Markvision Enterprise

SSL Configuration White Paper

September 2015 **www.lexmark.com**

740 West New Circle Road
Lexington, KY 40550

Abstract

This white paper describes the steps required to secure communication between the Markvision (Tomcat) Server and a user's Web browser using the SSL (Secure Socket Layer) protocol.

Edition: September 2015

For Lexmark technical support, visit **support.lexmark.com**.

For information on supplies and downloads, visit **www.lexmark.com**.

If you don't have access to the Internet, you can contact Lexmark by mail:

Lexmark International, Inc.
Bldg. 004-2/CSC
740 New Circle Road NW
Lexington, KY 40550
USA

Trademarks

Lexmark, the Lexmark logo, and *Open the possibilities*, and Markvision Enterprise are trademarks of Lexmark International, Inc., registered in the United States and/or other countries. Other trademarks are the property of their respective owner.

Table of Contents

1 Overview

Although Transport Layer Security (TLS) is recommended because of a known vulnerability in Secure Socket Layer (SSL), SSL can still be used with Tomcat. SSL is a common security protocol that uses data encryption and certificate authentication to protect communication between a server and a client. For Markvision Enterprise, SSL can be used to protect sensitive information shared between the Markvision (Tomcat) Server and a user's Web browser, such as:

- Device passwords
- Security policies
- Device authentication information (LDAP, Kerberos, etc.)
- Markvision Enterprise user credentials and passwords

SSL enables the Tomcat server and the Web browser to encrypt this data before sending it and decrypt it upon receipt. SSL also requires the server to present the browser with a certificate that proves the server is who it claims to be. This certificate can either be self-signed (not recommended for production) or approved by a trusted third-party Certificate Authority (CA).

This document describes the steps required to create an SSL keystore and self-signed certificate, obtain a certificate from a CA (if desired), and configure the Tomcat server's SSL Connector. The document also explains how to configure the server to automatically redirect users to the secure Markvision Enterprise Web page (if desired).

1.1 Audience

This document is intended for server/Web administrators who are familiar with the following.

- Basic Tomcat SSL configuration
    - For detailed information, see the online documentation for Apache Tomcat 8.0 SSL Configuration HOW-TO
- Basic SSL terminology (certificate, key, keystore, alias, etc.) Basic keytool commands
    - Keytool is the utility used to create and manage keys and certificates. It is part of the Java Runtime Environment (JRE).

2 Configuring the Markvision Server with SSL

The Markvision Server can be run as a standalone Tomcat server or it can be fronted with a Web server (IIS or Apache).

If you are running a standalone Tomcat server, you will first need to use keytool to create a keystore and an SSL certificate (either self-signed or certified by a CA). You will then need to configure the SSL Connector settings in the Tomcat server's main configuration file (server.xml).

If you are running your Tomcat server as an application server fronted by a Web server, you should follow the instructions for configuring SSL with your Web server and ignore the following instructions. In this case, the Web server will handle all encryption and decryption of traffic from your browser. Requests between the Web server and the Tomcat server will not be encrypted and will not require you to make any modifications to the Tomcat.

2.1 Creating the Keystore and Self-Signed Certificate

To create a keystore containing a self-signed certificate:

1. In keytool, enter:
   `%JAVA_HOME%\bin\keytool -genkey -alias [alias] -keyalg RSA -keystore [keystore_path]`
   Substitute your desired alias for [alias]and the desired location for your keystore for `[keystore_path]`.

   > **NOTE:**
   >
   > To avoid potential problems during upgrades, Lexmark recommends using the Markvision installation root directory (C:/Program Files/Lexmark) for the keystore path. A location outside the Markvision installation may also be used.

2. Enter a keystore password.
3. Fill in the required information for your certificate (your name, company name, etc.).
4. When prompted for the key password (the password for this specific certificate), press Enter. Pressing Enter sets the key password to the same value as the keystore password. The two passwords **MUST** be the same. If the two passwords are not the same, the server will not be able to access your certificate.

