keytool -keystore cacerts -importcert -alias <common name> -file cacert.cer.
  The default password of the jdk keystore is `changeit`
  When prompted to trust this certificate, respond by typing y.Run the following command to ensure the CA certificate has been successfully imported:
  keytool -list -keystore cacerts

3.19.5 OpenScape Contact Media Service

Certificate format: PEM - For the CA certificate, server certificate and private key (non-encrypted)

Purpose: secure communication for

- OpenScape Contact Media Service web interface for configuration.
- Communication between OpenScape Contact Media Service and the OpenScape Contact Center main server - in this interface the OpenScape Contact Media Service is always the TLS server.

- Communication between OpenScape Contact Media Service and the Communication System - in this interface the OpenScape Contact Media Service is always the TLS client.

The certificates for the OpenScape Contact Media Service are installed by means of the web interface, by selecting Security Configuration and then Certificates.

To install the certificate for the web interface, the certificate files and private key file shall be selected and uploaded to the corresponding fields Public Certificate, Private Key and CA Certificate for HTTPS Certificates.

NOTICE: If the certificate was not generated by a well-known Certificate Authority, the CA Certificate must be installed in the browser.

To install the certificate for the interface to the OpenScape Contact Center Main Server, the certificate files and private key file shall be selected and uploaded to the corresponding fields Public Certificate, Private Key and CA Certificate for RPC/TLS Certificates.

NOTICE: if the certificate was not generated by a well-known Certificate Authority, the CA Certificate must be installed in the Windows certificate repository of the OpenScape Contact Center Main Server.

To install the certificate for the interface to the Communication Systems (OpenScape Voice, OpenScape 4000 and OpenScape Business), the Certificate Authority certificate file shall be selected and uploaded to the corresponding field CA Certificate for SIP/TLS Certificates.

3.19.6 Facebook Connector

For OpenScape Contact Center V11, during the installation of the Facebook Connector, the user is requested to install the certificates. All the certificate and key files must be copied into a folder:

For the Webhook interface

- **ca.crt** (CA certificate),
- **server.crt** (server certificate)
- **key.crt** (private key)

For the Administration Web interface

- **ca.pem** (CA certificate),
- **server.pem** (server certificate)
- **key.pem** (private key)

The path of the folder must be entered in the Certificates Configuration screen.

NOTICE: If the certificates must be changed after installation the procedure described in the following chapters, must be followed for each specific interface.

3.19.6.1 Webhook Interface

Certificate format: PEM (.crt extension) - For the CA certificate, Connector server certificate and Connector private key (non encrypted)

Purpose: Secure communications between Facebook Connector and Facebook Server.

NOTICE: This certificate is used for Facebook Webhook feature, which requires that the Facebook Connector has a web server to receive the notifications about published ports and received messages to the monitored Company Facebook Pages. Facebook requires that this connection uses HTTPS and that the web server provides a valid certificate.

- The certificates must be renamed to `ca.crt` (CA Certificate), `server.crt` (server certificate) and `key.crt` (private key).
- The certificate files and the Key files must be copied to `...\social-media-hub-api\cert`, replacing the old certificate files.
- The Facebook Connector must be restarted.

3.19.6.2 Administrator Web Interface

Certificate format: PEM - For the CA Certificate, Connector server certificate and Connector private key (non encrypted).

Purpose: Secure communications between the Facebook Connector and the Facebook Server.

- The certificates must be renamed to `ca.pem` (CA certificate), `server.pem` (server certificate) and `key.pem` (private key).
- The cerificate files and Key files must be copied to `...\facebook-connector\cert`, replacing the old certificate files.
- The Facebook Connector must be restarted.

3.19.7 Twitter Connector

Certificate format: PKCS#12 - for the CA certificate, Connector server certificate and Connector private key with password.

Purpose: Secure communications for the Webhook connection from the Twitter server, for Webhook connection from the Application Server and for the administration of the Twitter Connector.

