Security

Core Security Concepts

If you are an administrator and are looking to optimize the security of your product, you should have a good understanding of the following security concepts.

HTTPS

Hypertext Transfer Protocol Secure (HTTPS) uses Secure Sockets Layer (SSL) or its subsequent standardization, Transport Layer Security (TLS), to encrypt the data transmitted over a channel. Several vulnerabilities have been found in SSL, so now supports TLS only.

Note TLS is loosely referred to as SSL often, so we will also follow this convention.

SSL employs a mix of privacy, authentication, and data integrity to secure the transmission of data between a client and a server. To enable these security mechanisms, SSL relies upon certificates, private-public key exchange pairs, and Diffie-Hellman key agreement parameters.

SSL Certificates

SSL certificates and private-public key pairs are a form of digital identification for user authentication and the verification of a communication partner’s identity. Certificate Authorities (CAs), such as VeriSign and Thawte, issue certificates to identify an entity (either a server or a client). A client or server certificate includes the name of the issuing authority and digital signature, the serial number, the name of the client or server that the certificate was issued for, the public key, and the certificate's expiration date. A CA uses one or more signing certificates to create SSL certificates. Each signing certificate has a matching private key that is used to create the CA signature. The CA makes signed certificates (with the public key embedded) readily available, enabling anyone to use them to verify that an SSL certificate was actually signed by a specific CA.

In general, setting up certificates involve the following steps:

1. Generating an identity certificate for a server.

2. Installing the identity certificate on the server.
3. Installing the corresponding root certificate on your client or browser.

The specific tasks you need to complete will vary, depending on your environment.

1-Way SSL Authentication

This authentication method is used when a client needs assurance that it is connecting to the right server (and not an intermediary server), making it suitable for public resources like online banking websites. Authentication begins when a client requests access to a resource on a server. The server on which the resource resides then sends its server certificate (also known as an SSL certificate) to the client in order to verify its identity. The client then verifies the server certificate against another trusted object: a server root certificate, which must be installed on the client or browser. After the server has been verified, an encrypted (and therefore secure) communication channel is established. At this point, the server prompts for the entry of a valid username and password in an HTML form. Entering user credentials after an SSL connection is established protects them from being intercepted by an unauthorized party. Finally, after the username and password have been accepted, access is granted to the resource residing on the server.

Note A client might need to store multiple server certificates to enable interaction with multiple servers.

To determine whether you need to install a root certificate on your client, look for a lock icon in your browser's URL field. If you see this icon, this generally indicates that the necessary root certificate has already been installed. This is usually the case for server certificates signed by one of the bigger Certifying Authorities (CAs), because root certificates from these CAs are included with popular browsers.

If your client does not recognize the CA that signed a server certificate, it will indicate that the connection is not secure. This is not necessarily a bad thing. It just indicates that the identity of the server you want to connect has not been verified. At this point, you can do one of two things: First, you can install the necessary root certificate on your client or browser. A lock icon in your browser's URL field will indicate the certificate was installed successfully. And second, you can install a self-signed certificate on your client. Unlike a root certificate, which is signed by a trusted CA, a self-signed certificate is signed by the person or entity that created it. While you can use a self-signed certificate to create an encrypted channel, understand that it carries an inherent amount of risk because the identity of the server you are connected with has not been verified.

Install Certificates

This section contains information about installing security certificates on the Cisco WAE server, Cisco WAE Coordinated Maintenance, and Cisco WAE Live.

Install a Certificate for the Cisco WAE Server

Cisco WAE comes with a default certificate. Because this certificate is not from a “trusted CA", the browser shows an unsecured connection warning. This is the expected behavior. The warning can be removed by applying an appropriate Certificate Authority (CA) issued certificate.

Step 1 Create a private server key and store it in a secure location. For example:

```
# openssl genrsa -out server.key 2048
```

Step 2 Create the Certificate Signing Request (CSR). The CSR is used by CA to create a certificate that identifies your website as secure. For example:

```
# openssl req -sha256 -new -key server.key -out server.csr
```

Step 3 Submit the CSR to the Certificate Authority to obtain your Certificate (for example, server.crt).

Note WAE supports server.crt in the PEM format only. To convert the server certificate from DER to PEM format, you can use the command

```
sudo openssl x509 -inform der -in <input certificate filename> -out <output certificate filename>
```

Step 4 Modify the *<WAE_run_directory>*/wae.conf by changing <key-file/> and <cert-file/> elements to point to the location of the server.key and server.crt files.