2.1.1 Creating the Certificate Signing Request

The steps in the previous section created a self-signed certificate that the Tomcat server will present to a user's browser to prove its identity. However, because the certificate has not been verified by a trusted third-party CA (i.e. Verisign or Thawte), when a user attempts to access Markvision Enterprise securely via SSL, they will see a security error indicating an invalid certificate:

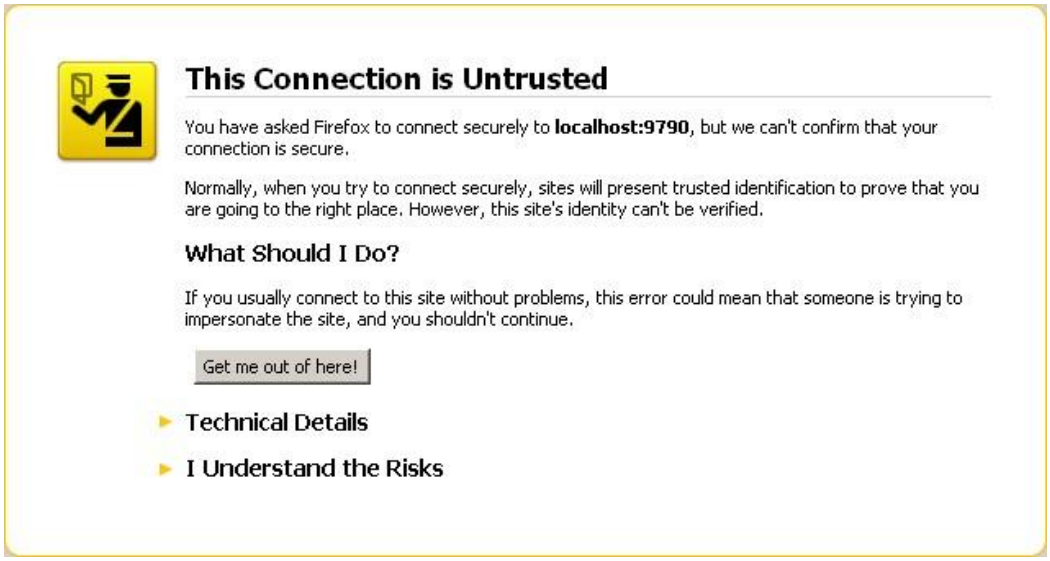

Figure 1: Connection is Untrusted Dialog

Users can still access the secure Markvision Enterprise Web page if they choose to confirm the security exception with their specific browser. To avoid the security exception, you will need to obtain a verified SSL certificate from a trusted third-party CA. To do this, you must create a Certificate Signing Request (CSR) that each CA will use to generate the verified SSL certificate. If you want to continue using a self- signed certificate, skip to Section 2.2.

1. In keytool, enter:
   `keytool -certreq -keyalg RSA -alias [alias] -file [my_cert_filename].csr - keystore [keystore_filename]`
   Substitute your alias, your desired certificate filename, and your keystore filename for `[alias]`, `[my_cert_filename]`, and `[keystore_filename]`.
   Keytool generates a CSR file called [`my_cert_filename].csr`.
2. Follow the instructions on the CA's Web site to submit your CSR.
   The CA generates a verified SSL certificate.
3. Follow the instructions on the CA's Web site to download your certificate.

2.1.2 Importing the Certificates

Before you can import your new certificate into your keystore, you must first download an additional certificate from the CA called a Root Certificate. Follow the instructions on the CA's Web site to download this certificate. After you have downloaded the Root Certificate:

Import the Root Certificate into your keystore:

1. In keytool, enter:
   `keytool -import -alias root -keystore [keystore_filename] -trustcacerts -file [root_cert_filename]`
   Substitute your keystore filename for [keystore_filename]and the Root Certificate filename for `[root_cert_filename]`.

Then, import your certificate into your keystore:

2. In keytool, enter:
   `keytool -import -alias [alias] -keystore [keystore_filename] -file [my_cert_filename]`
   Substitute your alias, keystore filename, and certificate filename for `[alias]`, `[keystore_filename]`, and `[my_cert_filename]`.

Your keystore and certificate are now certified by a trusted third-party CA.

2.2 Configuring the SSL Connector

If you are running a standalone Tomcat server, you must configure the server to use SSL. This is done by modifying the SSL Connector settings in the server's main configuration file (server.xml).

- **Note** – These steps are not necessary if you are running your Tomcat behind a Web server that is already handling all SSL communication.