To configure certificate in Twitter Connector execute the following steps:

- Open the files **application_oscc.properties** and **application_twitter.properties** in the installation folder. The default is:
  `C:\Program Files\OpenScape\Contact Center\OpenMedia\TwitterConnector`
- Fill the parameters:
  server.ssl.key-store with path to the keystore containing the certificate**;** **server.ssl.key-store-password** with the password used to generate the certificate.
- Save changes.
- Restart Twitter Connector service.

3.19.8 Agent Portal Java

NOTICE: For the communication with the OpenScape Contact Center Main Server, refer to section OpenScape Contact Center - Network Communication.

3.19.8.1 E-mail IMAP Client

The CA certificate which was used to sign the certificate of the corporate e-mail server must be installed in the Java trusted keystore.

Certificate format: DER (extension .cer) - For the CA certificate.

Purpose: This certificate shall be used by the Application Server to validate the certificate which is received from the Corporate E-mail Server.

In order to install the CA Certificate, follow this procedure:

- Obtain the CA Certificate from the Corporate E-mail Server.
- Identify the Java instance which is being used by Tomcat in the Application Server. Verify environment variable JRE_HOME
- Move to the JRE_HOME folder, as for example:
  `cd "\Program Files (x86)\Java\jre1.8.0_161"`
- Execute the following command to install the certificate in the trusted keystore of Java
  `bin\keytool.exe -keystore lib\security\cacerts -importcert -alias <aliasname> -file <file.cer>alias - enter a name which identifies the certificate in the keystore`
- Restart the service Agent Portal Java.

3.19.8.2 LDAPS Client

The CA certificate which was used to sign the certificate of the LDAP server must be installed in the Java trusted keystore.

Certificate format: DER (extension .cer) - For the CA certificate.

Purpose: This certificate shall be used by the Application Server to validate the certificate which is received from the LDAP Server.

In order to install the CA Certificate, follow this procedure:

- Obtain the CA Certificate from the LDAP Server
- Identify the Java instance which is being used by Tomcat in the Application Server. Verify environment variable JRE_HOME.
- Move to the JRE_HOME folder, as for example:
  `cd "\Program Files (x86)\Java\jre1.8.0_161"`
- Execute the following command to install the certificate in the trusted keystore of Java:
  `bin\keytool.exe -keystore lib\security\cacerts -importcert -alias <aliasname> -file <file.cer>alias - enter a name which identifies the certificate in the keystore`
- Restart the service Agent Portal Java.

3.19.8.3 Email Relay Server

The Email Relay Server is used to be a relay between the OpenScape Contact Center components and the Corporate Email Server.

The following OpenScape Contact Center components can connect to the Corporate Email Server via the Email Relay Server: OSCC Email Server, Application Server, Agent Portal Java, Client Desktop Application and Manager Application.

The Email Relay Server plays the role of TLS server for the connection with OpenScape Contact Center components and it plays the role of TLS client for the connection with the Corporate Email Server. In order to play the role of a TLS Server, a Certificate must be installed in the Email Relay.

Certificate format: Java Keystore or PKCS12.

In order to configure the certificate keystore, three parameters are required to be set on the XML configuration file (`emailrelay.xml`):

- **keystoreFile**: the absolute path for the keystore file;
- keystoreType: the type of the keystore file (such as JKS or PKCS12);
- **keystorePassword**: the password of the keystore file.

> ***INFO:*** The CA which was used to sign the certificate must be trusted on the TLS server and also on the TLS client.

In order to install the CA Certificate on the Email Relay Server, follow this procedure:

- Obtain the CA Certificate from the Email Relay Server.Identify the Java instance which is being used by Email Relay Server. Move to the Java home folder, as for example:`cd "\Program Files (x86)\Java\jre1.8.0_161""`
- Execute the following command to install the certificate in the trusted keystore of Java:

  ```
  bin\keytool.exe -keystore lib\security\cacerts -
  importcert -alias <aliasname> -file <file.cer> alias -
  <enter a name which identifies the certificate in the
  keystore>
  ```

Regarding to the clients, the following procedure shall be followed:

- OSCC Email Server, Client Desktop Application and Manager Application - the CA certificate shall be installed as a Trusted Certificate in Windows.Agent Portal Java
- Agent Portal Web

For more information refer to section E-mail IMAP Client.

