Oracle® Communications 5G Automated Testing Suite Guide

Release 1.2.2

F37945-01

September 2020

Oracle Communications 5G Automated Testing Suite Guide , Release 1.2.2

F37945-01

Contents

List of Figures

List of Tables

What's New in This Guide

This section shares the list of new features introduced in every ATS release. For more release specific information, you can refer to its release notes.

Release 1.2.2

Following new features are added to ATS 1.2.2 release:

Table Features

Feature	NRF	NSSF	Policy	SCP	UDR
Service Mesh Support	Yes	No	Yes	Yes	Yes
RBAC authorization	Same as Previous Release (Role Binding)	Same as Previous Release (Cluster Role Binding)	Enhanced Policy ATS to support Role Binding from this release onwards.	Same as Previous Release (Role Binding)	Same as Previous Release (Role Binding)
New TestCases added to Jenkins Pipeline	Not Applicable	Not Applicable	New test cases are added.	No new test cases are added.	Not Applicable

Release 1.2.1

Following new features are added to ATS 1.2.1 release:

Table Features

Feature	NRF	NSSF	Policy	SCP	UDR
Service Mesh Support	Yes	No	No	No	Yes. It includes envoy side car injection.
RBAC authorization	Same as Previous Release (Role Binding)	Same as Previous Release (Cluster Role Binding)	Same as Previous Release (Cluster Role Binding)	Same as Previous Release (Role Binding)	Enhanced SLF ATS to support Role Binding from this release onwards.
New TestCases added to Jenkins Pipeline	No new test cases are added.	Not Applicable	Not Applicable	Not Applicable	No new test cases are added.

ATS Release 1.2.0

Following new features are added to ATS 1.2.0 release:

Table Features

Feature	NRF	NSSF	Policy	SCP	UDR
NF Specific login support - Only NF specific pipelines are visible	Yes	Yes	Yes	Yes	Yes
RBAC authorization	Role Binding	Cluster Role Binding	Cluster Role Binding	Role Binding	Cluster Role Binding

Table (Cont.) Features

Feature	NRF	NSSF	Policy	SCP	UDR
One-click start of <NF> New-feature and Regression pipelines for **'All'** option after successfully logging into the Jenkins GUI.	Yes **Example:** • **New-Features One-Click** - To run **"All"** testcases in **NRF-NewFeatures** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/NRF-NewFeatures/build** link to open Jenkins and click **"Build"**. • **Regression One-Click** - To run **"All"** testcases in **NRF-Regression** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/NRF-Regression/build** link to open Jenkins and click **"Build"**.	Yes **Example:** • **New-Features One-Click** - To run **"All"** test cases in **NSSF-NewFeatures** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/NSSF-NewFeatures/build** link to open Jenkins and once opened click **"Build"**.	Yes **Example:** • **New-Features One-Click** - To run **"All"** testcases in **Policy-NewFeatures** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/Policy-NewFeatures/build** link to open Jenkins and click **"Build"**. • **Regression One-Click** - To run **"All"** testcases in **Policy-Regression** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/Policy-Regression/build** link to open Jenkins and click **"Build"**.	Yes **Example:** • **New-Features One-Click** - To run **"All"** testcases in **SCP-NewFeatures** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/SCP-NewFeatures/build** link to open Jenkins and click **"Build"**. • **Regression One-Click** - To run **"All"** testcases in **SCP-Regression** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/SCP-Regression/build** link to open Jenkins and click **"Build"**.	Yes **Example:New-Features One-Click** - To run **"All"** testcases in **SLF-NewFeatures** pipeline, use **http://<Jenkins_IP>:<Jenkins_Port>/job/SLF-NewFeatures/build** link to open Jenkins and click **"Build"**.
Allows to deploy ATS using either **Helm2** or **Helm3** helm versions	Yes	Yes. Helm 3 is supported via workaround available in NSSF ATS Installation procedure.	Yes	Yes	Yes

Table (Cont.) Features

Feature	NRF	NSSF	Policy	SCP	UDR
One-time configuration on Jenkins GUI to execute the test cases.	Yes	Yes	Yes	Yes	Yes
No need to login to ATS pod through CLI. Users can perform all operations through Jenkins GUI.	Yes	Yes	Yes	Yes	Yes
Allows to execute Sanity cases to validate the NF and ATS deployment.	Yes	Yes	No	No	No
Does not display **Skipped cases** in the console output when executing the cases using **Single/ Multiple** feature files option.	Yes	No	Yes	Yes	Yes
The **Documentation** section shows all the testcases according to the service operation supported by NF.	Yes	Yes	Yes	Yes	Yes
Default re-run count for failed scenarios	0	0	0	0	2

Table (Cont.) Features

Feature	NRF	NSSF	Policy	SCP	UDR
New TestCases added to Jenkins Pipeline	Provides a total of **169 scenarios** clubbed together in **64 feature files** of NRF ATS - 1.7.0 New Feature pipeline.	Provides a total of **150 scenarios** clubbed together in **26 feature files** in NSSF ATS - 1.4 New Feature pipeline.	Provides a total of **13 scenarios** clubbed together in **5 feature files** of Policy ATS - 1.7.1 New Feature pipeline.	Provides a total of **25 scenarios** clubbed together in **4 feature files** of SCP ATS - 1.7.0 New Feature pipeline.	Provides a total of **45 scenarios** clubbed together in **4 feature files** of SLF ATS - 1.7.0 New Feature pipeline.
Previous Release TestCases	Provides a total of **163 scenarios** clubbed together in **80 feature files** of NRF ATS - 1.6.1 Regression pipeline.	Not applicable. This is the first release of NSSF-ATS.	Provides 28 Feature files in PCF-Regression pipeline.	Provides 65 cases in 9 feature files.	Not applicable. This is the first release of UDR-ATS.
Backward Compatibility	ATS is **NOT** backward compatible. It means NRF ATS - 1.6.1 will work only with NRF 1.6.1. For NRF 1.5.0, user still need to use ATS - 1.0.0 version.	Not applicable. This is the first release of NSSF-ATS.	ATS is **NOT** backward compatible. It means PCF ATS - 1.1.0 will work only with PCF 1.6.1. For PCF 1.5.0, user still need to use ATS - 1.0.0 version.	ATS is **NOT** backward compatible. It means SCP ATS - 1.7.0 works only with SCP 1.7.0	Not applicable. This is the first release of UDR-ATS.
Supports NF with TLS Enabled (server side) and Disabled mode	Not applicable	Not applicable	Yes. Policy ATS supports Policy with TLS Enabled (server side) and Disabled mode	Not applicable	Not applicable
Test cases delivered in ATS Release 1.0.0 and 1.1.0 are added to its respective Regression Pipeline. User can run any pipeline but not parallel.	Yes	Not applicable	Yes	Yes	Not applicable

1

Understanding Automated Testing Suite (ATS)

In this chapter, you will get an overview about ATS, its need and its features.

Automated Testing Suite Overview

Automated Testing Suite (ATS) allows you to execute software test cases using an automated testing tool and then, compares the actual results with the expected or predicted results. In this process, there is no intervention from the user.

ATS for 5G Network Functions

For 5G Network Functions (NFs), ATS is built using **Oracle Linux 7-slim** as the base image. **Jenkins** is a part of the ATS image and it provides a GUI interface to the users to test either a single NF or multiple NFs independently in the same environment.

Along with the NF docker images, user are provided with the ATS image, simulator images, and test cases for the specific NF. All these are handed over to the customer as a fully automated suite so that they can directly perform Lab deployment and testing. You can combine it with any other **Continuous Integration (CI) pipeline** with minimal changes. Since, 5G ATS uses Jenkins as GUI.

Why Automated Testing Suite in 5G NFs?

Through Automated Testing Suite (ATS), Oracle Communications aims at providing an end-to-end solution to its customers for deploying and testing its 5G-NFs.

This guide covers implementation of ATS in 5G NFs like,

- Network Repository Function (NRF)
- Policy Control Function (PCF)
- Service Communication Proxy (SCP)
- Network Slice Selection Function (NSSF)
- Unified Data Repository (UDR)

ATS Features

The ATS features are as follows:

- Provides an end-to-end solution to the customers for testing Oracle Communications 5G-NFs. The ATS package includes:
  - Test scripts and docker images of test container.
    - The docker images have complete framework and libraries installed, which is common for all NFs working with BDD framework.

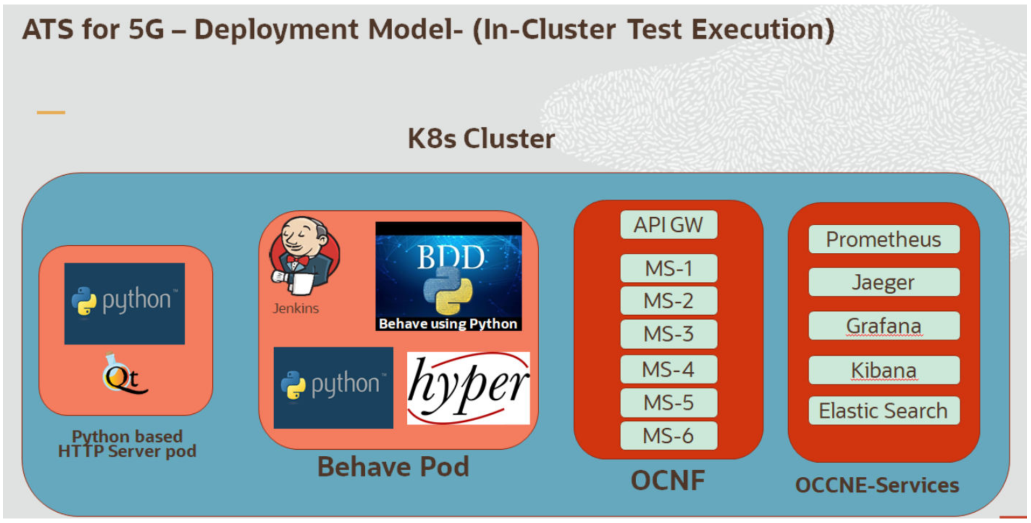
  - Docker image of HTTP Server simulator
  - Helm chart to deploy the ATS (delivered as a tar file)
  - Readme text file (.txt file)
- Enables all the NF teams with the basic environment, framework and a GUI (Jenkins) to execute all the functional test cases.

Deployment Model (In-Cluster)

According to **In-Cluster deployment model**, ATS can co-exist in the same cluster where the NFs are deployed. This deployment model is useful for In-Cluster testing.

> **Note:**
>
> The ATS 1.2.1 package supports in-cluster deployment.

Figure 1-1 In-Cluster Deployment Model



> **Note:**
>
> GO Language is used to create stubs for Policy ATS and SCP ATS.

2 NF ATS Installation Procedure

In this chapter, you will learn to install ATS for different network function platforms like,

- NRF
- NSSF
- Policy
- SCP
- SLF

NRF ATS Installation Procedure

The NRF ATS installation procedure covers three steps:

1. Locating and downloading ATS and Simulator Images
2. Preparing to deploy ATS and Stub Pod in Kubernetes Cluster
3. Deploying ATS and Stub Pod in Kubernetes Cluster

Locating and Downloading ATS Images

The steps to locate and download ATS Images are as follows:

1. Login to My Oracle Support using the appropriate credentials.
2. Select **Patches & Updates** tab.
3. In the **Patch Search** console, select **Product or Family (Advanced)** tab.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.
5. Select *Oracle Communications Cloud Native Core Network Repository Function <release_number>* in **Release** field.
6. Click **Search**. The **Patch Advanced Search Results** list appears.
7. Select the required ATS patch from the list. The **Patch Details** window appears.
8. Click **Download**. The **File Download** window appears.
9. Click the **<p********_<release_number>_Tekelec>.zip** file.
10. Extract the ATS release package zip file to download the ATS images to the system where network function must be installed.
11. The `ocats-nrf` directory has following files:

```
ocats-nrf-tools-pkg-1.7.2.0.0.tgz
ocats-nrf-tools-pkg-1.7.2.0.0-README.txt
ocats-nrf-custom-configtemplates-1.7.2.0.0.zip
ocats-nrf-custom-configtemplates-1.7.2.0.0-README.txt
```

12. The `ocats-nrf-tools-pkg-1.7.2.0.0-README.txt` file contains all the information required for the package.
13. The `ocats-nrf-tools-pkg-1.7.2.0.0.tgz` file has following images and charts packaged as tar files:

    ```
    ocats-nrf-tools-pkg-1.7.2.0.0.tgz

          |

          |_ _ _ocats-nrf-pkg-1.7.2.0.0.tgz

          |        |_ _ _ _ _ _ ocats-nrf-1.7.2.tgz (Helm Charts)

          |        |_ _ _ _ _ _ ocats-nrf-image-1.7.2.tar (Docker
    Images)

          |        |_ _ _ _ _ _ OCATS-NRF-Readme.txt

          |

          |_ _ _ocstub-python-pkg-1.7.2.0.0.tgz

                   |_ _ _ _ _ _ ocstub-1.7.2.tgz (Helm Charts)

                   |_ _ _ _ _ _ ocstub-python-image-1.7.2.tar (Docker
    Images)

                   |_ _ _ _ _ _ OCSTUB-PYTHON-Readme.txt
    ```

    In addition to the above images and charts, the **ocats-nrf-custom-configtemplates-1.7.2.0.0.zip** file is also there in the same location. The **ocats-nrf-custom-configtemplates-1.7.2.0.0-README.txt** file contains the information about the content of this zip file. The content of the zip file is as follows:

    ```
    ocats-nrf-custom-configtemplates-1.7.2.0.0.zip

          |

          |_ _ _ocats-nrf-custom-values.yaml (Custom values file for
    installation)

          |

          |_ _ _ocats-nrf-custom-serviceaccount.yaml (Template to
    create custom service account)

          |

          |_ _ _ocstub-python-custom-values.yaml (Custom values file
    for stub installation
    ```

14. The user can copy the tar file from here to the OCCNE/OCI/Kubernetes cluster where they want to deploy ATS.

Preparing to Deploy ATS and Stub Pod in Kubernetes Cluster

The steps to deploy ATS and Stub Pod in Kubernetes Cluster are as follows:

1. Execute the following command to extract tar file content.
   `tar -xvf ocats-nrf-tools-pkg-1.7.2.0.0.tgz`

   The output of this command is:

   ```
   ocats-nrf-pkg-1.7.2.0.0.tgz
   ocstub-python-pkg-1.7.2.0.0.tgz
   ```

2. Execute the given command to extract final helm charts and docker images of ATS.
   `tar -xvf ocats-nrf-pkg-1.7.2.0.0.tgz`

   The output of this command is:

   ```
   ocats-nrf-image-1.7.2.tar
   ocats-nrf-1.7.2.tgz
   OCATS-NRF-Readme.txt
   ```

   > **Note:**
   >
   > The `OCATS-NRF-Readme.txt` file contains all the information required for the package.

3. Execute the following command to untar the ocstub package .
   `tar -xvf ocstub-python-pkg-1.7.2.0.0.tgz`

   The output of this command is:

   ```
   ocstub-python-image-1.7.2.tar
   ocstub-python-1.7.2.tgz
   OCSTUB-PYTHON-Readme.txt
   ```

4. Execute the following command to extract the content of the custom values file:
   `unzip ocats-nrf-custom-configtemplates-1.7.2.0.0.zip`

   The output of this command is:

   ```
   ocats-nrf-custom-values.yaml (Custom yaml file for deployment of
   OCATS-NRF)
   ocats-nrf-custom-serviceaccount.yaml (Custom yaml file for service
   account creation to help customer if required)
   ocstub-python-custom-values.yaml (Custom yaml file for deployment
   of OCSTUB-PYTHON)
   ```

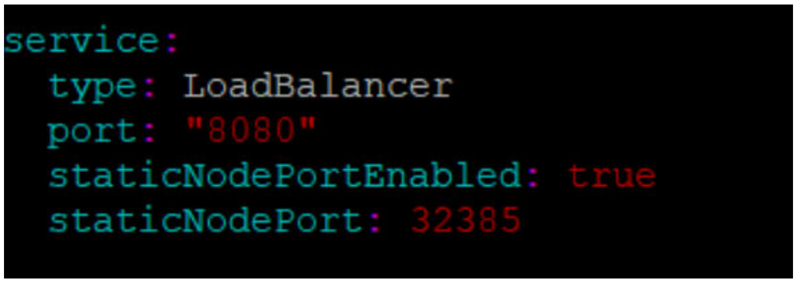
5. In your cluster, load the ATS docker image, '**`ocats-nrf-image-1.7.2.tar`**' and Stub docker image, '**`ocstub-python-image-1.7.2.tar`**' and push it to your registry.

   ```
   docker load -i ocats-nrf-image-1.7.2.tar
   docker push <registry>/ocats-nrf:1.7.2
   ```

```
docker load -i ocstub-python-image-1.7.2.tar
docker push <registry>/ocstub-python:1.7.2
```

6. Update the image name and tag in the `ocats-nrf-custom-values.yaml` and `ocstub-python-custom-values.yaml` file as required.
   For this, you need to open the `ocats-nrf-custom-values.yaml` and `ocstub-python-custom-values.yaml` file and update the `image.repository` and `image.tag`
7. ATS supports static port. By default, this feature is not available. To enable this feature:
   - In the **ocats-nrf-custom-values.yaml** file under service section, set the **staticNodePortEnabled** parameter value to 'true' and **staticNodePort** parameter value with valid nodePort.
   - A sample screen is given below:

Figure 2-1 ocats-nrf-custom-values.yaml - service section



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: true
  staticNodePort: 32385
```

Enabling Service Mesh for ATS

Note:

This procedure is applicable only if you want to enable service mesh.

To enable service mesh for ATS, perform the following steps:

1. If the service mesh is not enabled at the global level for the namespace, execute the following command to enable it before deploying ATS.

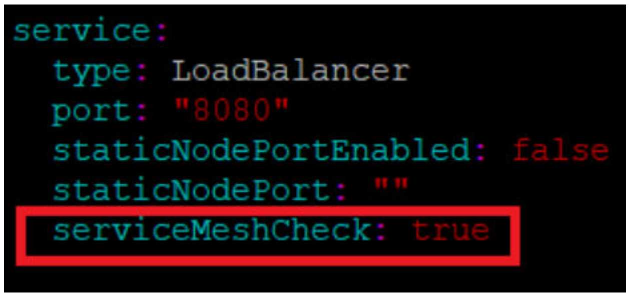
   ```
   kubectl label --overwrite namespace <namespace_name> istio-
   injection=enabled
   ```

   Example

   ```
   kubectl label --overwrite namespace ocnrf istio-injection=enabled
   ```

2. Under the **service** section of the **ocats-nrf-custom-values.yaml** file , set the **serviceMeshCheck** parameter **true**. By default, this parameter is set to false. A snippet of service section in the yaml file is given below:

Figure 2-2 Enabling Service Mesh



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: false
  staticNodePort: ""
  serviceMeshCheck: true
```

Deploying ATS and Stub Pod in Kubernetes Cluster

> **Note:**
>
> **It is important to ensure that all the three components; ATS, Stub and NRF are in the same namespace.**

You need two Stubs for the NRF tests to be executed. The service name for the stubs should be **notify-stub-service** and **notify-stub-service02**.

ATS and Stub supports Helm2 and Helm3 for deployment.

If the namespace does not exists, execute the following command to create a namespace:

```
kubectl create namespace ocnrf
```

Using Helm 2 for ATS:

```
helm install ocats-nrf-1.7.2.tgz --name <release_name> --namespace
<namespace_name> -f <values-yaml-file>

For example -
helm install ocats-nrf-1.7.2.tgz --name ocats --namespace ocnrf -f
ocats-nrf-custom-values.yaml
```

Using Helm 2 for Stubs:

```
helm install ocstub-python-1.7.2.tgz --set service.name=<stub-service-
name>
 --name <release_name> --namespace <namespace_name> -f <values-yaml-
file>

For example -
helm install ocstub-python-1.7.2.tgz --set service.name=notify-stub-
service --name
ocstub --namespace ocnrf -f ocstub-python-custom-values.yaml
helm install ocstub-python-1.7.2.tgz --set service.name=notify-stub-
service02 --name
ocstub1 --namespace ocnrf -f ocstub-python-custom-values.yaml
```

Using Helm 3 for ATS:

```
helm3 install -name <release_name> ocats-nrf-1.7.2.tgz --namespace
<namespace_name> -f <values-yaml-file>
```

Example:`helm3 install -name ocats ocats-nrf-1.7.2.tgz --namespace ocnrf -f ocats-nrf-custom-values.yaml`

Using Helm 3 for Stubs:

```
helm3 install -name <release_name> ocstub-python-1.7.2.tgz --set
service.name=<stub-service-name> --namespace <namespace_name> -f <values-
yaml-file>
```

Example: `helm3 install -name ocstub ocstub-python-1.7.2.tgz --set service.name=notify-stub-service --namespace ocnrf -f ocstub-python-custom-values.yaml`

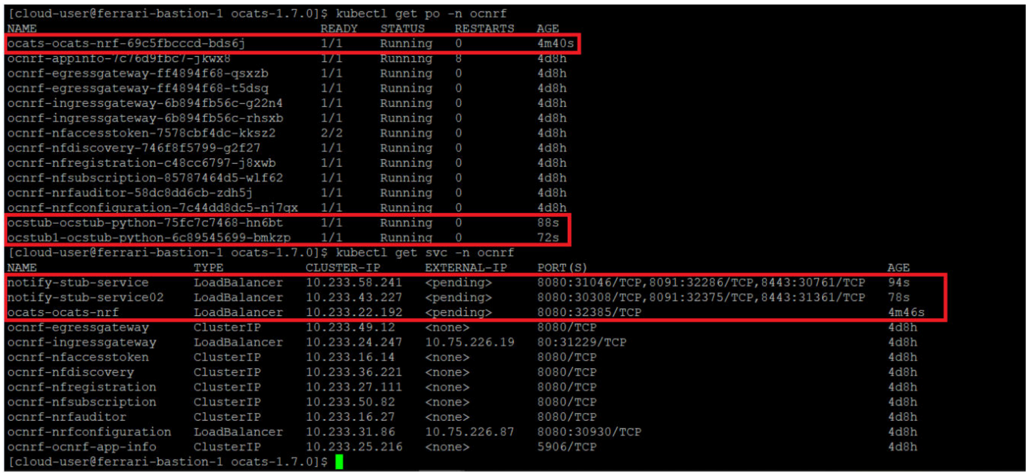
```
helm3 install -name ocstub1 ocstub-python-1.7.2.tgz --set
service.name=notify-stub-service02 --namespace ocnrf -f ocstub-python-
custom-values.yaml
```

Execute the following command to verify ATS deployment.
`helm status <release_name>`

Once ATS and Stub are deployed, execute the following command to check the pod and service deployment.

```
Checking Pod Deployment:
kubectl get pod -n ocnrf
Checking Service Deployment:
kubectl get service -n ocnrf
```

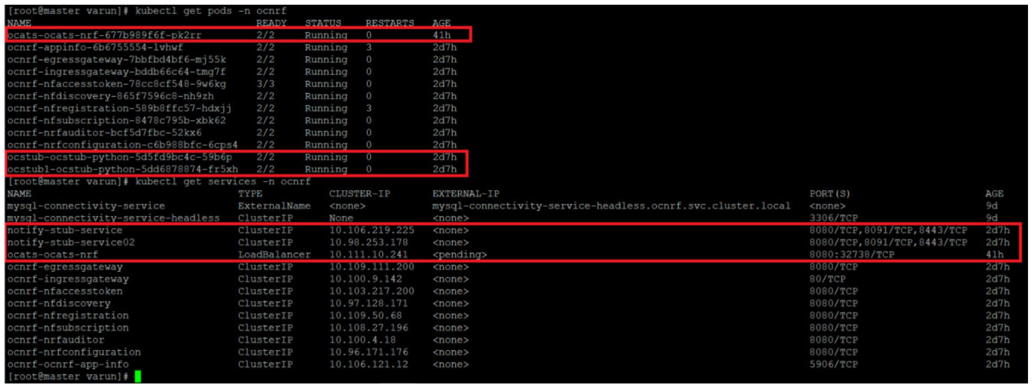
Figure 2-3 Checking Pod and Service Deployment without Service Mesh



```
[cloud-user@ferrari-bastion-1 ocats-1.7.0]$ kubectl get po -n ocnrf
NAME                                      READY   STATUS    RESTARTS   AGE
ocats-ocats-nrf-69c5fbcccd-bds6j          1/1     Running   0          4m40s
ocnrf-appinfo-7c76d9fbc7-jkwx8            1/1     Running   8          4d8h
ocnrf-egressgateway-ff4894f68-qsxzb       1/1     Running   0          4d8h
ocnrf-egressgateway-ff4894f68-t5dsq       1/1     Running   0          4d8h
ocnrf-ingressgateway-6b894fb56c-g22n4     1/1     Running   0          4d8h
ocnrf-ingressgateway-6b894fb56c-rhsxb     1/1     Running   0          4d8h
ocnrf-nfaccesstoken-7578cbf4dc-kksz2      2/2     Running   0          4d8h
ocnrf-nfdiscovery-746f8f5799-g2f27        1/1     Running   0          4d8h
ocnrf-nfregistration-c48cc6797-j8xwb      1/1     Running   0          4d8h
ocnrf-nfsubscription-85787464d5-wlf62     1/1     Running   0          4d8h
ocnrf-nrfauditor-58dc8dd6cb-zdh5j         1/1     Running   0          4d8h
ocnrf-nrfconfiguration-7c44dd8dc5-nj7qx   1/1     Running   0          4d8h
ocstub-ocstub-python-75fc7c7468-hn6bt     1/1     Running   0          88s
ocstub1-ocstub-python-6c89545699-bmkzp    1/1     Running   0          72s
[cloud-user@ferrari-bastion-1 ocats-1.7.0]$ kubectl get svc -n ocnrf
NAME                     TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)                                        AGE
notify-stub-service      LoadBalancer   10.233.58.241   <pending>      8080:31046/TCP,8091:32286/TCP,8443:30761/TCP   94s
notify-stub-service02    LoadBalancer   10.233.43.227   <pending>      8080:30308/TCP,8091:32375/TCP,8443:31361/TCP   78s
ocats-ocats-nrf          LoadBalancer   10.233.22.192   <pending>      8080:32385/TCP                                 4m46s
ocnrf-egressgateway      ClusterIP      10.233.49.12    <none>         8080/TCP                                       4d8h
ocnrf-ingressgateway     LoadBalancer   10.233.24.247   10.75.226.19   80:31229/TCP                                   4d8h
ocnrf-nfaccesstoken      ClusterIP      10.233.16.14    <none>         8080/TCP                                       4d8h
ocnrf-nfdiscovery        ClusterIP      10.233.36.221   <none>         8080/TCP                                       4d8h
ocnrf-nfregistration     ClusterIP      10.233.27.111   <none>         8080/TCP                                       4d8h
ocnrf-nfsubscription     ClusterIP      10.233.50.82    <none>         8080/TCP                                       4d8h
ocnrf-nrfauditor         ClusterIP      10.233.16.27    <none>         8080/TCP                                       4d8h
ocnrf-nrfconfiguration   LoadBalancer   10.233.31.86    10.75.226.87   8080:30930/TCP                                 4d8h
ocnrf-ocnrf-app-info     ClusterIP      10.233.25.216   <none>         5906/TCP                                       4d8h
[cloud-user@ferrari-bastion-1 ocats-1.7.0]$
```

If ATS is deployed with side car of service mesh, you need to ensure that both ATS and Stub pods have 2 containers in ready state and shows **"2/2"**. A sample screen is shown below:

Figure 2-4   ATS and Stub Deployed with Service Mesh



```
[root@master varun]# kubectl get pods -n ocnrf
NAME                                    READY   STATUS    RESTARTS   AGE
ocats-ocats-nrf-677b989f6f-pk2rr        2/2     Running   0          41h
ocnrf-appinfo-6b6755554-lvhwf           2/2     Running   3          2d7h
ocnrf-egressgateway-7bbfbd4bf6-mj55k    2/2     Running   0          2d7h
ocnrf-ingressgateway-bddb66c64-tmg7f    2/2     Running   0          2d7h
ocnrf-nfaccesstoken-78cc8cf548-9w6kg    3/3     Running   0          2d7h
ocnrf-nfdiscovery-865f7596c8-nh9zh      2/2     Running   0          2d7h
ocnrf-nfregistration-589b8ffc57-hdxjj   2/2     Running   3          2d7h
ocnrf-nfsubscription-8478c795b-xbk62    2/2     Running   0          2d7h
ocnrf-nrfauditor-bcf5d7fbc-52kx6        2/2     Running   0          2d7h
ocnrf-nrfconfiguration-c6b988bfc-6cps4  2/2     Running   0          2d7h
ocstub-ocstub-python-5d5fd9bc4c-59b6p   2/2     Running   0          2d7h
ocstub1-ocstub-python-5dd6878874-fr5xh  2/2     Running   0          2d7h
[root@master varun]# kubectl get services -n ocnrf
NAME                                 TYPE           CLUSTER-IP       EXTERNAL-IP                                            PORT(S)                       AGE
mysql-connectivity-service           ExternalName   <none>           mysql-connectivity-service-headless.ocnrf.svc.cluster.local   <none>                        9d
mysql-connectivity-service-headless  ClusterIP      None             <none>                                                 3306/TCP                      9d
notify-stub-service                  ClusterIP      10.106.219.225   <none>                                                 8080/TCP,8091/TCP,8443/TCP    2d7h
notify-stub-service02                ClusterIP      10.98.253.178    <none>                                                 8080/TCP,8091/TCP,8443/TCP    2d7h
ocats-ocats-nrf                      LoadBalancer   10.111.10.241    <pending>                                              8080:32738/TCP                41h
ocnrf-egressgateway                  ClusterIP      10.109.111.200   <none>                                                 8080/TCP                      2d7h
ocnrf-ingressgateway                 ClusterIP      10.100.9.142     <none>                                                 80/TCP                        2d7h
ocnrf-nfaccesstoken                  ClusterIP      10.103.217.200   <none>                                                 8080/TCP                      2d7h
ocnrf-nfdiscovery                    ClusterIP      10.97.128.171    <none>                                                 8080/TCP                      2d7h
ocnrf-nfregistration                 ClusterIP      10.109.50.68     <none>                                                 8080/TCP                      2d7h
ocnrf-nfsubscription                 ClusterIP      10.108.27.196    <none>                                                 8080/TCP                      2d7h
ocnrf-nrfauditor                     ClusterIP      10.100.4.18      <none>                                                 8080/TCP                      2d7h
ocnrf-nrfconfiguration               ClusterIP      10.96.171.176    <none>                                                 8080/TCP                      2d7h
ocnrf-ocnrf-app-info                 ClusterIP      10.106.121.12    <none>                                                 5906/TCP                      2d7h
[root@master varun]#
```

NSSF ATS Installation Procedure

The NSSF ATS installation procedure covers two steps:

1. Locating and downloading ATS and Simulator Images
2. Deploying ATS and Stub Pod in Kubernetes Cluster as per NSSF

Locating and Downloading ATS Images

1. Login to My Oracle Support using the appropriate credentials.
2. Select **Patches & Updates** tab.
3. In the **Patch Search** console, select **Product or Family (Advanced)** tab.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.
5. Select *Oracle Communications Cloud Native Core Network Slice Selection Function <release_number>* in **Release** field.
6. Click **Search**. The **Patch Advanced Search Results** list appears.
7. Select the required ATS patch from the list. The **Patch Details** window appears.
8. Click **Download**. The **File Download** window appears.
9. Click the **<p********_<release_number>_Tekelec>.zip** file.
10. Extract the ATS release package zip file to download the ATS images to the system where network function must be installed.
11. The `ocats-nssf` directory has the following files:
    - `ocats-nssf-tools-pkg-1.4.0.0.tgz`
    - `ocats-nssf-tools-pkg-1.4.0.0-README.txt`

> **Note:**
>
> The `ocats-nssf-tools-pkg-1.4.0.0-README.txt` file contains all the information required for the package.

12. The `ocats-nssf-tools-pkg-1.4.0.0-README.txt` file contains all the information required for the package.
13. The `ocats-nssf-tools-pkg-1.4.0.0.tgz` file has the following images and charts packaged as tar files:

```
ocats-nssf-tools-pkg-1.4.0.0.tgz
|
|_ _ _ocats-nssf-pkg-1.4.0.0.tgz
| |_ _ _ _ _ _ ocats-nssf-1.4.tgz (Helm Charts)
| |_ _ _ _ _ _ ocats-nssf-image-1.4.tar (Docker Images)
| |_ _ _ _ _ _ Readme.txt
```

14. The user can copy the tar file from here and copy in their OCCNE/OCI/Kubernetes cluster where they want to deploy ATS.

Deploying ATS in Kubernetes Cluster

The steps to deploy ATS in Kubernetes Cluster are as follows:

1. Execute the following command to extract tar file content:
   `tar -xvf ocats-nssf-tools-pkg-1.4.0.0.tgz`

   The output of this command is:

   `ocats-nssf-pkg-1.4.0.0.tgz`
2. Execute the following command to extract final helm charts and docker images of ATS:
   `tar -xvf ocats-nssf-pkg-1.4.0.0.tgz`

   The output of this command is:

   `ocats-nssf-image-1.4.tar ocats-nssf-1.4.tgz`

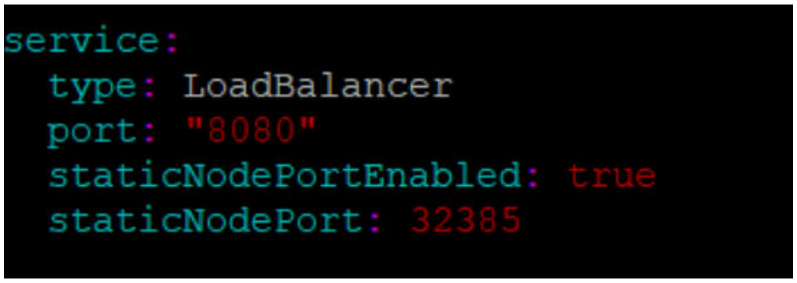
   `Readme.txt`
3. In your cluster, load the ATS image, '`ocats-nssf-image-<version>.tar`' and push to your registry.
   `docker load -i ocats-nssf-image-<version>.tar`
   a. Execute the following command to grep the image.
      `docker images | grep ocats-nssf`
   b. Copy the Image ID from the output of the grep command and change the tag to your registry.
      Example:

      ```
      docker tag <Image_ID> <your-registry-name/ocats-nssf:<tag>>
      ```

      ```
      docker push <your-registry-name/ocats-nssf:<tag>>
      ```
4. Untar the helm charts, `ocats-nssf-<version>.tgz`
   `tar -xvf ocats-nssf-<version>.tgz`
5. Update the image name and tag in the `ocats-nssf/values.yaml` file as required. For this, you need to open the `values.yaml` file and update the `image.repository` and `image.tag`.
6. ATS supports static port. By default, this feature is not available. To enable this feature:

- In the **ocats-nssf/values.yaml** file under service section, set the value of **staticNodePortEnabled** parameter as true and provide a valid nodePort value for **staticNodePort**.
- A sample screen is given below:

Figure 2-5   ocats-nssf/values.yaml - service section



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: true
  staticNodePort: 32385
```

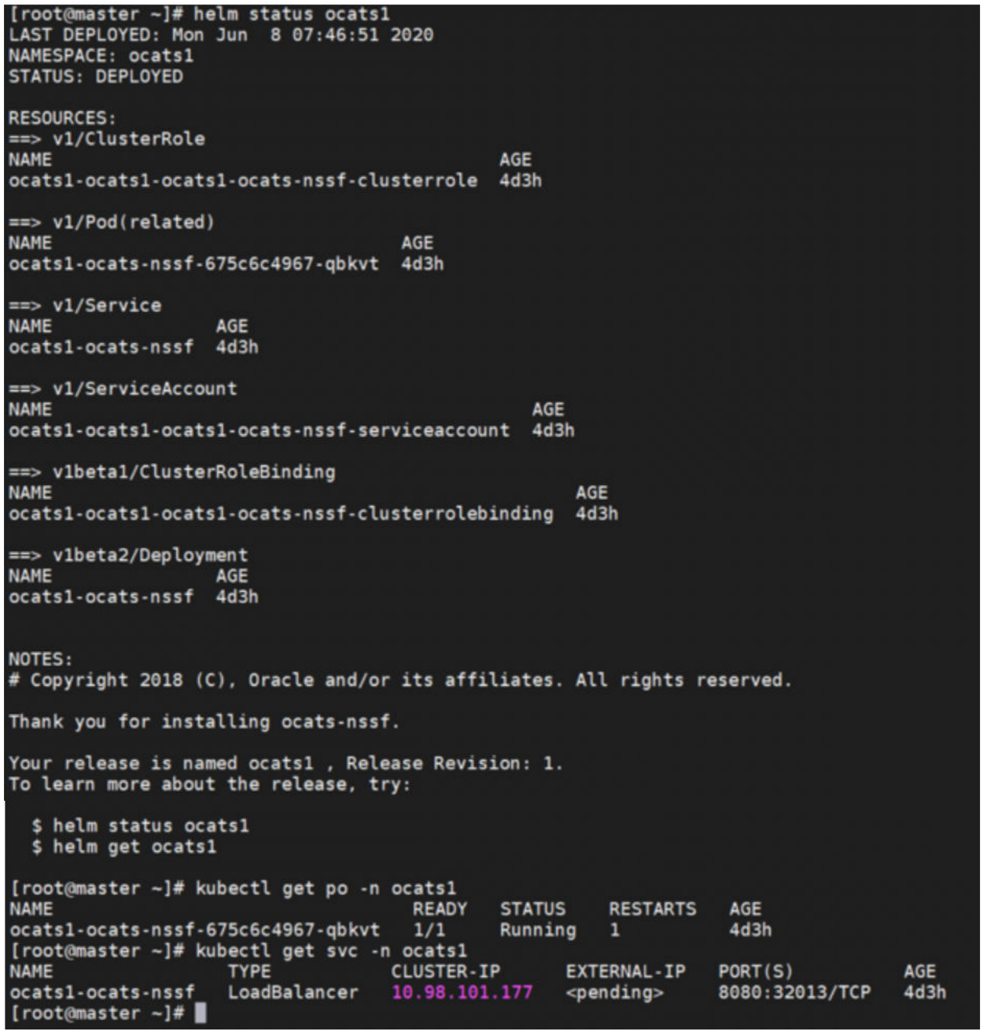
7. Deploy ATS using the updated helm charts after performing the previous **step 5**.
   `helm install ocats-nssf --name <release_name> --namespace <namespace_name> -f ocats-nssf/values.yaml`

   Example: `helm install ocats-nssf --name ocats --namespace ocnssf -f ocats-nssf/values.yaml`

   If this command returns an error like, <`Error: validation failed: unable to recognize "": no matches for kind "Deployment" in version "apps/v1beta2"`> then, open the `template/deployment.yml` file and change the apiVersoin to **apiVersion: apps/v1**.
8. Execute the following command to verify the ATS deployment:
   `helm status <release_name>`

   A sample screen showing ATS Helm release is given below:

Figure 2-6 ATS Helm Release



```
[root@master ~]# helm status ocats1
LAST DEPLOYED: Mon Jun  8 07:46:51 2020
NAMESPACE: ocats1
STATUS: DEPLOYED

RESOURCES:
==> v1/ClusterRole
NAME                                         AGE
ocats1-ocats1-ocats1-ocats-nssf-clusterrole  4d3h

==> v1/Pod(related)
NAME                                AGE
ocats1-ocats-nssf-675c6c4967-qbkvt  4d3h

==> v1/Service
NAME               AGE
ocats1-ocats-nssf  4d3h

==> v1/ServiceAccount
NAME                                            AGE
ocats1-ocats1-ocats1-ocats-nssf-serviceaccount  4d3h

==> v1beta1/ClusterRoleBinding
NAME                                                AGE
ocats1-ocats1-ocats1-ocats-nssf-clusterrolebinding  4d3h

==> v1beta2/Deployment
NAME               AGE
ocats1-ocats-nssf  4d3h


NOTES:
# Copyright 2018 (C), Oracle and/or its affiliates. All rights reserved.

