
OPNFV Functest Documentation

Release master

Functest <opnfv-tech-discuss@lists.opnfv.org>

Jan 28, 2022

Contents

1	functest	3
1.1	functest package	3
2	Indices and tables	29
	Python Module Index	31
	Index	33

Contents:

1.1 functest package

1.1.1 Subpackages

functest.core package

Submodules

functest.core.cloudify module

functest.core.singlevm module

Ease deploying a single VM reachable via ssh

It offers a simple way to create all tenant network resources + a VM for advanced testcases (e.g. deploying an orchestrator).

```
class functest.core.singlevm.SingleVm1 (**kwargs)
    Bases: functest.core.singlevm.VmReady1
```

Deploy a single VM reachable via ssh (scenario1)

It inherits from TenantNetwork1 which creates all network resources and completes it by booting a VM attached to that network.

It ensures that all testcases inheriting from SingleVm1 could work without specific configurations (or at least read the same config data).

```
check_console_loop = 6
```

```
check_console_regex = ' login: '
```

```
clean ()
```

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

connect (*vm1*)

Connect to a virtual machine via ssh

It first adds a floating ip to the virtual machine and then establishes the ssh connection.

Returns: - (fip, ssh) - None on error

create_floating_ip_timeout = 120

execute ()

Say hello world via ssh

It can be overridden to execute any command.

Returns: echo exit codes

prepare ()

Create the security group and the keypair

It can be overridden to set other rules according to the services running in the VM

Raises: Exception on error

run (***kwargs*)

Boot the new VM

Here are the main actions: - add a new ssh key - boot the VM - create the security group - execute the right command over ssh

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

ssh_connect_loops = 6

ssh_connect_timeout = 1

username = 'cirros'

class `functest.core.singlevm.SingleVm2` (***kwargs*)

Bases: `functest.core.singlevm.SingleVm1`

Deploy a single VM reachable via ssh (scenario2)

It creates new user/project before creating and configuring all tenant network resources and vms required by advanced testcases.

It ensures that all testcases inheriting from SingleVm2 could work without specific configurations (or at least read the same config data).

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

class `functest.core.singlevm.VmReady1` (***kwargs*)

Bases: `functest.core.tenantnetwork.TenantNetwork1`

Prepare a single VM (scenario1)

It inherits from TenantNetwork1 which creates all network resources and prepares a future VM attached to that network.

It ensures that all testcases inheriting from SingleVm1 could work without specific configurations (or at least read the same config data).

boot_vm (*name=None, **kwargs*)

Boot the virtual machine

It allows booting multiple machines for the child testcases. It forces the same configuration for all subtestcases.

Returns: vm

Raises: exception on error

check_regex_in_console (*name, regex=' login: ', loop=6*)

Wait for specific message in console

Returns: True or False on errors

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

clean_orphan_security_groups ()

Clean all security groups which are not owned by an existing tenant

It lists all orphan security groups in use as debug to avoid misunderstanding the testcase results (it could happen if cloud admin removes accounts without cleaning the virtual machines)

count_active_hypervisors ()

Count all hypervisors which are up.

count_hypervisors ()

Count hypervisors.

create_flavor (*name=None*)

Create flavor

It allows creating multiple flavors for the child testcases. It forces the same configuration for all subtestcases.

Returns: flavor

Raises: exception on error

create_flavor_alt (*name=None*)

Create flavor

It allows creating multiple alt flavors for the child testcases. It forces the same configuration for all subtestcases.

Returns: flavor

Raises: exception on error

create_server_timeout = 180

extra_alt_properties = {}

extra_properties = {}

filename = '/home/opnfv/functest/images/cirros-0.4.0-x86_64-disk.img'

filename_alt = '/home/opnfv/functest/images/cirros-0.4.0-x86_64-disk.img'

flavor_alt_disk = 1

flavor_alt_extra_specs = {}

flavor_alt_ram = 1024

```

flavor_alt_vcpus = 1
flavor_disk = 1
flavor_extra_specs = {}
flavor_ram = 512
flavor_vcpus = 1
image_alt_format = 'qcow2'
image_format = 'qcow2'

```

publish_image (*name=None*)

Publish image

It allows publishing multiple images for the child testcases. It forces the same configuration for all sub-testcases.