4 Importing CA Certificates into a Web Browser

Once CA certificates have been replaced solution wide it may be necessary to also install these CA certificates into the user's web browser. Unify products such as OpenScape Desktop Web Embedded Enterprise Edition, OpenScape Desktop Personal Edition, and OpenScape Web Collaboration make use of the end user's web browser for secure connections. Therefore, to allow Unify products to fully trust the connection, the CA certificates must be imported into the web browser.

All settings assume the browser is running on Microsoft Windows. OpenScape Web Embedded Enterprise, Personal Editions and Fusion only run on Microsoft Windows.

4.1 Importing CA Certificates into Internet Explorer

Step by Step

1) Open **Internet Options**. How this is done may vary based on the version of Internet Explorer being used.

2) Click on the **Content** tab and then click on **Certificates** button.

3) Click on the **Trusted Root Certification Authorities** tab and then click on **Import**..

4) Click on **Next**. On the next screen click on **Browse** and locate the root certificate to be imported. Once selected click on **Next**.

5) On the next screen select **Place all certificates in the following store**. The **Certificate store** should be of type **Trusted Root Certification Authorities**. If it is not then click on Browse and select this.

6) Click on **Next** and then **Finish**. A success message should pop-up. The new root certificate should now show in the list.

INFO: If you are only using IP addresses in your solution, then Internet Explorer may still show a certificate error even after the CA certificate has been imported – even if the IP address is placed in the subject alternative name (SAN) field of the certificate. To avoid this, place the IP address of the server in the common name (CN) field of the certificate. However, the best practice is to use FQDNs in certificates. If FQDNs are used, then no special changes need to be made to the certificate.

4.2 Importing CA Certificates into Firefox

Step by Step

1) Open **Firefox Options**. How this is done may vary based on the version of Firefox being used.

2) Click on **Advanced** > **Encryption** > **View Certificates** button.

3) The **Certificate Manager** window opens. Click on the **Authorities** tab and then click on **Import**.

4) Locate the root certificate to be imported and then click on **Open**.

5) A window will open asking for what purposes this certificate should be trusted. The option **Trust this CA to identify websites** must be checked. It is safe to check all options.

6) Click on **OK**. The new root certificate should now show in the list. Click **OK** and then **OK** to close the Firefox options.

4.3 Importing CA Certificates into Chrome

Chrome on Windows uses the Internet Explorer facilities for certificate management. Follow the instructions for importing CA certificates into Internet Explorer.

4.4 Browser Proxy Settings and FQDNs

If your web browser uses a proxy to connect to the internet AND you use a proxy exception list to access local resources, then you may need to add FQDNs to the proxy exception list in order to access these local resources. Example, you want to access your server at www.example.com which is a local server. This FQDN may resolve to the IP address 10.10.10.10 which is already in your proxy exception list. However, when trying to access this resource via the FQDN, the browser may not first resolve the IP address and thus will not use the proxy exception list. Therefore, it may be necessary to also add the FQDN to proxy exception list.

5 Formatting Certificates

This section details how to convert certificates to different formats. These commands are executed on a Linux host. OpenSSL comes installed on all Linux distributions. The keytool command is specific to Java. Commands using the keytool can also be executed on all Linux distributions.

Note: You are not required to use these commands to convert certificates. Third party tools are also allowed.