Thank you for installing ocats-nssf.

Your release is named ocats1 , Release Revision: 1.
To learn more about the release, try:

  $ helm status ocats1
  $ helm get ocats1

[root@master ~]# kubectl get po -n ocats1
NAME                                READY   STATUS    RESTARTS   AGE
ocats1-ocats-nssf-675c6c4967-qbkvt  1/1     Running   1          4d3h
[root@master ~]# kubectl get svc -n ocats1
NAME               TYPE          CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
ocats1-ocats-nssf  LoadBalancer  10.98.101.177  <pending>     8080:32013/TCP   4d3h
[root@master ~]#
```

Policy ATS Installation Procedure

The Policy ATS installation procedure covers two steps:

1. Locating and downloading the ATS images.
2. Deploying ATS images.

This includes installation of stubs (nf1stub, nf2stub and nf3stub) and ATS in Policy's namespace. The release of ATS supports in-cluster deployment of Policy and ATS with both TLS (server side) enabled and disabled mode.

Note:

Restart the Nrf-client pod of Policy for UDR and CHF discovery as part of each test case.

Locating and Downloading ATS Images

To locate and download the ATS Images:

1. Login to My Oracle Support using the appropriate credentials.
2. Select **Patches & Updates** tab.
3. In the **Patch Search** console, select **Product or Family (Advanced)** tab.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.
5. Select *Oracle Communications Cloud Native Core Policy Control Function <release_number>* in **Release** field.
6. Click **Search**. The **Patch Advanced Search Results** list appears.
7. Select the required ATS patch from the list. The **Patch Details** window appears.
8. Click **Download**. The **File Download** window appears.
9. Click the **<p********_<release_number>_Tekelec>.zip** file.
10. Extract the ATS release package zip file to download the ATS images to the system where network function must be installed.
11. The `ocats-policy` directory has following file:
    - `ocats-policy-tools-1.7.4.0.0.tgz`
12. The `ocats-policy-tools-1.7.4.0.0.tgz` file has following images and charts packaged as tar files:

```
ocats-policy-tools-1.7.4.0.0.tgz

|

|_ _ _ocats-policy-pkg-1.7.4.0.0.tgz

|       |_ _ _ _ _ _ ocats-policy-1.7.4.tgz (Helm Charts)

|       |_ _ _ _ _ _ ocats-policy-images-1.7.4.tar (Docker Images)

|

|_ _ _ocstub-pkg-1.1.0.0.0.tgz

        |_ _ _ _ _ _ ocstub-go-1.1.0.tgz(Helm Charts)

        |_ _ _ _ _ _ ocstub-go-image-1.1.0.tar (Docker Images)
```

13. The user can copy the tar file from here to their Kubernetes cluster where, they want to deploy ATS.

Deploying ATS in Kubernetes Cluster

To deploy ATS in Kubernetes Cluster:

1. Execute the following command to extract the tar file content:
   `tar -xvf ocats-policy-tools-1.7.4.0.0.tgz`

The output of this command is:

```
ocats-policy-pkg-1.7.4.0.0.tgz
ocstub-pkg-1.7.4.0.0.tgz
```

2. Go to the `ocats-policy-tools-1.7.4.0.0` folder and execute the following command to extract the final helm charts and docker images of ATS.
   `tar -xvf ocats-policy-pkg-1.7.4.0.0.tgz`

   The output of this command is:

   ```
   ocats-policy-1.7.4.tgz
   ocats-policy-image-1.7.4.tar
   ```

3. In your cluster, execute the given command to load the ATS image.
   `docker load --input ocats-policy-image-1.7.4.tar`
4. Execute the following commands to tag and push the ATS images

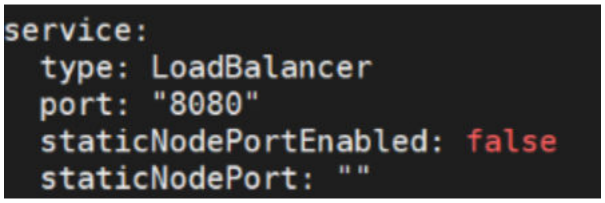
   ```
   docker tag ocatspolicy:1.7.4 <registry>/ocatspolicy:1.7.4
   docker push <registry>/ocatspolicy:1.7.4
   ```

   Example:

   ```
   docker tag ocats-policy-images:1.7.4 localhost:5000/ocats-policy-
   images:1.7.4
   docker push localhost:5000/ocats-policy-images:1.7.4
   ```

5. Untar the helm charts, `ocats-policy-1.7.4.tgz`
   `tar -xvf ocats-policy-1.7.4.tgz`
6. Update the registry name, image name and tag in the `ocats-policy/values.yaml` file as required.
   For this, you need to open the `values.yaml` file and update the `image.repository` and `image.tag`
7. ATS supports static port. By default, this feature is not available. To enable this feature:
   - In the **ocats-policy/values.yaml** file under service section, set the value of **staticNodePortEnabled** parameter as true and provide a valid nodePort value for **staticNodePort**.
   - A sample screen is given below:

   Figure 2-7 ocats-policy/values.yaml-service section

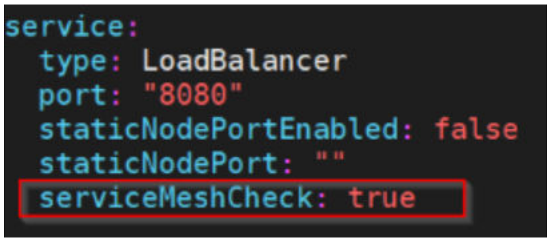
   

   ```
   service:
     type: LoadBalancer
     port: "8080"
     staticNodePortEnabled: false
     staticNodePort: ""
   ```

8. To enable service mesh feature:

a. Under the **service** section of the **values.yaml** file, there is a parameter, '**serviceMeshCheck**'. By default, this feature is set to **false**. To get ASM support, set this parameter to **true**. A snippet of service section in the yaml file is shown below:

Figure 2-8 Service Mesh Check Enabled



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: false
  staticNodePort: ""
  serviceMeshCheck: true
```

b. If you do not enable ASM at global level for the namespace, then execute the following command to enable it before deploying the ATS.
`kubectl label --overwrite namespace <namespace_name> istio-injection=enabled`

Example: `kubectl label --overwrite namespace ocpcf istio-injection=enabled`

9. Deploy ATS using the updated helm charts (refer to **step 5** for helm charts).

> **Note:**
>
> You need to ensure that all the three components ATS, Stub and PCF are deployed in the same namespace.

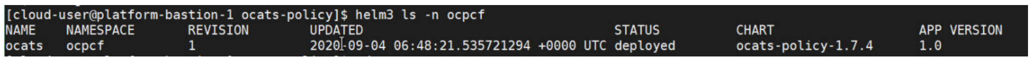
Using Helm 2 `helm install ocats-policy --name <release_name> --namespace <namespace_name> -f ocats-policy/values.yaml`

Example: `helm install ocats-policy --name ocats --namespace ocpcf -f ocats-policy/values.yaml`

Using Helm 3 `helm3 install -name <release_name> ocats-policy-1.7.4.tgz --namespace <namespace_name> -f <values-yaml-file>`
Example: `helm3 install -name ocats ocats-policy-1.7.4.tgz --namespace ocpcf -f ocats-policy/values.yaml`

10. Execute the following command to verify ATS deployment.
`helm status <release_name>`

Figure 2-9 Verifying ATS Deployment in Policy Namespace



```
[cloud-user@platform-bastion-1 ocats-policy]$ helm3 ls -n ocpcf
NAME    NAMESPACE       REVISION        UPDATED                                 STATUS          CHART                   APP VERSION
ocats   ocpcf           1               2020-09-04 06:48:21.535721294 +0000 UTC deployed        ocats-policy-1.7.4      1.0
```

Deploying Stub Pod in Kubernetes Cluster

To deploy Stub Pod in Kubernetes cluster:

1. Go to the `ocats-policy-tools-1.7.4.0.0` folder and execute the command to extract the ocstub tar file content.
   ```
   tar -xvf ocstub-pkg-1.1.0.0.0.tgz
   ```
   The output of this command is:
   ```
   ocstub-go-1.1.0.tgz
   ```
   ```
   ocstub-go-images-1.1.0.tar
   ```
2. In your cluster, execute the following command to load the STUB image and then, push it to your registry.
   ```
   docker load --input ocstub-go-image-1.1.0.tar
   ```
3. Untar the helm charts, `ocstub-go-1.1.0.tgz`.
   ```
   tar -xvf ocstub-go-1.1.0.tgz
   ```
4. Update the registry name, image name and tag (if required) in the `ocstub-go/values.yaml` file as required.
   Open the **values.yaml** file and update the `image.repository` and `image.tag`
5. If required, change the `apiVersion` to `apps/v1` in the **ocstub-go/templates/deployment.yaml** file as shown below.
   ```
   apiVersion: apps/v1
   ```
6. Deploy Stub.
   Using Helm 2: `helm install ocstub-go --set service.name=<service> --name <name> --namespace <namespace_name> -f ocstub-go/values.yaml`

   Example:
   ```
   helm install ocstub-go --set service.name=nf1stub --name nf1stub
   --namespace ocats -f ocstub-go/values.yaml

   helm install ocstub-go --set service.name=nf2stub --name nf2stub
   --namespace ocats -f ocstub-go/values.yaml

   helm install ocstub-go --set service.name=nf3stub --name nf3stub
   --namespace ocats -f ocstub-go/values.yaml
   ```

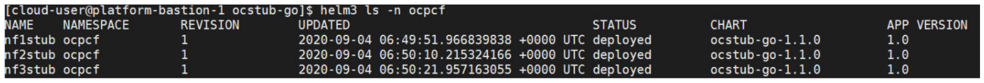
   Using Helm 3:`helm3 install -name <release_name> ocstub-go-1.1.0.tgz --set service.name=<stub-service-name> --namespace <namespace_name> -f <valuesyaml-file>`

   Example:
   ```
   helm3 install -name nf1stub ocstub-go-1.1.0.tgz --set
   service.name=nf1stub
   --namespace ocats -f ocstub-go/values.yaml

   helm3 install -name nf2stub ocstub-go-1.1.0.tgz --set
   service.name=nf2stub
   --namespace ocats -f ocstub-go/values.yaml

   helm3 install -name nf3stub ocstub-go-1.1.0.tgz --set
   service.name=nf3stub
   --namespace ocats -f ocstub-go/values.yaml
   ```

Figure 2-10 Stub - Checking Helm Status



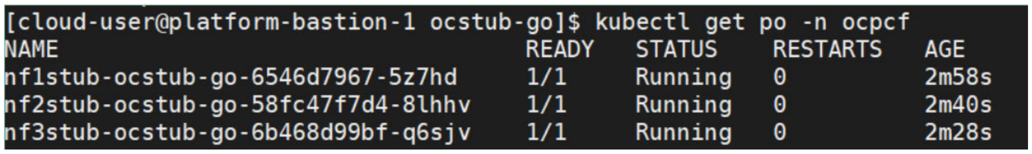
```
[cloud-user@platform-bastion-1 ocstub-go]$ helm3 ls -n ocpcf
NAME    NAMESPACE       REVISION        UPDATED                                 STATUS          CHART           APP VERSION
nf1stub ocpcf           1               2020-09-04 06:49:51.966839838 +0000 UTC deployed        ocstub-go-1.1.0 1.0
nf2stub ocpcf           1               2020-09-04 06:50:10.215324166 +0000 UTC deployed        ocstub-go-1.1.0 1.0
nf3stub ocpcf           1               2020-09-04 06:50:21.957163055 +0000 UTC deployed        ocstub-go-1.1.0 1.0
```

7. Similarly, execute the following commands to install all the stubs.
   `helm install ocstub-go --set service.name=nf2stub --name nf2stub --namespace ocats -f ocstub-go/values.yaml`

   `helm install ocstub-go --set service.name=nf3stub --name nf3stub --namespace ocats -f ocstub-go/values.yaml`
8. Execute the following command to check the Stub deployment.
   `helm status <release_name>`

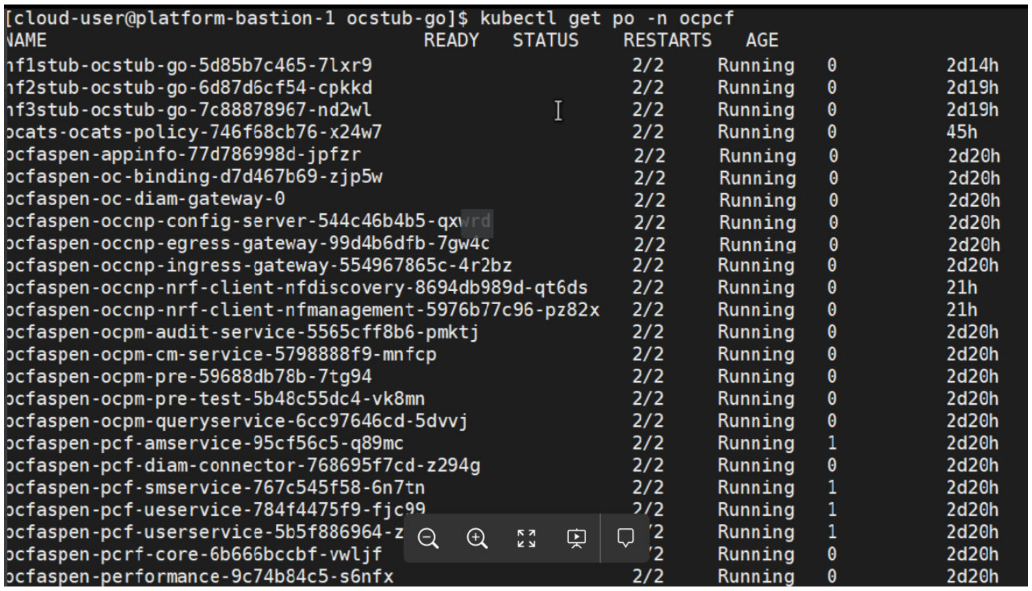
A sample screen showing stubs deployment is given below:

Figure 2-11 Stubs After Installation



```
[cloud-user@platform-bastion-1 ocstub-go]$ kubectl get po -n ocpcf
NAME                                READY   STATUS    RESTARTS   AGE
nf1stub-ocstub-go-6546d7967-5z7hd   1/1     Running   0          2m58s
nf2stub-ocstub-go-58fc47f7d4-8lhhv  1/1     Running   0          2m40s
nf3stub-ocstub-go-6b468d99bf-q6sjv  1/1     Running   0          2m28s
```

Figure 2-12 Policy Namespace



```
[cloud-user@platform-bastion-1 ocstub-go]$ kubectl get po -n ocpcf
NAME                                                  READY   STATUS    RESTARTS   AGE
nf1stub-ocstub-go-5d85b7c465-7lxr9                      2/2     Running   0          2d14h
nf2stub-ocstub-go-6d87d6cf54-cpkkd                      2/2     Running   0          2d19h
nf3stub-ocstub-go-7c88878967-nd2wl                      2/2     Running   0          2d19h
ocats-ocats-policy-746f68cb76-x24w7                     2/2     Running   0          45h
ocfaspen-appinfo-77d786998d-jpfzr                       2/2     Running   0          2d20h
ocfaspen-oc-binding-d7d467b69-zjp5w                     2/2     Running   0          2d20h
ocfaspen-oc-diam-gateway-0                              2/2     Running   0          2d20h
ocfaspen-occnp-config-server-544c46b4b5-qxwrd           2/2     Running   0          2d20h
ocfaspen-occnp-egress-gateway-99d4b6dfb-7gw4c           2/2     Running   0          2d20h
ocfaspen-occnp-ingress-gateway-554967865c-4r2bz         2/2     Running   0          2d20h
ocfaspen-occnp-nrf-client-nfdiscovery-8694db989d-qt6ds  2/2     Running   0          21h
ocfaspen-occnp-nrf-client-nfmanagement-5976b77c96-pz82x 2/2     Running   0          21h
ocfaspen-ocpm-audit-service-5565cff8b6-pmktj            2/2     Running   0          2d20h
ocfaspen-ocpm-cm-service-5798888f9-mnfcp                2/2     Running   0          2d20h
ocfaspen-ocpm-pre-59688db78b-7tg94                      2/2     Running   0          2d20h
ocfaspen-ocpm-pre-test-5b48c55dc4-vk8mn                 2/2     Running   0          2d20h
ocfaspen-ocpm-queryservice-6cc97646cd-5dvvj             2/2     Running   0          2d20h
ocfaspen-pcf-amservice-95cf56c5-q89mc                   2/2     Running   1          2d20h
ocfaspen-pcf-diam-connector-768695f7cd-z294g            2/2     Running   0          2d20h
ocfaspen-pcf-smservice-767c545f58-6n7tn                 2/2     Running   1          2d20h
ocfaspen-pcf-ueservice-784f4475f9-fjc99                 2/2     Running   1          2d20h
ocfaspen-pcf-userservice-5b5f886964-z                   2       Running   1          2d20h
ocfaspen-pcrf-core-6b666bccbf-vwljf                     2       Running   0          2d20h
ocfaspen-performance-9c74b84c5-s6nfx                    2/2     Running   0          2d20h
```

SCP ATS Installation Procedure

The SCP ATS installation procedure covers two steps:

1. Locating and downloading the ATS images.
2. Deploying ATS images.

Locating and Downloading ATS Images

The steps to locate and download ATS Images are as follows:

1. Login to My Oracle Support using the appropriate credentials.
2. Select **Patches & Updates** tab.
3. In the **Patch Search** console, select **Product or Family (Advanced)** tab.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.
5. Select *Oracle Communications Cloud Native Core Service Communication Proxy <release_number>* in **Release** field.
6. Click **Search**. The **Patch Advanced Search Results** list appears.
7. Select the required ATS patch from the list. The **Patch Details** window appears.
8. Click **Download**. The **File Download** window appears.
9. Click the **<p********_<release_number>_Tekelec>.zip** file.
10. Extract the ATS release package zip file to download the ATS images to the system where network function must be installed.
11. The `ocats-scp` directory has following files:

    ```
    ocats-scp-pkg-1.7.3.0.0.tgz
    ocats-scp-pkg-1.7.3.0.0-README.txt
    ocats-scp-custom-configtemplates-1.7.3.0.0.zip
    ocats-scp-custom-configtemplates-1.7.3.0.0-readme.txt
    ```

    Note:

    The `ocats-scp-pkg-1.7.3.0.0-README.txt` file contains all the information required for the package.

    The `ocats-scp-pkg-1.7.3.0.0.tgz` file has following images and charts packaged as tar files:

    ```
    ocats-scp-pkg-1.7.3.0.0.tgz

          |

          |_ _ _ocats-scp-pkg-1.7.3.0.0.tgz

          |        |_ _ _ _ _ _ ocats-scp-1.7.3.tgz (Helm Charts)

          |        |_ _ _ _ _ _ ocats-scp-images-1.7.3.tar (Docker
    Images)

          |        |_ _ _ _ _ _ Readme-internal.txt
    ```

The `ocats-scp-custom-configtemplates-1.7.3.0.0.zip` file has following images and charts packaged as tar files:

```
ocats-scp-custom-configtemplates-1.7.3.0.0.zip

        |_ _ _ _ _ _ _ ocats-scp-custom-serviceaccount-1.7.3.yaml
(Template to create custom service account)

        |_ _ _ _ _ _ _ ocats_scp_values_1.7.3.yaml (Custom values
file for installation)
```

The user can copy the tar file from here to their kubernetes cluster where, they want to deploy ATS.

Deploying ATS in Kuberbetes Cluster

The steps to deploy ATS in Kubernetes Cluster are as follows:

> **Note:**
>
> Deploy ATS and SCP in the same namespace.

> **Note:**
>
> ATS is deployed with role binding by default instead of cluster role binding.

1. Execute the following command to extract the tar file content.
   `tar -xvf ocats-scp-pkg-1.7.3.0.0.tgz`

   The output of this command is:

   ```
   ocats-scp-1.7.3.tgz
   ocats-scp-images-1.7.3.tar
   Readme-internal.txt
   ```

   The `ocats-scp-images-1.7.3.tar` file contains ocats-scp:1.7.3 (ATS Image) and ocats-gostub:1.7.3 (stub image).
2. In your cluster, execute the given command to load the ATS image and then, push it to your registry.
   `docker load --input ocats-scp-images-1.7.3.tar`
3. Execute the following command to extract the zip file content.

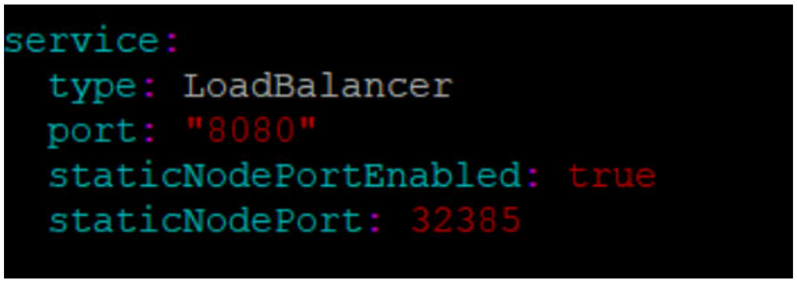
   ```
   Unzip "ocats-scp-custom-configtemplates-1.7.3.0.0.zip"
   ```

   The output of this command is:

   ```
   ocats_scp_values_1.7.3.yaml
   ocats-scp-custom-serviceaccount-1.7.3.yaml
   ```

4. Update the image name and tag in the **ocats_scp_values_1.7.3.yaml** file as required.
   For this, you need to open the `ocats_scp_values_1.7.3.yaml` file and update the `image.repository` and `image.tag`
5. ATS supports static port. By default, this feature is not available. To enable this feature:
   - In the **ocats_scp_values_1.7.3.yaml** file under service section, set the value of **staticNodePortEnabled** parameter as true and provide a valid nodePort value for **staticNodePort**.
   - A sample screen is given below:

   Figure 2-13 ocats_scp_values_1.7.3.yaml - service section



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: true
  staticNodePort: 32385
```

> **Note:**
>
> You can enable static node port at the time of deployment.

6. The **lbDeployments** section of the helm deployment file in SCP ATS is updated with following annotations, wherein
   - 8080 port is added to open ATS Jenkins GUI
     `traffic.sidecar.istio.io/excludeInboundPorts: "8080"`
   - 8091 port is added to fetch soothsayer pod metrics
     `traffic.sidecar.istio.io/excludeOutboundPorts: "8091"`

> **Note:**
>
> **THE ATS USER SHOULD NOT MODIFY THESE PORTS.**

7. Execute the following command to deploy ATS.
   Using Helm 2: `helm2 install ocats-scp-1.7.3.tgz --name <release_name> --namespace <namespace_name> -f ocats_scp_values_1.7.3.yaml`

   Example: `helm2 install ocats-scp-1.7.3.tgz --name ocats-scp --namespace scpsvc-f ocats_scp_values_1.7.3.yaml`

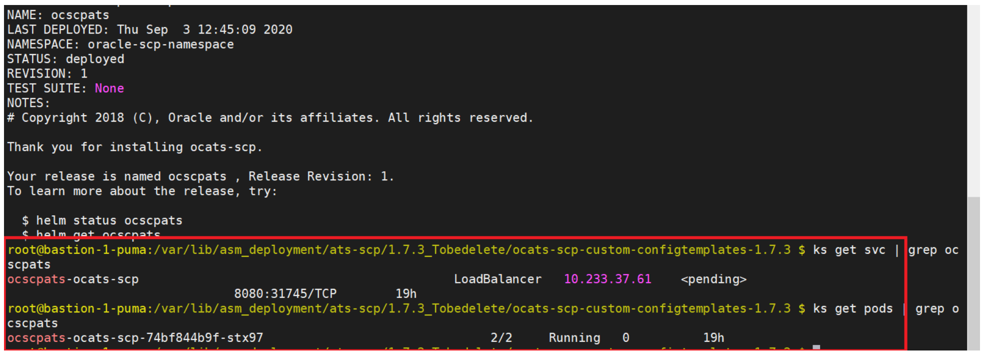
   Using Helm 3: `helm3 install <release_name> ocats-scp-1.7.3.tgz -n <namespace_name> -f ocats_scp_values_1.7.3.yaml`

   Example: `helm3 install ocscp-ats ocats-scp-1.7.3.tgz -n scpsvc -f ocats_scp_values_1.7.3.yaml`

> **Note:**
>
> If you have two Helm versions on your system then, you have to specify version number in the Helm commands you are executing. If there is only one Helm version then there is no need to mention version number.

8. Verify ATS deployment by executing the given command.
   `helm3 status <release_name> -n <namespace_name>`

Figure 2-14 Helm Status Image



```
NAME: ocscpats
LAST DEPLOYED: Thu Sep  3 12:45:09 2020
NAMESPACE: oracle-scp-namespace
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
# Copyright 2018 (C), Oracle and/or its affiliates. All rights reserved.

Thank you for installing ocats-scp.

Your release is named ocscpats , Release Revision: 1.
To learn more about the release, try:

  $ helm status ocscpats
  $ helm get ocscpats
root@bastion-1-puma:/var/lib/asm_deployment/ats-scp/1.7.3_Tobedelete/ocats-scp-custom-configtemplates-1.7.3 $ ks get svc | grep oc
scpats
ocscpats-ocats-scp                                          LoadBalancer   10.233.37.61    <pending>
                              8080:31745/TCP        19h
root@bastion-1-puma:/var/lib/asm_deployment/ats-scp/1.7.3_Tobedelete/ocats-scp-custom-configtemplates-1.7.3 $ ks get pods | grep o
cscpats
ocscpats-ocats-scp-74bf844b9f-stx97                            2/2     Running   0          19h
```

SLF ATS Installation Procedure

The SLF ATS installation procedure covers two steps:

1. Locating and downloading the ATS images.
2. Deploying ATS images.

Locating and Downloading ATS Images

The steps to locate and download ATS Images are as follows:

1. Login to My Oracle Support using the appropriate credentials.
2. Select **Patches & Updates** tab.
3. In the **Patch Search** console, select **Product or Family (Advanced)** tab.
4. Enter *Oracle Communications Cloud Native Core - 5G* in **Product** field and select the product from the Product drop-down.
5. Select *Oracle Communications Cloud Native Core Unified Data Repository <release_number>* in **Release** field.
6. Click **Search**. The **Patch Advanced Search Results** list appears.
7. Select the required ATS patch from the list. The **Patch Details** window appears.
8. Click **Download**. The **File Download** window appears.
9. Click the **<p********_<release_number>_Tekelec>.zip** file.
10. Extract the ATS release package zip file to download the ATS images to the system where network function must be installed.

11. The `ocats_slf` directory has following files:

```
ocats-slf-pkg-1.7.1.0.0.tgz

      |_ _ _ _ _ _ ocats-slf-1.7.1.tgz (Helm Charts)

     |_ _ _ _ _ _ ocats-slf-images-1.7.1.tar (Docker Images)
```

    The user can copy the tar file from here to their kubernetes cluster where they want to deploy ATS.

Preparing to Deploy ATS in Kuberbetes Cluster

The steps to deploy ATS in Kubernetes Cluster are as follows:

Note:

Deploy ATS and SLF in the same namespace.

1. Execute the following command to extract the tar file content.
   `tar -xvf ocats-slf-pkg-1.7.1.0.0.tgz`

   The output of this command is:

   ```
   ocats-slf-1.7.1.tgz
   ocats-slf-images-1.7.1.tar
   ```

   The `ocats-slf-images-1.7.1.tar` file contains ocats-slf-images-1.7.1 (ATS Image).
2. In your cluster, execute the given command to load the ATS image.
   `docker load --input ocats-slf-images-1.7.1.tar`
3. Execute the following command to tag and push the ATS image to your registry.

   ```
   docker tag ocats-slf-images-1.7.1:1.7.1 <registry>/ocats-slf-
   images-1.7.1:1.7.1
   docker push <registry>/ocats-slf-images-1.7.1:1.7.1
   ```

   Example:

   ```
   docker tag ocats-slf-images-1.7.1:1.7.1 localhost:5000/ocats-slf-
   images-1.7.1:1.7.1
   docker push localhost:5000/ocats-slf-images-1.7.1:1.7.1
   ```

4. Execute the following command to untar the helm charts (ocats-slf-1.7.1.tgz) and update the registry name, image name and tag (if required) in the **ocats-slf/values.yaml** file.

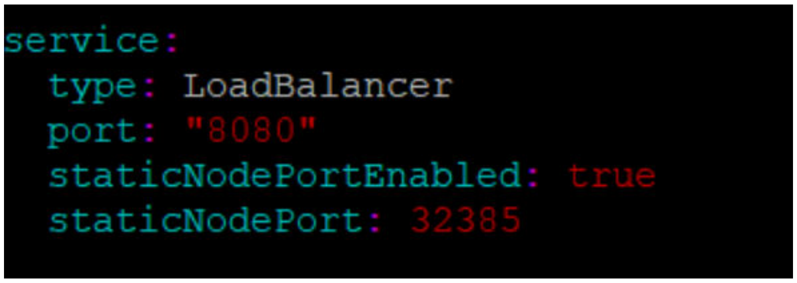
   ```
   tar -xvf ocats-slf-1.7.1.tgz
   ```

The list of content in ocats-slf is:

```
ocats-slf
├── Chart.yaml
├── destination-rule-ats.yaml
├── policy.yaml
├── templates
│   ├── deployment.yaml
│   ├── _helpers.tpl
│   ├── ingress.yaml
│   ├── NOTES.txt
│   ├── serviceaccount.yaml
│   └── service.yaml
└── values.yaml
```

5. ATS supports static port. By default, this feature is not available. To enable this feature:
   - In the **ocats-slf/values.yaml** file under service section, add the **staticNodePortEnabled** parameter as true and **staticNodePort** parameter with valid nodePort value. A sample screen is given below:

Figure 2-15 ocats-slf/values.yaml - service section



```
service:
  type: LoadBalancer
  port: "8080"
  staticNodePortEnabled: true
  staticNodePort: 32385
```

Enabling Service Mesh at Namespace Level

Note:

This section is applicable only if you are planning to deploy ATS on service mesh enabled system.

Execute the following command to enable service mesh at the namespace level.

```
kubectl label --overwrite namespace <namespace_name> istio-
injection=enabled
```

Example:

```
kubectl label --overwrite namespace ocudr istio-injection=enabled
```

Deploying ATS Pod in Kubernetes Cluster

You can deploy ATS Pod in Kubernetes cluster using Helm 2 or Helm 3 commands.

Using Helm 2

Execute the following command to deploy ATS.

