Installing a Public CA Certificate on RSA Security Analytics 10.4.0.2 and Above

This document provides instructions for configuring and installing a CA-signed SSL certificate on an RSA Security Analytics 10.4.0.2 or above server running the Jetty 9 service. Refer to Appendix A at the end of this document to verify the appropriate security-analytics-web-server and jettyuax packages that should be installed on the appliance.

NOTE: These instructions differ from the process found in the knowledge base article 26817, as new changes have been implemented in version 10.4.0.2 that require additional steps to be performed.

To verify whether or not the appropriate packages are installed, issue the command below.

- `rpm -qa | grep security-analytics`
- `rpm -qa | grep jetty`

```
[root@SA-Server ~]# rpm -qa | grep security-analytics
security-analytics-web-server-10.4.0.2.12064-5.noarch
[root@SA-Server ~]# rpm -qa | grep jetty
jettyuax-9.0.7-64.noarch
```

While the Security Analytics server has a self-signed SSL certificate installed by default, users that have their own Certificate Authority (CA) may wish to use their own CA-signed SSL certificate instead. The process for doing this will be explained in the sections below.

Preliminary Steps

Before commencing with the procedure, there are some preparatory measures that first must be taken.

1. Connect to the SA server via SSH as the root user and create a working directory under the /root home directory.

   ```
   mkdir /root/cert
   [root@CSO-SAServer-VM-01 ~]# mkdir /root/cert
   ```

2. Verify that the **keytool** is in your search path.

```
whereis keytool
[root@CSO-SAServer-VM-01 ~]# whereis keytool
keytool: /usr/bin/keytool /usr/share/man/man1/keytool.1.gz
```

3. Make a full backup of the **/opt/rsa/jetty9/etc** directory.

```
cp -r /opt/rsa/jetty9/etc /opt/rsa/jetty9/etc_orig
[root@CSO-SAServer-VM-01 ~]# cp -r /opt/rsa/jetty9/etc /opt/rsa/jetty9/etc_orig
[root@CSO-SAServer-VM-01 ~]# ls /opt/rsa/jetty9/
README.txt   etc          lib                          notice.html  start.ini
VERSION.txt  etc_orig     license-eplv10-aslv20.html   resources    start.jar
bin          jetty.state  logs                         start.d      webapps
```

4. Make a blank keystore by renaming the **/opt/rsa/jetty9/etc/keystore** file.

```
mv /opt/rsa/jetty9/etc/keystore /opt/rsa/jetty9/etc/keystore_orig
[root@CSO-SAServer-VM-01 ~]# mv /opt/rsa/jetty9/etc/keystore /opt/rsa/jetty9/etc/keystore_orig
```

NOTE: This step will result in the SA Server becoming temporarily inaccessible!

Generate the Certificate Signing Request (CSR)

This section will provide instructions for generating a Public/Private key pair for Jetty9, which will then be used to create a Certificate Signing Request (CSR). This is a two-step process that must be followed carefully in order to avoid issues. The password will also need to be noted as it will be referenced in later steps.