PEM to PKCS#12

Note: PKCS#12 keystores are also called pfx keystores.

```
openssl pkcs12 -export -in mycert.crt -inkey mykey.key -out
keystore.p12 -name "tomcat" -CAfile mycacert.crt -caname
myca –chain
```

or, without the CA certificates

```
openssl pkcs12 -export -in mycert.crt -inkey mykey.key -out
keystore.p12 -name "tomcat"
```

PKCS#12 to PEM

Output the client certificate(s) and key(s) into one file, with the key not encrypted (-nodes).Add the –nokeys command for certificate only or the –nocerts command for key only.

```
openssl pkcs12 -in keystore.p12 -nokeys -clcerts -out
cert.pem
```

```
openssl pkcs12 -in keystore.p12 -nocerts -clcerts -nodes-out
key.pem
```

PKCS#12 to CAs

Output the certificate authority certificates into one file.

```
openssl pkcs12 -in keystore.p12 –cacerts -out cert.pem
–nodes
```

PKCS#8 to PKCS#1

```
openssl rsa -in server.key -out server_new.key
```

Note: Do not use this command on the .pem file which contains the key and the certificate. Extract the private key from the .pem file to a new file and execute the command on this new file. Then you can import the new PKCS#1 key to the .pem file.

PEM to DER

Use ‘x509’ for a certificate and ‘rsa’ for a key.

```
openssl x509 -outform DER -in pemcert.crt -out dercert.crt
```

DER to PEM

Use ‘x509’ for a certificate and ‘rsa’ for a key.

```
openssl x509 -inform DER -in dercert.crt -out pemcert.crt
```

Non-Encrypted Key to Encrypted Key

```
openssl rsa -in nocrypt.pem –des3 -out crypted.pem
```

Encrypted Key to Non-Encrypted Key

```
openssl rsa -in crypted.pem -out nocrypt.pem
```

Viewing Certificates and Keys

```
openssl x509 -in cert.pem -noout -text
```

Viewing Certificates in a Keystore

Note: It’s difficult to determine if a keystore is JKS formatted or PKCS#12. Try the command below with and without the –storetype command.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -list
-v -storetype pkcs12 -keystore tomcat_keystore.p12
```

Import PKCS#12 to JKS Keystore

This is a long command.Anywhere you see ‘password’ you should use your password.

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool -
importkeystore –deststorepass password –destkeypass password
-destkeystore keystore.jks -srckeystore keystore.p12 -
srcstoretype PKCS12 -srcstorepass password -alias tomcat
```

Import CA certificate to a JKS Truststore

```
/opt/siemens/share/ibm-java-<version>/jre/bin/keytool
–import –trustcacerts –alias <ca_name> –file
<path_to_ca_cert> -keystore <path_to_truststore>
```

5.1 Example Certificate File with Concatenated Certificate Chain

Below is an example certificate file with a PEM encoded certificate, private key, and certificate authority trust chain (if any).

NOTES:

- There are no new lines between each entry.
- The key has to be in RSA format. If not, then it has to be converted.
- The EOL (End of Line) of the certificate has to be Unix-style. This mean every line has to end on LF (Line Feed) instead Of CR - LF (Carriage Return - Line Feed).