```
helm install --name <release_name> --namespace <namespace_name> -f
<values-yaml-file> ocats-slf
```

Example: `helm install --name ocats-slf --namespace ocudr -f ocats-slf/values.yaml ocats-slf`

Using Helm 3

Execute the following command to deploy ATS.

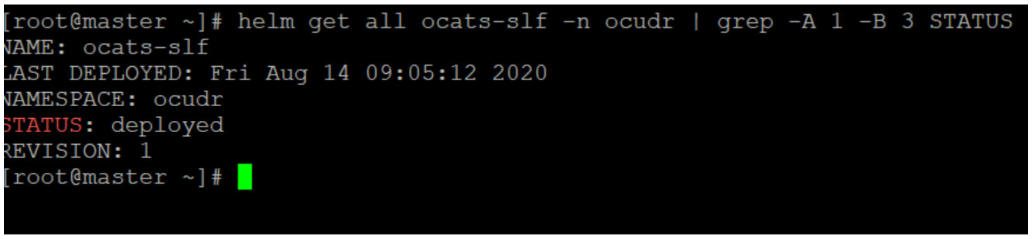
```
helm3 install -name <release_name> --namespace <namespace_name> -f
<values-yaml-file> ocats-slf
```

Example: `helm3 install -name ocats-slf --namespace ocudr -f ocats-slf/values.yaml ocats-slf`

To verify ATS deployment, execute the following command:

```
helm status <release_name>
```

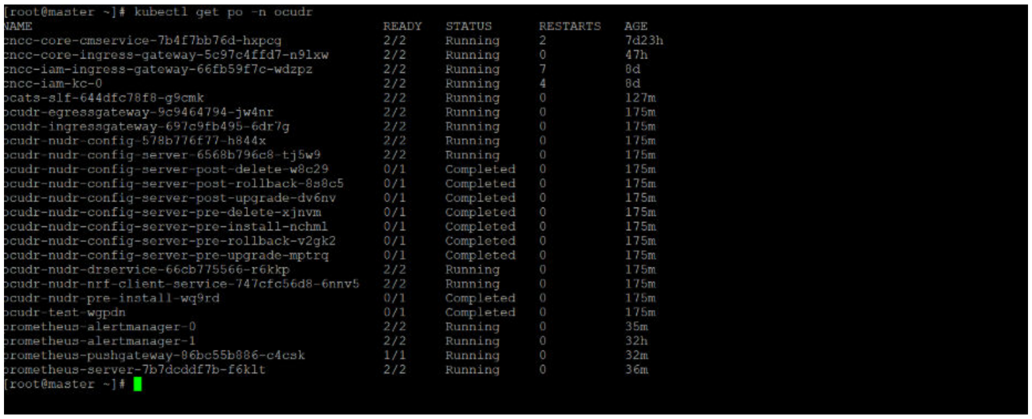
Figure 2-16 Verifying ATS Deployment



```
[root@master ~]# helm get all ocats-slf -n ocudr | grep -A 1 -B 3 STATUS
NAME: ocats-slf
LAST DEPLOYED: Fri Aug 14 09:05:12 2020
NAMESPACE: ocudr
STATUS: deployed
REVISION: 1
[root@master ~]#
```

Below is a sample screen showing UDR and ATS installed in the SLF namespace:

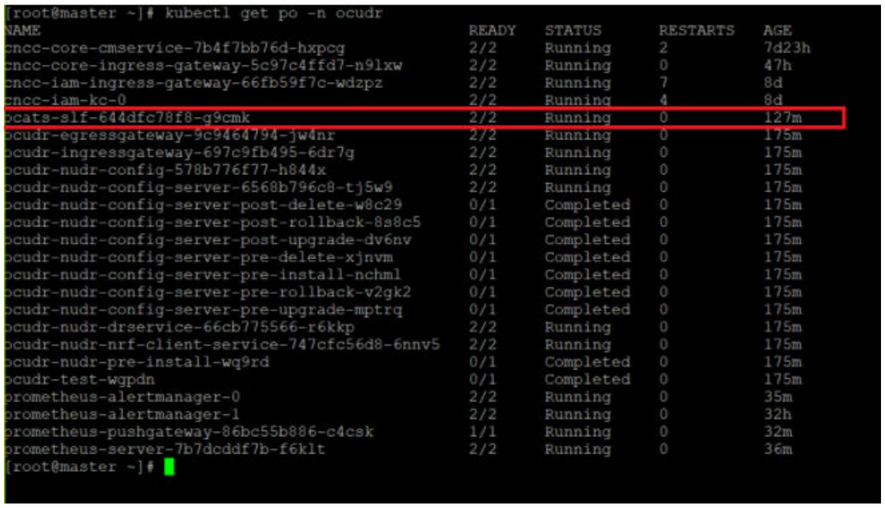
Figure 2-17 ATS and SLF Deployed in Same Namespace



```
[root@master ~]# kubectl get po -n ocudr
NAME                                                READY   STATUS      RESTARTS   AGE
cncc-core-cmservice-7b4f7bb76d-hxpcq                2/2     Running     2          7d23h
cncc-core-ingress-gateway-5c97c4ffd7-n9lxw          2/2     Running     0          47h
cncc-iam-ingress-gateway-66fb59f7c-wdzpz            2/2     Running     7          8d
cncc-iam-kc-0                                       2/2     Running     4          8d
ocats-slf-644dfc78f8-g9cmk                          2/2     Running     0          127m
ocudr-egressgateway-9c9464794-jw4nr                 2/2     Running     0          175m
ocudr-ingressgateway-697c9fb495-6dr7g               2/2     Running     0          175m
ocudr-nudr-config-578b776f77-h844x                  2/2     Running     0          175m
ocudr-nudr-config-server-6568b796c8-tj5w9           2/2     Running     0          175m
ocudr-nudr-config-server-post-delete-w8c29          0/1     Completed   0          175m
ocudr-nudr-config-server-post-rollback-8s8c5        0/1     Completed   0          175m
ocudr-nudr-config-server-post-upgrade-dv6nv         0/1     Completed   0          175m
ocudr-nudr-config-server-pre-delete-xjnvm           0/1     Completed   0          175m
ocudr-nudr-config-server-pre-install-nchml          0/1     Completed   0          175m
ocudr-nudr-config-server-pre-rollback-v2gk2         0/1     Completed   0          175m
ocudr-nudr-config-server-pre-upgrade-mptrq          0/1     Completed   0          175m
ocudr-nudr-drservice-66cb775566-r6kkp               2/2     Running     0          175m
ocudr-nudr-nrf-client-service-747cfc56d8-6nnv5      2/2     Running     0          175m
ocudr-nudr-pre-install-wq9rd                        0/1     Completed   0          175m
ocudr-test-wgpdn                                    0/1     Completed   0          175m
prometheus-alertmanager-0                           2/2     Running     0          35m
prometheus-alertmanager-1                           2/2     Running     0          32h
prometheus-pushgateway-86bc55b886-c4csk             1/1     Running     0          32m
prometheus-server-7b7dcddf7b-f6klt                  2/2     Running     0          36m
[root@master ~]#
```

If the ATS deployment is done with side car of service mesh, you need to ensure that the ATS shows 2 containers in ready state as "2/2". A sample output of the command is given below:

Figure 2-18 ATS Deployed with Side Car of Service Mesh



```
[root@master ~]# kubectl get po -n ocudr
NAME                                               READY   STATUS      RESTARTS   AGE
cncc-core-cmservice-7b4f7bb76d-hxpcg               2/2     Running     2          7d23h
cncc-core-ingress-gateway-5c97c4ffd7-n9lxw         2/2     Running     0          47h
cncc-iam-ingress-gateway-66fb59f7c-wdzpz           2/2     Running     7          8d
cncc-iam-kc-0                                      2/2     Running     4          8d
ocats-slf-644dfc78f8-g9cmk                         2/2     Running     0          127m
ocudr-egressgateway-9c9464794-jw4nr                2/2     Running     0          175m
ocudr-ingressgateway-697c9fb495-6dr7g              2/2     Running     0          175m
ocudr-nudr-config-578b776f77-h844x                 2/2     Running     0          175m
ocudr-nudr-config-server-6568b796c8-tj5w9          2/2     Running     0          175m
ocudr-nudr-config-server-post-delete-w8c29         0/1     Completed   0          175m
ocudr-nudr-config-server-post-rollback-8s8c5       0/1     Completed   0          175m
ocudr-nudr-config-server-post-upgrade-dv6nv        0/1     Completed   0          175m
ocudr-nudr-config-server-pre-delete-xjnvm          0/1     Completed   0          175m
ocudr-nudr-config-server-pre-install-nchml         0/1     Completed   0          175m
ocudr-nudr-config-server-pre-rollback-v2gk2        0/1     Completed   0          175m
ocudr-nudr-config-server-pre-upgrade-mptrq         0/1     Completed   0          175m
ocudr-nudr-drservice-66cb775566-r6kkp              2/2     Running     0          175m
ocudr-nudr-nrf-client-service-747cfc56d8-6nnv5     2/2     Running     0          175m
ocudr-nudr-pre-install-wq9rd                       0/1     Completed   0          175m
ocudr-test-wgpdn                                   0/1     Completed   0          175m
prometheus-alertmanager-0                          2/2     Running     0          35m
prometheus-alertmanager-1                          2/2     Running     0          32h
prometheus-pushgateway-86bc55b886-c4csk            1/1     Running     0          32m
prometheus-server-7b7dcddf7b-f6klt                 2/2     Running     0          36m
[root@master ~]#
```

Creating a Policy and Destination Rule

Following steps to create a policy are applicable only if a service mesh is enabled at the namespace level:

1. Edit the policy.yaml file as follows:
   - Change the **spec.targets.name** to **ocats-slf svc** name.
   - Change the namespace in which ocats-slf is deployed.

   The policy.yaml file snippet is given below:

   ```
   apiVersion: "authentication.istio.io/v1alpha1"
   kind: Policy
   metadata:
     name: ocats-slf
     namespace: myudr
   spec:
     targets:
     - name: ocats-slf
     peers:
     - mtls:
         mode: PERMISSIVE
   ```

2. Execute the following command to create a policy:
   `kubectl create -f policy.yaml`

   Output: `policy.authentication.istio.io/ocats-slf` is created.

Following steps to create a destination rule are applicable only if a service mesh is enabled at the namespace level:

1. If Service Mesh check is enabled, you need to create a destination rule to fetch the metrics from the Prometheus. This is so because in most of the deployments, Prometheus is kept outside of the service mesh and a destination

rule is required to communicate between TLS enabled entity (ATS) and non-TLS entity (Prometheus). To create a destination rule:

```
kubectl apply -f - <<EOF
apiVersion:networking.istio.io/v1alpha3
kind:DestinationRule
metadata:
  name:prometheus-dr
  namespace:myudr
spec:
  host:oso-prometheus-server.myudr.svc.cluster.local
  trafficPolicy:
    tls:
      mode:DISABLE
EOF
```

In the above rule,

- **name** indicates the name of destination rule.
- **namespace** indicates where the ATS (ocats-slf) is deployed.
- **host** indicates the hostname of the prometheus server. Change the **spec.host** value to **fqdn** of Prometheus server.

2. Execute the following command to create a policy:
`kubectl create -f destination-rule-ats.yaml`

   Output: `destinationrule.networking.istio.io/ocats-slf-dr` is created.

3

Executing NF Test Cases using ATS

In this chapter, you will learn to execute NF (NRF, PCF and SCP) Test Cases using ATS.

Executing NRF Test Cases using ATS

Prerequisite

To execute NRF Test Cases using NRF ATS 1.7.2, you need to ensure that following prerequisites are fulfilled.

- To execute Geo-Redundancy test cases, you need to deploy two NRF-1.7.2 with replication enabled. These test cases are executed separately as it requires two different NRFs.
- The user should create certificates/keys (public and private) for AccessToken micro-service before deploying NRF.
- Deploy NRF 1.7.2 with default helm configurations using helm charts.
- All micro-services of NRF should be up and running including Accesstoken micro-service.
- Deploy ATS using helm charts.
- The user **MUST** copy the public keys (RSA and ECDSA) created in the above step to the ATS pod at the **/var/lib/jenkins/ocnrf_tests/public_keys** location.
- Deploy Stub using helm charts.
- For NRF ATS 1.7.2, you need to deploy two stub servers for executing SLF and Forwarding functionality test cases. The service name for both the STUB servers should be **notify-stub-service** and **notify-stub-service02**.
- Ensure Prometheus service is up and running.
- Deploy ATS and Stubs in the same namespace as of OCNRF as default ATS deployment is with role binding.In addition, test stubs must be deployed in same namespace as NRF.
- User **MUST** not initiate a job in two different pipelines at the same time.
- If Service Mesh check is enabled, you need to create a destination rule to fetch the metrics from the Prometheus. This is so because in most of the deployments, Prometheus is kept outside of the service mesh and a destination rule is required to communicate between TLS enabled entity (ATS) and non-TLS entity (Prometheus). To create a rule:

```
kubectl apply -f - <<EOF
apiVersion:networking.istio.io/v1alpha3
kind:DestinationRule
metadata:
  name:prometheus-dr
  namespace:ocnrf
spec:
  host:oso-prometheus-server.ocnrf.svc.cluster.local
  trafficPolicy:
    tls:
      mode:DISABLE
EOF
```

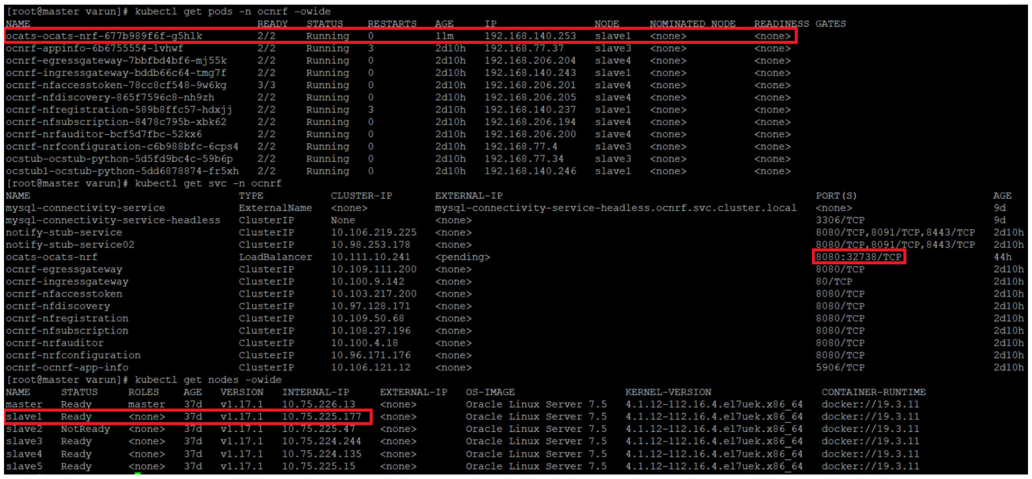
In the above rule,

- **name** indicates the name of destination rule.
- **namespace** indicates where the ATS is deployed.
- **host** indicates the hostname of the prometheus server.

Logging into ATS

Before logging into ATS, you need to ensure that ATS is deployed successfully using HELM charts. A sample screen is given below:

Figure 3-1 Verifying ATS Pod



```
[root@master varun]# kubectl get pods -n ocnrf -owide
NAME                                      READY   STATUS    RESTARTS   AGE     IP                NODE     NOMINATED NODE   READINESS GATES
ocats-ocats-nrf-677b989f6f-g5hlk          2/2     Running   0          11m     192.168.140.253   slave1   <none>           <none>
ocnrf-appinfo-6b6755554-lvhwf             2/2     Running   3          2d10h   192.168.77.37     slave3   <none>           <none>
ocnrf-egressgateway-7bbfbd4bf6-mj55k      2/2     Running   0          2d10h   192.168.206.204   slave4   <none>           <none>
ocnrf-ingressgateway-bddb66c64-tmg7f      2/2     Running   0          2d10h   192.168.140.243   slave1   <none>           <none>
ocnrf-nfaccesstoken-78cc8cf548-9w6kg      3/3     Running   0          2d10h   192.168.206.201   slave4   <none>           <none>
ocnrf-nfdiscovery-865f7596c8-nh9zh        2/2     Running   0          2d10h   192.168.206.205   slave4   <none>           <none>
ocnrf-nfregistration-589b8ffc57-hdxjj     2/2     Running   3          2d10h   192.168.140.237   slave1   <none>           <none>
ocnrf-nfsubscription-8478c795b-xbk62      2/2     Running   0          2d10h   192.168.206.194   slave4   <none>           <none>
ocnrf-nrfauditor-bcf5d7fbc-52kx6          2/2     Running   0          2d10h   192.168.206.200   slave4   <none>           <none>
ocnrf-nrfconfiguration-c6b988bfc-6cps4    2/2     Running   0          2d10h   192.168.77.4      slave3   <none>           <none>
ocstub-ocstub-python-5d5fd9bc4c-59b6p     2/2     Running   0          2d10h   192.168.77.34     slave3   <none>           <none>
ocstub1-ocstub-python-5dd6878874-fr5xh    2/2     Running   0          2d10h   192.168.140.246   slave1   <none>           <none>
[root@master varun]# kubectl get svc -n ocnrf
NAME                                   TYPE           CLUSTER-IP       EXTERNAL-IP                                                PORT(S)                           AGE
mysql-connectivity-service             ExternalName   <none>           mysql-connectivity-service-headless.ocnrf.svc.cluster.local   <none>                            9d
mysql-connectivity-service-headless    ClusterIP      None             <none>                                                     3306/TCP                          9d
notify-stub-service                    ClusterIP      10.106.219.225   <none>                                                     8080/TCP,8091/TCP,8443/TCP        2d10h
notify-stub-service02                  ClusterIP      10.98.253.178    <none>                                                     8080/TCP,8091/TCP,8443/TCP        2d10h
ocats-ocats-nrf                        LoadBalancer   10.111.10.241    <pending>                                                  8080:32738/TCP                    44h
ocnrf-egressgateway                    ClusterIP      10.109.111.200   <none>                                                     8080/TCP                          2d10h
ocnrf-ingressgateway                   ClusterIP      10.100.9.142     <none>                                                     80/TCP                            2d10h
ocnrf-nfaccesstoken                    ClusterIP      10.103.217.200   <none>                                                     8080/TCP                          2d10h
ocnrf-nfdiscovery                      ClusterIP      10.97.128.171    <none>                                                     8080/TCP                          2d10h
ocnrf-nfregistration                   ClusterIP      10.109.50.68     <none>                                                     8080/TCP                          2d10h
ocnrf-nfsubscription                   ClusterIP      10.108.27.196    <none>                                                     8080/TCP                          2d10h
ocnrf-nrfauditor                       ClusterIP      10.100.4.18      <none>                                                     8080/TCP                          2d10h
ocnrf-nrfconfiguration                 ClusterIP      10.96.171.176    <none>                                                     8080/TCP                          2d10h
ocnrf-ocnrf-app-info                   ClusterIP      10.106.121.12    <none>                                                     5906/TCP                          2d10h
[root@master varun]# kubectl get nodes -owide
NAME     STATUS     ROLES    AGE   VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE                  KERNEL-VERSION                   CONTAINER-RUNTIME
master   Ready      master   37d   v1.17.1   10.75.226.13    <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
slave1   Ready      <none>   37d   v1.17.1   10.75.225.177   <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
slave2   NotReady   <none>   37d   v1.17.1   10.75.225.47    <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
slave3   Ready      <none>   37d   v1.17.1   10.75.224.244   <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
slave4   Ready      <none>   37d   v1.17.1   10.75.224.135   <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
slave5   Ready      <none>   37d   v1.17.1   10.75.225.15    <none>        Oracle Linux Server 7.5   4.1.12-112.16.4.el7uek.x86_64   docker://19.3.11
```

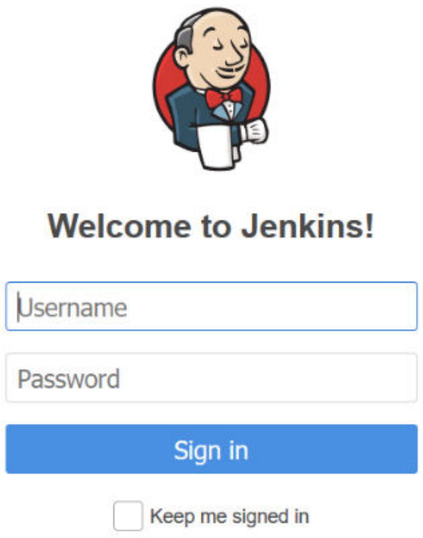
You can use the external IP of the worker node and nodeport of the ATS service as `<Worker-Node-IP>:<Node-Port-of-ATS>`

Note:

In the **Verifying ATS Pod** screen, **slave1** is the node where ATS is deployed, **32728** is the ATS nodeport and **10.75.225.177** is the worker node IP, highlighed in red. For more details on ATS deployment, refer to NRF ATS Installation Procedure.

To login to ATS, open a browser and provide the IP Address and port details as <Worker-Node-IP>:<Node-Port-of-ATS>. As per above screen, it is **10.75.225.177:32728**. The following screen appears:

Figure 3-2 ATS Login

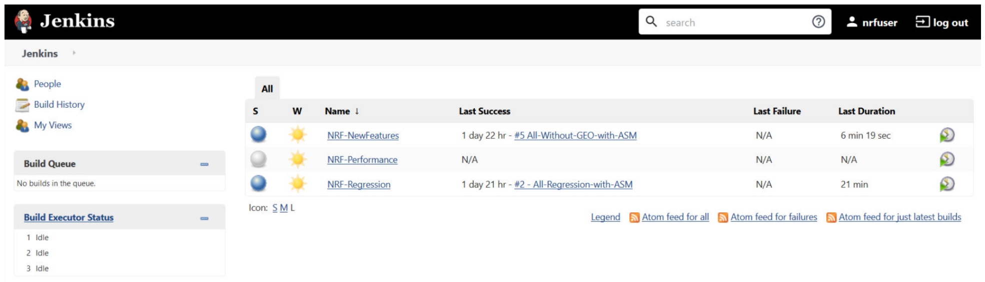


Executing ATS

To execute ATS:

1. Enter the **username** as 'nrfuser' and **password** as 'nrfpasswd'. Click **Sign in**. The following screen appears.

   Figure 3-3 NRF Pre-Configured Pipelines

   No builds in the queue.

   1 Idle
   2 Idle
   3 Idle

S	W	Name ↓	Last Success	Last Failure	Last Duration
●	☀	NRF-NewFeatures	1 day 22 hr - #5 All-Without-GEO-with-ASM	N/A	6 min 19 sec
○	☀	NRF-Performance	N/A	N/A	N/A
●	☀	NRF-Regression	1 day 21 hr - #2 - All-Regression-with-ASM	N/A	21 min

   Icon: S M L

   Legend Atom feed for all Atom feed for failures Atom feed for just latest builds

   NRF ATS has three pre-configured pipelines.

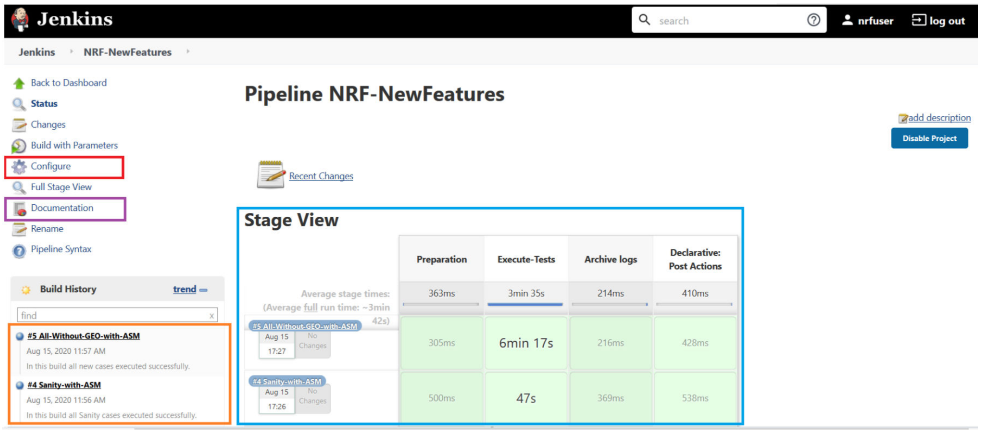
   - **NRF-NewFeatures:** This pipeline has all the test cases, which are delivered as part of NRF ATS - 1.7.0
   - **NRF-Performance:** This pipeline is not operational as of now. It is reserved for future releases of ATS.
   - **NRF-Regression:** This pipleine has all the test cases, which were delivered in NRF ATS - 1.6.1

NRF-NewFeatures Pipeline

After identifying the NRF pipelines, the user needs to do one-time configuration in ATS as per NRF deployment. In this pipeline, all the new testcases related to NRF are executed. To configure its parameters:

1. Click **NRF-NewFeatures** in the Name column. Following screen appears:

Figure 3-4   Configuring NRF-New Features



#5 All-Without-GEO-with-ASM
Aug 15, 2020 11:57 AM
In this build all new cases executed successfully.

#4 Sanity-with-ASM
Aug 15, 2020 11:56 AM
In this build all Sanity cases executed successfully.

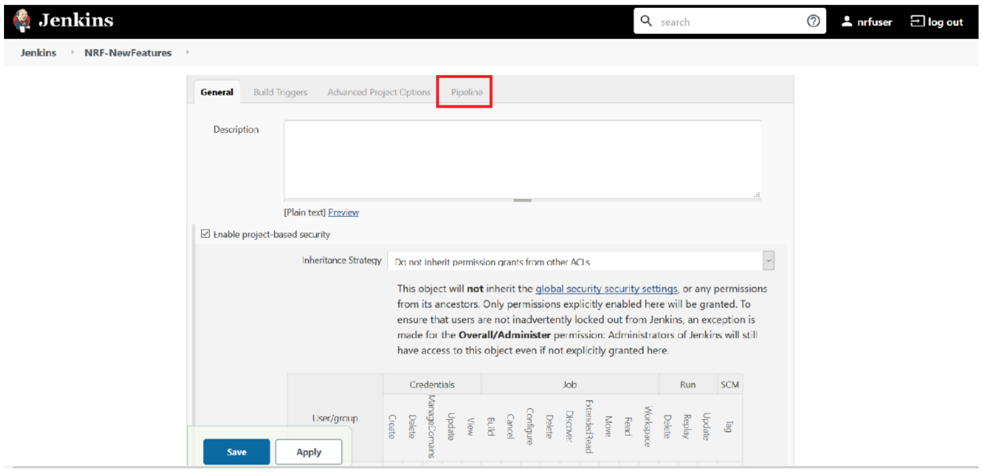
	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times: (Average full run time: ~3min 42s)	363ms	3min 35s	214ms	410ms
#5 All-Without-GEO-with-ASM Aug 15 17:27 No Changes	305ms	6min 17s	216ms	428ms
#4 Sanity-with-ASM Aug 15 17:26 No Changes	500ms	47s	369ms	538ms

In the above screen:

- Click **Configure** to navigate to the screen where configuration needs to be done.
- Click **Documentation** to navigate to the screen that has documented test cases, which are part of this NRF release.
- Click the blue dots inside **Build History** box to reach the success console logs of the "Sanity", "All-Without-GEO" and "All-GEO" respectively.
- The **Stage View** represents the already executed pipeline for the customer reference.

2. Click **Configure**. User **MUST** wait for the page to load completely. Once the page loads completely, click the **Pipeline** tab as shown below:
MAKE SURE THAT THE SCREEN SHOWN BELOW LOADS COMPLETELY BEFORE YOU PERFORM ANY ACTION ON IT. ALSO, DO NOT MODIFY ANY CONFIGURATION OTHER THAN DISCUSSED BELOW.

Figure 3-5 Pipeline Option



Description

[Plain text] Preview

☑ Enable project-based security

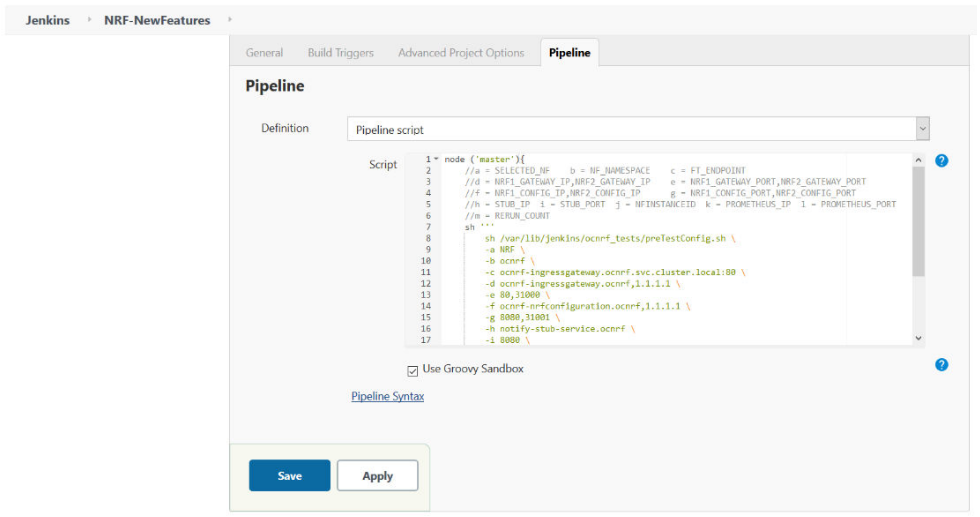
Inheritance Strategy: Do not inherit permission grants from other ACLs

This object will **not** inherit the global security security settings, or any permissions from its ancestors. Only permissions explicitly enabled here will be granted. To ensure that users are not inadvertently locked out from Jenkins, an exception is made for the **Overall/Administer** permission: Administrators of Jenkins will still have access to this object even if not explicitly granted here.

User/group	Credentials					Job										Run			SCM
	Create	Delete	ManageDomains	Update	View	Build	Cancel	Configure	Delete	Discover	ExtendedRead	Move	Read	Workspace		Delete	Replay	Update	Tag

3. The **Pipeline** section of the configuration page appears as shown below:

Figure 3-6 Pipeline Section



Pipeline

Definition: Pipeline script

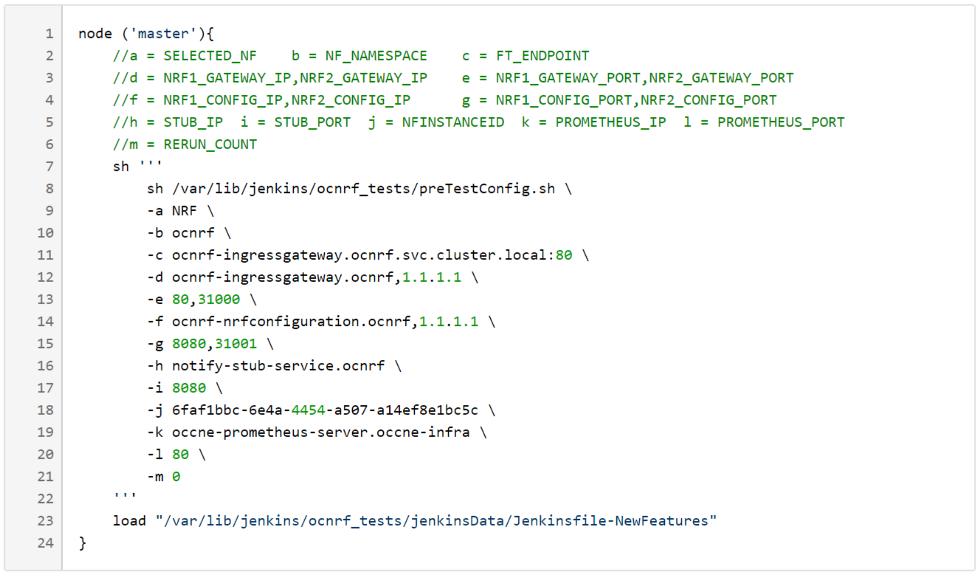
Script:

```
node ('master'){
    //a = SELECTED_NF    b = NF_NAMESPACE    c = FT_ENDPOINT
    //d = NRF1_GATEWAY_IP,NRF2_GATEWAY_IP    e = NRF1_GATEWAY_PORT,NRF2_GATEWAY_PORT
    //f = NRF1_CONFIG_IP,NRF2_CONFIG_IP      g = NRF1_CONFIG_PORT,NRF2_CONFIG_PORT
    //h = STUB_IP  i = STUB_PORT  j = NFINSTANCEID  k = PROMETHEUS_IP  l = PROMETHEUS_PORT
    //m = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocnrf_tests/preTestConfig.sh \
        -a NRF \
        -b ocnrf \
        -c ocnrf-ingressgateway.ocnrf.svc.cluster.local:80 \
        -d ocnrf-ingressgateway.ocnrf,1.1.1.1 \
        -e 80,31000 \
        -f ocnrf-nrfconfiguration.ocnrf,1.1.1.1 \
        -g 8080,31001 \
        -h notify-stub-service.ocnrf \
        -i 8080 \
```

☑ Use Groovy Sandbox

In the above screen, you can change the values of the '**Pipeline script**'. The content of the pipeline script is as follows:

Figure 3-7 Pipeline Script



```
1   node ('master'){
2       //a = SELECTED_NF    b = NF_NAMESPACE    c = FT_ENDPOINT
3       //d = NRF1_GATEWAY_IP,NRF2_GATEWAY_IP    e = NRF1_GATEWAY_PORT,NRF2_GATEWAY_PORT
4       //f = NRF1_CONFIG_IP,NRF2_CONFIG_IP      g = NRF1_CONFIG_PORT,NRF2_CONFIG_PORT
5       //h = STUB_IP  i = STUB_PORT  j = NFINSTANCEID  k = PROMETHEUS_IP  l = PROMETHEUS_PORT
6       //m = RERUN_COUNT
7       sh '''
8           sh /var/lib/jenkins/ocnrf_tests/preTestConfig.sh \
9           -a NRF \
10          -b ocnrf \
11          -c ocnrf-ingressgateway.ocnrf.svc.cluster.local:80 \
12          -d ocnrf-ingressgateway.ocnrf,1.1.1.1 \
13          -e 80,31000 \
14          -f ocnrf-nrfconfiguration.ocnrf,1.1.1.1 \
15          -g 8080,31001 \
16          -h notify-stub-service.ocnrf \
17          -i 8080 \
18          -j 6faf1bbc-6e4a-4454-a507-a14ef8e1bc5c \
19          -k occne-prometheus-server.occne-infra \
20          -l 80 \
21          -m 0
22      '''
23      load "/var/lib/jenkins/ocnrf_tests/jenkinsData/Jenkinsfile-NewFeatures"
24  }
```

Note:

The User **MUST NOT** change any other value apart from **line number 9** to **line 21**.

You can change the parameter values from **"a"** - to - **"m"** as per user requirement. The parameter details are available as comments from line number **2** - to - **6**.

