

Run Apex Policy for O-RU & O-DU use case

This page describes how to run the stand-alone ONAP policy framework and deploy the apex policy designed for O-RU and O-DU use case.

Assumption:

It is assumed that the DMaaP message router is already running as a docker container with the hostname "onap-dmaap" and connected to the docker network named "nonrtric-docker-net". It is also assumed that a topic named "unauthenticated.SEC_FAULT_OUTPUT" has already been created in the message router.

Start the ONAP Policy framework

The ONAP Policy framework can be run in a stand-alone mode using the docker-compose in OSC nonrtric repo:

```
git clone "https://gerrit.o-ran-sc.org/r/nonrtric"
git checkout dawn --track origin/dawn
cd nonrtric/docker-compose/docker-compose-policy-framework
```

Start all the containers by running this command:

```
docker-compose up -d
```

Run SDNR-simulator

The apex policy will make a REST call to the SDNR-simulator for sending the configuration messages. The code for SDNR simulator is available in the nonrtric repo. Run the following commands to build the docker image for SDNR-simulator:

```
cd nonrtric/test/usecases/oruclosedlooprecovery/scriptversion/simulators
docker build . -f Dockerfile-sdnr-sim -t sdnr-simulator:1.0.0
```

The SDNR-simulator can then be started using the following command:

```
docker run --rm --name sdnr-sim --network nonrtric-docker-net -e MR-HOST="http://onap-dmaap" -e MR-PORT="3904"
sdnr-simulator:1.0.0
```

Create/Deploy apex policy for O-RU & O-DU use case

All the ingredients of apex policy designed for O-RU & O-DU use case are available here:

```
cd nonrtric/test/usecases/oruclosedlooprecovery/apexpolicyversion/LinkMonitor
```

In order to deploy the apex policy with default config, only the contents under /deployment directory are needed. However, if something needs to be changed in the policy config, refer to the section named "Workflow for updating the policy config" at the end of this page.

- Before creating the apex policy, it is good to perform a healthcheck call:

```
curl -k -u 'healthcheck:zb!XztG34' -X GET "https://localhost:6868/policy/pap/v1/components/healthcheck"
```

This should give the following response:

```
{"pdps":{"xacml":[{"instanceId":"policy-xacml-pdp","pdpState":"ACTIVE","healthy":"HEALTHY"}],"drools":[]},"apex":{"instanceId":"policy-apex-pdp","pdpState":"ACTIVE","healthy":"HEALTHY","message":"Pdp Heartbeat"}},{"healthy":false,"api":{"name":"Policy API","url":"https://policy-api:6969/policy/api/v1/healthcheck","healthy":true,"code":200,"message":"alive"},"pap":{"name":"Policy PAP","url":"https://policy-pap:6969/policy/pap/v1/healthcheck","healthy":true,"code":200,"message":"alive"}}
```

Make sure that the policy-apex-pdp, api and pap are healthy.

Note: It might take a few minutes before the policy-apex-pdp reports its health status. If health is not reported after waiting for a few minutes, check the logs of policy-apex-pdp docker container to find any error messages.

- Create the policy via Policy API:

```
curl -k -u 'healthcheck:zb!XztG34' -X POST "https://localhost:6969/policy/api/v1/policytypes/onap.policies.native.Apex/versions/1.0.0/policies" -H "Accept: application/json" -H "Content-Type: application/json" -d @./deployment/ToscaPolicy.json
```

- Make sure that the following REST call returns the policy:

```
curl -k -u 'healthcheck:zb!XztG34' -X GET "https://localhost:6969/policy/api/v1/policytypes/onap.policies.native.Apex/versions/1.0.0/policies/onap.policies.native.apex.LinkMonitor/versions/1.0.0"
```

- Deploy the policy to apex-pdp via Policy PAP:

```
curl -k -u 'healthcheck:zb!XztG34' -X POST "https://localhost:6868/policy/pap/v1/pdps/policies" -H "Accept: application/json" -H "Content-Type: application/json" -d @./deployment/DeployPolicyPAP.json
```

This should give the following response:

```
{"message":"Use the policy status url to fetch the latest status. Kindly note that when a policy is successfully undeployed, it will no longer appear in policy status response.", "uri":"/policy/pap/v1/policies/status"}
```

- Check the status of deployed policy:

```
curl -k -u 'healthcheck:zb!XztG34' -X GET "https://localhost:6868/policy/pap/v1/policies/status"
```

This should give the following response:

```
[{"pdpGroup":"defaultGroup","pdpType":"apex","pdpld":"policy-apex-pdp","policy":{"name":"onap.policies.native.apex.LinkMonitor","version":"1.0.0"},"policyType":{"name":"onap.policies.native.Apex","version":"1.0.0"},"deploy":true,"state":"SUCCESS"}, {"pdpGroup":"defaultGroup","pdpType":"xacml","pdpld":"policy-xacml-pdp","policy":{"name":"SDNC_Policy.ONAP_NF_NAMING_TIMESTAMP","version":"1.0.0"},"policyType":{"name":"onap.policies.Naming","version":"1.0.0"},"deploy":true,"state":"SUCCESS"}]
```

- Send the LinkFailureEvent to DMAAP Message Router:

```
curl -X POST -H accept:application/json -H Content-Type:application/json "http://localhost:3904/events/unauthenticated.SEC_FAULT_OUTPUT/" -d @./events/LinkFailureEvent.json
```

- The SDNR-simulator should receive the following call if the apex policy is working:

```
"POST /rests/data/network-topology:network-topology/topology=topology-netconf/node=HCL-O-DU-1123/yang-ext:mount/o-ran-sc-du-hello-world:network-function/du-to-ru-connection=ERICSSON-O-RU-11225 HTTP/1.1" 200
```

- The logs of policy-apex-pdp docker container should show some messages exchanged related to the LinkFailureEvent that was sent.

```
docker logs policy-apex-pdp
```

Workflow for updating the policy config

In case some changes need to be made in the policy config (located in /config directory), a new ToscaPolicy.json file (located in /deployment directory) has to be generated for deploying the apex policy. The new file can be created by using the apexCLIToscaEditor.sh script in the policy-apex-pdp docker container via the following steps:

- Run the policy-apex-pdp docker container and mount the LinkMonitor directory (located in the nonrtric repo) into the docker container:

```
docker run -it --rm --name policy-apex-pdp --network nonrtric-docker-net -v <<Path_to_nonrtric_repo>>/nonrtric/test/usecases/oruclosedlooprecovery/apexpolicyversion/LinkMonitor/:/home/apexuser/examples/LinkMonitor/ onap/policy-apex-pdp:latest
```

- Get into the docker container and create the updated ToscaPolicy.json file:

```
docker exec -it policy-apex-pdp sh
```

```
/opt/app/policy/apex-pdp/bin/apexCLIToscaEditor.sh -c /home/apexuser/examples/LinkMonitor/models  
/LinkMonitorModelJavascript_0.0.1.apex -ot /home/apexuser/examples/LinkMonitor/deployment/ToscaPolicy.json -ac  
/home/apexuser/examples/LinkMonitor/config/LinkMonitorConfigDmaap2RestJsonEvent.json -t /home/apexuser/examples  
/LinkMonitor/tosca/ToscaTemplate.json
```

- After running the above command, the ToscaPolicy.json file is automatically updated in the /deployment directory of nontrac in the host where docker is running.