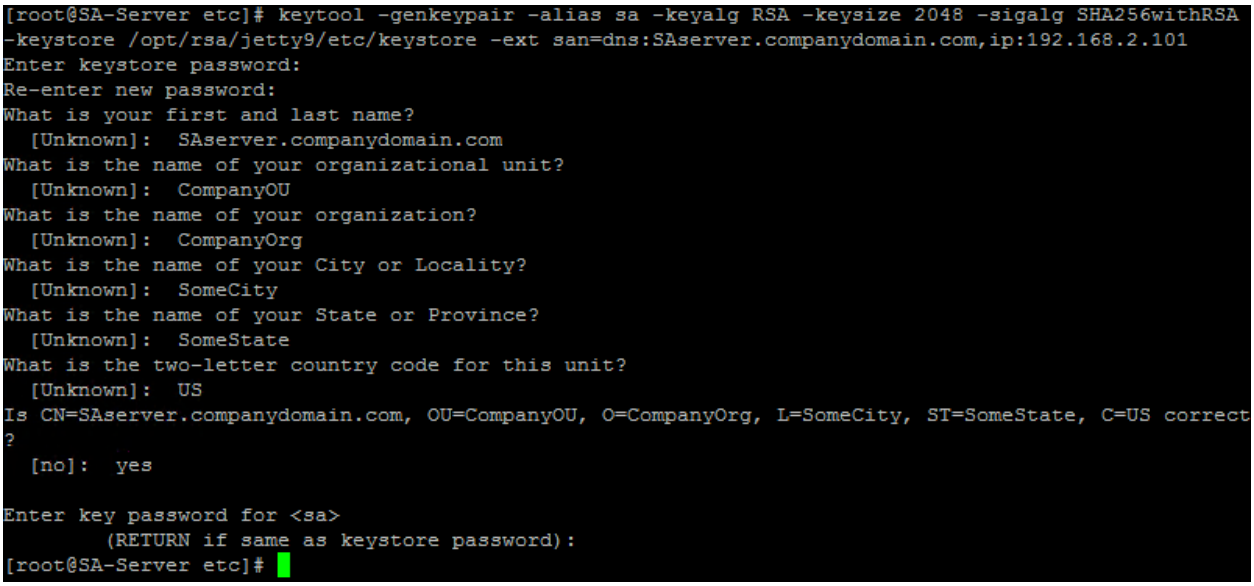
In order to perform the steps in this section, you will need the following information:

- Select an alias name to identify the certificate that will be created.
  (In this example, the alias **sa** will be used.)
- Select a password for the keystore, as well as for the individual key pair if they are to be different.
  (In this example, the password **netwitness** will be used for both passwords.)
- The hostname for the SA server will also be needed. The FQDN and/or IP address of the server may be additionally added using the -ext option for each.
  (In this example, the hostname will be **SAserver**, the FQDN will be **SAserver.companydomain.com**, and the SA Server IP address will be **10.25.55.168**.)

Follow the steps below to generate the public/private key pair and the Certificate Signing Request.

1. Issue the command below to generate the key pair, using the information requested above.

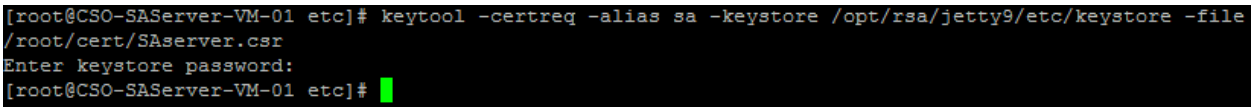
```
keytool -genkeypair -alias <aliasName> -keyalg RSA -keysize 2048 -
sigalg SHA256withRSA -keystore /opt/rsa/jetty9/etc/keystore -ext
san=dns:<serverFQDN>,ip:<serverIP>
```



(Note that, in this example, the FQDN is also used when prompted for the first and last name.)

2. In order to create a Certificate Signing Request (CSR) file, issue the command below, using the alias and password from the previous step.

```
keytool -certreq -alias <aliasName> -keystore
/opt/rsa/jetty9/etc/keystore -ext san=dns:<serverFQDN>,ip:<serverIP>
-file /root/cert/SAserver.csr
```



Have the CSR signed by the Certificate Authority (CA) Administrator

In order for the certificate to be signed, the CSR must be provided to the Certificate Authority (CA) administrator, who will then generally provide two or three cert files in return. These files will be the cert for the SA server (usually named after the server itself and therefore named **SAserver.cer** in this example), the CA public cert (usually named **root.cer**), and in some cases also an intermediate cert (usually named **intermediate.cer**).

NOTE: The .cer file format is the only accepted format at this time. If you or your CA administrator are not familiar with this format or need further assistance with this process, please contact support.

Once the appropriate files have been obtained, follow the steps below to import them into the SA server keystore.