Returns: image

Raises: exception on error

publish_image_alt (*name=None*)

Publish alternative image

It allows publishing multiple images for the child testcases. It forces the same configuration for all sub-testcases.

Returns: image

Raises: exception on error

run (***kwargs*)

Boot the new VM

Here are the main actions: - publish the image - create the flavor

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

visibility = 'private'

class `functest.core.singlevm.VmReady2` (***kwargs*)

Bases: `functest.core.singlevm.VmReady1`

Deploy a single VM reachable via ssh (scenario2)

It creates new user/project before creating and configuring all tenant network resources, flavors, images, etc. required by advanced testcases.

It ensures that all testcases inheriting from SingleVm2 could work without specific configurations (or at least read the same config data).

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

functest.core.tenantnetwork module

Ease deploying tenant networks

It offers a simple way to create all tenant network resources required by a testcase (including all Functest ones):

- TenantNetwork1 selects the user and the project set as env vars

- TenantNetwork2 creates a user and project to isolate the same resources

This classes could be reused by more complexed scenarios (Single VM)

```
class functest.core.tenantnetwork.NewProject (cloud, case_name, guid)
```

Bases: object

Ease creating new projects/users

```
clean ()
```

Remove projects/users

```
create ()
```

Create projects/users

```
get_environ ()
```

Get new environ

```
class functest.core.tenantnetwork.TenantNetwork1 (**kwargs)
```

Bases: xtesting.core.testcase.TestCase

Create a tenant network (scenario1)

It creates and configures all tenant network resources required by advanced testcases (subnet, network and router).

It ensures that all testcases inheriting from TenantNetwork1 could work without network specific configurations (or at least read the same config data).

```
cidr = '192.168.120.0/24'
```

```
clean ()
```

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

```
create_network_resources ()
```

Create all tenant network resources

It creates a router which gateway is the external network detected. The new subnet is attached to that router.

Raises: exception on error

```
static get_default_role (cloud, member='Member')
```

Get the default role

It also tests the role in lowercase to avoid possible conflicts.

```
static get_external_network (cloud)
```

Return the configured external network name or the first retrieved external network name

```
static get_public_auth_url (cloud)
```

Get Keystone public endpoint

```
run (**kwargs)
```

Run the test case.

It allows running TestCase and getting its execution status.

The subclasses must override the default implementation which is false on purpose.

The new implementation must set the following attributes to push the results to DB:

- result,
- start_time,

- stop_time.

Args: kwargs: Arbitrary keyword arguments.

shared_network = False

class `functest.core.tenantnetwork.TenantNetwork2` (**kwargs)

Bases: `functest.core.tenantnetwork.TenantNetwork1`

Create a tenant network (scenario2)

It creates new user/project before creating and configuring all tenant network resources required by a testcase (subnet, network and router).

It ensures that all testcases inheriting from TenantNetwork2 could work without network specific configurations (or at least read the same config data).

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

Module contents

functest.opnfv_tests package

Subpackages

functest.opnfv_tests.openstack package

Subpackages

functest.opnfv_tests.openstack.api package

Submodules

functest.opnfv_tests.openstack.api.connection_check module

Verify the connection to OpenStack Services

class `functest.opnfv_tests.openstack.api.connection_check.ConnectionCheck` (**kwargs)

Bases: `xtesting.core.testcase.TestCase`

Perform simplest queries

func_list = ['get_network_extensions', 'list_aggregates', 'list_domains', 'list_endpoints']

run (**kwargs)

Run all read operations to check connections

Module contents

functest.opnfv_tests.openstack.cinder package

Submodules

functest.opnfv_tests.openstack.cinder.cinder_test module

CinderCheck testcase.

class `functest.opnfv_tests.openstack.cinder.cinder_test.CinderCheck (**kwargs)`
 Bases: `functest.core.singlevm.SingleVm2`

CinderCheck testcase implementation.

Class to execute the CinderCheck test using 2 Floating IPs to connect to the VMs and one data volume

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

execute ()

Execute CinderCheck testcase.