```
a - Name of the NF to be tested in capital (NRF).
b - Namespace in which the NRF is deployed
c - endPointIP:endPointPort value used while deploying the NRF
using the helm chart
d - Comma separated values of NRF1 and NRF2 ingress gateway service
(ocnrf-ingressgateway.ocnrf,1.1.1.1) - this is also known as as
cluster_domain.
A dummy value of NRF2 ingress gateway (1.1.1.1) is provided for the
reference.
e - Comma separated values of NRF1 and NRF2 port of ingressgateway
service (80,31000)
 - A dummy value of NRF2 ingress gateway port (31000) is provided
for the reference.
f - Comma separated values of NRF1 and NRF2 configuration service
(ocnrf-nrfconfiguration.ocnrf,1.1.1.1) - this is also known as as
cluster_domain.
A dummy value of NRF2 configuration service (1.1.1.1) is provided
for the reference.
g - Comma separated values of NRF1 and NRF2 port of configuration
service (8080,31001)
 - A dummy value of NRF2 configuration microservice port (31001) is
provided for the
reference.
```

```
h - Name_of_stub_service.namespace (notify-stub-service.ocnrf)
i - Port of stub service (8080)
j - NRF_Instance ID (6faf1bbc-6e4a-4454-a507-a14ef8e1bc5c)
k - Name_of_Prometheus_service.namespace (occne-prometheus-
server.occne-infra)
l - Port of Prometheus service (80)
m - Number of times the re-run of failed case is allowed (default
as 0).
```

Note:

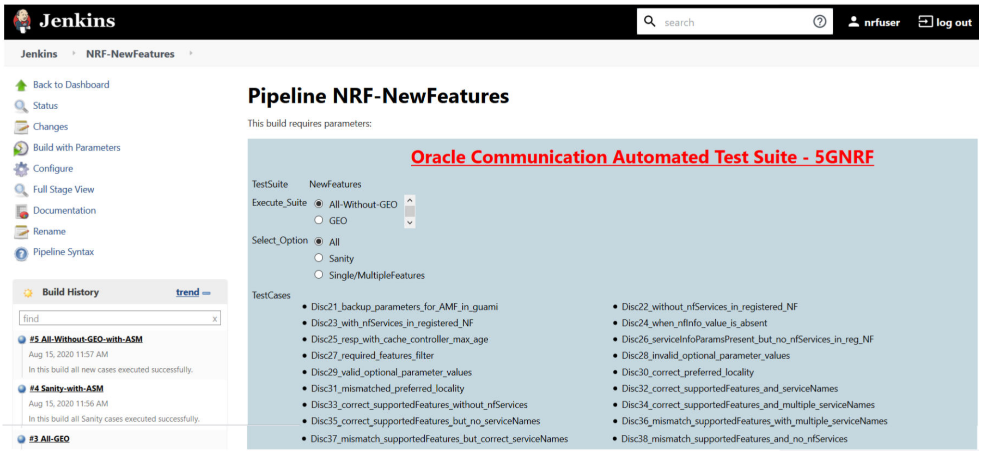
You need not to change any value if

- OCCNE cluster is used
- NRF, ATS and Stub are deployed in the ocnrf namespace
- Any GEO-Redundancy case is not executed.

If any GEO-Redundancy case is executed, you have to provide pipeline script values for NRF-2 in d, e, f and g options as per deployment.

4. Click **Save** after making neccesary changes. The NRF-NewFeatures screen appears. Click **Build with Parameters**. Following screen appears:

Figure 3-8   Pipeline NRF-NewFeatures



#5 All-Without-GEO-with-ASM
Aug 15, 2020 11:57 AM
In this build all new cases executed successfully.

#4 Sanity-with-ASM
Aug 15, 2020 11:56 AM
In this build all Sanity cases executed successfully.

#3 All-GEO

Pipeline NRF-NewFeatures

This build requires parameters:

Oracle Communication Automated Test Suite - 5GNRF

TestSuite NewFeatures

Execute_Suite [✓ All-Without-GEO] GEO

Select_Option [✓ All] Sanity Single/MultipleFeatures

TestCases

- Disc21_backup_parameters_for_AMF_in_guami
- Disc22_without_nfServices_in_registered_NF
- Disc23_with_nfServices_in_registered_NF
- Disc24_when_nfInfo_value_is_absent
- Disc25_resp_with_cache_controller_max_age
- Disc26_serviceInfoParamsPresent_but_no_nfServices_in_reg_NF
- Disc27_required_features_filter
- Disc28_invalid_optional_parameter_values
- Disc29_valid_optional_parameter_values
- Disc30_correct_preferred_locality
- Disc31_mismatched_preferred_locality
- Disc32_correct_supportedFeatures_and_serviceNames
- Disc33_correct_supportedFeatures_without_nfServices
- Disc34_correct_supportedFeatures_and_multiple_serviceNames
- Disc35_correct_supportedFeatures_but_no_serviceNames
- Disc36_mismatch_supportedFeatures_with_multiple_serviceNames
- Disc37_mismatch_supportedFeatures_but_correct_serviceNames
- Disc38_mismatch_supportedFeatures_and_no_nfServices

In the above screen, you have **Execute_Suite** options to execute NRF test cases either:

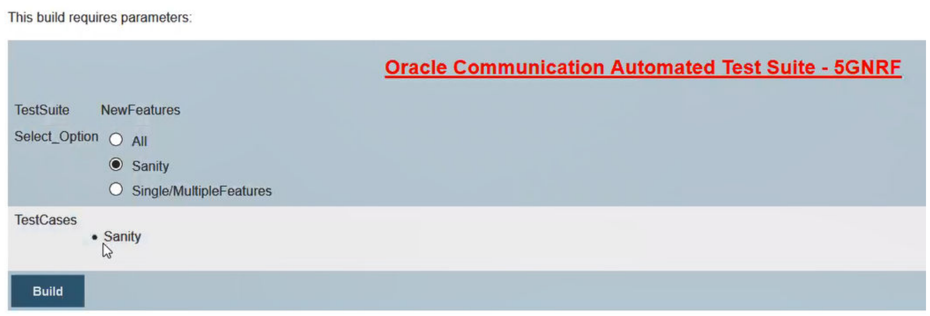
- **All-Without-GEO**: This is the default option. It executes all the testcases except GEO-Redundancy.
- **GEO**: It executes all the GEO-Redundancy cases.

In the above screen, there are three **Select_Option**(s), which are:

- **All:** This is the default option. It executes all the NRF test cases. User just need to scroll down and click **Build** to execute all the test cases.

- **Sanity:** It is recommended to execute Sanity before executing any test case. This helps to ensure that all the deployments are done properly. When you select Sanity, the following screen appears:

Figure 3-9 Build Requires Parameters - Sanity


This build requires parameters:

Oracle Communication Automated Test Suite - 5GNRF

TestSuite NewFeatures

Select_Option All [✓ Sanity] Single/MultipleFeatures

TestCases
- Sanity

Click **Build** to execute all the sanity test cases.

> **Note:**
>
> Sanity option is not available when Execute_Suite is set to GEO.

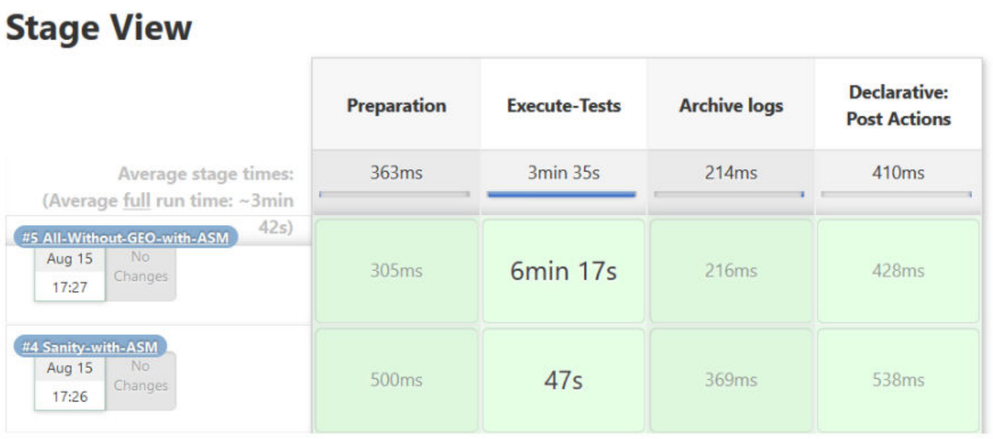
- **Single/MultipleFeatures:** This option allows you to select any number of test cases that you want to execute from the list of total test cases available for execution. After selecting the test cases, scroll-down and click **Build**. The selected NRF test cases are executed.

The NRF testcases are divided into following NRF Service operations:

- **NRF Sanity** - This feature file contains all the basic sanity test cases for NRF ATS to validate NRF deployment. It is advisable for users to execute these test cases before starting a complete suite.
- **Discovery** - These feature files are listed with a prefix as **"Disc"**.
- **Registration** - These feature files are listed with a prefix as **"Reg"**.
- **NRF SLF** - These feature files are lited with a prefix as **"SLF"**.
- **GEO Redundancy** - These feature files are lited with a prefix as **"GEO"**.

The following screen shows successful execution of **Sanity**, **All-Without-GEO** and **GEO** test cases.

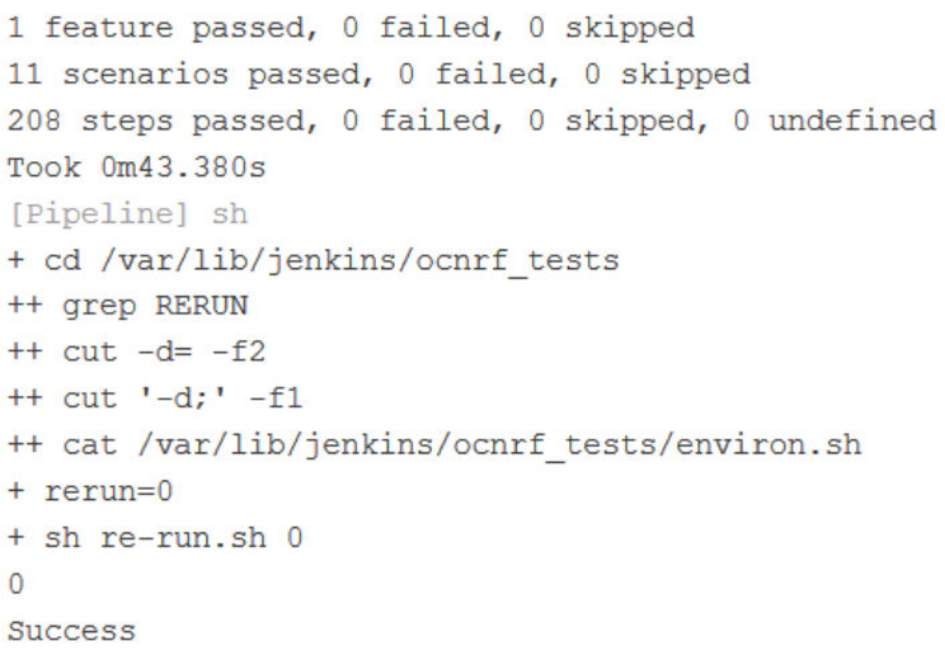
Figure 3-10 Sample Screen



	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times	363ms	3min 35s	214ms	410ms
#5 All-Without-GEO-with-ASM	305ms	6min 17s	216ms	428ms
#4 Sanity-with-ASM	500ms	47s	369ms	538ms

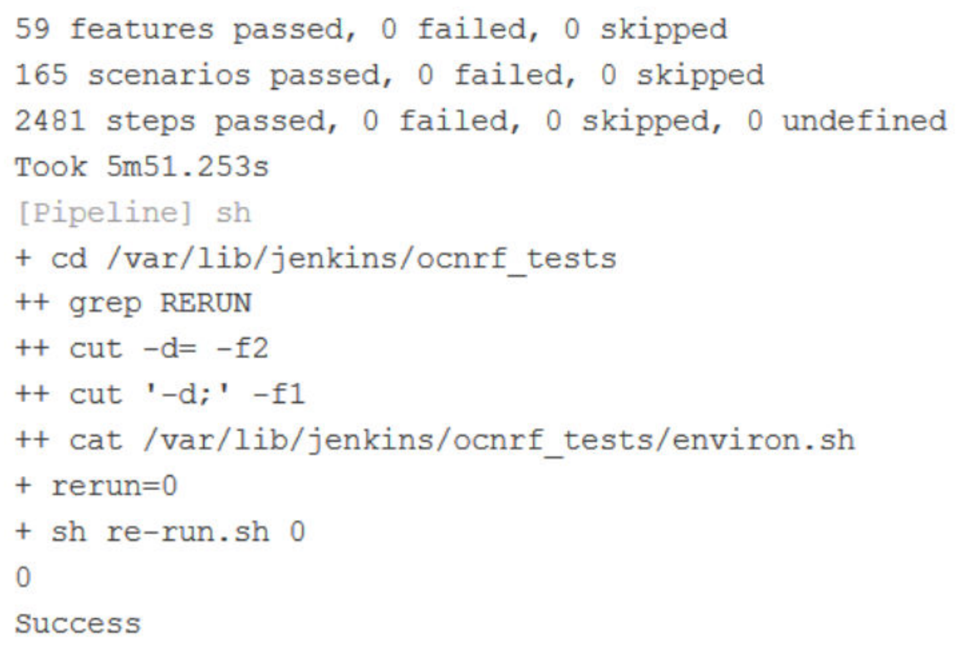
The following screens show the results for **Sanity**, **All-Without-GEO** and **GEO** test cases in the same order as they are executed.

Figure 3-11 Test Cases Result - Sanity with Service Mesh



```
1 feature passed, 0 failed, 0 skipped
11 scenarios passed, 0 failed, 0 skipped
208 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m43.380s
[Pipeline] sh
+ cd /var/lib/jenkins/ocnrf_tests
++ grep RERUN
++ cut -d= -f2
++ cut '-d;' -f1
++ cat /var/lib/jenkins/ocnrf_tests/environ.sh
+ rerun=0
+ sh re-run.sh 0
0
Success
```

Figure 3-12   Test Cases Result - All-Without-Geo-with Service Mesh



```
59 features passed, 0 failed, 0 skipped
165 scenarios passed, 0 failed, 0 skipped
2481 steps passed, 0 failed, 0 skipped, 0 undefined
Took 5m51.253s
[Pipeline] sh
+ cd /var/lib/jenkins/ocnrf_tests
++ grep RERUN
++ cut -d= -f2
++ cut '-d;' -f1
++ cat /var/lib/jenkins/ocnrf_tests/environ.sh
+ rerun=0
+ sh re-run.sh 0
0
Success
```

NRF-NewFeatures Documentation

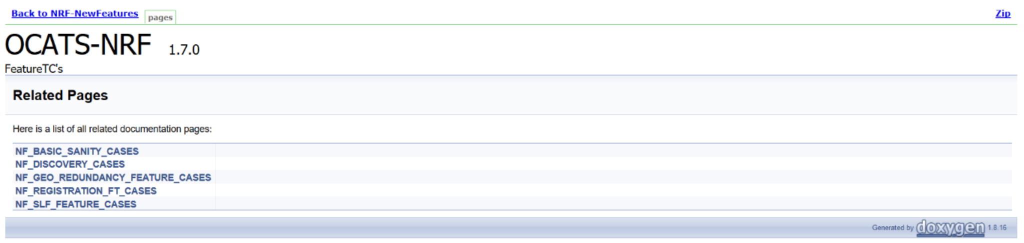
To view NRF test cases, go to NRF-NewFeatures pipeline and click **Documentation** link in the left navigation pane. It shows all the test cases provided as part of NRF ATS -1.7.0 along with sanity cases.

Note:

No new test cases are added as part of NRF-ATS 1.7.2 release.

The following screen shows all the documentation features:

Figure 3-13   NRF-NewFeatures Documentation



OCATS-NRF 1.7.0

FeatureTC's

Related Pages

Here is a list of all related documentation pages:

NF_BASIC_SANITY_CASES
NF_DISCOVERY_CASES
NF_GEO_REDUNDANCY_FEATURE_CASES
NF_REGISTRATION_FT_CASES
NF_SLF_FEATURE_CASES

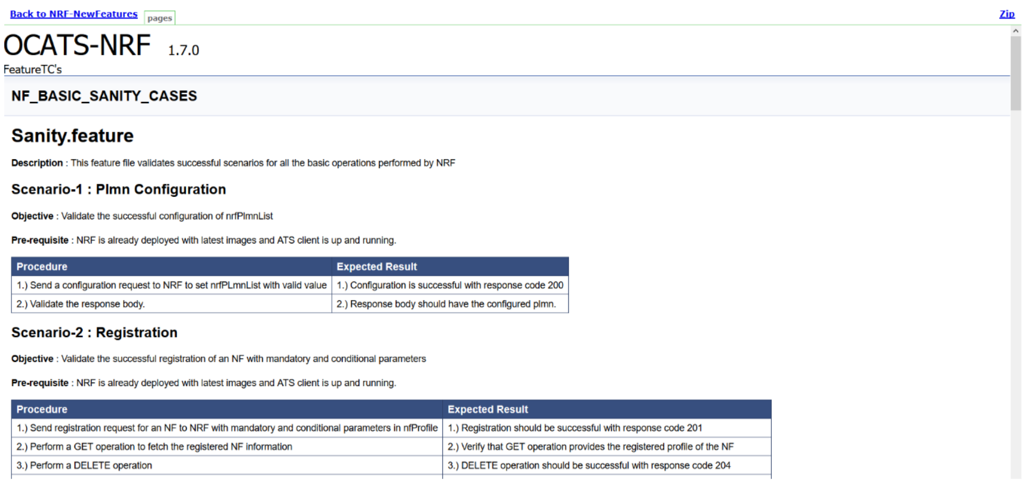
Each one of the documentation feature is described below:

- **NF_BASIC_SANITY_CASES** - Lists all the sanity cases, which are useful to identify whether all the NRF functionality works fine.

- **NF_DISCOVERY_CASES** - Lists all the discovery microservice related cases.
- **NF_GEO_REDUNDANCY_FEATURE_CASES** - Lists all the Geo-Redundancy related cases.
- **NF_REGISTRATION_FT_CASES** - Lists all the registration related cases.
- **NF_SLF_FEATURE_CASES** - Lists all the SLF related cases.

Click any functionality to view its test cases and scenarios of each test case. A sample screen is as follows:

Figure 3-14 Sample Feature - NF_BASIC_SANITY_CASES



OCATS-NRF 1.7.0

FeatureTC's

NF_BASIC_SANITY_CASES

Sanity.feature

Description : This feature file validates successful scenarios for all the basic operations performed by NRF

Scenario-1 : Plmn Configuration

Objective : Validate the successful configuration of nrfPlmnList

Pre-requisite : NRF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send a configuration request to NRF to set nrfPLmnList with valid value	1.) Configuration is successful with response code 200
2.) Validate the response body.	2.) Response body should have the configured plmn.

Scenario-2 : Registration

Objective : Validate the successful registration of an NF with mandatory and conditional parameters

Pre-requisite : NRF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send registration request for an NF to NRF with mandatory and conditional parameters in nfProfile	1.) Registration should be successful with response code 201
2.) Perform a GET operation to fetch the registered NF information	2.) Verify that GET operation provides the registered profile of the NF
3.) Perform a DELETE operation	3.) DELETE operation should be successful with response code 204

Based on the functionalities covered under Documentation, the **Build Requires Parameters** screen displays test cases. To navigate back to the Pipeline NRF-NewFeatures screen, click **Back to NRF-NewFeatures** link available on top left corner of the screen.

NRF-Regression Pipeline

This pre-configured pipeline contains all the test cases that were provided as part of NRF ATS 1.7.0. However, some test cases are updated as per new implementation of NRF.

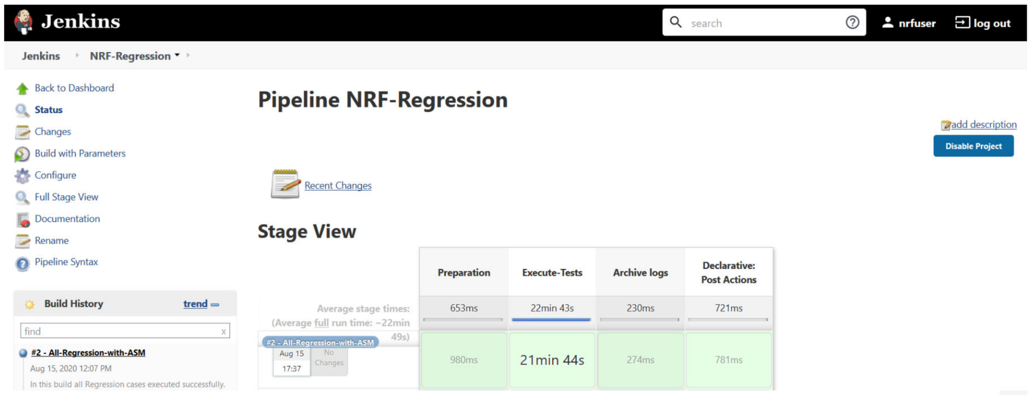
The configuration method and parameters are same as the **NewFeatures** pipeline. Only difference in this pipeline is that it does not have **Sanity** and **GEO** option. Thus while configuring this pipeline, you need not to provide any information for NRF2.

The NRF-Regression test cases are divided into following service operations:

- **AccessToken** - These feature files are listed with a prefix as **"oAuth"**.
- **Configuration** - These feature files are listed with a prefix as **"Config"**.
- **Discovery** - These feature files are listed with a prefix as **"Disc"**.
- **NRF Forwarding** - These feature files are listed with a prefix as **"Forwarding"**.
- **NRF Functional** - These feature files are listed with a prefix as **"Feat"**.
- **Registration** - These feature files are listed with a prefix as **"Reg"** and **"Upd"**. These are related to update operation of registered profiles.

- **NRF SLF** - These feature files are listed with a prefix as **"SLF"**.
- **Subscription** - These feature files are listed with a prefix as **"Subs"**.

Figure 3-15 NRF-Regression



Pipeline NRF-Regression

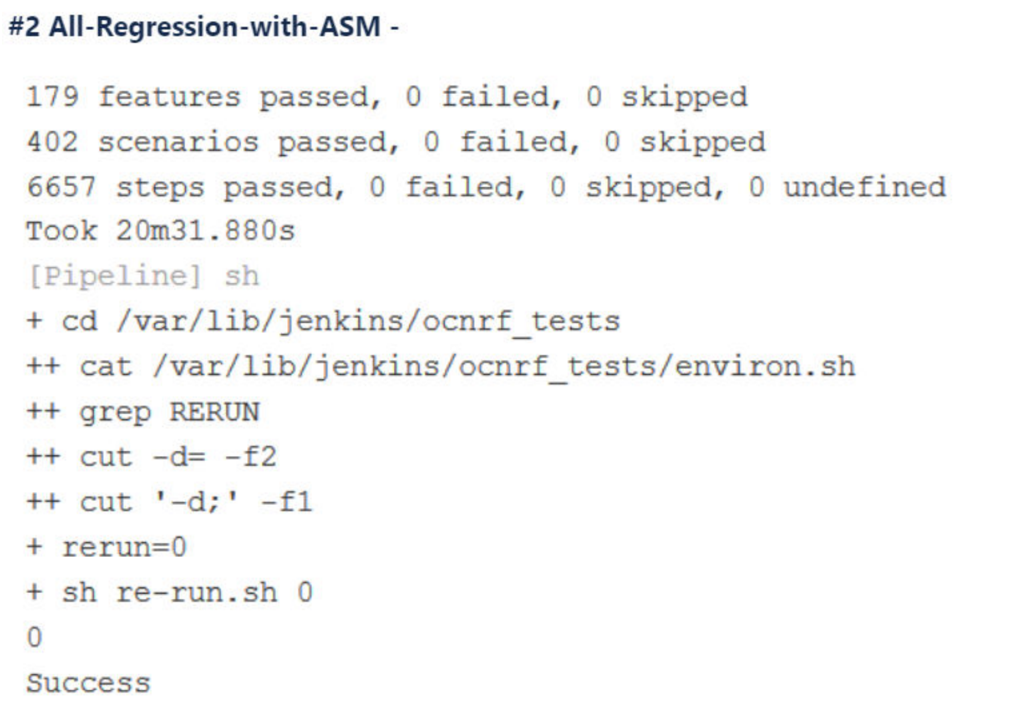
Stage View

	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times: (Average full run time: ~22min 49s)	653ms	22min 43s	230ms	721ms
#2 - All-Regression-with-ASM Aug 15 17:37 No Changes	980ms	21min 44s	274ms	781ms

#2 - All-Regression-with-ASM
Aug 15, 2020 12:07 PM
In this build all Regression cases executed successfully.

The following screen shows full successful execution as part of ATS image.

Figure 3-16 NRF-Regression with Service Mesh Result



#2 All-Regression-with-ASM -

```
179 features passed, 0 failed, 0 skipped
402 scenarios passed, 0 failed, 0 skipped
6657 steps passed, 0 failed, 0 skipped, 0 undefined
Took 20m31.880s
[Pipeline] sh
+ cd /var/lib/jenkins/ocnrf_tests
++ cat /var/lib/jenkins/ocnrf_tests/environ.sh
++ grep RERUN
++ cut -d= -f2
++ cut '-d;' -f1
+ rerun=0
+ sh re-run.sh 0
0
Success
```

NRF-Regression Documentation

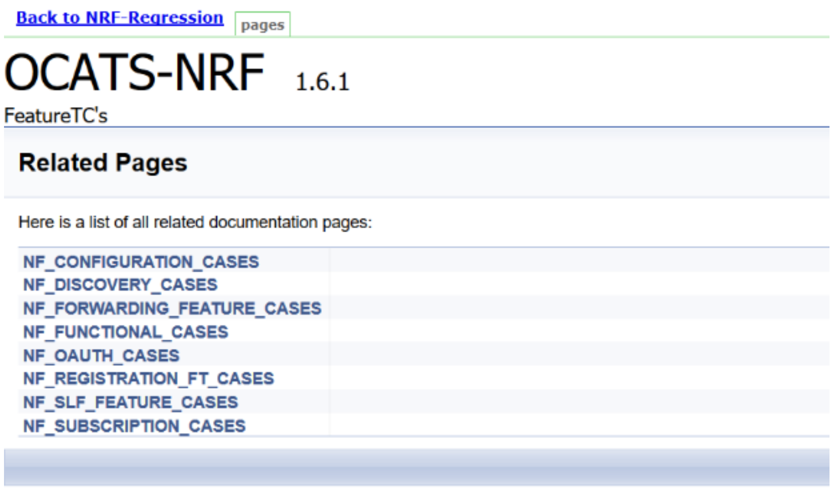
Click **Documentation** in the left navigation pane of the NRF-Regression pipeline to view all the test cases provided till NRF ATS 1.6.1.

The NRF test cases are divided into multiple groups based on following functionalities:

- **NF_CONFIGURATION_CASES** - Lists the cases related to NRF configuration.
- **NF_DISCOVERY_CASES** - Lists all the discovery microservice related cases.
- **NF_FORWARDING_FEATURE_CASES** - Lists all the forwarding related cases.
- **NF_FUNCTIONAL_CASES** - Lists all the functional cases.
- **NF_OAUTH_CASES** - Lists all the accesstoken related cases.
- **NF_REGISTRATION_CASES** - Lists all the registration related cases.
- **NF_SLF_FEATURE_CASES** - Lists all the SLF related cases.
- **NF_SUBSCRIPTION_CASES** - Lists all subscription related cases.

Following screen appears:

Figure 3-17 NRF-Regression Documentation


Back to NRF-Regression
pages

OCATS-NRF 1.6.1
FeatureTC's

Related Pages

Here is a list of all related documentation pages:

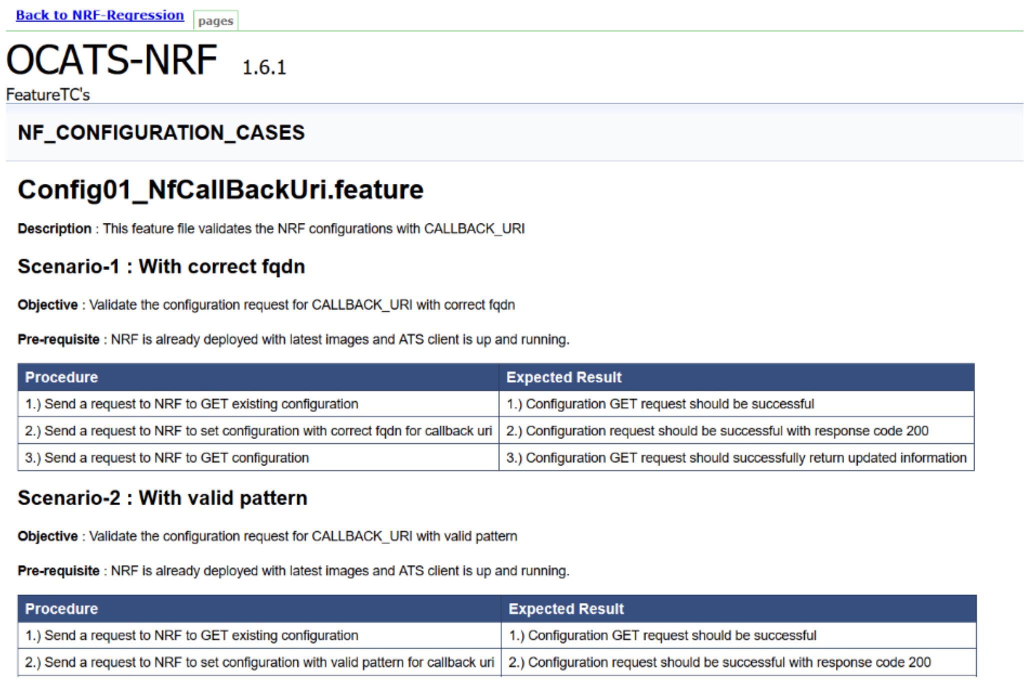
NF_CONFIGURATION_CASES
NF_DISCOVERY_CASES
NF_FORWARDING_FEATURE_CASES
NF_FUNCTIONAL_CASES
NF_OAUTH_CASES
NF_REGISTRATION_FT_CASES
NF_SLF_FEATURE_CASES
NF_SUBSCRIPTION_CASES

The NRF-Regression test cases are divided into multiple groups based on the functionality.

- **NF_CONFIGURATION_CASES** - Lists all the cases related to NRF configuration.
- **NF_DISCOVERY_CASES** - Lists all the discovery microservice related cases.
- **NF_FORWARDING_FEATURE_CASES** - Lists all the forwarding related cases.
- **NF_FUNCTIONAL_CASES** - Lists all the functional cases.
- **NF_OAUTH_CASES** - Lists all the accesstoken related cases.
- **NF_REGISTRATION_CASES** - Lists all the registration related cases.
- **NF_SLF_FEATURE_CASES** - Lists all the SLF related cases.
- **NF_SUBSCRIPTION_CASES** - Lists all subscription related cases.

A sample screen showing documentation for NRF ATS - 1.6.1 is given below:

Figure 3-18 Sample Screen: NRF-Regression Documentation



Back to NRF-Regression pages

OCATS-NRF 1.6.1

FeatureTC's

NF_CONFIGURATION_CASES

Config01_NfCallBackUri.feature

Description : This feature file validates the NRF configurations with CALLBACK_URI

Scenario-1 : With correct fqdn

Objective : Validate the configuration request for CALLBACK_URI with correct fqdn

Pre-requisite : NRF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send a request to NRF to GET existing configuration	1.) Configuration GET request should be successful
2.) Send a request to NRF to set configuration with correct fqdn for callback uri	2.) Configuration request should be successful with response code 200
3.) Send a request to NRF to GET configuration	3.) Configuration GET request should successfully return updated information

Scenario-2 : With valid pattern

Objective : Validate the configuration request for CALLBACK_URI with valid pattern

Pre-requisite : NRF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send a request to NRF to GET existing configuration	1.) Configuration GET request should be successful
2.) Send a request to NRF to set configuration with valid pattern for callback uri	2.) Configuration request should be successful with response code 200

Executing NSSF Test Cases using ATS

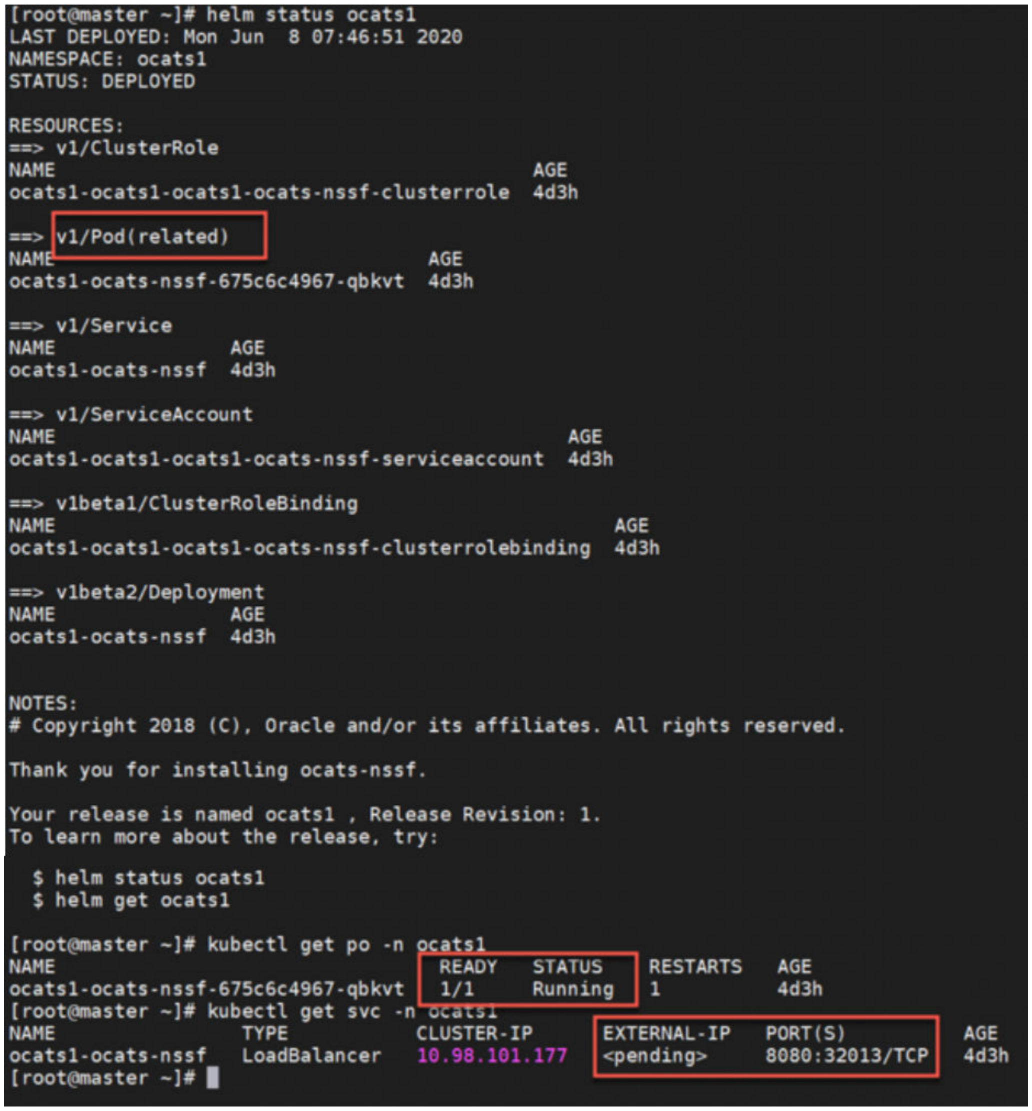
To execute NSSF Test Cases using NRF ATS 1.4, you need to ensure that following prerequisites are fulfilled.

- Before deploying NSSF, the user must create certificates/keys (public and private) for AccessToken microservice. The public keys (RSA and ECDSA) must be copied to the ATS pod at **/var/lib/jenkins/ocnssf_tests/public_keys** location.
- User must deploy NSSF 1.4 with default helm configurations using helm charts.
- All NSSF micro-services should be up and running including AccessToken microservice.

Logging into ATS

Before logging into ATS, you need to ensure that ATS is deployed successfully using HELM charts. A sample screen is given below:

Figure 3-19 Verifying ATS Deployment



```
[root@master ~]# helm status ocats1
LAST DEPLOYED: Mon Jun  8 07:46:51 2020
NAMESPACE: ocats1
STATUS: DEPLOYED

RESOURCES:
==> v1/ClusterRole
NAME                                         AGE
ocats1-ocats1-ocats1-ocats-nssf-clusterrole  4d3h

==> v1/Pod(related)
NAME                                AGE
ocats1-ocats-nssf-675c6c4967-qbkvt  4d3h

==> v1/Service
NAME               AGE
ocats1-ocats-nssf  4d3h

==> v1/ServiceAccount
NAME                                            AGE
ocats1-ocats1-ocats1-ocats-nssf-serviceaccount  4d3h

==> v1beta1/ClusterRoleBinding
NAME                                                AGE
ocats1-ocats1-ocats1-ocats-nssf-clusterrolebinding  4d3h

==> v1beta2/Deployment
NAME               AGE
ocats1-ocats-nssf  4d3h


NOTES:
# Copyright 2018 (C), Oracle and/or its affiliates. All rights reserved.

Thank you for installing ocats-nssf.

