Securing your Acrolinx Server with HTTPS

support.acrolinx.com/hc/en-us/articles/213614069-Securing-your-Acrolinx-Server-with-HTTPS

Applies To

Software	Version
Acrolinx Server	5.3

Download an archived version of this article for versions 4.7 , 5.0 and 5.1 , or 5.2 .

In a standard installation of Acrolinx, communication between the core server and the outside world is unencrypted. By "outside world" we mean people who connect to the core server with an Acrolinx integration, or with a web browser.

When integrations connect to the core server through the public internet, you might choose to use HTTPS, the internet standard for secure web communication. HTTPS communication ensures that data and passwords are unreadable during transmission.

Generally, we recommend that you always configure a cipher whitelist and disable unused and insecure SSL protocols .

By the way, setting up HTTPS, especially generating the proper public key certificate, is best done by a security professional. Some of the configuration steps aren't for the faint-hearted. In any event, the Acrolinx Support team is happy to assist you in securing Acrolinx with HTTPS.

Acrolinx only supports certificates signed by a third-party certificate authority (CA). Self-signed certificates in any form aren't supported.

Here's a checklist of what you'll need:

1. Your certificate (.crt) file signed by a third-party certificate authority.
2. The key (.key) file that belongs to your certificate.
3. All related (chain) certificate files.
4. OpenSSL installed to generate your Java keystore.

Once you have the .crt, .key, related certificate files, and you've installed OpenSSL, you can start. If you'd like help with the following steps, contact Acrolinx Support to schedule a web meeting.

Getting Started

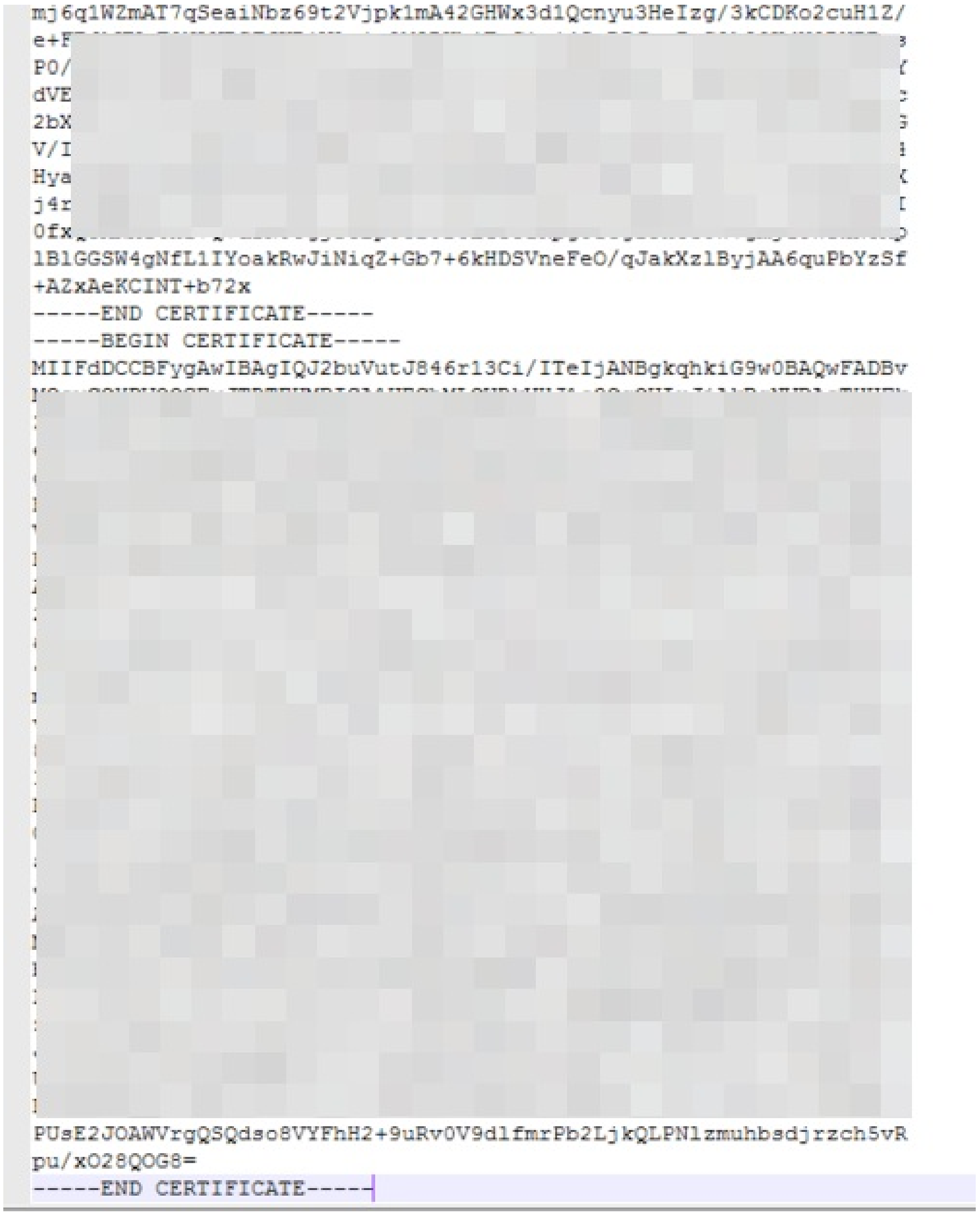
To secure Acrolinx for HTTPS communication, you need to define the protocol and port that you use for secure external communication. You also need to define the keystore and truststore details.