Sets up the OpenStack keypair, router, security group, and VM instance objects then validates cinder.
 :return: the exit code from the super.execute() method

prepare ()

Create the security group and the keypair

It can be overridden to set other rules according to the services running in the VM

Raises: Exception on error

volume_timeout = 60

Module contents

functest.opnfv_tests.openstack.patrole package

Submodules

functest.opnfv_tests.openstack.patrole.patrole module

class `functest.opnfv_tests.openstack.patrole.patrole.Patrole (**kwargs)`
 Bases: `functest.opnfv_tests.openstack.tempest.tempest.TempestCommon`

configure (**kwargs)

Create all openstack resources for tempest-based testcases and write tempest.conf.

run (**kwargs)

Boot the new VM

Here are the main actions: - publish the image - create the flavor

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

Module contents

functest.opnfv_tests.openstack.rally package

Submodules

functest.opnfv_tests.openstack.rally.rally module

Rally testcases implementation.

class `functest.opnfv_tests.openstack.rally.rally.RallyBase` (***kwargs*)
Bases: `functest.core.singlevm.VmReady2`

Base class form Rally testcases implementation.

apply_blacklist (*case_file_name, result_file_name*)
Apply blacklist.

blacklist_file = `'/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/s`

build_task_args (*test_name*)
Build arguments for the Rally task.

clean ()
Cleanup of OpenStack resources. Should be called on completion.

static clean_rally_conf (*rally_conf='etc/rally/rally.conf'*)
Clean Rally config

static clean_rally_logs (*rally_conf='etc/rally/rally.conf'*)
Clean Rally config

concurrency = 4

static create_rally_deployment (*environ=None*)
Create new rally deployment

excl_func ()
Exclude functionalities.

static excl_scenario ()
Exclude scenario.

static export_task (*file_name, export_type='html'*)
Export all task results (e.g. html or xunit report)
Raises: `subprocess.CalledProcessError`: if Rally doesn't return 0
Returns: None

static file_is_empty (*file_name*)
Determine is a file is empty.

static get_task_id (*tag*)
Get task id from command rally result.

Parameters *tag* –

Returns *task_id* as string

static get_verifier_deployment_id ()
Returns deployment id for active Rally deployment

static in_iterable_re (*needle, haystack*)
Check if given needle is in the iterable haystack, using regex.

Parameters

- **needle** – string to be matched

- **haystack** – iterable of strings (optionally regex patterns)

Returns True if needle is equal to any of the elements in haystack, or if a nonempty regex pattern in haystack is found in needle.

is_successful ()

The overall result of the test.

iterations_amount = 10

prepare_run (**kwargs)

Prepare resources needed by test scenarios.

prepare_task (test_name)

Prepare resources for test run.

rally_aar4_patch_path = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api'

rally_conf_path = '/etc/rally/rally.conf'

rally_dir = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/stable'

rally_scenario_dir = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/stable'

run (**kwargs)

Run testcase.

run_task (test_name)

Run a task.

run_tests (**kwargs)

Execute tests.

shared_network = True

stests = ['authenticate', 'glance', 'cinder', 'gnocchi', 'heat', 'keystone', 'neutron']

support_dir = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/stable'

task_dir = '/home/opnfv/functest/data/rally/task'

static task_succeed (json_raw)

Parse JSON from rally JSON results.

Parameters json_raw –

Returns Bool

task_timeout = 3600

temp_dir = '/home/opnfv/functest/data/rally/task/var'

template_dir = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/stable'

tenants_amount = 3

static update_keystone_default_role (rally_conf='/etc/rally/rally.conf')

Set keystone_default_role in rally.conf

static update_rally_logs (res_dir, rally_conf='/etc/rally/rally.conf')

Print rally logs in res dir

users_amount = 2

static verify_report (file_name, uuid, export_type='html')

Generate the verifier report (e.g. html or xunit report)

Raises: subprocess.CalledProcessError: if Rally doesn't return 0

Returns: None

```
visibility = 'public'
volume_service_type = 'volumev3'
volume_version = 3
```

```
class functest.opnfv_tests.openstack.rally.rally.RallyFull (**kwargs)
    Bases: functest.opnfv_tests.openstack.rally.rally.RallyBase
```