Your release is named ocats1 , Release Revision: 1.
To learn more about the release, try:

  $ helm status ocats1
  $ helm get ocats1

[root@master ~]# kubectl get po -n ocats1
NAME                                READY   STATUS    RESTARTS   AGE
ocats1-ocats-nssf-675c6c4967-qbkvt  1/1     Running   1          4d3h
[root@master ~]# kubectl get svc -n ocats1
NAME               TYPE          CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE
ocats1-ocats-nssf  LoadBalancer  10.98.101.177  <pending>     8080:32013/TCP  4d3h
[root@master ~]#
```

There are two ways to login to ATS Jenkins GUI.

- When an external load balancer (metalLB in case of OCCNE) is available and an external IP is provided to the ATS service, the user can login to ATS GUI using <External-IP>:8080.
- When an external IP is not provided to the ATS service, the user can open the browser and provide the external IP of the worker node and nodeport of the ATS service to login to ATS GUI.
  `<Worker-Node-IP>:<Node-Port-of-ATS>`

> **Note:**
>
> In the **Verifying ATS Deployment** screen, ATS nodeport is highlighted in red as 32013. For more details on ATS deployment, refer to **NSSF ATS Installation Procedure.**

Open a browser and provide IP and port details as <Worker-Node-IP>:<NodePort-of-ATS> (As per the above example: 10.98.101.171:32013). The ATS login screen appears.

Executing ATS

To execute ATS:

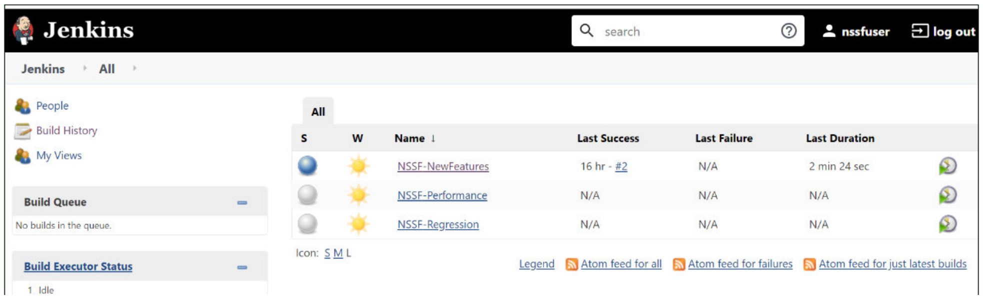
1. Enter the username as 'nssfuser' and password as 'nssfpasswd'. Click **Sign in**.

   > **Note:**
   >
   > If you want to modify your default login password, refer to Modifying Login Password

   The following screen appears showing pre-configured pipelines for NSSF individually (3 Pipelines).

   - **NSSF-New-Features:** This pipeline has all the test cases that are delivered as part of NSSF ATS - 1.4.
   - **NSSF-Performance:** This pipeline is not operational as of now. It is reserved for future releases of ATS.
   - **NSSF-Regression:** This pipleine has all the test cases of previous releases. As this is the first release of NSSF-ATS, this pipeline does not show any previous release test cases.

   Figure 3-20 Pre-Configured Pipelines

S	W	Name ↓	Last Success	Last Failure	Last Duration
●	☀	NSSF-NewFeatures	16 hr - #2	N/A	2 min 24 sec
●	☀	NSSF-Performance	N/A	N/A	N/A
●	☀	NSSF-Regression	N/A	N/A	N/A

   Icon: S M L

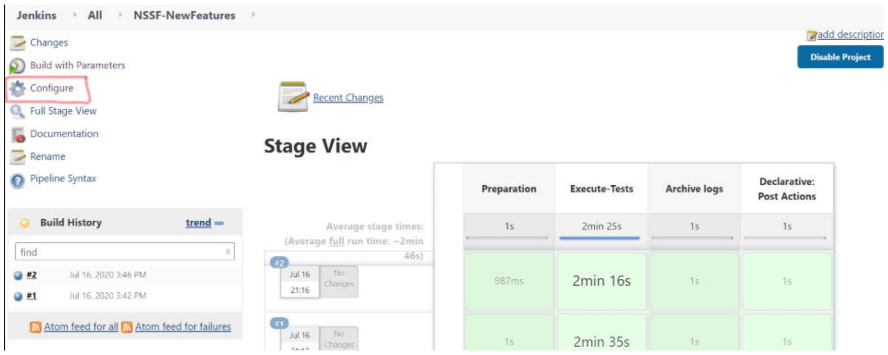
   Legend Atom feed for all Atom feed for failures Atom feed for just latest builds

   Each one of this pipeline is explained below:

   - **NSSF-NewFeatures Pipeline:** After identifying the NSSF pipelines, the user needs to do one-time configuration in ATS as per their SUT deployment. In this pipeline, all the new testcases related to NSSF are executed. To configure its parameters:

a. Click **NSSF-NewFeatures** in the Name column. The following screen appears:

Figure 3-21 NSSF-NewFeatures Pipeline



Stage View

	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times: (Average full run time: ~2min 46s)	1s	2min 25s	1s	1s
#2 Jul 16 21:16 No Changes	987ms	2min 16s	1s	1s
#1 Jul 16 No Changes	1s	2min 35s	1s	1s

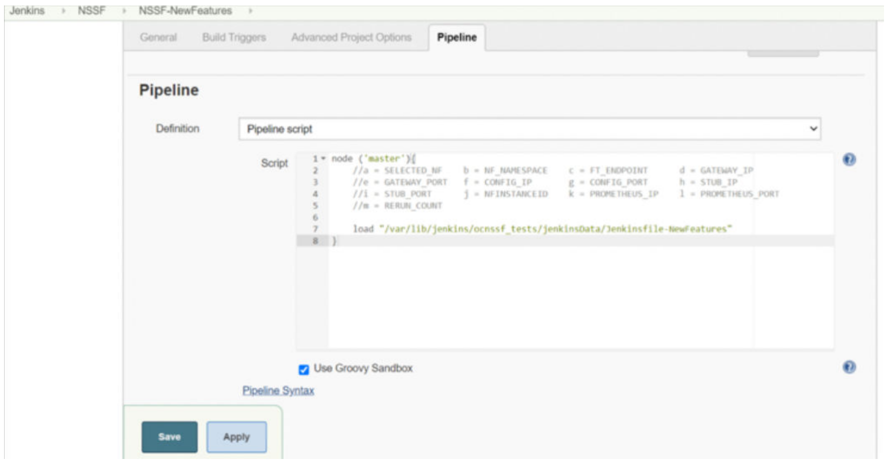
Build	Date
#2	Jul 16, 2020 3:46 PM
#1	Jul 16, 2020 3:42 PM

In the above screen:

- Click **Configure** to navigate to a screen where configuration needs to be done.
- Click **Documentation** to view the documented test cases.
- Click blue dots inside **Build History** box to view the success console logs of the "All" and "Sanity" respectively.
- The **Stage View** represents already executed pipeline for the customer reference.

b. Click **Configure**. Users MUST wait for the page to load completely. Once the page loads completely, click the **Pipeline** tab to reach the Pipeline configuration as shown below:
MAKE SURE THAT THE SCREEN SHOWN ABOVE LOADS COMPLETELY BEFORE YOU PERFORM ANY ACTION ON IT. ALSO, DO NOT MODIFY ANY CONFIGURATION OTHER THAN DISCUSSED BELOW.

Figure 3-22 NSSF Configure



Definition: Pipeline script

Script:

```
node ('master'){
    //a = SELECTED_NF    b = NF_NAMESPACE    c = FT_ENDPOINT     d = GATEWAY_IP
    //e = GATEWAY_PORT   f = CONFIG_IP       g = CONFIG_PORT     h = STUB_IP
    //i = STUB_PORT      j = NFINSTANCEID    k = PROMETHEUS_IP   l = PROMETHEUS_PORT
    //m = RERUN_COUNT

    load "/var/lib/jenkins/ocnssf_tests/jenkinsData/Jenkinsfile-NewFeatures"
}
```

☑ Use Groovy Sandbox

c. In the above screen, the values of the 'Pipeline script' needs to be changed. The content of the pipeline script is as follows:

```
node ('master'){
    //a = SELECTED_NF     b = NF_NAMESPACE    c =
FT_ENDPOINT      d = GATEWAY_IP
    //e = GATEWAY_PORT   f = CONFIG_IP       g =
CONFIG_PORT      h = STUB_IP
    //i = STUB_PORT      j = NFINSTANCEID    k =
PROMETHEUS_IP    l = PROMETHEUS_PORT
    //m = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocnssf_tests/preTestConfig.sh \
        -a NSSF \
        -b ocnssf \
        -c ocnssf-ingressgateway.ocnssf.svc.cluster.local:80
\
        -d ocnssf-ingressgateway.ocnssf \
        -e 80 \
        -f ocnssf-nssfconfiguration.ocnssf \
        -g 8080 \
        -h notify-stub-service.ocnssf \
        -i 8080 \
        -j 6faf1bbc-6e4a-4454-a507-a14ef8e1bc5c \
        -k occne-prometheus-server.occne-infra \
        -l 80 \
        -m 2
    '''
    load "/var/lib/jenkins/ocnssf_tests/jenkinsData/
Jenkinsfile-NewFeatures"
}
```

Note:

The User MUST NOT change any other value apart from line number 8 to line 20.

You can change only those parameters that are marked as "a" to "m" as per your requirement.

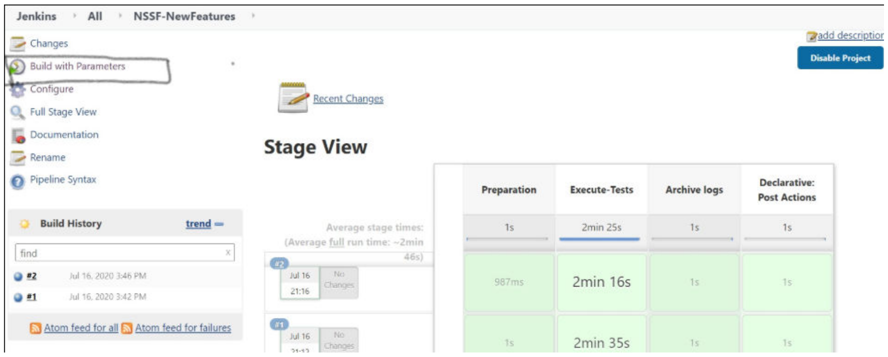
- a - Name of the NF to be tested in capital (NSSF).
- b - Namespace in which the NSSF is deployed
- c - endPointIP:endPointPort value used while deploying the NSSF using the helm chart
- d - Name_of_NSSF_ingressgateway_service.namespace (ocnssf-nssfconfiguration.ocnssf) - this is also known as as cluster_domain.
- e - Port of ingressgateway service (80)
- f - Name_of_NSSF_configuration_service.namespace (ocnssf-nssfconfiguration.ocnssf)
- g - Port of configuration service (8080)

- h - Name_of_stub_service.namespace (notify-stub-service.ocnssf)
- i - Port of stub service (8080)
- j - NSSF_Instance ID (6faf1bbc-6e4a-4454-a507-a14ef8e1bc5c)
- k - Name_of_Prometheus_service.namespace (occne-prometheus-server.occne-infra)
- l - Port of Prometheus service (80)
- m - Number of times the re-run of failed case is allowed (default as 2).

> **Note:**
>
> You do not have to change any value if OCCNE cluster is used and NSSF, ATS and STUB are deployed in ocnssf namespace.

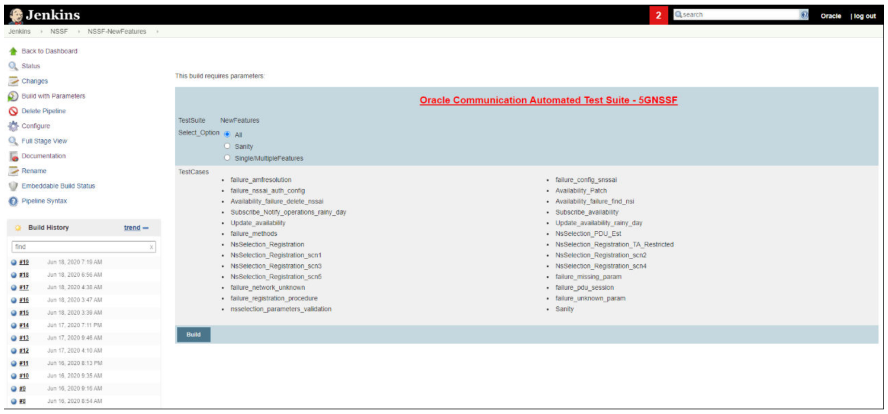
d. Click **Save** after making necessary changes. You are navigated back to the **Pipeline NSSF-NewFeatures** screen. Click **Build with Parameters** as shown below:

Figure 3-23 Build with Parameters



The following screen appears:

Figure 3-24 Build with Parameters Options



This build requires parameters:

Oracle Communication Automated Test Suite - 5GNSSF

TestSuite NewFeatures

Select_Option [✓ All] Sanity Single/MultipleFeatures

TestCases

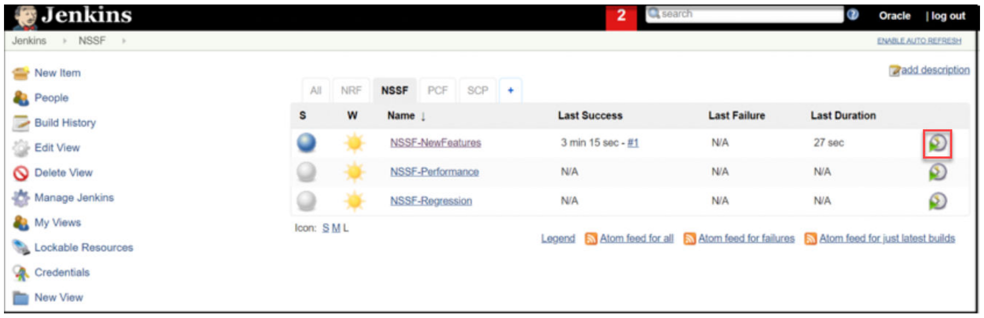
- failure_amfresolution
- failure_nssai_auth_config
- Availability_failure_delete_nssai
- Subscribe_Notify_operations_rainy_day
- Update_availability
- failure_methods
- NsSelection_Registration
- NsSelection_Registration_scn1
- NsSelection_Registration_scn3
- NsSelection_Registration_scn5
- failure_network_unknown
- failure_registration_procedure
- nsselection_parameters_validation
- failure_config_snssai
- Availability_Patch
- Availability_failure_find_nsi
- Subscribe_availability
- Update_availability_rainy_day
- NsSelection_PDU_Est
- NsSelection_Registration_TA_Restricted
- NsSelection_Registration_scn2
- NsSelection_Registration_scn4
- failure_missing_param
- failure_pdu_session
- failure_unknown_param
- Sanity

Executing NSSF Test Cases

To execute NSSF test cases:

1. Click the **Schedule a Build with parameters** icon present on the NSSF-NewFeatures screen in the extreme right column corresponding to NSSF-NewFeatures row as shown below:

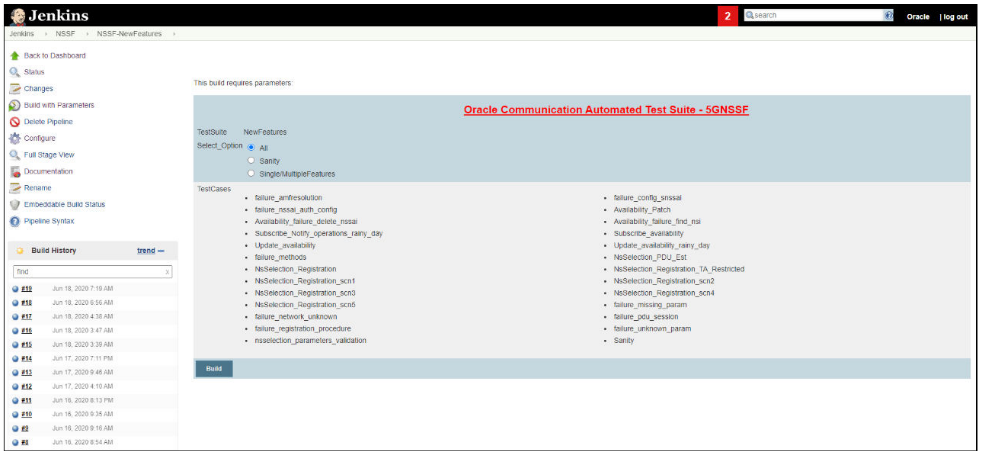
Figure 3-25 Schedule a Build with Parameters



S	W	Name ↓	Last Success	Last Failure	Last Duration	
●	☀	NSSF-NewFeatures	3 min 15 sec - #1	N/A	27 sec	⏲
○	☀	NSSF-Performance	N/A	N/A	N/A	⏲
○	☀	NSSF-Regression	N/A	N/A	N/A	⏲

2. The following screen appears:

Figure 3-26 Build Screen



#19	Jun 18, 2020 7:19 AM
#18	Jun 18, 2020 6:56 AM
#17	Jun 18, 2020 4:38 AM
#16	Jun 18, 2020 3:47 AM
#15	Jun 18, 2020 3:39 AM
#14	Jun 17, 2020 7:11 PM
#13	Jun 17, 2020 9:46 AM
#12	Jun 17, 2020 4:10 AM
#11	Jun 16, 2020 8:13 PM
#10	Jun 16, 2020 9:35 AM
#9	Jun 16, 2020 9:16 AM
#8	Jun 16, 2020 8:54 AM

This build requires parameters:

Oracle Communication Automated Test Suite - 5GNSSF

TestSuite NewFeatures

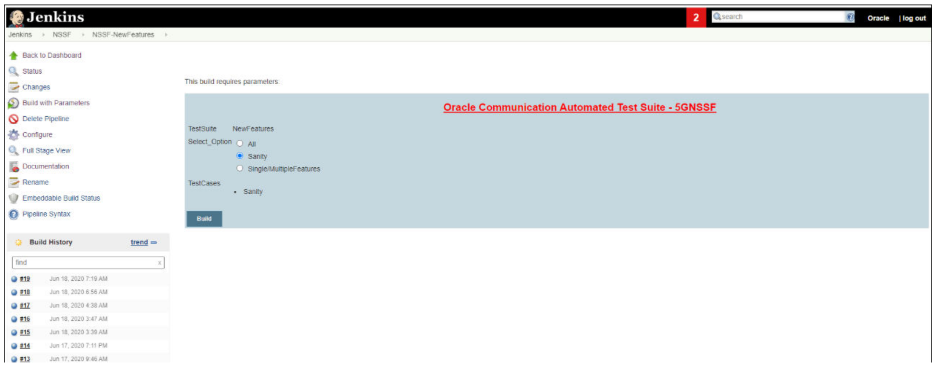
Select_Option [✓ All] Sanity Single/MultipleFeatures

TestCases
- failure_amfresolution
- failure_nssai_auth_config
- Availability_failure_delete_nssai
- Subscribe_Notify_operations_rainy_day
- Update_availability
- failure_methods
- NsSelection_Registration
- NsSelection_Registration_scn1
- NsSelection_Registration_scn3
- NsSelection_Registration_scn5
- failure_network_unknown
- failure_registration_procedure
- nsselection_parameters_validation
- failure_config_snssai
- Availability_Patch
- Availability_failure_find_nsi
- Subscribe_availability
- Update_availability_rainy_day
- NsSelection_PDU_Est
- NsSelection_Registration_TA_Restricted
- NsSelection_Registration_scn2
- NsSelection_Registration_scn4
- failure_missing_param
- failure_pdu_session
- failure_unknown_param
- Sanity

In the above screen, there are three **Select_Option(s)**, which are:

- **All:** By default, all the NSSF test cases are selected for execution. User just needs to scroll down and click **Build** to execute all the test cases.
- **Sanity:** It is recommended to execute Sanity before executing any test case. This helps to ensure that all the deployments are done properly or not. When you select Sanity, the following screen appears.

Figure 3-27 Select_Option(s) - Sanity



Jenkins › NSSF › NSSF-NewFeatures ›

Back to Dashboard
Status
Changes
Build with Parameters
Delete Pipeline
Configure
Full Stage View
Documentation
Rename
Embeddable Build Status
Pipeline Syntax

Build History trend

#19	Jun 18, 2020 7:19 AM
#18	Jun 18, 2020 6:56 AM
#17	Jun 18, 2020 4:38 AM
#16	Jun 18, 2020 3:47 AM
#15	Jun 18, 2020 3:39 AM
#14	Jun 17, 2020 7:11 PM
#13	Jun 17, 2020 9:46 AM

This build requires parameters:

Oracle Communication Automated Test Suite - 5GNSSF

TestSuite NewFeatures
Select_Option All [✓ Sanity] Single/MultipleFeatures
TestCases
- Sanity

Build

Click **Build** to execute all the sanity test cases.

- **Single/MultipleFeature:** This option allows you to select any number of test cases that you want to execute from the list of total test cases available for execution. After selecting the test cases, scroll-down and click **Build**. The selected NSSF test cases are executed.

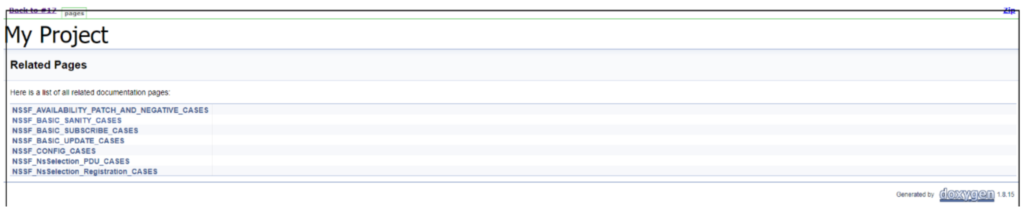
The NSSF test cases are divided into NSSF Service operations as follows:

- **Availability Update:** These feature files are listed with a prefix as "Update".
- **Configuration:** These feature files are listed with a prefix as "failure".
- **Registration:** These feature files are listed with a prefix as "NsSelection_Registration".
- **PDU Session:** These feature files are listed with a prefix as "NsSelection_PDU".
- **NSSF Sanity:** This feature file contains all the basic sanity cases for NSSF ATS 1.6.1.
- **Subscription:** These feature files are listed with a prefix as "Subscribe".

NewFeatures - Documentation

To view NSSF functionalities, go to NSSF-NewFeatures pipeline and click the **Documentation** link in the left navigation pane. The following screen appears:

Figure 3-28 NSSF - Documentation



Back to #17 pages Zip

My Project

Related Pages

Here is a list of all related documentation pages:

NSSF_AVAILABILITY_PATCH_AND_NEGATIVE_CASES
NSSF_BASIC_SANITY_CASES
NSSF_BASIC_SUBSCRIBE_CASES
NSSF_BASIC_UPDATE_CASES
NSSF_CONFIG_CASES
NSSF_NsSelection_PDU_CASES
NSSF_NsSelection_Registration_CASES

Generated by doxygen 1.8.15

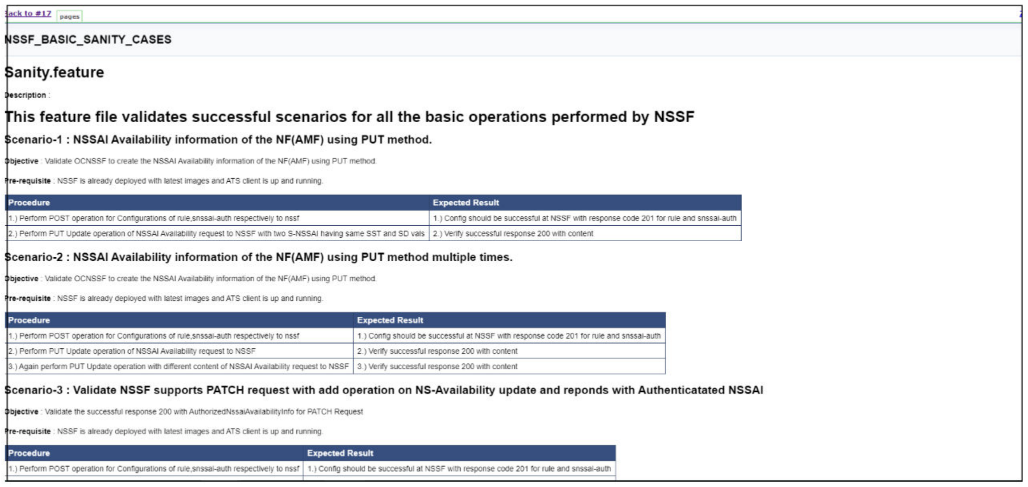
Each one of the documentation features is described below:

- **NSSF_BASIC_SANITY_CASES** - Lists all the sanity cases, which are useful to identify whether all the NSSF functionality works fine.

- **NSSF_CONFIG_CASES -** Lists all the test cases related to NSSF configuration.
- **NSSF_BASIC_UPDATE_CASES -** Lists all the test cases relaed to Availability Update.
- **NSSF_AVAILABILITY_PATCH_AND_NEGATIVE_CASES -** Lists all the test cases related to Availability Patch and other negative scenarios.
- **NSSF_NsSelection_REGISTRATION_CASES -** Lists all the test cases related to NsSelection registration.
- **NSSF_NsSelection_PDU_CASES -** Lists all the test cases related to NsSelection PDU related cases.
- **NSSF_BASIC_SUBSCRIBE_CASES -** Lists all the test cases related to subscription.

You can click any functionality to view its test cases and scenarios of each test case. A sample screen is given below:

Figure 3-29 NSSF_BASIC_SANITY_CASES



ack to #17 pages

NSSF_BASIC_SANITY_CASES

Sanity.feature

Description :

This feature file validates successful scenarios for all the basic operations performed by NSSF

Scenario-1 : NSSAI Availability information of the NF(AMF) using PUT method.

Objective : Validate OCNSSF to create the NSSAI Availability information of the NF(AMF) using PUT method.

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth
2.) Perform PUT Update operation of NSSAI Availability request to NSSF with two S-NSSAI having same SST and SD vals	2.) Verify successful response 200 with content

Scenario-2 : NSSAI Availability information of the NF(AMF) using PUT method multiple times.

Objective : Validate OCNSSF to create the NSSAI Availability information of the NF(AMF) using PUT method.

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth
2.) Perform PUT Update operation of NSSAI Availability request to NSSF	2.) Verify successful response 200 with content
3.) Again perform PUT Update operation with different content of NSSAI Availability request to NSSF	3.) Verify successful response 200 with content

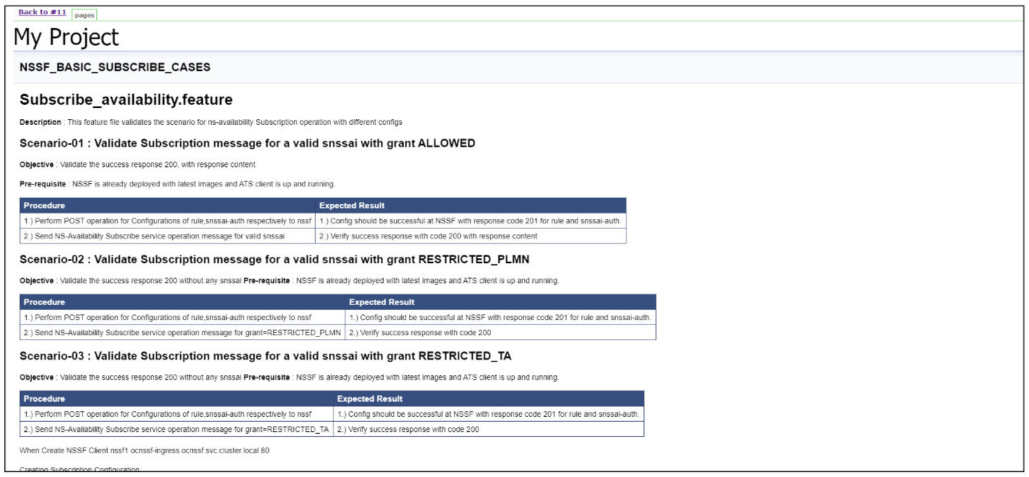
Scenario-3 : Validate NSSF supports PATCH request with add operation on NS-Availability update and reponds with Authenticatated NSSAI

Objective : Validate the successful response 200 with AuthorizedNssaiAvailabilityInfo for PATCH Request

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth

Figure 3-30 NSSF_BASIC_SUBSCRIBE_CASES



Back to #11 pages

My Project

NSSF_BASIC_SUBSCRIBE_CASES

Subscribe_availability.feature

Description : This feature file validates the scenario for ns-availability Subscription operation with different configs

Scenario-01 : Validate Subscription message for a valid snssai with grant ALLOWED

Objective : Validate the success response 200, with response content

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth.
2.) Send NS-Availability Subscribe service operation message for valid snssai	2.) Verify success response with code 200 with response content

Scenario-02 : Validate Subscription message for a valid snssai with grant RESTRICTED_PLMN

Objective : Validate the success response 200 without any snssai **Pre-requisite** : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth.
2.) Send NS-Availability Subscribe service operation message for grant=RESTRICTED_PLMN	2.) Verify success response with code 200

Scenario-03 : Validate Subscription message for a valid snssai with grant RESTRICTED_TA

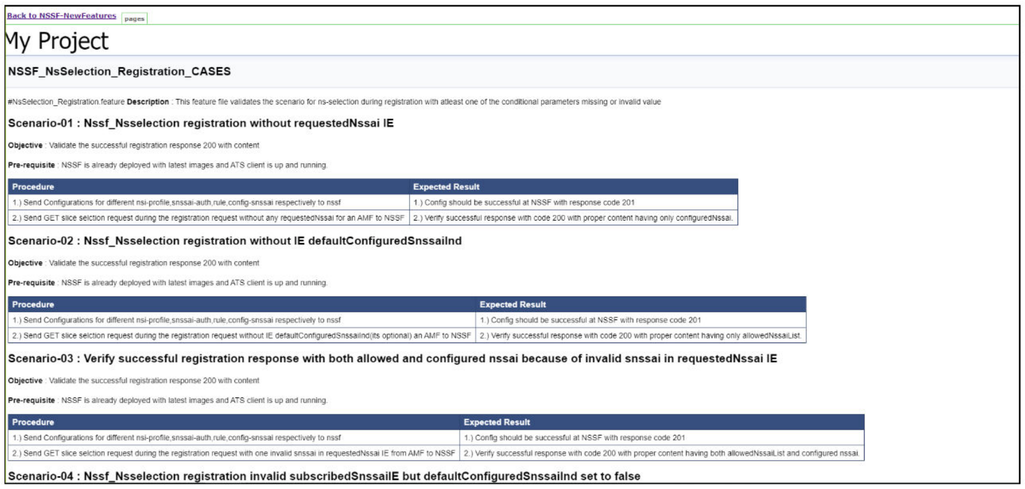
Objective : Validate the success response 200 without any snssai **Pre-requisite** : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Perform POST operation for Configurations of rule,snssai-auth respectively to nssf	1.) Config should be successful at NSSF with response code 201 for rule and snssai-auth.
2.) Send NS-Availability Subscribe service operation message for grant=RESTRICTED_TA	2.) Verify success response with code 200

When Create NSSF Client nssf1 ocnssf-ingress.ocnssf.svc.cluster.local 80

Creating Subscription Configuration

Figure 3-31   NSSF_NsSelection_Registration_CASES



Back to NSSF-NewFeatures pages

My Project

NSSF_NsSelection_Registration_CASES

#NsSelection_Registration.feature **Description** : This feature file validates the scenario for ns-selection during registration with atleast one of the conditional parameters missing or invalid value

Scenario-01 : Nssf_Nsselection registration without requestedNssai IE

Objective : Validate the successful registration response 200 with content

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send Configurations for different nsi-profile,snssai-auth,rule,config-snssai respectively to nssf	1.) Config should be successful at NSSF with response code 201
2.) Send GET slice selction request during the registration request without any requestedNssai for an AMF to NSSF	2.) Verify successful response with code 200 with proper content having only configuredNssai.

Scenario-02 : Nssf_Nsselection registration without IE defaultConfiguredSnssaiInd

Objective : Validate the successful registration response 200 with content

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send Configurations for different nsi-profile,snssai-auth,rule,config-snssai respectively to nssf	1.) Config should be successful at NSSF with response code 201
2.) Send GET slice selction request during the registration request without IE defaultConfiguredSnssaiInd(its optional) an AMF to NSSF	2.) Verify successful response with code 200 with proper content having only allowedNssaiList.

Scenario-03 : Verify successful registration response with both allowed and configured nssai because of invalid snssai in requestedNssai IE

Objective : Validate the successful registration response 200 with content

Pre-requisite : NSSF is already deployed with latest images and ATS client is up and running.

Procedure	Expected Result
1.) Send Configurations for different nsi-profile,snssai-auth,rule,config-snssai respectively to nssf	1.) Config should be successful at NSSF with response code 201
2.) Send GET slice selction request during the registration request with one invalid snssai in requestedNssai IE from AMF to NSSF	2.) Verify successful response with code 200 with proper content having both allowedNssaiList and configured nssai.

Scenario-04 : Nssf_Nsselection registration invalid subscribedSnssaiIE but defaultConfiguredSnssaiInd set to false

Executing Policy Test Cases using ATS

This ATS-Policy release is a converged release comprising of scenarios (test cases) from PCF, PCRF and Converged. ATS 1.2.2 is compatible with Policy 1.7.4 with TLS Enabled (server side) and Disabled Mode, CNPCRF and converged policy.

To execute Policy Test Cases, you need to ensure that following prerequisites are fulfilled.

Prerequisites

- Deploy OCCNP.
- Install Go-STUB for PCF and Converged Policy.
- **PCF**
  - **PCF with TLS not available:** In the PCF's custom values file, check if the following parameters are configured with the respective values:

    ```
    ingress-gateway:
                     enableIncomingHttps: false
              egress-gateway:
                     enableOutgoingHttps: false
    ```

  - **PCF with TLS Enabled:** In the PCF's custom values file, check if the following parameters are configured with the respective values:

    ```
     ingress-gateway:
                     enableIncomingHttps: true
      egress-gateway:
                     enableOutgoingHttps: true/false
    ```

    You also need to ensure that PCF is deployed with corresponding certificates.

    This scenario has two options:

      * **Client without TLS Enabled:** In this case, PCF is deployed with TLS enabled without generating any certificate in the ATS pod.
      * **Client with TLS Security Enabled:** In this case, PCF and ATS both have required certificates. For more details, refer to the **Enabling Https support for Egress and Ingress Gateway** section in this topic.
    – In the **-application-config** configmap, configure the following parameters with the respective values:
      * primaryNrfApiRoot=http://nf1stub.<namespace_gostubs_are_deployed_in>.svc:8080
        Example: primaryNrfApiRoot=http://nf1stub.ocats.svc:8080
      * nrfClientSubscribeTypes=UDR,CHF
      * supportedDataSetId=POLICY (Comment out the supportedDataSetId)

> **Note:**
>
> You can configure these values at the time of Policy deployment also.

> **Note:**
>
> Execute the following command to get all configmaps in your namespace.
> `kubectl get configmaps -n <Policy_namespace>`

- **CN-PCRF**
  Execute the following command to set the Log level to Debug in Diam-GW POD:

  ```
  kubectl edit statefulset <diam-gw pod name> -n <namespace>
  ```
- **Converged Policy:** It is same as PCF. You can refer to PCF explanation given above.
- Prometheus server should be installed in cluster.
- Database cluster should be in a running state with all the required tables. You need to ensure that there are no previous entries in database before executing test cases.
- ATS should be deployed in same namespace as Policy using Helm Charts.
- User **MUST NOT** initiate a job in two different pipelines at the same time.
- If you enable Service Mesh check, then you need to create a destination rule for fetching the metrics from the Prometheus. In most of the deployments, Prometheus is kept outside the service mesh so you need a destination rule to communicate between TLS enabled entity (ATS) and non-TLS entity (Prometheus). You can create a destination rule as follows:

  ```
  kubectl apply -f - <<EOF

  apiVersion:networking.istio.io/v1alpha3
  kind:DestinationRule
  ```

```
metadata:
  name:prometheus-dr
  namespace:ocats
spec:
  host:oso-prometheus-server.pcf.svc.cluster.local
  trafficPolicy:
    tls:
      mode:DISABLE
EOF
```

In the destination rule:

- name indicates the name of destination rule.
- namespace indicates where the ATS is deployed.
- host indicates the hostname of the prometheus server.

Enabling TLS in ATS Pod

You can enable TLS in ATS pod after successful deployment of PCF (TLS enabled server side) and ATS. To enable TLS in ATS Pod:

1. Execute the following command to copy the caroot.cer generated while PCF deployment to ATS pod in "cert" directory.

   ```
   kubectl cp  <path_to_file>/caroot.cer <namespace>/<ATS-Pod-name>:
   /var/lib/jenkins/cert/ -n <namespace>
   ```

   Example:

   ```
   kubcetl cp cert/caroot.cer ocpcf/ocpcf-ocats-pcf-56754b9568-rkj8z:
   /var/lib/jenkins/cert/
   ```

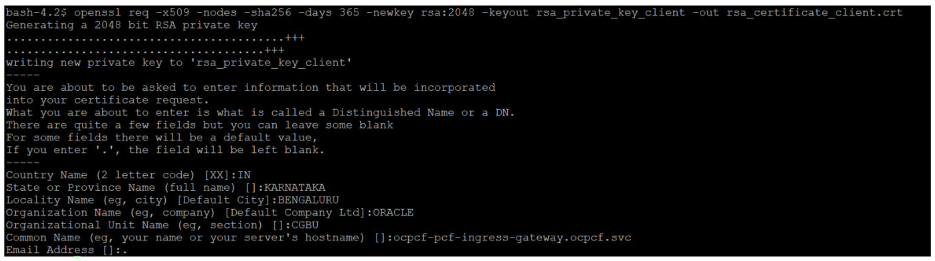
2. Execute the following command to login to your ATS Pod.

   ```
    kubectl exec -it <ATS-Pod-name> bash -n <namespace>
   ```

3. Execute the following commands from cert directory to create private key and certificates:

   a. ```
      openssl req -x509 -nodes -sha256 -days 365 -newkey rsa:2048
      -keyout
      rsa_private_key_client -out rsa_certificate_client.crt
      ```

      **Figure 3-32    Command 1**

      ```
      bash-4.2$ openssl req -x509 -nodes -sha256 -days 365 -newkey rsa:2048 -keyout rsa_private_key_client -out rsa_certificate_client.crt
      Generating a 2048 bit RSA private key
      ....+++
      ....+++
      writing new private key to 'rsa_private_key_client'

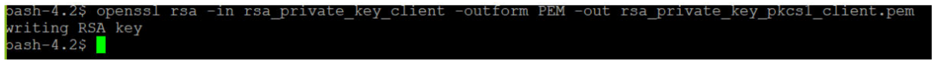
      You are about to be asked to enter information that will be incorporated
      into your certificate request.
      What you are about to enter is what is called a Distinguished Name or a DN.
      There are quite a few fields but you can leave some blank
      For some fields there will be a default value,
      If you enter '.', the field will be left blank.

      Country Name (2 letter code) [XX]:IN
      State or Province Name (full name) []:KARNATAKA
      Locality Name (eg, city) [Default City]:BENGALURU
      Organization Name (eg, company) [Default Company Ltd]:ORACLE
      Organizational Unit Name (eg, section) []:CGBU
      Common Name (eg, your name or your server's hostname) []:ocpcf-pcf-ingress-gateway.ocpcf.svc
      Email Address []:.
      ```
      ```

> **Note:**
>
> You need to provide appropriate values and specify fqdn of PCF Ingress Gateway service i.e. <ingress-servicename>.<pcf_namespace>.svc in Common Name.

b.
```
openssl rsa -in rsa_private_key_client -outform PEM -out
      rsa_private_key_pkcs1_client.pem
```

Figure 3-33 Command 2

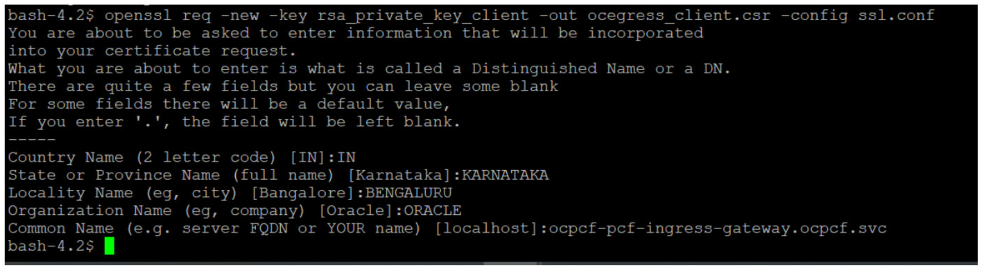


```
bash-4.2$ openssl rsa -in rsa_private_key_client -outform PEM -out rsa_private_key_pkcs1_client.pem
writing RSA key
bash-4.2$
```

c.
```
openssl req -new -key rsa_private_key_client -out
ocegress_client.csr -config
ssl.conf
```

> **Note:**
>
> You can either use or copy the ssl.conf file, which was used while deploying PCF to ATS pod for this step.

Figure 3-34 Command 3

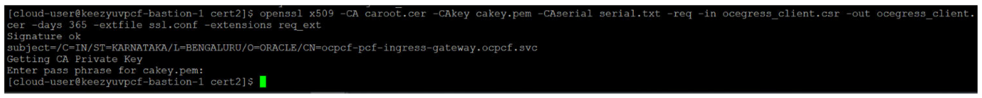


```
bash-4.2$ openssl req -new -key rsa_private_key_client -out ocegress_client.csr -config ssl.conf
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [IN]:IN
State or Province Name (full name) [Karnataka]:KARNATAKA
Locality Name (eg, city) [Bangalore]:BENGALURU
Organization Name (eg, company) [Oracle]:ORACLE
Common Name (e.g. server FQDN or YOUR name) [localhost]:ocpcf-pcf-ingress-gateway.ocpcf.svc
bash-4.2$
```

4. Execute the following command to copy the ocegress_client.csr to the bastion.