Step 5 Restart the Cisco WAE server.

```
# sudo supervisorctl stop wae:*
# sudo supervisorctl start wae:*
```

Install a Certificate for Cisco WAE Live

Cisco WAE Live includes a default certificate that causes the browser to indicate that the certificate is not trusted. This is the expected behavior. The warning can be removed by applying an appropriate CA issued certificate.

To install a CA certificate for Cisco WAE Live, do the following:

Before you begin

Note This procedure is only applicable for Cisco WAE Live 7.1.1 and later.

- You must be an administrator with Cisco WAE user privileges to perform this task.
- The tool `keytool` is deployed with jdk/jre. Make sure the `keytool path` is included in `PATH`.

Note The previous example is applicable if your shell is sh, ksh, or bash. Use equivalent commands for other shells.

- Log out and in again, or enter the following command using the appropriate profile filename.

```
# source ~/.profile
```

Step 1 In order to obtain a certificate from the Certificate Authority (CA) of your choice, you have to create a Certificate Signing Request (CSR). To create a CSR follow these steps:

a) Delete the default certificate. For example:

```
# keytool -storepass changeit -delete -alias cisco -keystore
$CARIDEN_HOME/lib/web/apache-tomcat-8.5.53/conf/keystore
```

b) Create a local self-signed Certificate. For example:

```
# keytool -storepass changeit -genkey -alias tomcat -keyalg RSA -keystore
$CARIDEN_HOME/lib/web/apache-tomcat-8.5.53/conf/keystore
```

c) Create the CSR. For example:

```
# keytool -storepass changeit -certreq -keyalg RSA -alias tomcat -file certreq.csr -keystore
$CARIDEN_HOME/lib/web/apache-tomcat-8.5.53/conf/keystore
```

d) Submit the CSR to a Certificate Authority to obtain your certificate.

e) (Optional) Restart Cisco WAE Live to use the new certificate immediately.

```
# embedded_web_server -action stop
# embedded_web_server -action start
```

Step 2 Install the certificate.

a) Download a Chain Certificate (also called a Root Certificate) from the CA you obtained the certificate from.

b) Import the Chain Certificate into the keystore.

```
# keytool -storepass changeit -import -alias root -keystore
$CARIDEN_HOME/lib/web/apache-tomcat-8.5.53/conf/keystore -trustcacerts -file
<filename_of_the_chain_certificate>
```

c) Import the new certificate.

```
# keytool -storepass changeit -import -alias tomcat -keystore
$CARIDEN_HOME/lib/web/apache-tomcat-8.5.53/conf/keystore -file <your_certificate_filename>
```

d) Restart Cisco WAE Live.

```
# embedded_web_server -action stop
# embedded_web_server -action start
```

Install a Certificate for the LDAP Server

Cisco WAE supports authentication and authorization of foreign users using Lightweight Directory Access Protocol (LDAP).

To use LDAPS protocol, get the SSL certificate and add it to a keystore.

Step 1 Save the self signed certificate to cert.pem file using the following command:

```
# openssl s_client -connect <ldap-host>:<ldap-ssl-port> </dev/null 2>/dev/null | sed -n
'/^-----BEGIN/,/^-----END/ { p }' > cert.pem
```

Step 2 Get the default key-store path by running the following command from `WAE_RUN` directory.

```
# $WAE_ROOT/lib/exec/test-java-ssl-conn <ldap-host> <ldap-ssl-port> 2>1 | grep "trustStore is:"
```

Running the above command helps you find the directory from where certs are picked up. It may be a directory similar to:

```
trustStore is: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.102-4.b14.el7.x86_64/jre/lib/security/cacerts
```

Step 3 Import cert into default key-store using following command:

```
# sudo keytool -import -keystore <default-key-store-path> -storepass changeit -noprompt -file cert.pem
```

Install a Certificate for the EPN-M Server

Install the certificate when using the Cisco Evolved Programmable Network Manager (Cisco EPN Manager) agent for L1 collection.

Step 1 Save the self signed certificate to cert.pem file using the following command:

```
# openssl s_client -connect <epnm-host>:<epnm-port> </dev/null 2>/dev/null | sed -n
'/^-----BEGIN/,/^-----END/ { p }' > cert.pem
```

Step 2 Get the default key-store path using the following command. Typically the default key-store path is /etc/pki/java/cacerts for CentOS 7 with open-jdk

```
# $WAE_ROOT/lib/exec/test-java-ssl-conn <epnm-host> <epnm-port> 2>1 | grep "trustStore is:"
```

Step 3 Import cert into default key-store using following command:

```
# sudo keytool -import -keystore <default-key-store-path> -storepass changeit -noprompt -file cert.pem
```