1. Stop all your Acrolinx servers.

2. Import your certificate into a Java keystore:
   a) Combine all certificate files (chain certificates) you received together with the <CERTIFICATE_NAME>.crt file into a new <CERTIFICATE_NEW_NAME>.crt file as follows:

   - Open your main <CERTIFICATE_NAME>.crt file and select Save As… to create a <CERTIFICATE_NEW_NAME>.crt file.
   - Open each chain certificate file, Select All, Copy and then paste into your <CERTIFICATE_NEW_NAME>.crt file directly underneath the existing -----END CERTIFICATE----- line.
   - Repeat for all chain certificates.

   Your certificate file should now look like this:



```
mj6q1WZmAT7qSeaiNbz69t2Vjpk1mA42GHWx3d1Qcnyu3HeIzg/3kCDKo2cuH1Z/
[redacted]
1B1GGSW4gNfL1IYoakRwJiNiqZ+Gb7+6kHDSVneFeO/qJakXz1ByjAA6quPbYzSf
+AZxAeKCINT+b72x
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFdDCCBFygAwIBAgIQJ2buVutJ846r13Ci/ITeIjANBgkqhkiG9w0BAQwFADBv
[redacted]
PUsE2JOAWVrgQSQdso8VYFhH2+9uRv0V9d1fmrPb2LjkQLPN1zmuhbsdjrzch5vR
pu/xO28QOG8=
-----END CERTIFICATE-----
```

   b) Combine your <CERTIFICATE_NEW_NAME>.crt file and the <KEY_NAME>.key file into a single PKCS12 file - To do this, open a command-line shell and enter the following command:

```
openssl pkcs12 -inkey <KEY_NAME>.key -in <CERTIFICATE_NAME>.crt -export -out
combined.pkcs12
```

c) Create a <PASSWORD> when asked, verify it when prompted and make a note of it.

d) Copy the file combined.pkcs12 from wherever it was generated into the <INSTALL_DIR>/server/bin directory of your core server.

If you use a Windows server, you might enter a command that resembles this:

```
copy  C:\files\combined.pkcs12 C:\Program Files\Acrolinx\AcrolinxIQ\server\bin
```

If you use a Unix-based server, you might enter a command that resembles this:

```
cp /Users/<USERNAME>/Documents/combined.pkcs12 /home/admin/Acrolinx/server/bin
```

e) Navigate to your server's <INSTALL_DIR>/server/bin directory (the location to which you just copied your combined.pkcs12 file) and import the PKCS12 file into your Java keystore using the following command:

Here's a Windows example:

```
cd C:\Program Files\Acrolinx\AcrolinxIQ\server\bin
keytool.exe -v -importkeystore -srckeystore combined.pkcs12 -srcstoretype
PKCS12  -srcstorepass <PASSWORD> -destkeystore myKeystore -deststoretype JKS -
deststorepass <KEYSTORE_PASSWORD> -destkeypass <KEYSTORE_PASSWORD>
```