```
openssl x509 -CA caroot.cer -CAkey cakey.pem -CAserial serial.txt
-req
      -in ocegress_client.csr -out ocegress_client.cer -days 365
-extfile
    ssl.conf -extensions req_ext
```

Figure 3-35 Copying ocegress_client.csr to bastion



```
[cloud-user@keezyuvpcf-bastion-1 cert2]$ openssl x509 -CA caroot.cer -CAkey cakey.pem -CAserial serial.txt -req -in ocegress_client.csr -out ocegress_client.
cer -days 365 -extfile ssl.conf -extensions req_ext
Signature ok
subject=/C=IN/ST=KARNATAKA/L=BENGALURU/O=ORACLE/CN=ocpcf-pcf-ingress-gateway.ocpcf.svc
Getting CA Private Key
Enter pass phrase for cakey.pem:
[cloud-user@keezyuvpcf-bastion-1 cert2]$
```

5. Copy the **ocegress_client.cer** from Bastion to the ATS Pod.
6. Restart the ingress and egress gateway pods from the Bastion.

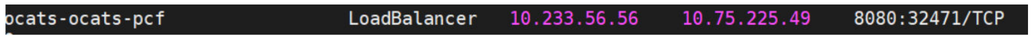
Logging into ATS

Before logging into ATS Jenkins GUI, it is important to get the nodeport of the service, '-ocats-Policy'. Execute the following command to get the nodeport:

```
kubectl get svc -n <Policy_namespace>
```

Example: `kubectl get svc -n ocats`

Figure 3-36 Policy Nodeport



```
ocats-ocats-pcf                LoadBalancer   10.233.56.56     10.75.225.49     8080:32471/TCP
```

To login to Jenkins, open the Web Browser and type the URL: http://<Worker-Node-IP>:<Node-Port-of-ATS>. In the above screen, 32471 is the nodeport. **Example:** http://10.75.225.49:32471

Note:

For more information on ATS deployment in PCF, refer to Policy ATS Installation Procedure.

Executing ATS

To execute ATS:

1. Enter the **username** as "policyuser" and **password** as "policypasswd". Click **Sign in**.

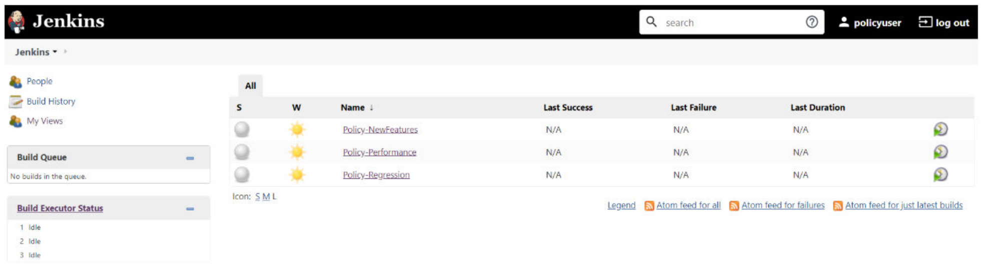
   Note:

   If you want to modify your default login password, refer to Modifying Login Password

   The following screen appears showing policy pre-configured pipelines:

   - **Policy-NewFeatures:** This pipeline has all the test cases, which are delivered as part of Policy ATS - 1.7.4
   - **Policy-Performance:** This pipeline is not operational as of now. It is reserved for future releases of ATS.
   - **Policy-Regression:** This pipleine has all the test cases, which were delivered in Policy ATS - 1.7.0

Figure 3-37 Pre-Configured Pipelines



S	W	Name ↓	Last Success	Last Failure	Last Duration
		Policy-NewFeatures	N/A	N/A	N/A
		Policy-Performance	N/A	N/A	N/A
		Policy-Regression	N/A	N/A	N/A

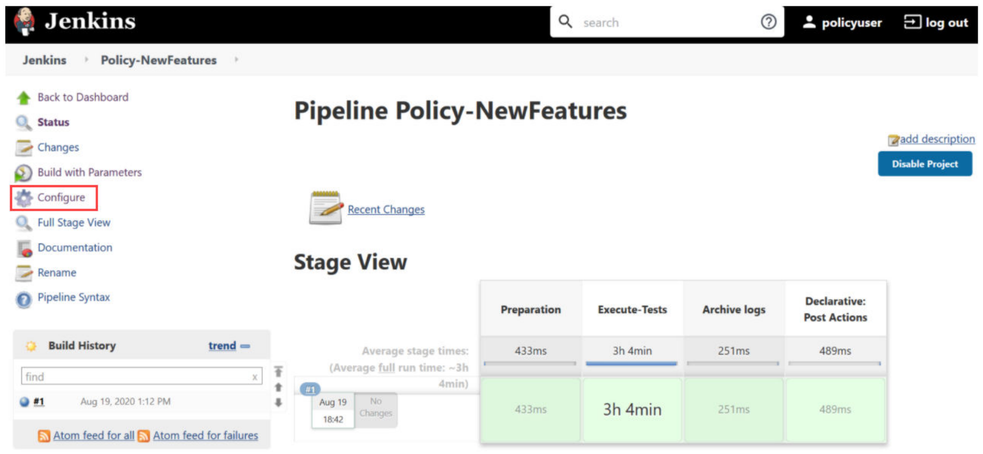
The pre-configured pipelines are explained below:

Policy-New Features Pipeline

This is a pre-configured pipeline where all the Policy test cases are executed. To configure its parameters, which is a one time activity:

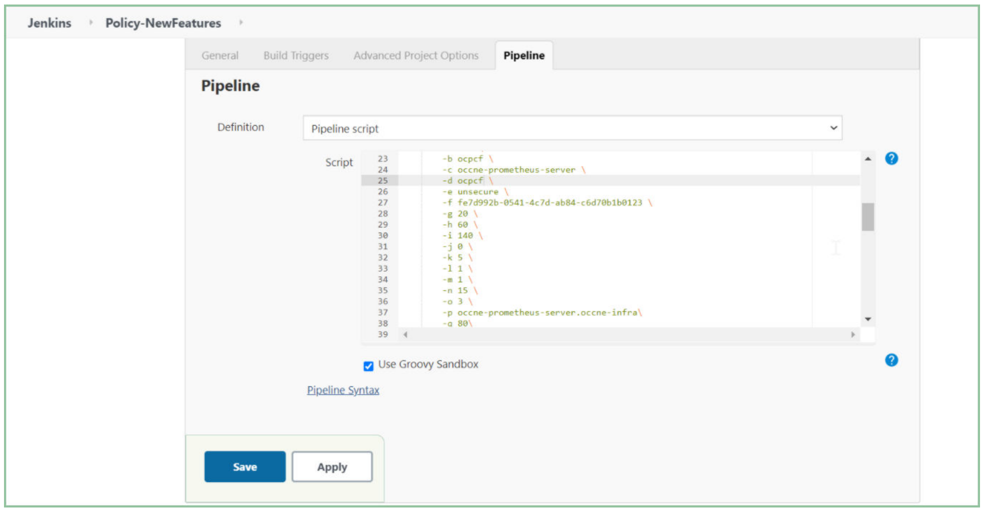
1. Click **Policy-NewFeatures** in the Name column and then, click **Configure** in the left navigation pane as shown below:

Figure 3-38 Policy-NewFeatures Configure



2. The Policy-NewFeatures, **General** tab appears. Make sure that the screen loads completely.
3. Scroll-down to the end. The control moves from **General** tab to the **Pipeline** tab as shown below:

Figure 3-39   Policy - Pipeline Script



```
        -b ocpcf \
        -c occne-prometheus-server \
        -d ocpcf \
        -e unsecure \
        -f fe7d992b-0541-4c7d-ab84-c6d70b1b0123 \
        -g 20 \
        -h 60 \
        -i 140 \
        -j 0 \
        -k 5 \
        -l 1 \
        -m 1 \
        -n 15 \
        -o 3 \
        -p occne-prometheus-server.occne-infra\
        -q 80\
```

In the **Script** area of the Pipeline section, you can change value of the following parameters:

- **b:** Change this parameter to update the namespace where Policy was deployed in your bastion.
- **d:** Change this parameter to update the namespace where your gostubs are deployed in your bastion.
- **e:** Set this parameter as 'unsecure', if you intend to run ATS in TLS disabled mode. Else, set this parameter as 'secure'.
- **g:** Set this parameter to more than 30 secs. The default wait time for the pod is 30 secs. Every TC requires restart of the nrf-client-management pod.
- **h:** Set this parameter to more than 60 secs. The default wait time to add a configured policy to the database is 60 secs.
- **i:** Set this parameter to more than 140 secs. The default wait time for Nf_Notification Test Cases is given as 140 secs.
- **k:** Use this parameter to set the waiting time to initialize Test Suite 8.
- **l:** Use this parameter to set the waiting time to get response from Stub 9.
- **m:** Use this parameter to set the waiting time after adding Policy Configuration 10.
- **n:** Use this parameter to set the waiting time after adding Policy 11.
- **o:** Use this parameter to set the waiting time before sending next message 12.
- **p:** Use this parameter to set Prometheus Server IP 13.
- **q:** Use this parameter to set Prometheus Server Port.

Note:

DO NOT MODIFY ANYTHING OTHER THAN THESE PARAMETER VALUES.

- Click **Save** after updating the parameters value. The Policy-NewFeatures Pipeline screen appears.

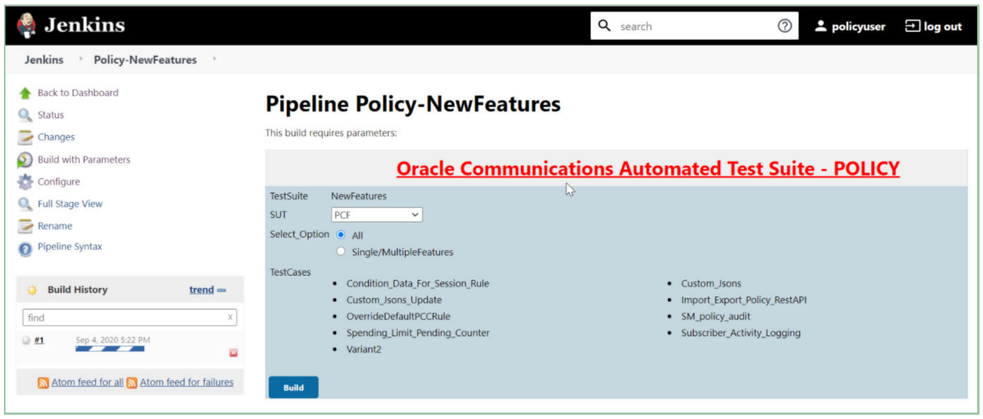
> **Note:**
>
> It is advisable to save the pipeline script in your local machine that you can refer at the time of ATS pod restart.

Executing Policy Test Cases

To execute Policy test cases:

1. Click the **Build with Parameters** link available in the left navigation pane of the Policy-NewFeatures Pipeline screen. The following screen appears.

Figure 3-40 Policy - Build with Parameters



Pipeline Policy-NewFeatures

This build requires parameters:

Oracle Communications Automated Test Suite - POLICY

TestSuite NewFeatures

SUT PCF

Select_Option [✓ All] Single/MultipleFeatures

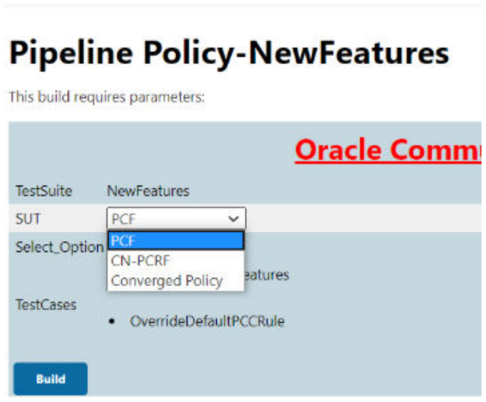
TestCases

- Condition_Data_For_Session_Rule
- Custom_Jsons_Update
- OverrideDefaultPCCRule
- Spending_Limit_Pending_Counter
- Variant2
- Custom_Jsons
- Import_Export_Policy_RestAPI
- SM_policy_audit
- Subscriber_Activity_Logging

In the above screen, you can select SUT as either PCF, CN-PCRF or Converged Policy. It also has two **Select_Option**(s), which are:

- **All:** By default, all the Policy test cases are selected for execution. User just need to scroll down and click **Build** to execute all the test cases.
- **Single/MultipleFeatures:** This option allows you to select any number of test cases that you want to execute from the list of total test cases available for execution. After selecting the test cases, scroll-down and click **Build**. The selected Policy test cases are executed.

Figure 3-41 SUT Options


Pipeline Policy-NewFeatures

This build requires parameters:

Oracle Comm

TestSuite NewFeatures

SUT PCF

Select_Option

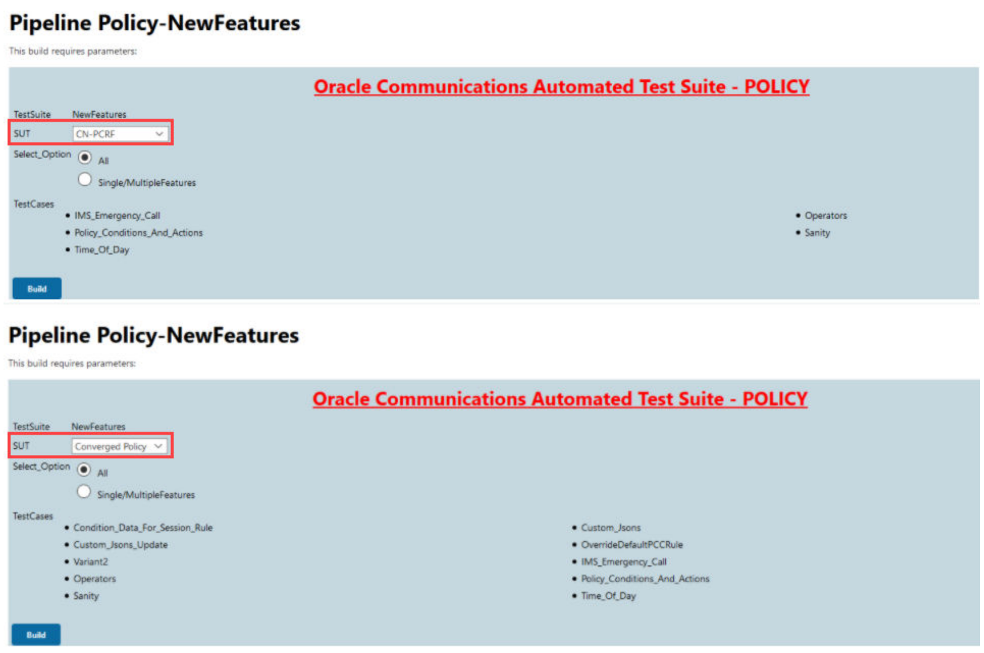
- PCF
- CN-PCRF
- Converged Policy

TestCases

- OverrideDefaultPCCRule

Based on your selection, related test cases appear.

Figure 3-42 Test Cases based on SUT


Pipeline Policy-NewFeatures

This build requires parameters:

Oracle Communications Automated Test Suite - POLICY

TestSuite NewFeatures

SUT CN-PCRF

Select_Option [✓ All] Single/MultipleFeatures

TestCases

- IMS_Emergency_Call
- Policy_Conditions_And_Actions
- Time_Of_Day
- Operators
- Sanity

Pipeline Policy-NewFeatures

This build requires parameters:

Oracle Communications Automated Test Suite - POLICY

TestSuite NewFeatures

SUT Converged Policy

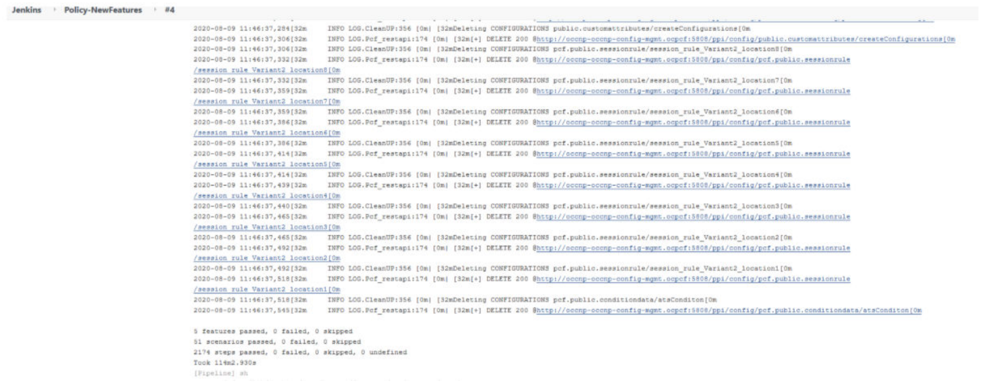
Select_Option [✓ All] Single/MultipleFeatures

TestCases

- Condition_Data_For_Session_Rule
- Custom_Jsons_Update
- Variant2
- Operators
- Sanity
- Custom_Jsons
- OverrideDefaultPCCRule
- IMS_Emergency_Call
- Policy_Conditions_And_Actions
- Time_Of_Day

Go to **Build** → **Console Output** to view the test result output as shown below:

Figure 3-43   Sample: Test Result Output in Console

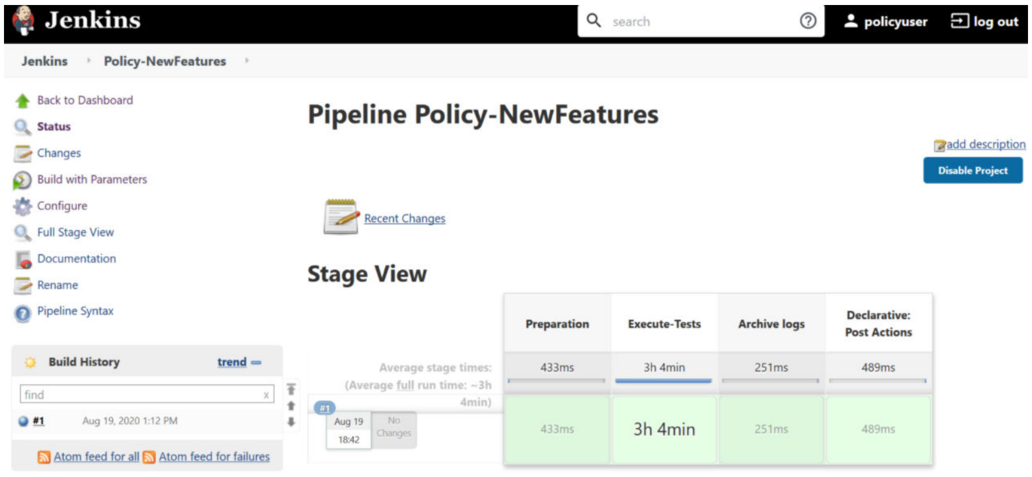


Jenkins › Policy-NewFeatures › #4

```
2020-08-09 11:46:37,284[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS public.customattributes/createConfigurations[0m
2020-08-09 11:46:37,306[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/public.customattributes/createConfigurations[0m
2020-08-09 11:46:37,306[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location8[0m
2020-08-09 11:46:37,332[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location8[0m
2020-08-09 11:46:37,332[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location7[0m
2020-08-09 11:46:37,359[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location7[0m
2020-08-09 11:46:37,359[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location6[0m
2020-08-09 11:46:37,386[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location6[0m
2020-08-09 11:46:37,386[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location5[0m
2020-08-09 11:46:37,414[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location5[0m
2020-08-09 11:46:37,414[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location4[0m
2020-08-09 11:46:37,439[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location4[0m
2020-08-09 11:46:37,440[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location3[0m
2020-08-09 11:46:37,465[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location3[0m
2020-08-09 11:46:37,465[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location2[0m
2020-08-09 11:46:37,492[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location2[0m
2020-08-09 11:46:37,492[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.sessionrule/session_rule_Variant2_location1[0m
2020-08-09 11:46:37,518[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.sessionrule
/session rule Variant2 location1[0m
2020-08-09 11:46:37,518[32m     INFO LOG.CleanUP:356 [0m| [32mDeleting CONFIGURATIONS pcf.public.conditiondata/atsConditon[0m
2020-08-09 11:46:37,545[32m     INFO LOG.Pcf_restapi:174 [0m| [32m[+] DELETE 200 @http://occnp-occnp-config-mgmt.ocpcf:5808/ppi/config/pcf.public.conditiondata/atsConditon[0m

5 features passed, 0 failed, 0 skipped
51 scenarios passed, 0 failed, 0 skipped
2174 steps passed, 0 failed, 0 skipped, 0 undefined
Took 114m2.930s
[Pipeline] sh
```

Figure 3-44   Sample Output of Build Status



Pipeline Policy-NewFeatures

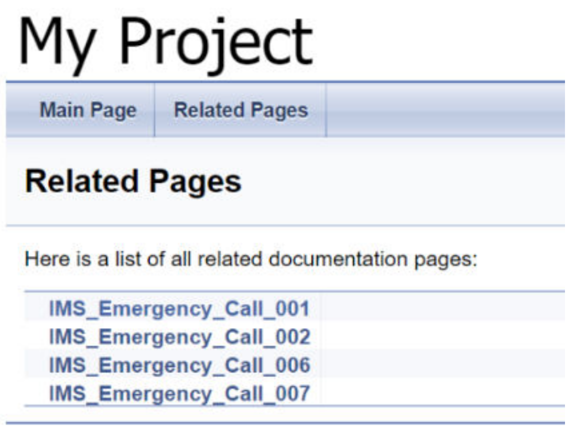
Stage View

	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times: (Average full run time: ~3h 4min)	433ms	3h 4min	251ms	489ms
#1 Aug 19 18:42 No Changes	433ms	3h 4min	251ms	489ms

NewFeatures - Documentation

To view Policy functionalities, go to Policy-NewFeatures pipeline and click **Documentation** link in the left navigation pane.

Figure 3-45   Policy-NewFeatures Feature List



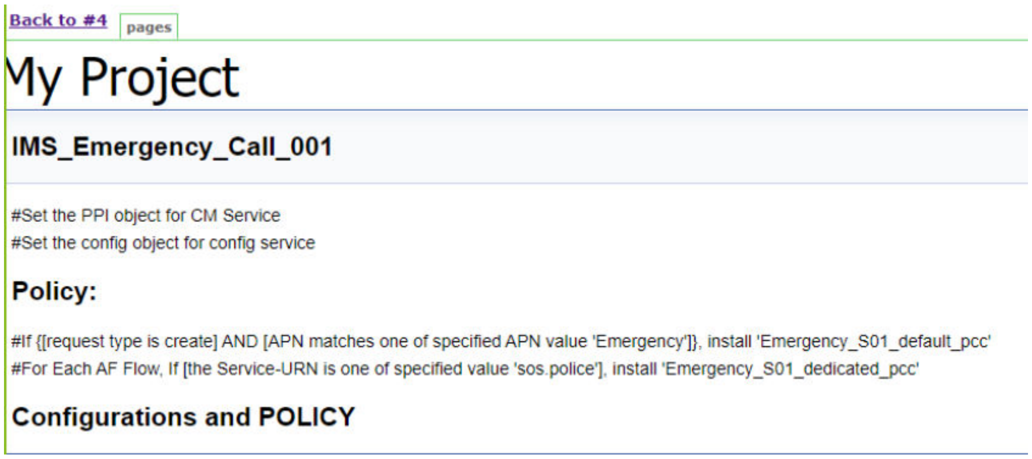
Related Pages

Here is a list of all related documentation pages:

- IMS_Emergency_Call_001
- IMS_Emergency_Call_002
- IMS_Emergency_Call_006
- IMS_Emergency_Call_007

You can click any functionality to view its test cases and scenarios of each test case. For example, on click of IMS_Emergency_Call_001, the following test description appears:

Figure 3-46 IMS_Emergency_Call_001



My Project

IMS_Emergency_Call_001

#Set the PPI object for CM Service
#Set the config object for config service

Policy:

#If {[request type is create] AND [APN matches one of specified APN value 'Emergency']}, install 'Emergency_S01_default_pcc'
#For Each AF Flow, If [the Service-URN is one of specified value 'sos.police'], install 'Emergency_S01_dedicated_pcc'

Configurations and POLICY

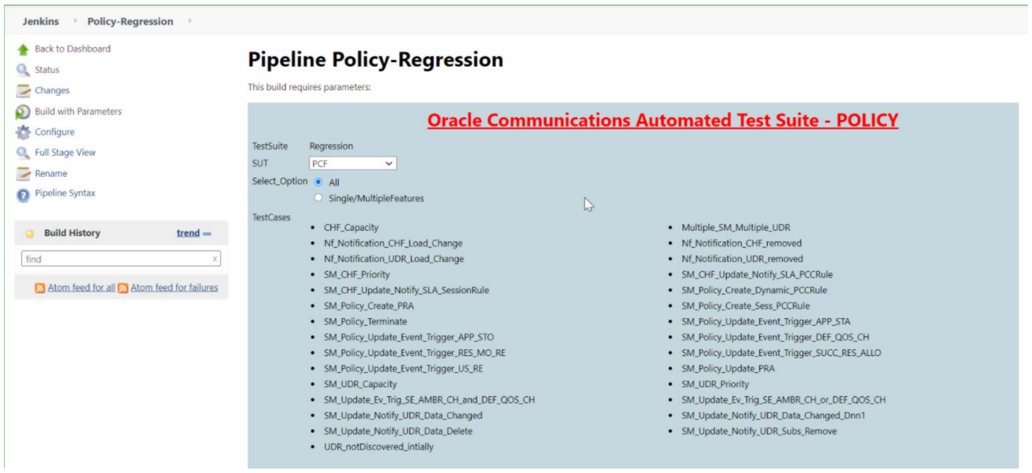
Based on the functionalities covered under Documentation, the **Build Requires Parameters** screen displays test cases. To navigate back to the Pipeline Policy-NewFeatures screen, click **Back to Policy-NewFeatures** link available on top left corner of the screen.

PCF-Regression Pipeline

This pre-configured pipeline has all the test cases of previous releases. For example, as part of Release 1.7.4, this pipeline has all the test cases that were released as part of release 1.7.0

To view Regression pipeline details, click **Build with Parameters** in the left navigation pane. The following screen appears:

Figure 3-47 Policy-Regression



Pipeline Policy-Regression

This build requires parameters:

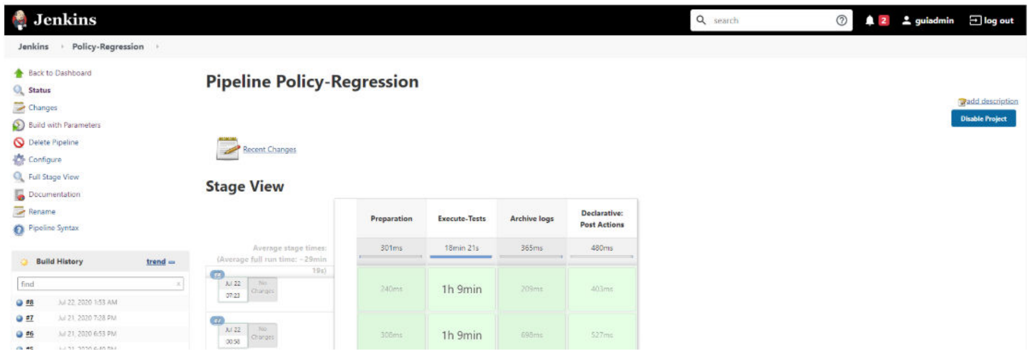
Oracle Communications Automated Test Suite - POLICY

TestSuite Regression
SUT PCF
Select_Option [✓ All] Single/MultipleFeatures

TestCases
- CHF_Capacity
- Nf_Notification_CHF_Load_Change
- Nf_Notification_UDR_Load_Change
- SM_CHF_Priority
- SM_CHF_Update_Notify_SLA_SessionRule
- SM_Policy_Create_PRA
- SM_Policy_Terminate
- SM_Policy_Update_Event_Trigger_APP_STO
- SM_Policy_Update_Event_Trigger_RES_MO_RE
- SM_Policy_Update_Event_Trigger_US_RE
- SM_UDR_Capacity
- SM_Update_Ev_Trig_SE_AMBR_CH_and_DEF_QOS_CH
- SM_Update_Notify_UDR_Data_Changed
- SM_Update_Notify_UDR_Data_Delete
- UDR_notDiscovered_intially
- Multiple_SM_Multiple_UDR
- Nf_Notification_CHF_removed
- Nf_Notification_UDR_removed
- SM_CHF_Update_Notify_SLA_PCCRule
- SM_Policy_Create_Dynamic_PCCRule
- SM_Policy_Create_Sess_PCCRule
- SM_Policy_Update_Event_Trigger_APP_STA
- SM_Policy_Update_Event_Trigger_DEF_QOS_CH
- SM_Policy_Update_Event_Trigger_SUCC_RES_ALLO
- SM_Policy_Update_PRA
- SM_UDR_Priority
- SM_Update_Ev_Trig_SE_AMBR_CH_or_DEF_QOS_CH
- SM_Update_Notify_UDR_Data_Changed_Dnn1
- SM_Update_Notify_UDR_Subs_Remove

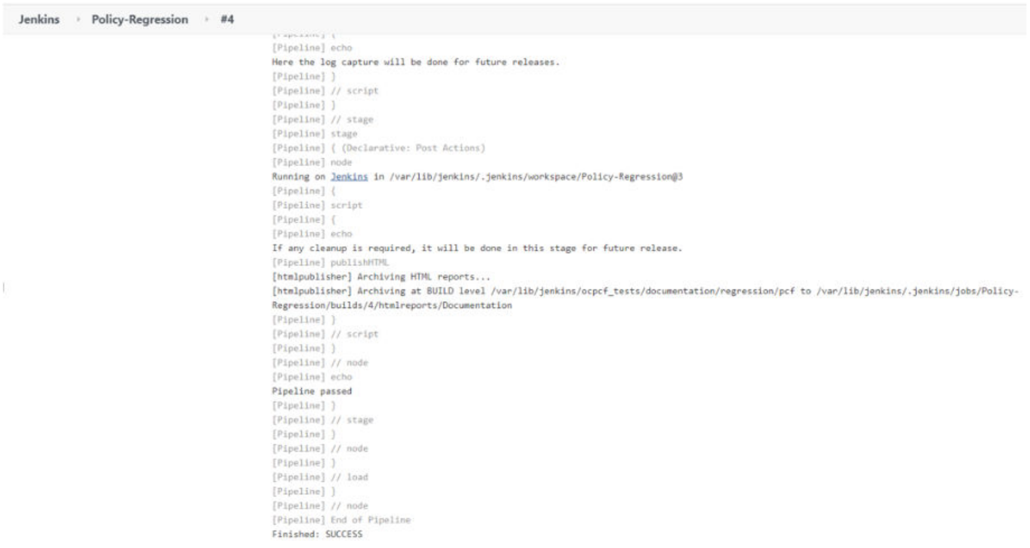
Click **Build**. The build output appears as shown below:

Figure 3-48   Policy-Regression Build Output



The console output is as follows:

Figure 3-49   Policy-Regression Console Output


Jenkins › Policy-Regression › #4

```
[Pipeline] echo
Here the log capture will be done for future releases.
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/.jenkins/workspace/Policy-Regression@3
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] echo
If any cleanup is required, it will be done in this stage for future release.
[Pipeline] publishHTML
[htmlpublisher] Archiving HTML reports...
[htmlpublisher] Archiving at BUILD level /var/lib/jenkins/ocpcf_tests/documentation/regression/pcf to /var/lib/jenkins/.jenkins/jobs/Policy-
Regression/builds/4/htmlreports/Documentation
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // node
[Pipeline] echo
Pipeline passed
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // load
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

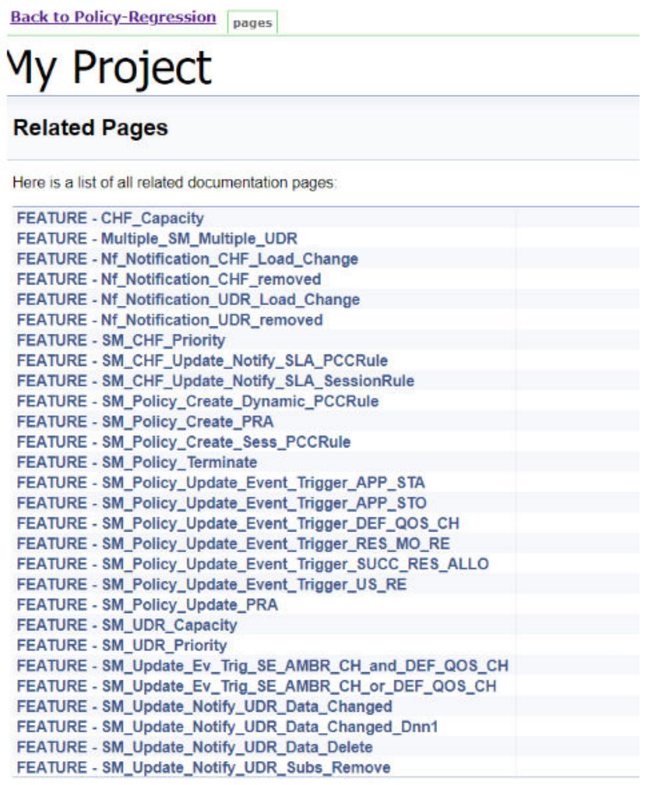
Note:

The regression pipeline does not have any sanity option. However, you should perform all the steps as performed in NewFeatures pipeline. Configure the pipeline script changes to provide environment variables.