1. Use SCP to move the files to the /root/cert directory on the SA server.

2. Issue the commands below to import files into the keystore.

```
keytool -import -trustcacerts -keystore /opt/rsa/jetty9/etc/keystore -
alias <name> -file /root/cert/<name>.cer
```

```
[root@CSO-SAServer-VM-01 ~]# keytool -import -trustcacerts -keystore /opt/rsa/jetty9/etc/keystore
-alias root -file /root/cert/root.cer
[root@CSO-SAServer-VM-01 ~]# keytool -import -trustcacerts -keystore /opt/rsa/jetty9/etc/keystore
-alias intermediate -file /root/cert/intermediate.cer
[root@CSO-SAServer-VM-01 ~]# keytool -import -trustcacerts -keystore /opt/rsa/jetty9/etc/keystore
-alias sa -file /root/cert/SAserver.cer
```

NOTE: For SAserver.cer, make sure the alias is the same as the key pair. (The alias **sa** is used in this example.)

3. Verify that the certificates have been successfully imported into the keystore using the command below.

```
keytool -list -keystore /opt/rsa/jetty9/etc/keystore
```

```
[root@CSO-SAServer-VM-01 ~]# keytool -list -keystore /opt/rsa/jetty9/etc/keystore
Enter keystore password:

Keystore type: JKS
Keystore provider: SUN

Your keystore contains 3 entries

sa, Dec 3, 2014, PrivateKeyEntry,
Certificate fingerprint (SHA1): C7:3E:0F:7D:D3:88:EE:F3:06:2D:80:D1:3C:85:4D:0E:93:A7:C1:A7
root, Dec 3, 2014, trustedCertEntry,
Certificate fingerprint (SHA1): DE:28:F4:A4:FF:E5:B9:2F:A3:C5:03:D1:A3:49:A7:F9:96:2A:82:12
intermediate, Dec 3, 2014, trustedCertEntry,
Certificate fingerprint (SHA1): 23:8C:8E:2C:D1:96:75:6B:28:67:48:9A:2E:1E:4B:1A:68:44:2C:61
```

Edit the jetty-ssl.xml file to point to the newly created keystore and use its obfuscated passwords

As best practices recommend that no passwords be stored in XML files in a plaintext format, this section provides instructions for obtaining and using the OBF-format strings for the passwords in the jetty-ssl.xml file. This will also allow Security Analytics to recognize the new keystore that was created.

NOTE: If the public/private key pair was given a password that is different from the keystore password, then this process will need to be performed twice.

In version 10.4.0.2, the default keystore for the Jetty web server was changed from /opt/rsa/jetty9/etc/keystore to /opt/rsa/carlos/keystore in order to share the same keystore with Puppet. To enforce this change, a Puppet module was created to house a new version of the jetty-ssl.xml file, which is pushed to the /opt/rsa/jetty9/etc directory during the puppet catalog run.

Different from the procedure for earlier versions, it is the jetty-ssl.xml file in the Puppet module that will need to be edited to reflect the keystore changes.

As with the previous examples, the password **netwitness** will be used.

1. Stop the puppetmaster service with the command below to prevent any unwanted changes while the jetty-ssl.xml file is updated.

   ```
   service puppetmaster stop
   ```

   ```
   [root@SA-Server ~]# service puppetmaster stop
   Stopping puppetmaster:                                     [  OK  ]
   ```

2. Make a backup of the jetty-ssl.xml file.

   ```
   cp /etc/puppet/modules/saserver/files/jetty-ssl.xml /root/cert/jetty-ssl.xml_orig
   ```

   ```
   [root@SA-Server ~]# cp /etc/puppet/modules/saserver/files/jetty-ssl.xml /root/cert/jetty-ssl.xml_orig
   ```

3. Issue the command below to identify the OBF-format strings for the password. The command will return the password that was entered, as well as the OBF-format string and its MD5 hash. **Be sure to surround the password with single quotes if special characters are used.**

   ```
   java -cp /opt/rsa/jetty9/lib/jetty-util*
   org.eclipse.jetty.util.security.Password '<password>'
   ```

   ```
   [root@SA-Server ~]# java -cp /opt/rsa/jetty9/lib/jetty-util* org.eclipse.jetty.util.security.Password 'netwitness'
   ```