-----BEGIN CERTIFICATE----- MIIE/TCCAuWgAwIBAgIBFTANBgkqhkiG9w0BAQsFADBYMQswCQYDVQQGEwJVUzEQ MA4GA1UECBMHRmxvcmlkYTETMBEGA1UEBxMKQm9jYSBSYXRvbjEOMAwGA1UEChMF VW5pZnkxEjAQBgNVBAMTCUFuc2libGVDQTAeFw0xNDAxMDIyMTQ4MzNaFw0yNDAx MDIyMTQ4MzNaMGAxCzAJBgNVBAYTAlVTMRAwDgYDVQQIEwdGbG9yaWRhMRMwEQYD VQQHEwpCb2NhIFJhdG9uMQ4wDAYDVQQKEwVVbmlmeTEaMBgGA1UEAxMRYWNjLTAx LWJyaWFuUzgtdXMwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDIUEYR. -----END CERTIFICATE-----	*The server or client certificate.*
-----BEGIN RSA PRIVATE KEY----- MIIEpAIBAAKCAQEAyFBGEeLesPStSyIGZLjYAifgmHm4MOsQswSUuIqXJT+yIKym igTzg14kQS2tY2B3NPBiZrrkVAA7rxR9Ua2qs6sJB/i9LkaEsenVFvGyy2SoJcxm ZbANZAiKl3L5L9eURxWIkwNDkP1bNbwtdXamkD+mujMRCHKXpPIWWJtlkbqmhE0C 3JI+ADbEUXH1IDUydoQYG74bz/Jr4yWyVz+0Zla0NOPuGx+KTT73T1V5oXGMIxlU kuE1AohNauaBNLs2cX5WRL/Vwauz7ftBLDlMJGMScvmRHO0DW7f+oG+pHd0+Gnx9 gmYQS/ualvan5ny+3jrrjcX0aGhwI4BkxafLtQIDAQABAoIBAB1jYZRYM5HEo/QE -----END RSA PRIVATE KEY-----	*The server or client key.*
-----BEGIN CERTIFICATE----- MIIE/TCCAuWgAwIBAgIBFTANBgkqhkiG9w0BAQsFADBYMQswCQYDVQQGEwJVUzEQ MA4GA1UECBMHRmxvcmlkYTETMBEGA1UEBxMKQm9jYSBSYXRvbjEOMAwGA1UEChMF VW5pZnkxEjAQBgNVBAMTCUFuc2libGVDQTAeFw0xNDAxMDIyMTQ4MzNaFw0yNDAx MDIyMTQ4MzNaMGAxCzAJBgNVBAYTAlVTMRAwDgYDVQQIEwdGbG9yaWRhMRMwEQYD VQQHEwpCb2NhIFJhdG9uMQ4wDAYDVQQKEwVVbmlmeTEaMBgGA1UEAxMRYWNjLTAx LWJyaWFuUzgtdXMwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDIUEYR. -----END CERTIFICATE-----	*The intermediate CA certificate which signed the server/client certificate, and then the intermediate CA certificate which signed the previous intermediate CA.* *There could be further intermidiate CAs, each signing the previous.*
-----BEGIN CERTIFICATE----- MIIE/TCCAuWgAwIBAgIBFTANBgkqhkiG9w0BAQsFADBYMQswCQYDVQQGEwJVUzEQ MA4GA1UECBMHRmxvcmlkYTETMBEGA1UEBxMKQm9jYSBSYXRvbjEOMAwGA1UEChMF VW5pZnkxEjAQBgNVBAMTCUFuc2libGVDQTAeFw0xNDAxMDIyMTQ4MzNaFw0yNDAx MDIyMTQ4MzNaMGAxCzAJBgNVBAYTAlVTMRAwDgYDVQQIEwdGbG9yaWRhMRMwEQYD VQQHEwpCb2NhIFJhdG9uMQ4wDAYDVQQKEwVVbmlmeTEaMBgGA1UEAxMRYWNjLTAx LWJyaWFuUzgtdXMwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDIUEYR. -----END CERTIFICATE-----	*The root CA certificate.*

The administrator can check the **server.pem / client.pem** certificates of OSV using the following:

```
# /unisphere/srx3000/srx/bin/check_certs
```

6 Creating Your Own Root CA

The following sections detail how to create your own simple CA and certificates. These sections use OpenSSL commands executed on a Linux operating system. All Linux operating systems should come with OpenSSL by default.

You can use other third party PKI tools to create a CA and manage certificates. They will be much easier to use and manage.

NOTICE: Starting with OpenScape Solution V8 (incl. UC V7) the use of SHA2 (SHA-256) signatures in certificates is recommended.