1. Open `$MVE_INSTALL/tomcat/conf/server.xml` and find the SSL Connector. For example:

```
<!-- Define a SSL HTTP/1.1 Connector
           on port 8443
    This connector uses the JSSE
    configuration, when using APR, the
    connector should be using the OpenSSL
    style configuration described in the APR
    documentation -->

<Connector port="8443" protocol="HTTP/1.1"
        SSLEnabled="true" maxThreads="150"
        scheme="https" secure="true"
        clientAuth="false"
        sslProtocols="TLS" />
```

2. Ensure that the Connector port=number is set to the port you plan to use for your SSL connection. The "default" is 8443.

3. Uncomment the SSL Connector entry and enter your keystore and alias information.

```
<Connector port="8443"
       protocol="org.apache.coyote.http11.Http11NioProtocol"
       SSLEnabled="true" maxThreads="150"
       scheme="https" secure="true"
       clientAuth="false"
       sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
       keystoreFile="C:/Program Files/Lexmark/keystore_filename"
       keystorePass="keystore_password"
       alias="alias"
       ciphers="TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_
       RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA256,
       TLS_RSA_WITH_AES_128_CBC_SHA" />
```

NOTE:

These cipher suites have been selected in order to address known SSL vulnerabilities such as logjam and FREAK.

Substitute your keystore filename, password, and alias for `keystore_filename`, `keystore_password`, and `alias`.

NOTE:

To avoid potential problems during upgrades, Lexmark recommends using the Markvision installation root directory (`C:/Program Files/Lexmark`) for the keystore path. A location outside the Markvision installation may also be used.

NOTE:

As stated in the comment shown in Step 1, this SSL Connector uses the JSSE (Java Secure Socket Extension) configuration. If you wish to use the APR (Apache Portable Runtime) configuration instead, you will need to configure the SSL Connector according to the OpenSSL style configuration.

If you have installed APR but configure the SSL Connector using JSSE, you will see an error message when you attempt to use SSL. Note that Lexmark recommends updating the OpenSSL libraries to the latest compatible version with Apache Tomcat in order to obtain security patches.

4. Restart the Tomcat server.

5. Test your SSL configuration by entering the secure URL into your browser. For example, `https://localhost:8443/mve`.

3 Redirecting to the Secure Web Page

If you are running a standalone Tomcat server, you can configure the server's Web.xmlfile to automatically redirect users to the secure Markvision Enterprise Web page when they enter the unsecure address into their browser. For example, if a user entered http://localhost:9788, they would automatically be redirected to https://localhost:8443/mve.

> **NOTE:**
>
> These steps are not necessary if you are running your Tomcat behind a Web server that is already handling all SSL communication.

1. Open `$MVE_INSTALL/apps/dm-mve/WEB-INF/Web.xml` and uncomment the following:

```
<!--   <security-constraint>-->
<!--          <Web-resource-collection>-->
<!--                 <Web-resource-name>All Requests</Web-
resource-name>-->
<!--                 <url-pattern>/*</url-pattern>-->
<!--          </Web-resource-collection>-->
<!--          <user-data-constraint>-->
<!--                 <transport-
guarantee>CONFIDENTIAL</transport-guarantee>-->
<!--          </user-data-constraint>-->
<!--   </security-constraint>-->
```

2. Open `$MVE_INSTALL/tomcat/conf/server.xml` and ensure that the `redirectPort=` number for the non-SSL Connector matches the `Connector port=` number for the SSL Connector.

```
<Connector port="9788" maxHttpHeaderSize="8192"
            maxThreads="150" minSpareThreads="25"
            maxSpareThreads="75" enableLookups="false"
            redirectPort="8443"
             acceptCount="100"
            connectionTimeout="120000"
       disableUploadTimeout="true" URIEncoding="UTF-8"
       />

<Connector port="8443"
      protocol="org.apache.coyote.http11.Http11NioProtocol”
      SSLEnabled="true" maxThreads="150"
      scheme="https" secure="true"
      clientAuth="false"
      sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
      keystoreFile="C:/Program Files/Lexmark/keystore_filename"
      keystorePass="keystore_password"
      alias="alias"
      ciphers="TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_RSA_
      WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA256,TLS_RSA_W
      ITH_AES_128_CBC_SHA" />
```

3. Restart the Tomcat server.

4. Enter the unsecure address (for example, http://localhost:9788) into your browser and verify that you are redirected to the secure address (for example, https://localhost:8443/mve).