If keytool.exe isn't found, you'll need to state the path to it. This will depend where you have Java installed. For example:

```
C:\Program Files\Java\jdk1.8.0_131\bin\keytool.exe -v -importkeystore -
srckeystore combined.pkcs12 -srcstoretype PKCS12  -srcstorepass <PASSWORD> -
destkeystore myKeystore -deststoretype JKS -deststorepass <KEYSTORE_PASSWORD> -
destkeypass <KEYSTORE_PASSWORD>
```

Here's a Unix example:

```
cd /home/admin/Acrolinx/server/bin
keytool -v -importkeystore -srckeystore combined.pkcs12 -srcstoretype PKCS12  -
srcstorepass <PASSWORD> -destkeystore myKeystore -deststoretype JKS -
deststorepass <KEYSTORE_PASSWORD> -destkeypass <KEYSTORE_PASSWORD>
```

- The variable <PASSWORD> is the password you used when you created the PKCS12 file.
- The variable <KEYSTORE_PASSWORD> is your keystore password.

You've now generated a keystore in your core server's bin directory.

3. Open your core server properties file. You can find it under `/server/bin/coreserver.properties` . Add the following properties.

```
endpointHost=<SERVER_NAME>
endpointProtocol=https
```

<SERVER_NAME> has to be the domain name on your certificate, and not the machine name. To use HTTPS, the core server needs a DNS name that's accessible

to the integrations.

a) Update the endpointPort value if you use a different port for secure communication. By default, all integrations connect to the core server through port 8031. If your security policy requires secure communication to go through a specific port such as 443, add the extra endpointPort as follows:

```
endpointPort=443
```

4. Define the keystore and truststore details by adding the following properties:

   ```
   sslKeystore=<KEYSTORE_NAME>
    sslKeystorePassword=<PASSWORD>
    sslTruststore=<KEYSTORE_NAME>
    sslTruststorePassword=<PASSWORD>
   ```

   Example:

   ```
   sslKeystore=myKeystore
    sslKeystorePassword=pRu4Hatr
    sslTruststore=myKeystore
    sslTruststorePassword=pRu4Hatr
   ```

   In most cases, the value for sslTruststore is the same as the value for sslKeystore . If you leave out the sslTruststore properties completely, the values that you entered for the sslKeystore properties are automatically used as the sslTruststore settings as well. If you created a separate truststore, use that file name and password instead.

5. (Optional) If you use a different JVM to the one that is installed with Acrolinx and want to specify a different keystore algorithm, complete this step - otherwise proceed to step 6. Add the following additional property:

   ```
   ssl.KeyManagerFactory.algorithm=<ALGORITHM_NAME>
   ```

   For example, if you use the IBM JVM, add the property as follows:

   ```
   ssl.KeyManagerFactory.algorithm=IbmX509
   ```

6. In your language server properties files (ls-0x.properties) located in <CONFIG_DIR>/server/bin/, configure the endpointHost and coreServerEndpointUrl to use the domain name:

   ```
   endpointHost=<EXAMPLE.COM>
   coreServerEndpointUrl=https://<EXAMPLE.COM>:<PORT_NUMBER>/internal
   ```

7. Save and start all your Acrolinx servers.
8. Copy the file <INSTALL_DIR>/server/bin/analyticsserver.properties and place the copy into <CONFIG_DIR>/server/bin
   Open the file <CONFIG_DIR>/server/bin/analyticsserver.properties and set the following properties:

```
com.acrolinx.jreport.internal.host=<MACHINE_NAME>
com.acrolinx.jreport.coreserver.scheme=https
com.acrolinx.jreport.coreserver.host=<EXAMPLE.COM>
```

Save and start all your Acrolinx servers.

9. Move contents of <CONFIG_DIR>/server/certificates/untrusted to <CONFIG_DIR>/server/certificates/trusted.
   These files need to be *moved* , not simply copied. When they're all in <CONFIG_DIR>/server/certificates/trusted, your /untrusted directory should be empty.

10. Restart all servers.

Updating the Server Address in the Acrolinx Integrations

After you've secured the Acrolinx Server communication with HTTPS, Acrolinx users need to update the connection details in their integrations to use the new HTTPS address.

Instruct users to navigate to Options > Change Server... and update the **Server address** field to:

`https://<SERVER_NAME>:<PORT>`

Users need to access the Dashboard through the https address as well:

`https://<SERVER_NAME>:<PORT>/Dashboard.html`

Your Acrolinx Server is now secured with HTTPS and ready to use.