6.1 Simple Certificate Creation

Step by Step

1) Create the CA configuration file. Copy the contents of the below text into a file named **ca.cnf**:

```
#
#CA configuration file
#

HOME                                    = .

[req]
default_bits                            = 2048
default_md                              = sha256
distinguished_name                      = req_distinguished_name
x509_extensions                         = v3_ca

[ req_distinguished_name ]
countryName                             = Country Name (2 letter code)
countryName_default                     = US
countryName_min                         = 2
countryName_max                         = 2
StateOrProvinceName                     = State or province name (full name)
localityName                            = Locality (city)
organizationName                        = Organization Name (Company)
```

```
organizationalUnitName          = Organizational Unit Name (section)
commonName                      = Common Name (CA Name)
commonName_max                  = 64

[ v3_ca ]
keyUsage                        = keyCertSign, cRLSign
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid:always,issuer:always
basicConstraints                = CA:true

#EOF
```

2) Create the server configuration file. Copy the contents of the below text into a file named **server.cnf**:

```
#
# Server configuration file. If you are creating certificates for more than one server
# and are using subAltName then you should create a copy of this file for each server.
#

HOME                            = .

[req]
default_bits                    = 2048
default_md                      = sha256
distinguished_name              = req_distinguished_name
x509_extensions                 = v3_req

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = US
countryName_min                 = 2
countryName_max                 = 2
StateOrProvinceName             = State or province name (full name)
localityName                    = Locality (city)
organizationName                = Organization Name (Company)
organizationalUnitName          = Organizational Unit Name (section)
```

```
commonName                      = Common Name (CA Name)
commonName_max                  = 64

[ v3_ca ]
keyUsage                        = keyEncipherment, digitalSignature, nonRepudiation
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid:always,issuer:always
basicConstraints                = CA:false

# README: If you decide to use subjectAltName you must uncomment
# the following line and AT LEAST one of the lines under [alt_names].

# subjectAltName                = @alt_names

[alt_names]
# You can define more DNS, IP, URI names etc. . Just follow the order.

# DNS.1                         = www.pls.com
# DNS.1                         = www.fmm.com
# URI.1                         = https://plm.com
# IP.1                          = 192.168.1.168
# IP.2048                       = 001:0db8:85a3:0000:0000:8a2e:0370:7334

# EOF
```

3) Create the root CA certificate.

This single step will create a self-signed root CA that will be used to sign server certificates. You will be prompted to enter organization information such as country, state, city and etc. Be aware that the file names used in this command will be used in following steps.

```
openssl req -new -x509 -keyout cakey.pem -out cacert.pem
-nodes -days 3650 -config ca.cnf
```

4) The file server.cnf is a generic configuration file that can be used to create many server certificates.

You will be prompted to enter organization information such as country, state, city and etc. The final prompt will ask for the common name.

Create the key and signing request:

```
openssl req -new -keyout server1key.pem -out
server1req.pem -nodes -config server.cnf
```

Sign the request using the CA certificate and key created in step 3).

```
openssl x509 -req -CA cacert.pem -CAkey cakey.pem -
CAcreateserial -extfile server.cnf -extensions v3_req
-in server1req.pem -out server1cert.pem -days 1825
```

View your newly created certificate using the following command:

```
openssl x509 –text –noout –in server1cert.pem
```

5) Repeat step 4) to replace the certificate in the OpenScape Contact Center main server.

6) Add the root certificate file (cacert.cer) to the Windows Trusted Root Certification Authorities on the server which is running the OpenScape Contact Center server from.

Control Panel > Network and Sharing Center > Internet Options > Content > Certificates

7) Add the root certificate to the keystore of the JDK in the clients on which Agent Portal is running. Navigate to `jdk\jre\lib\security` folder and import the certificate via the following command:`keytool -keystore cacerts -importcert -alias <common name> -file cacert.cer`.

The default password of the jdk keystore is **changeit**.

- When prompted to trust this certificate, respond by typing `y`.
- Run the following command to ensure the CA certificate has been successfully imported:`keytool -list -keystore cacerts`

6.2 Creating a SAN Certificate

What if I need to create more certificates and want to include more IP addresses or DNS names in the certificate?

This extra information is stored in a field in the certificate named subject alternative name. Certificates that contain subject alternative name information are often referred to as SAN certificates.

A perfect example of this is the OpenScape Voice. The OpenScape Voice uses multiple signaling addresses across both nodes. Since we only need one certificate for both nodes of the OpenScape Voice, then it makes sense to include all of these signaling IP addresses in a single certificate. The same can be done for FQDNs. For example, lets assume the SIP, secure SIP, and CSTA IP addresses on the OpenScape Voice each have their own FQDNs; osv_sip.example.com, osv_ssip.example.com, and osv_csta.example.com. All of these FQDNs can be combined into the same certificate.

