Release D: Simple script for O-RU & O-DU use case

Ref: ~~NONRTRIC-487~~ - NONRTRIC - OSC D-Release O-DU & O-RU recovery use case DONE

- O-RU sends alarm as VES event to "unauthenticated.SEC_FAULT_OUTPUT"
- Poll DMaaP for alarms from "unauthenticated.SEC_FAULT_OUTPUT"
- Filter the alarms for "faultFields:alarmCondition" set to "28" ("C/U-plane logical Connection faulty"), with "faultFields:eventSeverity" anything but "NORMAL"
- Find O-RU identity in the alarm, provided in "commonEventHeader:sourceName", and then find the corresponding O-DU through hard coded mapping (for now)
- REST conf call to OAM Controller through SDNR, using endpoint:
  "/rests/data/network-topology:network-topology/topology=topology-netconf/node=[O-DU-ID]/yang-ext:mount/o-ran-sc-du-hello-world:network-function/du-to-ru-connection=[O-RU-ID]",
  to set the administrative state of the O-DU to UNLOCKED, providing data:
  {"o-ran-sc-du-hello-world:du-to-ru-connection": [{"name": "O-RU-ID", "administrative-state": "UNLOCKED"}]}
- (Poll DMaaP and trace out the alarm clear, "faultFields:eventSeverity" "NORMAL")

- DMaaP :
  - https://gerrit.o-ran-sc.org/r/gitweb?p=oam.git;a=blob;f=solution/dev/ves-test-collector/client-scripts-ves-v7/json/templates/fault.json;hb=HEAD
    - The @....@ entries will be replaced by the values from https://wiki.o-ran-sc.org/display/OAM/o-ran-fm%3Aalarm-notif+to+ves%3Afault#oranfm:alarmnotiftoves:fault-MappingtoVESfaultfields
  - The topics etc are described in https://wiki.o-ran-sc.org/display/OAM/VES+Events+and+Topics+on+Message+Router
- RestConf call:
  - The latest draft of the Yang file for the O-DU is seen at:
    - https://wiki.o-ran-sc.org/display/OAM/Use+case+driven+YANG+modules+for+O-DU and
    - https://wiki.o-ran-sc.org/download/attachments/20878535/o-ran-sc-du-hello-world.yang?api=v2 and
    - https://gerrit.o-ran-sc.org/r/c/scp/oam/modeling/+/5854

The script version is now available as a docker image in the Nexus repo, see Docker image.

The script has default values for all data that can be modified at startup with the following parameters:

```
usage: PROG [-h] [--mrHost MRHOST] [--mrPort MRPORT] [--mrTopic MRTOPIC] [--sdnrHost SDNRHOST] [--sdnrPort
SDNRPORT] [--oRuTooDuMapFile ORUTOODUMAPFILE] [--pollTime POLLTIME] [-v] [--version]

optional arguments:
-h, --help show this help message and exit
--mrHost MRHOST The URL of the MR host (default: http://message-router.onap)
--mrPort MRPORT The port of the MR host (default: 3904)
--mrTopic MRTOPIC The topic to poll messages from (default: unauthenticated.SEC_FAULT_OUTPUT)
--sdnrHost SDNRHOST The URL of the SNDR host (default: http://localhost)
--sdnrPort SDNRPORT The port of the SDNR host (default: 9990)
--oRuTooDuMapFile ORUTOODUMAPFILE
A file with the mapping between O-RU ID and O-DU ID as a dictionary (default: o-ru-to-o-du-map.txt)
--pollTime POLLTIME The time between polls (default: 10)
-v, --verbose Turn on verbose printing
--version show program's version number and exit
```

The container must be connected to the same network as the MR and SDNR are running in. The parameters to the application can be provided with the `-e PARAM_NAME=PARAM_VALUE` notation. Start the container by using the command, with available params listed:

```
docker run --network [NETWORK NAME] --name oru-app -e VERBOSE=on -e MR-HOST=[HOST NAME OF MR] -e MR-PORT=[PORT OF
MR] -e SDNR-HOST=[HOST NAME OF SDNR] -e SDNR-PORT=[PORT OF SDNR] oru-app
```