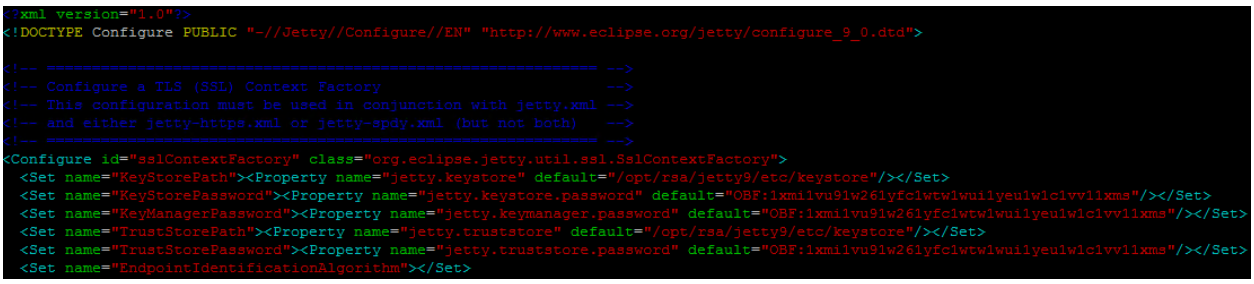
   The output to the command will look similar the following:

   ```
   netwitness
   OBF:1xmi1vu91w261yfc1wtw1wui1yeu1w1c1vv11xms
   MD5:724cfe133a94be51321a9ac2bff65f06
   ```

4. Issue the command `vi /etc/puppet/modules/saserver/files/jetty-ssl.xml` to edit the jetty-ssl.xml file and make the changes below.
   a. Replace the existing OBF strings for **jetty.keystore.password**, **jetty.keymanager.password**, and **jetty.truststore.password** with the string output from the previous step.
   b. Change the default value for **jetty.keystore** and **jetty.truststore** from /opt/rsa/carlos/keystore to **/opt/rsa/jetty9/etc/keystore** in order to point to the Jetty keystore.
   c. Remove the entire line for **jetty.keystore.alias** in the file.

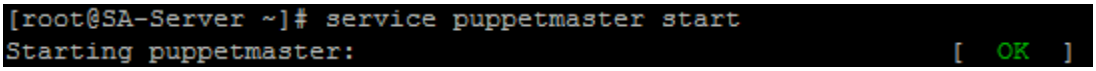
The necessary changes to the file are depicted in the image below.

After making the changes, the file should appear similar to the example below.



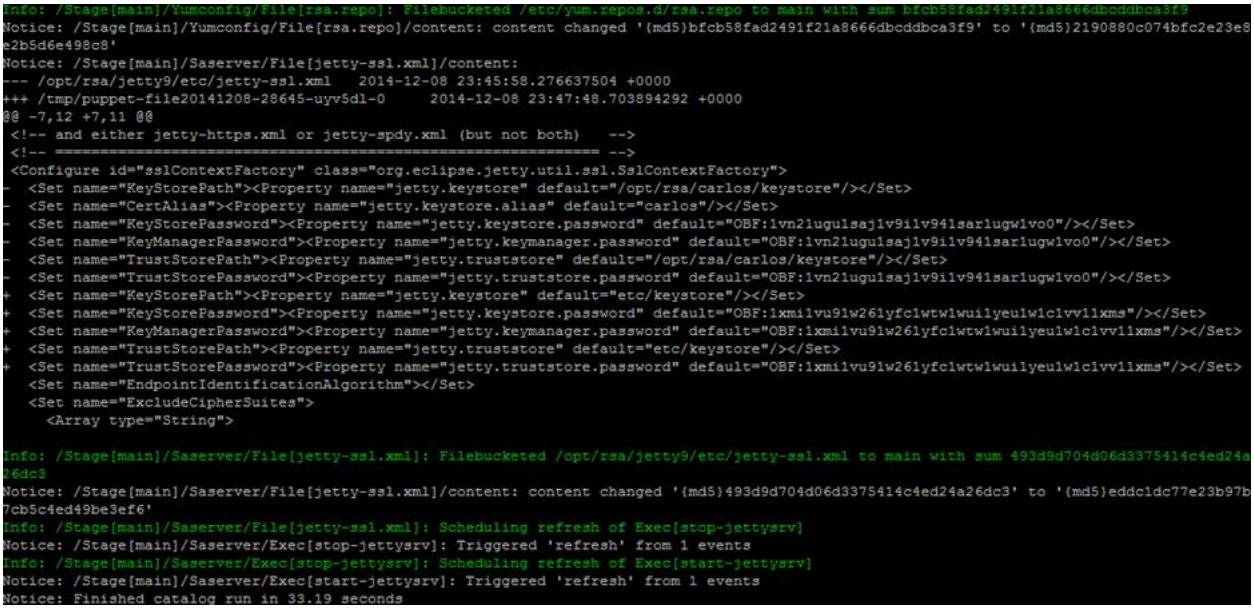
5. After making the change, start the puppetmaster service once again.

```
service puppetmaster start
```