Step by Step

1) To create a SAN certificate make a copy of the server.cnf file.

Open the file for editing and uncomment the subjectAltName field and at least one DNS or IP field.

In the example below we will uncomment the two DNS fields. This will allow us to create a SAN certificate with two FQDNs.

```
subjectAltName = @alt_names
[ alt_names ]
#You can define more DNS, IP, URI names, and etc.Just
follow the order.
DNS.1= www.foo.com
DNS.2= www.bar.org
URI.1= https://www.foo.com
#IP.1= 192.168.1.1
#IP.2= fd00:10:45:237::16
```

2) Repeat step 4) in previous section 'Simple Certificate Creation' to create the certificate. Be sure to use the new configuration file. When viewing your certificate, you will see your SAN information.

```
X509v3 Subject Alternative Name:
DNS:www.foo.com, DNS:www.bar.org, URI:https://
www.foo.com
```

7 Activating TLS in Web Browsers

Starting with version 7 of Unify products, Transport Layer Security (TLS) is the only supported secure communication protocol. Older Secure Sockets Layer protocols SSLv2 and SSLv3 are no longer supported. TLS has been an industry standard for many years there should be no problems with existing third party products communicating with Unify products. The most notable exception may be the use of older web browsers. All modern browsers support TLS by default. However, some older browsers support TLS but do not have it enabled by default. The instructions below detail how to enable TLS in our supported browsers.

Firefox:

Opening the Firefox options window will vary based on the operating system and/or version of Firefox being used. Once the options window has been opened navigate to Advanced > Encryption tab. Make sure “Use TLS 1.2” is checked.

Internet Explorer:

Opening the Internet Explorer internet options window will vary based on the version of Windows and Internet Explorer being used. Once the internet options window has been opened, navigate to the Advanced tab. Scroll down until you see the option “Use TLS 1.2”. Make sure this is checked.

8 Referenced Works

[1]	International Telecommunication Union, "X.509 : Information technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks, ITU-T Recommendation X.509," November 2008. [Online]. Available: http://www.itu.int/rec/T-REC-X.509-200811-I/en. [Accessed 21 February 2013].
[2]	D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley and W. Polk, "RFC 5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," May 2008. [Online]. Available: http://tools.ietf.org/html/rfc5280. [Accessed 21 February 2013].
[3]	M. I. Gallant, "PKCS #12 File Types: Portable Protected Keys in .NET," Microsoft, March 2004. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms867088.aspx. [Accessed 22 February 2013].
[4]	RSA Laboratories, "RSA Laboratories - PKCS #12: Personal Information Exchange Syntax Standard, version 1.1," 27 October 2012. [Online]. Available: http://www.rsa.com/rsalabs/node.asp?id=2138. [Accessed 21 February 2013].
[5]	Wikipedia contributors, "Public-key infrastructure," Wikipedia, The Free Encyclopedia, 21 January 2013. [Online]. Available: http://en.wikipedia.org/wiki/Public-key_infrastructure. [Accessed 21 February 2013].
[6]	T.Dierks and E.Rescorla, "RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2," August 2008. [Online]. Available: https://tools.ietf.org/html/rfc5246. [Accessed 21 February 2013].
[7]	Sun Microsystems, Inc., "Java Secure Socket Extension (JSSE) Reference Guide for the Java 2 Platform Standard Edition 5," 2010. [Online]. Available: http://docs.oracle.com/javase/1.5.0/docs/guide/security/jsse/JSSERefGuide.html#Stores. [Accessed 22 February 2013].
[8]	The Internet Engineering Task Force (IETF), "Public-Key Infrastructure (X.509) (pkix)," [Online]. Available: http://datatracker.ietf.org/wg/pkix/charter/. [Accessed 21 February 2013].