Regression - Documentation

Click **Documentation** in the left navigation pane of the Policy-Regression pipeline. Following screen appears:

Figure 3-50   Policy-Regression Documentation



Back to Policy-Regression pages

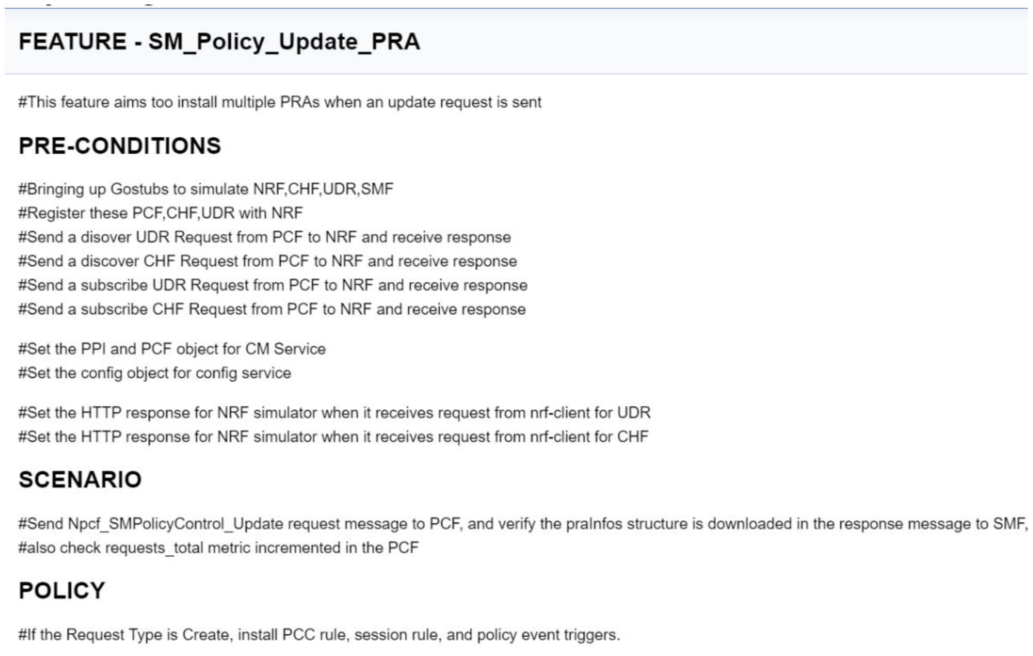
My Project

Related Pages

Here is a list of all related documentation pages:

FEATURE - CHF_Capacity
FEATURE - Multiple_SM_Multiple_UDR
FEATURE - Nf_Notification_CHF_Load_Change
FEATURE - Nf_Notification_CHF_removed
FEATURE - Nf_Notification_UDR_Load_Change
FEATURE - Nf_Notification_UDR_removed
FEATURE - SM_CHF_Priority
FEATURE - SM_CHF_Update_Notify_SLA_PCCRule
FEATURE - SM_CHF_Update_Notify_SLA_SessionRule
FEATURE - SM_Policy_Create_Dynamic_PCCRule
FEATURE - SM_Policy_Create_PRA
FEATURE - SM_Policy_Create_Sess_PCCRule
FEATURE - SM_Policy_Terminate
FEATURE - SM_Policy_Update_Event_Trigger_APP_STA
FEATURE - SM_Policy_Update_Event_Trigger_APP_STO
FEATURE - SM_Policy_Update_Event_Trigger_DEF_QOS_CH
FEATURE - SM_Policy_Update_Event_Trigger_RES_MO_RE
FEATURE - SM_Policy_Update_Event_Trigger_SUCC_RES_ALLO
FEATURE - SM_Policy_Update_Event_Trigger_US_RE
FEATURE - SM_Policy_Update_PRA
FEATURE - SM_UDR_Capacity
FEATURE - SM_UDR_Priority
FEATURE - SM_Update_Ev_Trig_SE_AMBR_CH_and_DEF_QOS_CH
FEATURE - SM_Update_Ev_Trig_SE_AMBR_CH_or_DEF_QOS_CH
FEATURE - SM_Update_Notify_UDR_Data_Changed
FEATURE - SM_Update_Notify_UDR_Data_Changed_Dnn1
FEATURE - SM_Update_Notify_UDR_Data_Delete
FEATURE - SM_Update_Notify_UDR_Subs_Remove

Figure 3-51   Sample: Regression Documentation - Feature



FEATURE - SM_Policy_Update_PRA

#This feature aims too install multiple PRAs when an update request is sent

PRE-CONDITIONS

#Bringing up Gostubs to simulate NRF,CHF,UDR,SMF
#Register these PCF,CHF,UDR with NRF
#Send a disover UDR Request from PCF to NRF and receive response
#Send a discover CHF Request from PCF to NRF and receive response
#Send a subscribe UDR Request from PCF to NRF and receive response
#Send a subscribe CHF Request from PCF to NRF and receive response

#Set the PPI and PCF object for CM Service
#Set the config object for config service

#Set the HTTP response for NRF simulator when it receives request from nrf-client for UDR
#Set the HTTP response for NRF simulator when it receives request from nrf-client for CHF

SCENARIO

#Send Npcf_SMPolicyControl_Update request message to PCF, and verify the praInfos structure is downloaded in the response message to SMF,
#also check requests_total metric incremented in the PCF

POLICY

#If the Request Type is Create, install PCC rule, session rule, and policy event triggers.

This screen shows only those functionalities whose test cases were released in previous releases.

Executing SCP Test Cases using ATS

To execute SCP Test Cases, you need to ensure that following prerequisites are fulfilled.

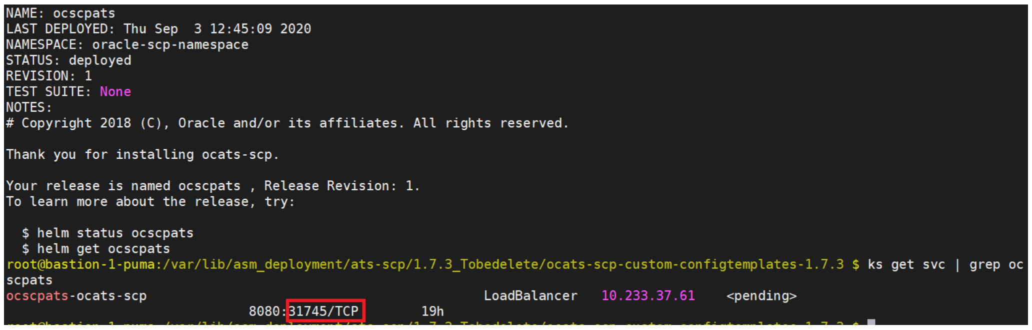
Prerequisites

- Deploy SCP 1.7.3 with following custom values in deployment file.
  - As you can provide NRF information only at time of deployment, Stub NRF details like nrf1svc and nrf2svc should also be provided at the time of deployment before executing these cases. **For Example:** If teststub namespace is scpsvc then SCP should have been deployed with primary nrf as *nrf1svc.scpsvc.svc.<clusterDomain>* and secondary nrf as *nrf2svc.scpsvc.svc.<clusterDomain>* for NRF test cases to work.
  - NRF details of SCP should specify port as 8080 in ipEndPoints. **Example:** ipEndPoints: [{"port": "8080"}]).
  - In the SCP deployment file, **servingScope** must have Reg1 and **servingLocalities** must have USEast and Loc9. In addition, the recommended auditInterval is 120 and guardTime is 10.
  - For ATS execution, you should deploy SCP with SCP-Worker replicas set to 1.
- Deploy ATS using helm charts.
- As you can deploy default ATS with role binding, it is important to deploy ATS and test stubs in the same namespace as SCP.

Logging into ATS

Before logging into ATS, you need to ensure that ATS is deployed successfully using HELM charts. A sample screen is given below:

Figure 3-52 Verifying ATS Deployment



```
NAME: ocscpats
LAST DEPLOYED: Thu Sep  3 12:45:09 2020
NAMESPACE: oracle-scp-namespace
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
# Copyright 2018 (C), Oracle and/or its affiliates. All rights reserved.

Thank you for installing ocats-scp.

Your release is named ocscpats , Release Revision: 1.
To learn more about the release, try:

  $ helm status ocscpats
  $ helm get ocscpats
root@bastion-1-puma:/var/lib/asm_deployment/ats-scp/1.7.3_Tobedelete/ocats-scp-custom-configtemplates-1.7.3 $ ks get svc | grep oc
scpats
ocscpats-ocats-scp                                       LoadBalancer   10.233.37.61    <pending>
                               8080:31745/TCP        19h
```

To login to ATS Jenkins GUI, open the browser and provide the external IP of the worker node and nodeport of the ATS service as `<Worker-Node-IP>:<Node-Port-of-ATS>`. The Jenkins login screen appears.

Note:

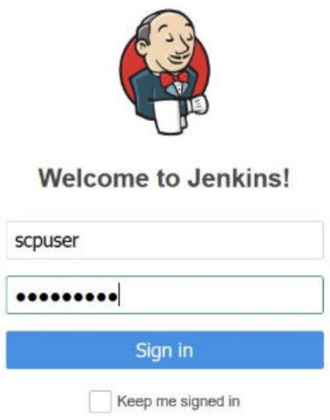
In the **Verifying ATS Deployment** screen, the ATS nodeport is highlighed in red as **31745**. For more details on ATS deployment, refer toSCP ATS Installation Procedure .

Executing ATS

To execute ATS:

1. Enter the **username** as "scpuser" and **password** as "scppasswd". Click **Sign in**. A sample screen is shown below.

   Figure 3-53   Logging into ATS GUI

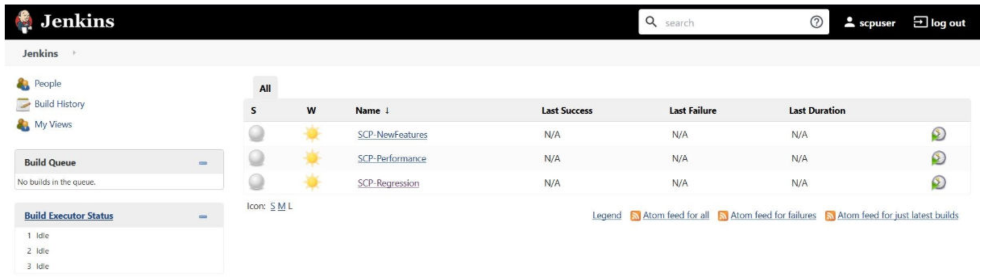
   

   Note:

   If you want to modify your default login password, refer to Modifying Login Password

2. Following screen appears showing pre-configured pipelines for SCP individually (3 Pipelines).
   - SCP-New-Features
   - SCP-Performance: This pipeline is not operational as of now. It is reserved for future releases of ATS.
   - SCP-Regression

Figure 3-54 ATS SCP First Logged-In Screen



Jenkins   search   scpuser   log out

Jenkins

People
Build History
My Views

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle
3 Idle

All

S	W	Name ↓	Last Success	Last Failure	Last Duration
		SCP-NewFeatures	N/A	N/A	N/A
		SCP-Performance	N/A	N/A	N/A
		SCP-Regression	N/A	N/A	N/A

Icon: S M L

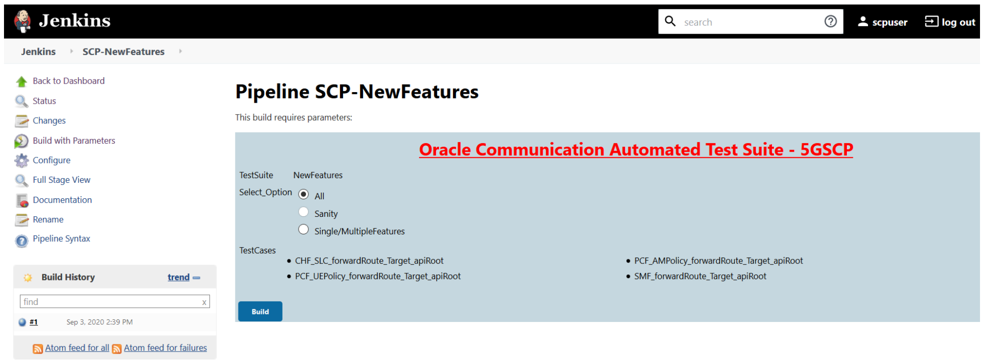
Legend Atom feed for all Atom feed for failures Atom feed for just latest builds

Pipeline SCP-NewFeatures

This is a pre-configured pipeline where all the SCP test cases are executed. To configure its parameters, which is a one time activity:

1. Click **SCP-NewFeatures** in the Name column. The following screen appears:

Figure 3-55 SCP-NewFeatures



Jenkins   search   scpuser   log out

Jenkins › SCP-NewFeatures ›

Back to Dashboard
Status
Changes
Build with Parameters
Configure
Full Stage View
Documentation
Rename
Pipeline Syntax

Build History   trend
find
#1   Sep 3, 2020 2:39 PM
Atom feed for all Atom feed for failures

Pipeline SCP-NewFeatures

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSCP

TestSuite   NewFeatures
Select_Option   (●) All   ( ) Sanity   ( ) Single/MultipleFeatures
TestCases
- CHF_SLC_forwardRoute_Target_apiRoot
- PCF_UEPolicy_forwardRoute_Target_apiRoot
- PCF_AMPolicy_forwardRoute_Target_apiRoot
- SMF_forwardRoute_Target_apiRoot

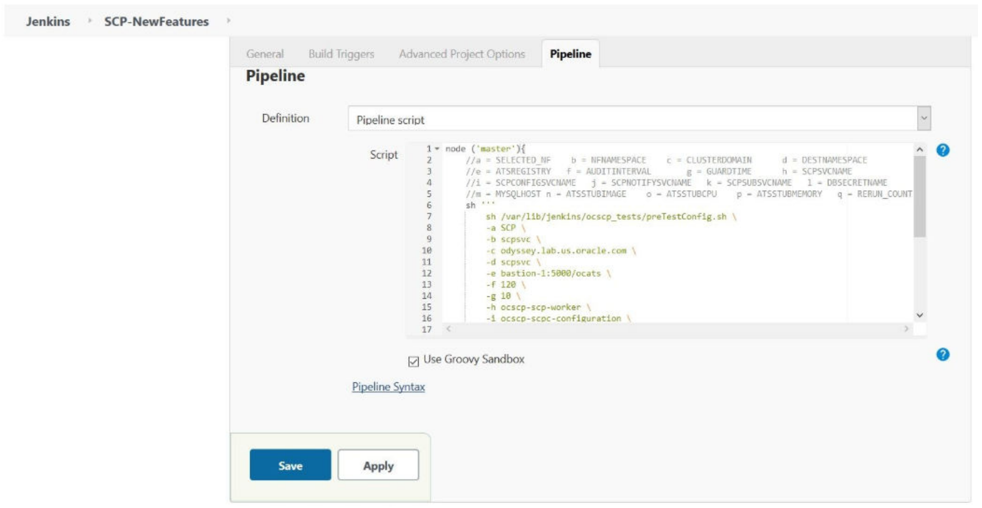
Build

2. Click **Configure** in the left navigation pane to provide input parameters. The SCP-NewFeatures Configure - General tab appears.

> **Note:**
>
> MAKE SURE THAT THE SCREEN SHOWN BELOW LOADS COMPLETELY BEFORE YOU PERFORM ANY ACTION ON IT. ALSO, DO NOT MODIFY ANY CONFIGURATION OTHER THAN DISCUSSED BELOW.

3. Scroll-down to the end. The control moves from **General** tab to the **Advanced Project Options** tab as shown below:

Figure 3-56   Advanced Project Options



Pipeline

Definition: Pipeline script

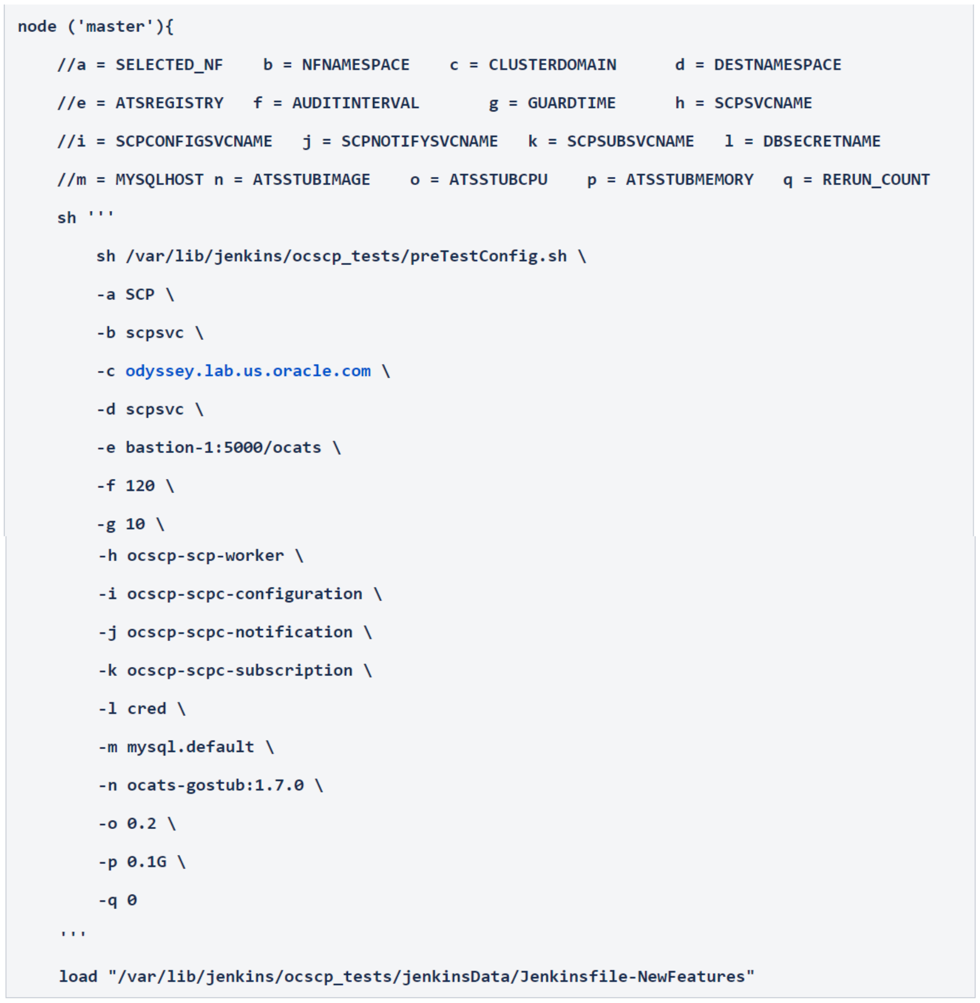
Script

```
node ('master'){
    //a = SELECTED_NF     b = NFNAMESPACE     c = CLUSTERDOMAIN      d = DESTNAMESPACE
    //e = ATSREGISTRY   f = AUDITINTERVAL      g = GUARDTIME       h = SCPSVCNAME
    //i = SCPCONFIGSVCNAME   j = SCPNOTIFYSVCNAME   k = SCPSUBSVCNAME   l = DBSECRETNAME
    //m = MYSQLHOST n = ATSSTUBIMAGE    o = ATSSTUBCPU    p = ATSSTUBMEMORY   q = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocscp_tests/preTestConfig.sh \
        -a SCP \
        -b scpsvc \
        -c odyssey.lab.us.oracle.com \
        -d scpsvc \
        -e bastion-1:5000/ocats \
        -f 120 \
        -g 10 \
        -h ocscp-scp-worker \
        -i ocscp-scpc-configuration \
```

☑ Use Groovy Sandbox

You can modify script pipeline parameters from **"-b"** to **"-q"** on the basis of your deployment environment and click **Save**. The content of the pipeline script is as follows:

Figure 3-57   SCP Pipeline Content



```
node ('master'){
    //a = SELECTED_NF     b = NFNAMESPACE     c = CLUSTERDOMAIN      d = DESTNAMESPACE
    //e = ATSREGISTRY   f = AUDITINTERVAL       g = GUARDTIME      h = SCPSVCNAME
    //i = SCPCONFIGSVCNAME    j = SCPNOTIFYSVCNAME    k = SCPSUBSVCNAME    l = DBSECRETNAME
    //m = MYSQLHOST n = ATSSTUBIMAGE     o = ATSSTUBCPU     p = ATSSTUBMEMORY    q = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocscp_tests/preTestConfig.sh \
        -a SCP \
        -b scpsvc \
        -c odyssey.lab.us.oracle.com \
        -d scpsvc \
        -e bastion-1:5000/ocats \
        -f 120 \
        -g 10 \
        -h ocscp-scp-worker \
        -i ocscp-scpc-configuration \
        -j ocscp-scpc-notification \
        -k ocscp-scpc-subscription \
        -l cred \
        -m mysql.default \
        -n ocats-gostub:1.7.0 \
        -o 0.2 \
        -p 0.1G \
        -q 0
    '''
    load "/var/lib/jenkins/ocscp_tests/jenkinsData/Jenkinsfile-NewFeatures"
```

The description of these parameters is as follows:

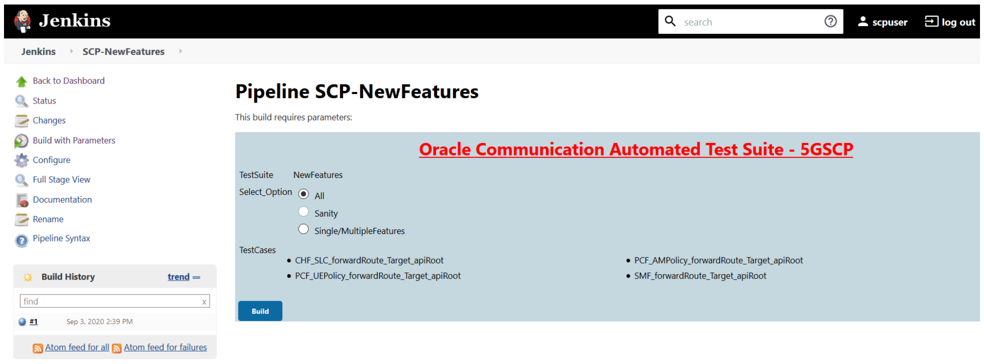
- -a - Selected NF
- -b - NameSpace in which SCP is Deployed
- -c - Kubernetes Cluster Domain where SCP is Deployed
- -d - Test Stubs NameSpace - must be same as SCP Namespace
- -e - Docker registry where test stub image is available
- -f - Audit Interval provided in SCP Deployment file
- -g - Guard Time provided SCP Deployment file
- -h - SCP-Worker microservice name as provided during deployment
- -i - SCPC-Configuration microservice name as provided during deployment
- -j - SCPC-Notification microservice name as provided during deployment
- -k - SCPC-Subscription microservice name as provided during deployment
- -l - DB Secret name as provided during deployment

- -m - Mysql Host name as provided during deployment
- -n - Test Stub Image Name with tag
- -o - Test Stub CPU requests and limit
- -p - Test Stub Memory requests and limit
- -q - re-run count

> **Note:**
>
> DO NOT MODIFY ANYTHING OTHER THAN THESE PARAMETERS.

4. Click the **Build with Parameters**. Following screen appears:

Figure 3-58 Build with Parameters Options



Pipeline SCP-NewFeatures

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSCP

TestSuite NewFeatures

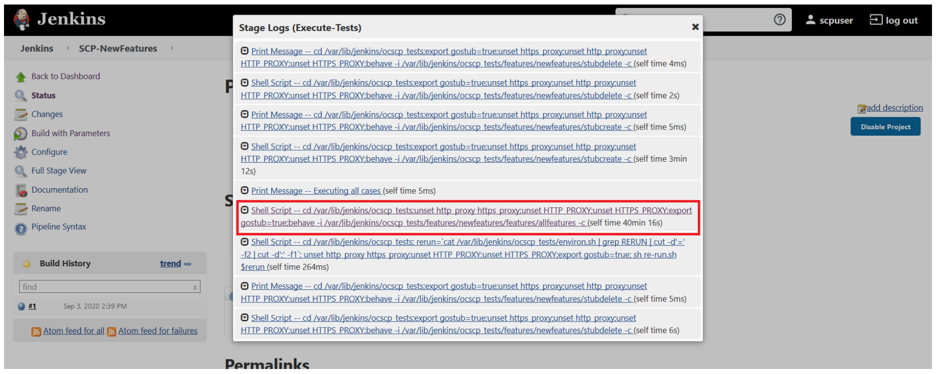
Select_Option [✓ All] Sanity Single/MultipleFeatures

TestCases
- CHF_SLC_forwardRoute_Target_apiRoot
- PCF_UEPolicy_forwardRoute_Target_apiRoot
- PCF_AMPolicy_forwardRoute_Target_apiRoot
- SMF_forwardRoute_Target_apiRoot

In the above screen, there are three **Select_Option**(s), which are:

- **All:** By default, all the SCP test cases are selected for execution. User just need to scroll down and click **Build** to execute all the test cases.
- **Sanity:** This option is **NOT AVAILABLE** for SCP.
- **Single/MultipleFeatures:** This option allows you to select any number of test cases that you want to execute from the list of total test cases available for execution. After selecting the test cases, scroll-down and click **Build**. The selected SCP test cases are executed.
- To check execution results and logs:
  – Click the execute-tests stage of pipeline and then logs.
  – Select the test execution step.
  – Double-click to open the execution logs console.

Figure 3-59 SCP-NewFeatures Stage Logs



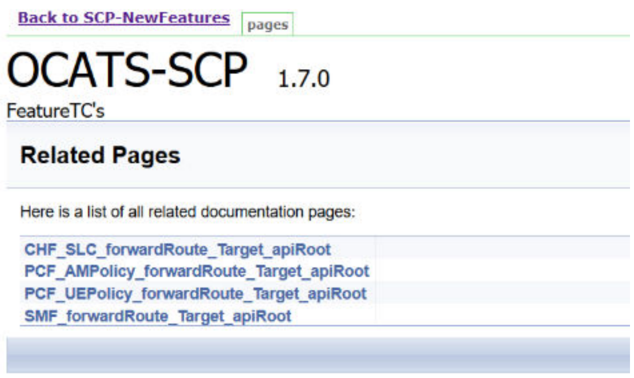
Stage Logs (Execute-Tests)

```
Print Message -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubdelete -c (self time 4ms)
Shell Script -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubdelete -c (self time 2s)
Print Message -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubcreate -c (self time 5ms)
Shell Script -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubcreate -c (self time 3min 12s)
Print Message -- Executing all cases (self time 5ms)
Shell Script -- cd /var/lib/jenkins/ocscp_tests;unset http_proxy https_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;export gostub=true;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/features/allfeatures -c (self time 40min 16s)
Shell Script -- cd /var/lib/jenkins/ocscp_tests; rerun=`cat /var/lib/jenkins/ocscp_tests/environ.sh | grep RERUN | cut -d'=' -f2 | cut -d';' -f1`; unset http_proxy https_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;export gostub=true; sh re-run.sh $rerun (self time 264ms)
Print Message -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubdelete -c (self time 5ms)
Shell Script -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/newfeatures/stubdelete -c (self time 6s)
```

NewFeatures - Documentation

This pipeline has the HTML report of all the feature files that you can test as part of SCP ATS release. To view SCP functionalities, go to SCP-NewFeatures pipeline and click **Documentation** link in the left navigation pane. The following screen appears:

Figure 3-60 SCP-NewFeatures-Documentation



Back to SCP-NewFeatures
pages

OCATS-SCP 1.7.0

FeatureTC's

Related Pages

Here is a list of all related documentation pages:

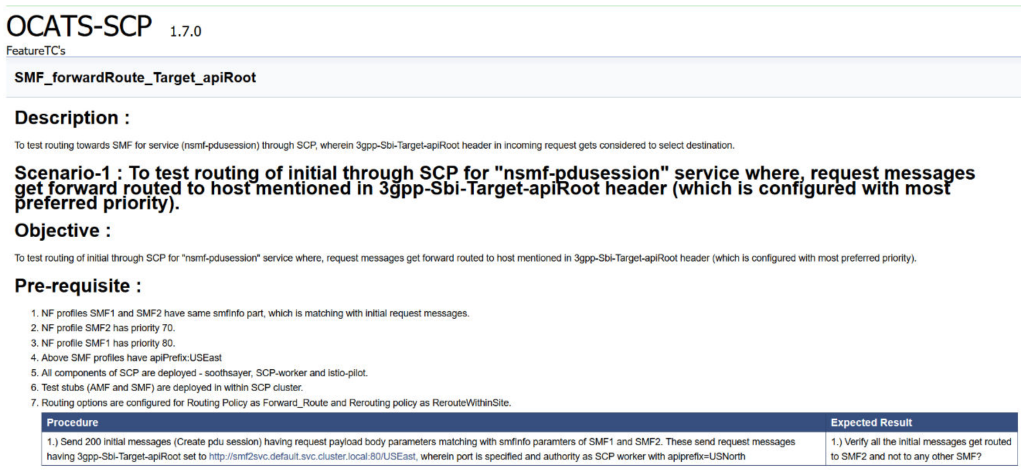
- CHF_SLC_forwardRoute_Target_apiRoot
- PCF_AMPolicy_forwardRoute_Target_apiRoot
- PCF_UEPolicy_forwardRoute_Target_apiRoot
- SMF_forwardRoute_Target_apiRoot

Note:

Documentation option appears only if New-Features pipeline is executed atleast once.

You can click any functionality to view its test cases and scenarios for each test case. For example, on click of SMF_forwardRoute_Target_apiRoot, the following screen appears:

Figure 3-61   Sample: SCP Functionality



OCATS-SCP 1.7.0
FeatureTC's

SMF_forwardRoute_Target_apiRoot

Description :

To test routing towards SMF for service (nsmf-pdusession) through SCP, wherein 3gpp-Sbi-Target-apiRoot header in incoming request gets considered to select destination.

Scenario-1 : To test routing of initial through SCP for "nsmf-pdusession" service where, request messages get forward routed to host mentioned in 3gpp-Sbi-Target-apiRoot header (which is configured with most preferred priority).

Objective :

To test routing of initial through SCP for "nsmf-pdusession" service where, request messages get forward routed to host mentioned in 3gpp-Sbi-Target-apiRoot header (which is configured with most preferred priority).

Pre-requisite :

1. NF profiles SMF1 and SMF2 have same smfInfo part, which is matching with initial request messages.
2. NF profile SMF2 has priority 70.
3. NF profile SMF1 has priority 80.
4. Above SMF profiles have apiPrefix:USEast
5. All components of SCP are deployed - soothsayer, SCP-worker and istio-pilot.
6. Test stubs (AMF and SMF) are deployed in within SCP cluster.
7. Routing options are configured for Routing Policy as Forward_Route and Rerouting policy as RerouteWithinSite.

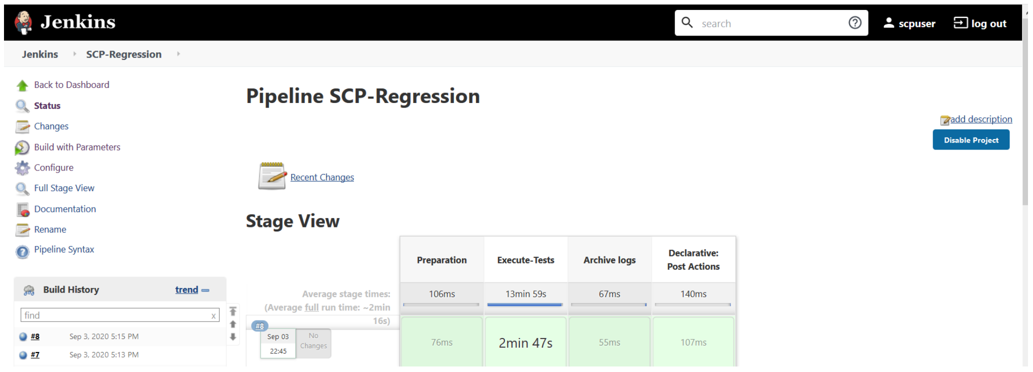
Procedure	Expected Result
1.) Send 200 initial messages (Create pdu session) having request payload body parameters matching with smfInfo paramters of SMF1 and SMF2. These send request messages having 3gpp-Sbi-Target-apiRoot set to http://smf2svc.default.svc.cluster.local:80/USEast, wherein port is specified and authority as SCP worker with apiprefix=USNorth	1.) Verify all the initial messages get routed to SMF2 and not to any other SMF?

Based on the functionalities covered under Documentation, the **Build Requires Parameters** screen displays test cases. To navigate back to the Pipeline SCP-NewFeatures screen, click **Back to SCP-NewFeatures** link available on top left corner of the screen.

SCP-Regression Pipeline

This pre-configured pipeline has all the test cases of previous releases. When you click SCP-Regression Pipeline, following screen appears:

Figure 3-62   SCP-Regression Pipeline



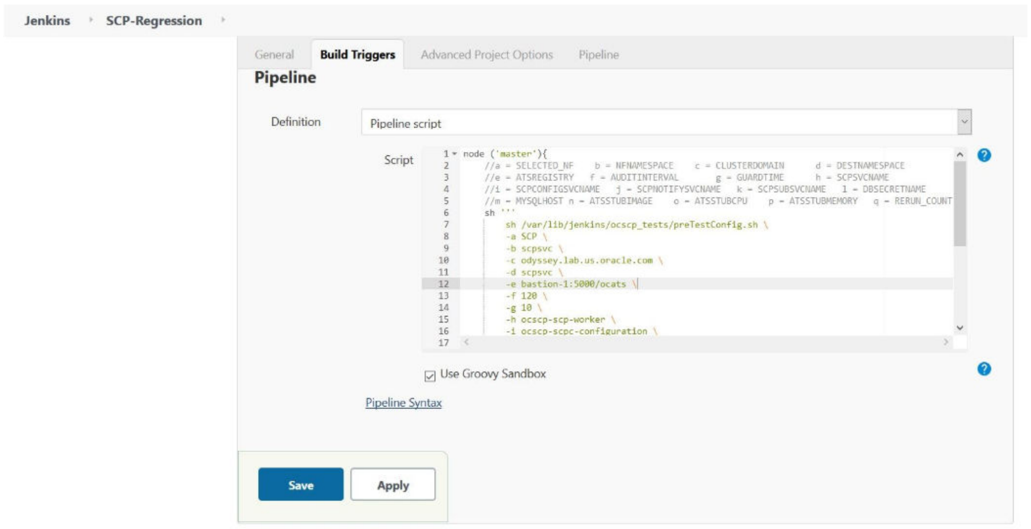
	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
Average stage times: (Average full run time: ~2min 16s)	106ms	13min 59s	67ms	140ms
#8 Sep 03 22:45 No Changes	76ms	2min 47s	55ms	107ms

#8	Sep 3, 2020 5:15 PM
#7	Sep 3, 2020 5:13 PM

If you are executing SCP pipeline for the first time, you have to set Input Parameters before execution. Subsequent execution does not require any input unless there is a need to change any configuration.

In the left navigation pane, click **Configure** to provide inputs parameters and scroll to bottom of the screen to pipeline script as displayed below.