6. Issue the command `puppet agent -t` to push the file changes to the /opt/rsa/jetty9/etc directory. **NOTE: This step will automatically restart the jettysrv service!**

When the command is executed, the changes to the jetty-ssl.xml file will be noted in the output, which will appear similar to the example below.



After Jetty has fully initialized, the new certificate chain will be in place, which can be confirmed by navigating to the Security Analytics user interface in a web browser.

Creating a Backup of the New Configuration

At this time, performing an upgrade of the Jetty server will cause the keystore and jetty-ssl.xml files to be reset to their respective default settings. Therefore, after the previous steps have been performed and it has been confirmed that the Security Analytics Web UI can be successfully accessed, a backup of the two files should be created, as shown below.

```
cp /opt/rsa/jetty9/etc/keystore /opt/rsa/jetty9/etc/keystore.bak
cp /opt/rsa/jetty9/etc/jetty-ssl.xml /opt/rsa/jetty9/etc/jetty-ssl.xml.bak
```

```
[root@CSO-SAServer-VM-01 ~]# cp /opt/rsa/jetty9/etc/keystore /opt/rsa/jetty9/etc/keystore.bak
[root@CSO-SAServer-VM-01 ~]# cp /opt/rsa/jetty9/etc/jetty-ssl.xml /opt/rsa/jetty9/etc/jetty-ssl.xml.bak
```

After performing an upgrade on the SA server, it will be necessary to overwrite the new keystore and jetty-ssl.xml (Puppet module) files with the backups that have been created in order to restore the new configuration.

For more information on this procedure, refer to the knowledge base article 29187.

Debugging and Troubleshooting

If you are unable to connect to Security Analytics with your browser, you may be able to determine the root cause by examining the logs that are stored in the **/opt/rsa/jetty9/logs/** directory. The log filenames are in the following format: YYYY_MM_DD.stderrout.log (i.e. 2014_02_10.stderrout.log)

Common issues:

- Using the wrong password for the keystore in jetty-ssl.xml
- Using the wrong password for the key manager in jetty-ssl.xml
- Using the wrong password for the trust store in jetty-ssl.xml
- Wrong path to the keystore in jetty-ssl.xml

To ensure that the certificate is configured correctly, verify that the keystore contains 2 or more entries using the command shown below.

```
keytool -list -keystore /opt/rsa/jetty9/etc/keystore
```

```
[root@CSO-SAServer-VM-01 ~]# keytool -list -keystore /opt/rsa/jetty9/etc/keystore
Enter keystore password:

Keystore type: JKS
Keystore provider: SUN

Your keystore contains 3 entries

sa, Dec 3, 2014, PrivateKeyEntry,
Certificate fingerprint (SHA1): C7:3E:0F:7D:D3:88:EE:F3:06:2D:80:D1:3C:85:4D:0E:93:A7:C1:A7
root, Dec 3, 2014, trustedCertEntry,
Certificate fingerprint (SHA1): DE:28:F4:A4:FF:E5:B9:2F:A3:C5:03:D1:A3:49:A7:F9:96:2A:82:12
intermediate, Dec 3, 2014, trustedCertEntry,
Certificate fingerprint (SHA1): 23:8C:8E:2C:D1:96:75:6B:28:67:48:9A:2E:1E:4B:1A:68:44:2C:61
```

