Configuration of SSL - Release E

DRAFT - This page contains information about the default certs in A1 Policy Management Service and how to update/replace them using Docker and Kubernetes.

General

The A1 Policy Management Service, the Information Coordinator Service and the DMaap Adapter has a default keystores and truststores that are built into the containers. The paths and passwords for these stores are located in a yaml file:
config/application.yaml

The default trust store includes a1simulator cert as a trusted cert which is located here:
https://gerrit.o-ran-sc.org/r/gitweb?p=sim/a1-interface.git;a=tree;f=near-rt-ric-simulator/certificate

The default trust store also includes a1controller cert as a trusted cert which is located here (keystore.jks file):
https://gerrit.o-ran-sc.org/r/gitweb?p=nonrtric.git;a=tree;f=sdnc-a1-controller/oam/installation/sdnc-a1/src/main/resources

By default, trust validation is not enabled (which means that a peer with any cert can connect/be connected to the components).

A1 Policy Management Service, configuration of SSL in Kubernetes

The keystore and truststore can be configured in Kubernetes. In the source code repository, the default files are located. Updating this in a running cluster can be done in more than one way, but this is probably the simplest.
First you need to create a directory with three files.

config/
application.yaml
keystore.jks
truststore.jks

The default application.yaml can be taken from the source code repository or by using command "kubectl describe configmap policymanagementservice-configmap -n nonrtric"
Then you need to create the keystore.jks and (if used, this is not used by default) a truststore.jks .
There is a README file in the source code repository that describes how the default keystore and truststore are created. This involves creating a CA cert used for signing.

If you are happy with just a self signed cert, it can be created using the keytool command. Example:

```
keytool -genkeypair -alias policy_agent -keyalg RSA -keysize 2048 -keystore keystore.jks -validity 3650 -
storepass <key store password>
```

The following parameters in the application.yaml needs to be updated (the non relevant parameters for this are omitted):
...

server:
ssl:
key-store-type: JKS
key-store-password: ***<key password>***
key-store: ***/opt/app/policy-agent/config/keystore.jks ---- Note that the path needs to be updated to this***
key-password: ***<key store password>***
key-alias: ***<key alias>***
 # Add the following 2 parameters if you want to enable trust validation for accesses towards the service (mainly the service NBI).
trust-store-password: <trust store password>
trust-store: /opt/app/policy-agent/config/truststore.jks
app:
webclient:
 # – Update the following parameters if you want to enable trust validation in the connections from the service towards the RAN
trust-store-used: ***true***
trust-store-password: ***<trust store password>***
trust-store: ***/opt/app/policy-agent/config/truststore.jks***
...

When this is done you can update the config map that holds the configuration for the service:

```
kubectl create configmap  policymanagementservice-configmap-config --from-file=./config --dry-run=client -n
nonrtric -o yaml | kubectl apply -f -
```

The three files file be placed in the configuration directory for the service. A POD restart is needed for the changes to take effect.

Enrichment Coordinator Service, configuration of SSL in Kubernetes

Configuration of the SSL for the ECS is done in the same was as for the A1 Policy Management Service (see above). There are two differences:

1. The name of the config map is *enrichmentservice-configmap.*
2. The file paths for the config directory for this component (where the config map is mounted) is */opt/app/enrichment-coordinator-service/config.*

So the application.yaml paths will then be:

*trust-store: /opt/app/**enrichment-coordinator-service**/config/truststore.jks*
*key-store:/opt/app/**enrichment-coordinator-service**/config/keystore.jks*

The command for updating the config map is:

```
kubectl create configmap  enrichmentservice-configmap --from-file=./config --dry-run=client -n nonrtric -o yaml
| kubectl apply -f -
```

DMaaP Adapter Service, configuration of SSL in Kubernetes

Configuration of the SSL is done in the same was as for the A1 Policy Management Service and the ECS (see above). There are two differences:

1. The name of the config map is dmaapadapterservice-*configmap-config.*
2. The file paths for the config directory for this component (where the config map is mounted) is */opt/app/enrichment-coordinator-service/config.*

So the application.yaml paths will then be:

*trust-store: /opt/app/**dmaap-adaptor-service**/config/truststore.jks*
*key-store:/opt/app/**dmaap-adaptor-service**/config/keystore.jks*

The command for updating the config map is:

```
kubectl create configmap  dmaapadapterservice-configmap-config --from-file=./config --dry-run=client -n
nonrtric -o yaml | kubectl apply -f -
```

A1 Policy Management Service, configuration of SSL in Docker

The default keystore, truststore, and application.yaml files can be overridden by mounting new files using the "volumes" field of docker-compose or docker run command.

Assuming that the keystore, truststore, and application.yaml files are located in the same directory as docker-compose, the volumes field should have these entries:

```
volumes:
    - ./new_keystore.jks:/opt/app/policy-agent/etc/cert/keystore.jks:ro

    - ./new_truststore.jks:/opt/app/policy-agent/etc/cert/truststore.jks:ro

    - ./new_application.yaml:/opt/app/policy-agent/config/application.yaml:ro
```

The target paths in the container should not be modified.

Example docker run command for mounting new files (assuming they are located in the current directory):

```
docker run -p 8081:8081 -p 8433:8433 --name=policy-agent-container --network=nonrtric-docker-net --volume "$PWD/new_keystore.jks:/opt/app/policy-agent/etc/cert/keystore.jks" --volume "$PWD/new_truststore.jks:/opt/app/policy-agent/etc/cert/truststore.jks" --volume "$PWD/new_application.yaml:/opt/app/policy-agent/config/application.yaml" o-ran-sc/nonrtric-policy-agent:2.2.0-SNAPSHOT
```