SDC Sanity

Introduction

SDC sanity Docker is a part of our system and its aim is to create a suite of tests that will perform a basic E2E flow (Onboarding).

Docker execution

Sanity Docker executed as part of our Docker run script and will be started on demand as part of other Dockers init process.

Executing example:

/data/scripts/docker_run.sh -r 1.1-stagging-latest -t

-t flag meaning start sanity docker

About onboarding E2E flow:

Onboarding E2E flow cover following SDC functionality:

- Onboard of VNF
- · Create VF from VSP
- Certify VF
- Create Service
- · Add VF to service
- · Certify Service
- Export TOSCA and validate it structure using external TOSCA parser

**as part of execution we open a connection to Titan and perform resources clean up both before and after tests execution (only resource with "ci" prefix will be deleted from the catalog)

List of VNFs/PNFs that proceed by onboarding flow:

- sample-signed-pnf-cms-includes-cert-1.0.1-SNAPSHOT.zip
- sample-signed-pnf-1.0.1-SNAPSHOT.zip
- sample-pnf-1.0.1-SNAPSHOT.csar
- sample-pnf-custom-type.csar
- vbng.zip
- vgmux.zip
- vfw.zip
- vvg.zip
- base_vvg.zipvLBMS.zip
- vLBIVIS.2vgw.zip
- infra.zip
- vLB.zip
- base_vfw.zip
- vbrgemu.zip
- resource-ZteEpcMmeVf-csar_fix.csar
- ZteEpcMmeVf.csa
- Huawei_vHSS.csar
- vSBC_update_v03.csar
- Huawei_vMME.csar
- vsp-vgw.csar
- database-substitution-mappings.csar

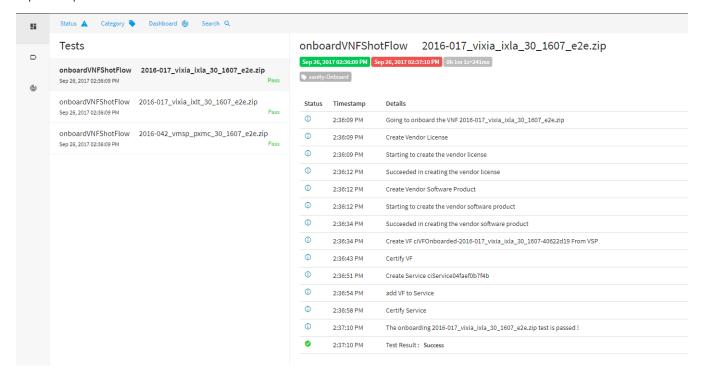
Execution time:

Current tests execution time is ~7min - this includes onboarding of 22 VNF/PNF running in 1 thread

Report location:

/data/logs/sdc-api-tests/ExtentReport/
/data/logs/sdc-ui-tests/ExtentReport/

Report example:



CSIT:

SDC plans to leverage our internal sanity docker as part of the CSframeworkork. we will be executing the docker from the CSIT and storing the results

Future inchantments:

SDC plans to add ui testing capabilities to our sanity docker by changing the docker os and adding Selenium.
 The UI testing will leverage the exciting java selenium test already available in the SDC code base.



2. SDC plans to change the flow of the tests so that the test input will be dynamically retrieved from the vnf repo instead of placing the input in the docker itself,

this way we will be able to re execute the test with updated inputs without creating a new Docker image..