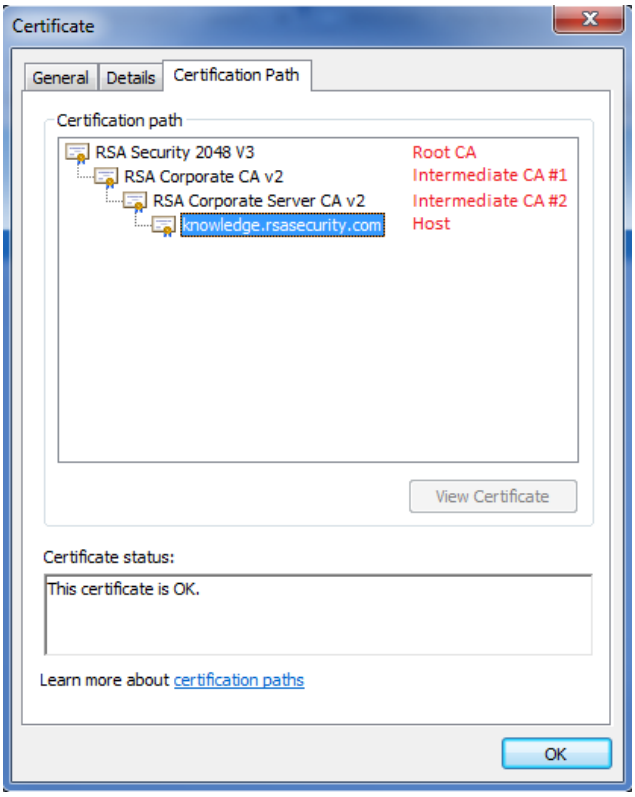
Verify the information below when examining the keytool output.

- The alias of your ROOT certificate should be of type 'trustedCertEntry'
- The alias of your INTERMEDIATE certificate should be of type 'trustedCertEntry'
- The alias of your SA Server certificate should be of type 'PrivateKeyEntry'

If the error message `"keytool error: java.lang.Exception: Failed to establish chain from reply"` is displayed when attempting to import the certificate for the SA Server, then one of the following two issues may have occurred:

- The SA Server certificate was imported before the root and intermediate certificates
- Multiple intermediate certificates are required (i.e. primary intermediate and secondary intermediate certificates) and must be imported in order before the SA Server and after importing the root certificate.

In order to illustrate how certificate chains function, refer to the screenshot below, which is the Certificate Path for the RSA SecurCare Online (SCOL) portal.



Notice that this chain has two intermediate certificates, which complete the chain between the Root CA and the host (being knowledge.rsasecurity.com).

Appendix A: RPM Packages Based on Security Analytics Version

Refer to the chart below to determine that the appliance has the appropriate RPM packages installs for the respective Security Analytics version.

	security-analytics-web-server	jettyuax
Security Analytics 10.3.1	security-analytics-web-server-10.3.1.5950-5.noarch	jettyuax-9.0.7-61.noarch
Security Analytics 10.3.2	security-analytics-web-server-10.3.2.6567-5.noarch	jettyuax-9.0.7-63.noarch
Security Analytics 10.3.3	security-analytics-web-server-10.3.3.7817-5.noarch	jettyuax-9.0.7-63.noarch
Security Analytics 10.3.4	security-analytics-web-server-10.3.4.10688-5.noarch	jettyuax-9.0.7-63.noarch
Security Analytics 10.3.5	security-analytics-web-server-10.3.5.12175-5.noarch	jettyuax-9.0.7-63.noarch
Security Analytics 10.4.0.0	security-analytics-web-server-10.4.0.0.11685-5.noarch	jettyuax-9.0.7-64.noarch
Security Analytics 10.4.0.1	security-analytics-web-server-10.4.0.1.11709-5.noarch	jettyuax-9.0.7-64.noarch
Security Analytics 10.4.0.2	security-analytics-web-server-10.4.0.2.12064-5.noarch	jettyuax-9.0.7-64.noarch
Security Analytics 10.4.1	security-analytics-web-server-10.4.1.0.14415-5.noarch	jettyuax-9.0.7-66.noarch

Document Last Modified: 24 March 2016