Figure 3-63   Regression - Pipeline Script



Pipeline

Definition: Pipeline script

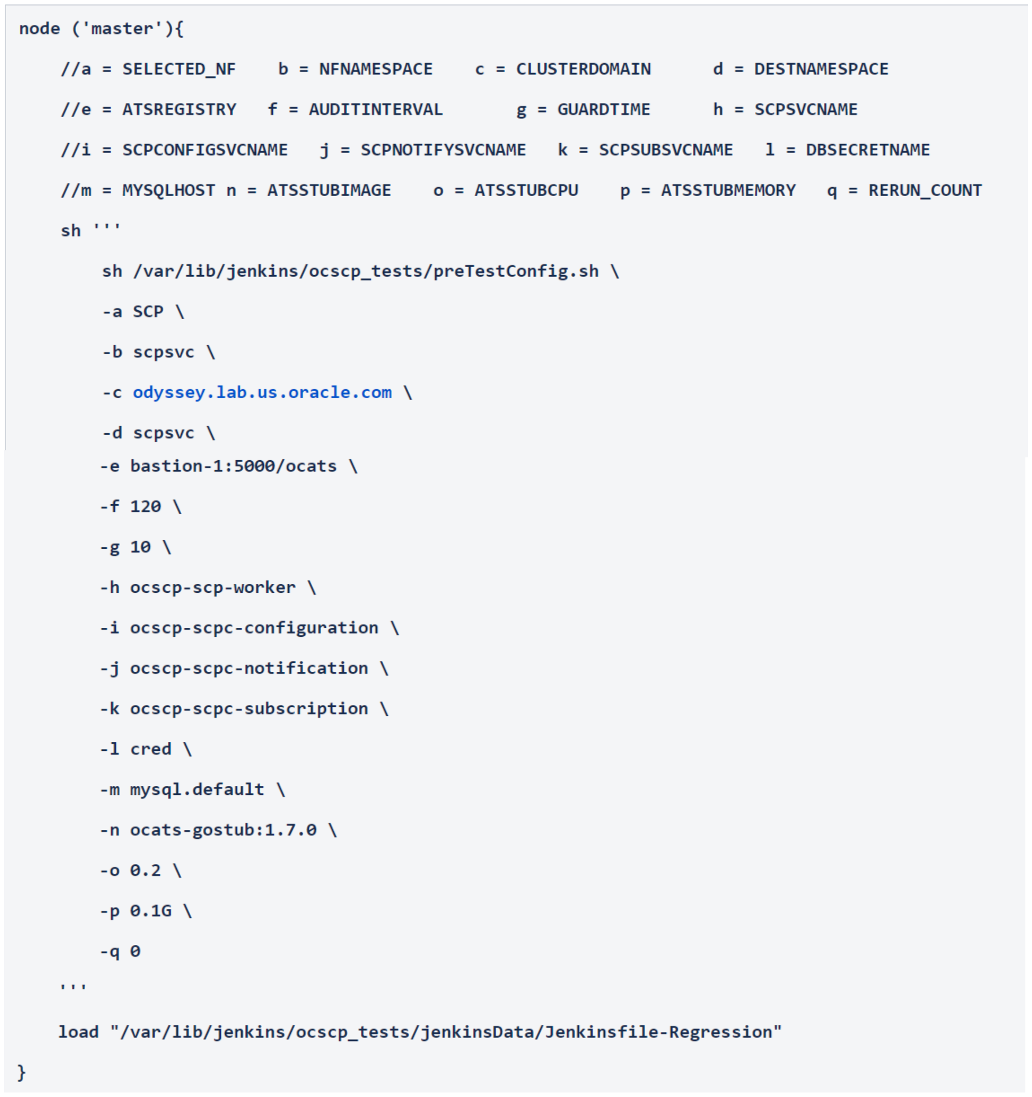
Script:

```
node ('master'){
    //a = SELECTED_NF    b = NFNAMESPACE    c = CLUSTERDOMAIN      d = DESTNAMESPACE
    //e = ATSREGISTRY    f = AUDITINTERVAL        g = GUARDTIME      h = SCPSVCNAME
    //i = SCPCONFIGSVCNAME   j = SCPNOTIFYSVCNAME   k = SCPSUBSVCNAME   l = DBSECRETNAME
    //m = MYSQLHOST n = ATSSTUBIMAGE    o = ATSSTUBCPU    p = ATSSTUBMEMORY   q = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocscp_tests/preTestConfig.sh \
        -a SCP \
        -b scpsvc \
        -c odyssey.lab.us.oracle.com \
        -d scpsvc \
        -e bastion-1:5000/ocats \
        -f 120 \
        -g 10 \
        -h ocscp-scp-worker \
        -i ocscp-scpc-configuration \
```

☑ Use Groovy Sandbox

You can change parameters from **"-b"** to **"-q"** as per deployment environment and click **Save**. The content of the pipeline script is as follows:

Figure 3-64 SCP-Regression Pipeline Script



```
node ('master'){
    //a = SELECTED_NF    b = NFNAMESPACE    c = CLUSTERDOMAIN      d = DESTNAMESPACE
    //e = ATSREGISTRY   f = AUDITINTERVAL      g = GUARDTIME      h = SCPSVCNAME
    //i = SCPCONFIGSVCNAME   j = SCPNOTIFYSVCNAME   k = SCPSUBSVCNAME   l = DBSECRETNAME
    //m = MYSQLHOST n = ATSSTUBIMAGE    o = ATSSTUBCPU    p = ATSSTUBMEMORY   q = RERUN_COUNT
    sh '''
        sh /var/lib/jenkins/ocscp_tests/preTestConfig.sh \
        -a SCP \
        -b scpsvc \
        -c odyssey.lab.us.oracle.com \
        -d scpsvc \
        -e bastion-1:5000/ocats \
        -f 120 \
        -g 10 \
        -h ocscp-scp-worker \
        -i ocscp-scpc-configuration \
        -j ocscp-scpc-notification \
        -k ocscp-scpc-subscription \
        -l cred \
        -m mysql.default \
        -n ocats-gostub:1.7.0 \
        -o 0.2 \
        -p 0.1G \
        -q 0
    '''
    load "/var/lib/jenkins/ocscp_tests/jenkinsData/Jenkinsfile-Regression"
}
```

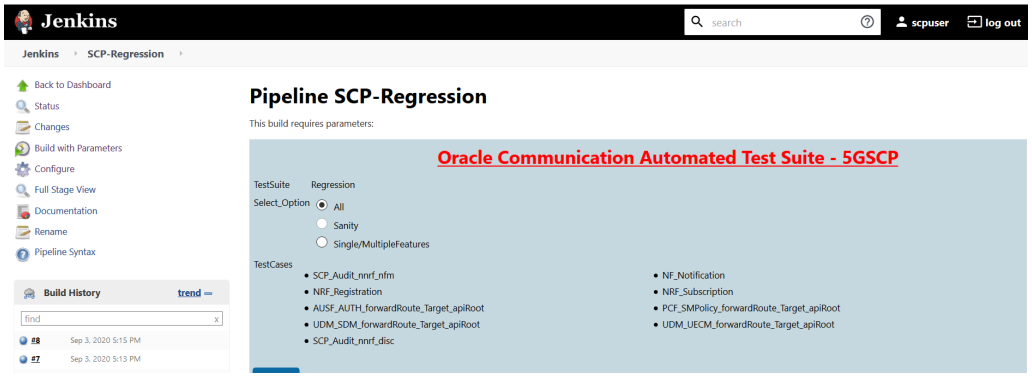
The description of parameters is as follows:

```
-a     - Selected NF
           -b     - NameSpace in which SCP is Deployed
           -c     - K8s Cluster Domain where SCP is Deployed
           -d     - Test Stubs NameSpace - Must be same as SCP Namespace
           -e     - Docker registry where test stub image is available
           -f     - Audit Interval provided in SCP Deployment file
           -g    - Guard Time provided SCP Deployment file
           -h    - SCP-Worker microservice name as provided during
deployment
           -i     - SCPC-Configuration microservice name as provided
during deployment
           -j     - SCPC-Notification microservice name as provided
during deployment
           -k    - SCPC-Subscription microservice name as provided
```

```
during deployment
          -l    - DB Secret name as provided during deployment
          -m  - Mysql Host name as provided during deployment
          -n   - Test Stub Image Name with tag
          -o   - Test Stub CPU requests and limit
          -p   - Test Stub Memory requests and limit
          -q   - re-run count
```

Click **Build with Parameters**. The following screen appears:

Figure 3-65 SCP Regression - Build with Parameters



Pipeline SCP-Regression

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSCP

TestSuite Regression

Select_Option [✓ All] Sanity Single/MultipleFeatures

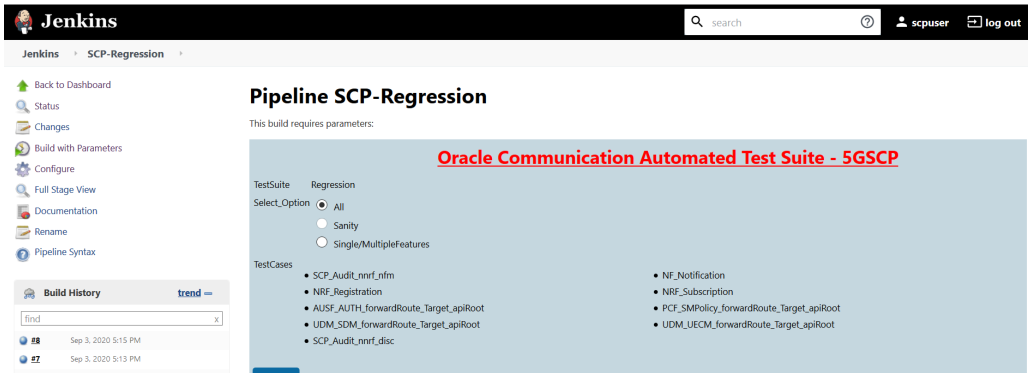
TestCases
- SCP_Audit_nnrf_nfm
- NRF_Registration
- AUSF_AUTH_forwardRoute_Target_apiRoot
- UDM_SDM_forwardRoute_Target_apiRoot
- SCP_Audit_nnrf_disc
- NF_Notification
- NRF_Subscription
- PCF_SMPolicy_forwardRoute_Target_apiRoot
- UDM_UECM_forwardRoute_Target_apiRoot

It has following three options:

- **All** - To execute all the test cases except **SCP_Audit_nnrf_disc**. If SCP is deployed with nnrf-disc for Audit or Registration with NRF is not enabled, then you should not use the All option. Instead, use Single/MultipleFeatures option to select appropriate cases for execution.
- **Sanity** - This option is not available for SCP.
- **Single/MultipleFeatures** - To execute selected test cases. You can select one or more test cases and execute using this option.

Select an appropriate option and click **Build** to start test execution.

Figure 3-66 SCP-Regression Build Option



Pipeline SCP-Regression

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSCP

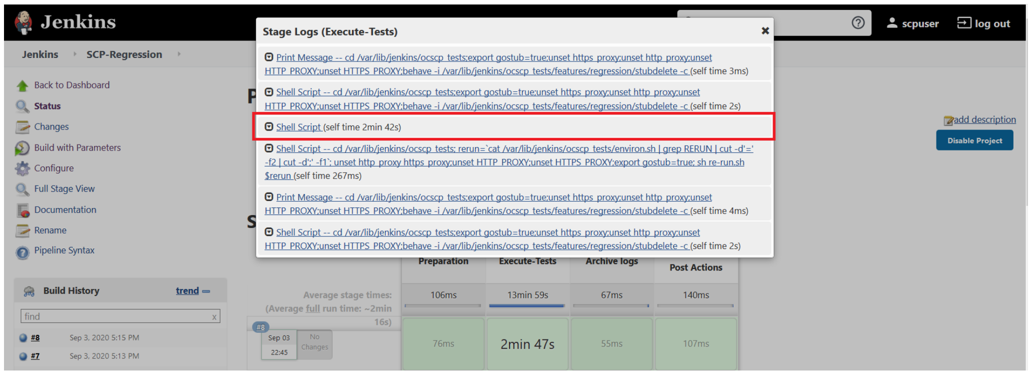
TestSuite Regression

Select_Option [✓ All] Sanity Single/MultipleFeatures

TestCases
- SCP_Audit_nnrf_nfm
- NRF_Registration
- AUSF_AUTH_forwardRoute_Target_apiRoot
- UDM_SDM_forwardRoute_Target_apiRoot
- SCP_Audit_nnrf_disc
- NF_Notification
- NRF_Subscription
- PCF_SMPolicy_forwardRoute_Target_apiRoot
- UDM_UECM_forwardRoute_Target_apiRoot

To check execution results and logs, click the execute-tests stage of pipeline and then logs. To open execution logs console, select test execution step and double-click the execution log.

Figure 3-67 SCP-Regression Stage Logs



Stage Logs (Execute-Tests)

Print Message -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/regression/stubdelete -c (self time 3ms)

Shell Script -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/regression/stubdelete -c (self time 2s)

Shell Script (self time 2min 42s)

Shell Script -- cd /var/lib/jenkins/ocscp_tests; rerun=`cat /var/lib/jenkins/ocscp_tests/environ.sh | grep RERUN | cut -d'=' -f2 | cut -d';' -f1`; unset http_proxy https_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;export gostub=true; sh re-run.sh $rerun (self time 267ms)

Print Message -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/regression/stubdelete -c (self time 4ms)

Shell Script -- cd /var/lib/jenkins/ocscp_tests;export gostub=true;unset https_proxy;unset http_proxy;unset HTTP_PROXY;unset HTTPS_PROXY;behave -i /var/lib/jenkins/ocscp_tests/features/regression/stubdelete -c (self time 2s)

	Preparation	Execute-Tests	Archive logs	Post Actions
Average stage times: (Average full run time: ~2min 16s)	106ms	13min 59s	67ms	140ms
#8 Sep 03 22:45 No Changes	76ms	2min 47s	55ms	107ms

#8	Sep 3, 2020 5:15 PM
#7	Sep 3, 2020 5:13 PM

Executing SLF Test Cases using ATS

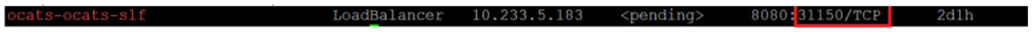
Logging into ATS

Before logging into ATS, you need to know the nodeport of the "-ocats-slf" service. To get the nodeport detail, execute the following command:

```
kubectl get svc -n <slf_namespace>
```

Example: `kubectl get svc -n ocats`

Figure 3-68 SLF Nodeport



```
ocats-ocats-slf                LoadBalancer   10.233.5.183   <pending>   8080:31150/TCP      2d1h
```

In the above screen, 31150 is the nodeport.

To login to ATS via Jenkins:

1. In the web browser, type **http://<Worker IP>:<port obtained above>** and press **Enter**.
   Example: http://10.75.225.49:31150

   The Login screen appears.

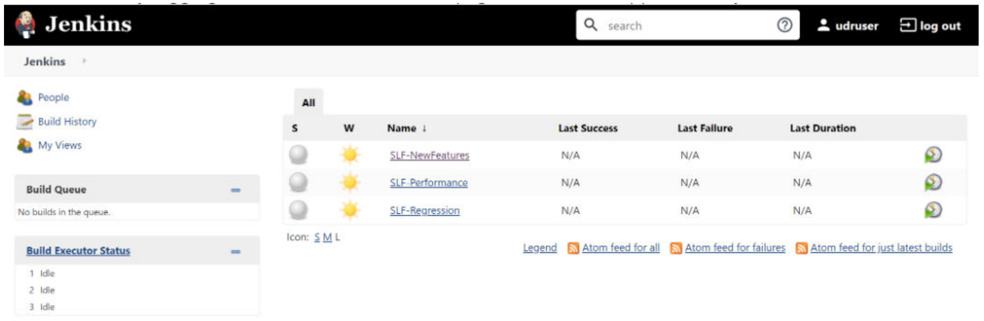
2. Enter the username as 'udruser' and password as 'udrpasswd'. Click **Sign in**. A screen with pre-configured pipelines for SLF appears (3 pipelines).

   - **SLF-New-Features:** This pipeline has all the test cases, which are delivered as part of SLF ATS - 1.7.1.
   - **SLF-Performance:** This pipeline is not operational as of now. It is reserved for future releases of ATS.

- **SLF-Regression:** This pipeline has all the test cases of previous releases. As this is the first release of SLF-ATS, this pipeline does not show any previous release test cases.

> **Note:**
>
> If you want to modify your default login password, refer to Modifying Login Password

Figure 3-69 SLF Pre-configured Pipelines



Jenkins | search | udruser | log out

Jenkins

People
Build History
My Views

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle
3 Idle

All

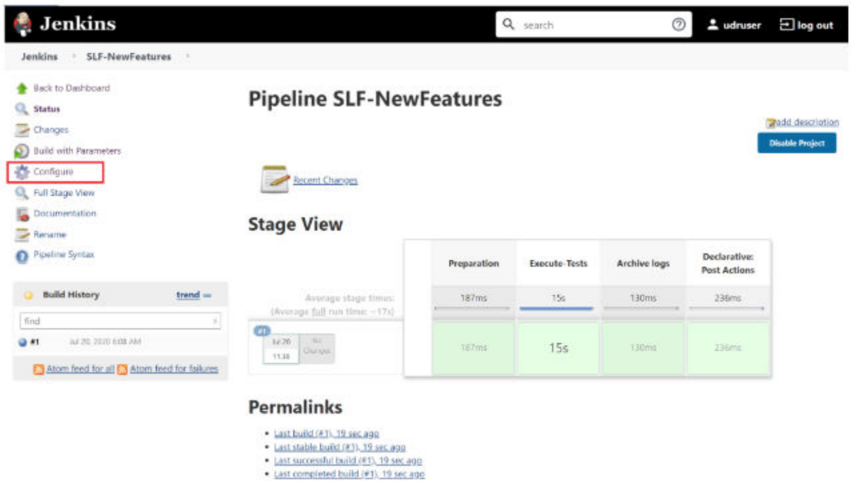
S	W	Name ↓	Last Success	Last Failure	Last Duration
●	☀	SLF-NewFeatures	N/A	N/A	N/A
●	☀	SLF-Performance	N/A	N/A	N/A
●	☀	SLF-Regression	N/A	N/A	N/A

Icon: S M L

Legend | Atom feed for all | Atom feed for failures | Atom feed for just latest builds

3. Click **SLF-NewFeatures**. The following screen appears:

Figure 3-70 SLF-NewFeatures Configure



Jenkins | search | udruser | log out

Jenkins › SLF-NewFeatures ›

Back to Dashboard
Status
Changes
Build with Parameters
Configure
Full Stage View
Documentation
Rename
Pipeline Syntax

Build History trend
find
#1 Jul 20, 2020 6:08 AM
Atom feed for all | Atom feed for failures

Pipeline SLF-NewFeatures

add description
Disable Project

Recent Changes

Stage View

Average stage times:
(Average full run time: ~17s)

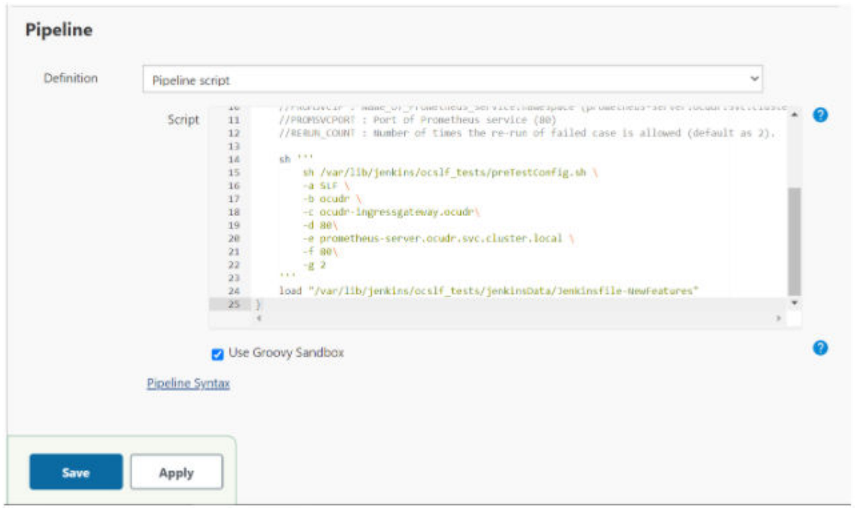
	Preparation	Execute-Tests	Archive logs	Declarative: Post Actions
	187ms	15s	130ms	236ms
#1 Jul 20 11:38 No Changes	187ms	15s	130ms	236ms

Permalinks

- Last build (#1), 19 sec ago
- Last stable build (#1), 19 sec ago
- Last successful build (#1), 19 sec ago
- Last completed build (#1), 19 sec ago

4. Click **Configure** in the left navigation pane. The **General** tab appears. User **MUST** wait for the page to load completely.
5. Once the page loads completely, click the **Advanced Project Options** tab. Scroll down to reach the **Pipeline** configuration as shown below:
   MAKE SURE THAT THE SCREEN SHOWN BELOW LOADS COMPLETELY BEFORE YOU PERFORM ANY ACTION ON IT. ALSO, DO NOT MODIFY ANY CONFIGURATION OTHER THAN DISCUSSED BELOW.

Figure 3-71   SLF Configuration Parameters



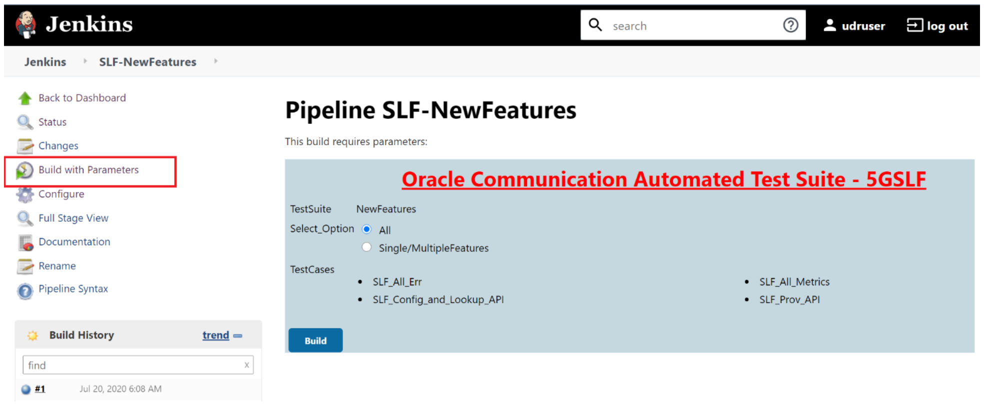
```
11   //PROMSVCPORT : Port of Prometheus service (80)
12   //RERUN_COUNT : Number of times the re-run of failed case is allowed (default as 2).
13
14   sh '''
15       sh /var/lib/jenkins/ocslf_tests/preTestConfig.sh \
16       -a SLF \
17       -b ocudr \
18       -c ocudr-ingressgateway.ocudr\
19       -d 80\
20       -e prometheus-server.ocudr.svc.cluster.local \
21       -f 80\
22       -g 2
23   '''
24   load "/var/lib/jenkins/ocslf_tests/jenkinsData/Jenkinsfile-NewFeatures"
25 }
```

You **SHOULD NOT** change any other value apart from **line number 12 to line 15**. It means the parameters marked as **"a"** - to - **"g"** can only be changed as per user requirement. The detail about these parameters is as follows:

- **a -** Name of the NF to be tested in capital (SLF).
- **b -** Namespace in which the udr is deployed.
- **c -** Name_of_UDR_ingressgateway_service.namespace (ocudr-ingressgateway.ocudr).
- **d -** Port of ingressgateway service (80).
- **e -** Name_of_Prometheus_service.namespace (prometheus-server.ocudr.svc.cluster.local).
- **f -** Port of Prometheus service (80).
- **g -** Number of times the re-run of failed case is allowed (default as 2).

6. Click **Save** after making neccesary changes. The SLF-NewFeatures screen appears.
7. Click **Build with Parameters**. The following screen appears:

Figure 3-72   SLF Build with Parameters



Pipeline SLF-NewFeatures

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSLF

TestSuite   NewFeatures

Select_Option   [✓ All]   Single/MultipleFeatures

TestCases
- SLF_All_Err
- SLF_Config_and_Lookup_API
- SLF_All_Metrics
- SLF_Prov_API

#1   Jul 20, 2020 6:08 AM

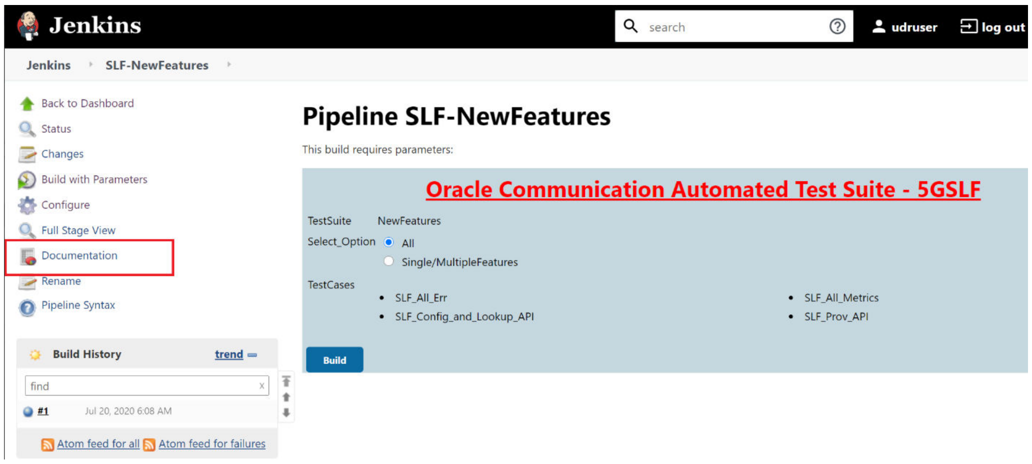
In the above screen, there are two **Select_Option**(s), which are:

- **All:** By default, all the SCP test cases are selected for execution. User just need to scroll down and click **Build** to execute all the test cases.
- **Single/MultipleFeatures:** This option allows you to select any number of test cases that you want to execute from the list of total test cases available for execution. After selecting the test cases, scroll-down and click **Build**. The selected SLF test cases are executed.

NewFeatures-Documentation

To view SLF functionalities, go to SLF-NewFeatures pipeline and click **Documentation** link in the left navigation pane as shown below:

Figure 3-73 SLF-NewFeatures Documentation Option



Pipeline SLF-NewFeatures

This build requires parameters:

Oracle Communication Automated Test Suite - 5GSLF

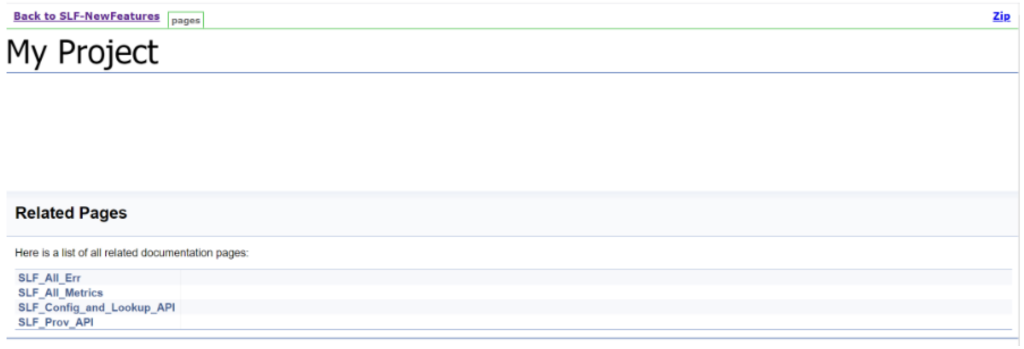
TestSuite NewFeatures

Select_Option (●) All ( ) Single/MultipleFeatures

TestCases
- SLF_All_Err
- SLF_Config_and_Lookup_API
- SLF_All_Metrics
- SLF_Prov_API

The following screen appears:

Figure 3-74 SLF-NewFeatures Documentation



My Project

Related Pages

Here is a list of all related documentation pages:

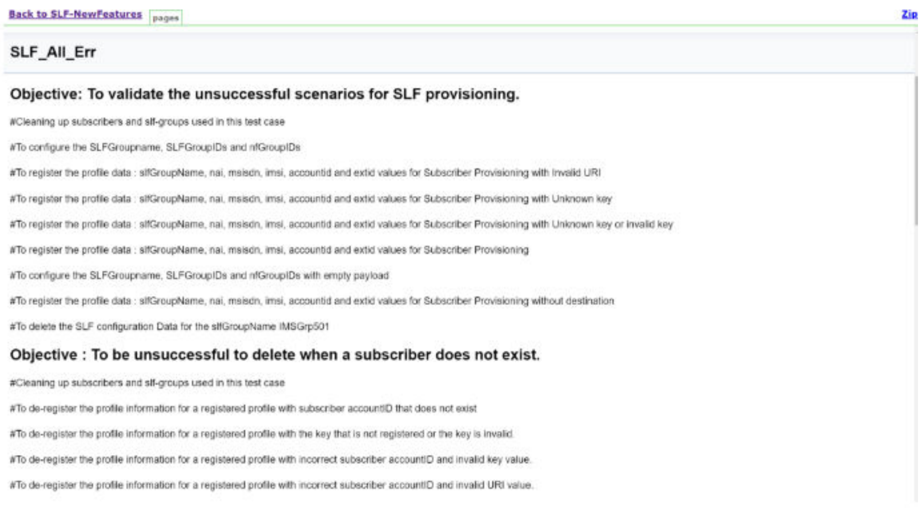
SLF_All_Err
SLF_All_Metrics
SLF_Config_and_Lookup_API
SLF_Prov_API

Note:

Documentation option appears only if New-Features pipeline is executed atleast once.

You can click any functionality to view its test cases and scenarios for each test case. For example, on click of SLF_All_Err, following screen appears:

Figure 3-75 Sample: SLF Test Case Description



SLF_All_Err

Objective: To validate the unsuccessful scenarios for SLF provisioning.

#Cleaning up subscribers and slf-groups used in this test case

#To configure the SLFGroupname, SLFGroupIDs and nfGroupIDs

#To register the profile data : slfGroupName, nai, msisdn, imsi, accountid and extid values for Subscriber Provisioning with Invalid URI

#To register the profile data : slfGroupName, nai, msisdn, imsi, accountid and extid values for Subscriber Provisioning with Unknown key

#To register the profile data : slfGroupName, nai, msisdn, imsi, accountid and extid values for Subscriber Provisioning with Unknown key or invalid key

#To register the profile data : slfGroupName, nai, msisdn, imsi, accountid and extid values for Subscriber Provisioning

#To configure the SLFGroupname, SLFGroupIDs and nfGroupIDs with empty payload

#To register the profile data : slfGroupName, nai, msisdn, imsi, accountid and extid values for Subscriber Provisioning without destination

#To delete the SLF configuration Data for the slfGroupName IMSGrp501

Objective : To be unsuccessful to delete when a subscriber does not exist.

#Cleaning up subscribers and slf-groups used in this test case

#To de-register the profile information for a registered profile with subscriber accountID that does not exist

#To de-register the profile information for a registered profile with the key that is not registered or the key is invalid

#To de-register the profile information for a registered profile with incorrect subscriber accountID and invalid key value.

#To de-register the profile information for a registered profile with incorrect subscriber accountID and invalid URI value.

Based on the functionalities covered under Documentation, the **Build Requires Parameters** screen displays test cases. To navigate back to the Pipeline SLF-NewFeatures screen, click **Back to SLF-NewFeatures** link available on top left corner of the screen.

A

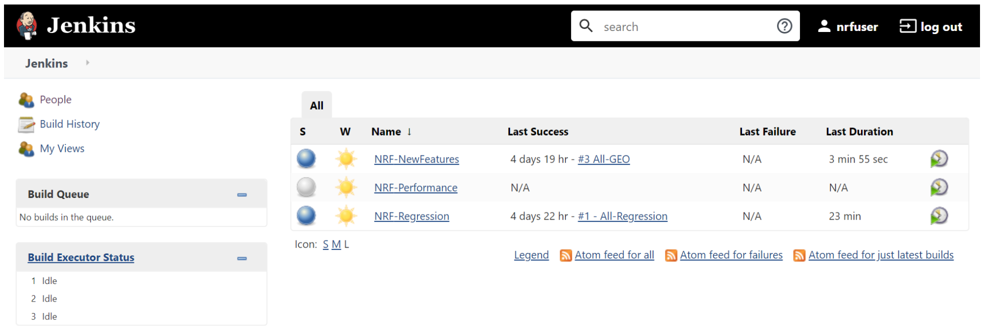
Modifying Login Password

You can login to ATS application using default login credentials. The default login credentials are shared for each NF in its respective chapter of this guide.

If the user wants to modify its login password, the ATS application allows to do so. To modify login password:

1. Login to ATS application using default login credentials. The home screen of respective NF appears showing its pre-configured pipelines.

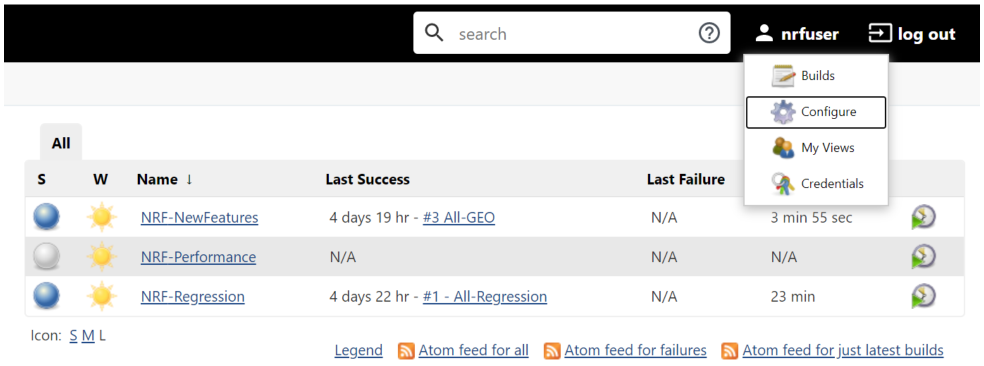
Figure A-1 Sample: NRF Home Screen



S	W	Name ↓	Last Success	Last Failure	Last Duration
		NRF-NewFeatures	4 days 19 hr - #3 All-GEO	N/A	3 min 55 sec
		NRF-Performance	N/A	N/A	N/A
		NRF-Regression	4 days 22 hr - #1 - All-Regression	N/A	23 min

2. Hover-over logged-in user name and click the down arrow. Click **Configure** as shown below.

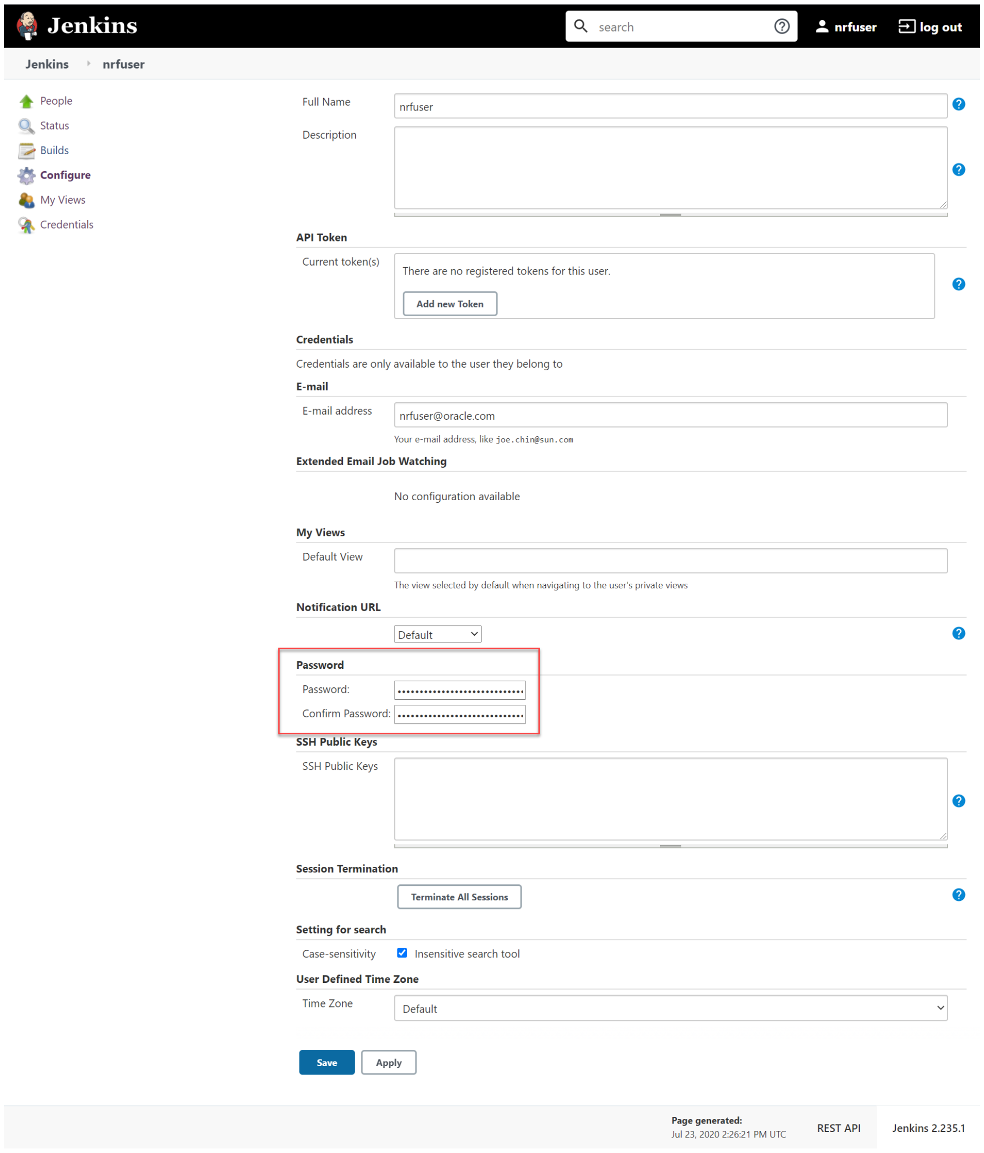
Figure A-2 Configure Option



S	W	Name ↓	Last Success	Last Failure	
		NRF-NewFeatures	4 days 19 hr - #3 All-GEO	N/A	3 min 55 sec
		NRF-Performance	N/A	N/A	N/A
		NRF-Regression	4 days 22 hr - #1 - All-Regression	N/A	23 min

3. The following screen appears.

Figure A-3 Logged-in User Detail



Full Name: nrfuser

Description:

API Token

Current token(s): There are no registered tokens for this user.

Credentials

Credentials are only available to the user they belong to

E-mail

E-mail address: nrfuser@oracle.com

Your e-mail address, like joe.chin@sun.com

Extended Email Job Watching

No configuration available

My Views

Default View:

The view selected by default when navigating to the user's private views

Notification URL

Default

Password

Password: ••••••••••••••••••••••••••••••

Confirm Password: ••••••••••••••••••••••••••••••

SSH Public Keys

SSH Public Keys:

Session Termination

Setting for search

Case-sensitivity: ✓ Insensitive search tool

User Defined Time Zone

Time Zone: Default

Page generated: Jul 23, 2020 2:26:21 PM UTC REST API Jenkins 2.235.1

4. In the **Password** section, enter the new password in the **Password** and **Confirm Password** fields and click **Save**.

Thus, a new password is set for the user.