Rally full testcase implementation.

```
task_timeout = 7200
```

```
class functest.opnfv_tests.openstack.rally.rally.RallyJobs (**kwargs)
    Bases: functest.opnfv_tests.openstack.rally.rally.RallyBase
```

Rally OpenStack CI testcase implementation.

```
apply_blacklist (case_file_name, result_file_name)
    Apply blacklist.
```

```
build_task_args (test_name)
    Build arguments for the Rally task.
```

```
prepare_run (**kwargs)
    Create resources needed by test scenarios.
```

```
prepare_task (test_name)
    Prepare resources for test run.
```

```
stests = ['neutron']
```

```
task_timeout = 7200
```

```
class functest.opnfv_tests.openstack.rally.rally.RallySanity (**kwargs)
    Bases: functest.opnfv_tests.openstack.rally.rally.RallyBase
```

Rally sanity testcase implementation.

Module contents

functest.opnfv_tests.openstack.refstack package

Submodules

functest.opnfv_tests.openstack.refstack.refstack module

Refstack testcase implementation.

```
class functest.opnfv_tests.openstack.refstack.refstack.Refstack (**kwargs)
    Bases: functest.opnfv_tests.openstack.tempest.tempest.TempestCommon
```

Refstack testcase implementation class.

```
generate_test_list (**kwargs)
    Generate test list based on the test mode.
```

Module contents

functest.opnfv_tests.openstack.shaker package

Submodules

functest.opnfv_tests.openstack.shaker.shaker module

Shaker wraps around popular system network testing tools like iperf, iperf3 and netperf (with help of flent). Shaker is able to deploy OpenStack instances and networks in different topologies. Shaker scenario specifies the deployment and list of tests to execute.

```
class functest.opnfv_tests.openstack.shaker.shaker.Shaker (**kwargs)
    Bases: functest.core.singlevm.SingleVm2

    Run shaker full+perf I2 and I3

    check_console_loop = 12

    check_requirements ()
        Check the requirements of the test case.
        It can be overridden on purpose.

    clean ()
        Clean the resources.
        It can be overridden if resources must be deleted after running the test case.

    create_server_timeout = 300

    execute ()

        Returns:
        • 0 if success
        • 1 on operation error

    filename = '/home/opnfv/functest/images/shaker-image-1.3.0+stretch.qcow2'
    flavor_disk = 3
    flavor_ram = 512
    flavor_vcpus = 1
    port = 9000

    prepare ()
        Create the security group and the keypair
        It can be overridden to set other rules according to the services running in the VM
        Raises: Exception on error

    quota_cores = -1
    quota_instances = -1
    shaker_timeout = '3600'
    ssh_connect_loops = 12
    username = 'debian'
```

Module contents

functest.opnfv_tests.openstack.tempest package

Submodules

functest.opnfv_tests.openstack.tempest.tempest module

Tempest testcases implementation.

```
class functest.opnfv_tests.openstack.tempest.tempest.TempestCommon (**kwargs)
    Bases: functest.core.singlevm.VmReady2

    TempestCommon testcases implementation class.

apply_tempest_blacklist (black_list)
    Exclude blacklisted test cases.

static backup_tempest_config (conf_file, res_dir)
    Copy config file to tempest results directory

check_extensions ()
    Check the mandatory network extensions.

check_requirements ()
    Check the requirements of the test case.

    It can be overridden on purpose.

check_services ()
    Check the mandatory services.

clean ()
    Cleanup all OpenStack objects. Should be called on completion.

static clean_rally_conf (rally_conf='etc/rally/rally.conf')
    Clean Rally config

configure (**kwargs)
    Create all openstack resources for tempest-based testcases and write tempest.conf.

static configure_tempest_update_params (tempest_conf_file, image_id=None, flavor_id=None, compute_cnt=1, image_alt_id=None, flavor_alt_id=None, admin_role_name='admin', cidr='192.168.120.0/24', domain_id='default')
    Add/update needed parameters into tempest.conf file

static configure_verifier (deployment_dir)
    Execute rally verify configure-verifier, which generates tempest.conf

static create_verifier ()
    Create new verifier

filename_alt = '/home/opnfv/functest/images/cirros-0.4.0-x86_64-disk.img'

generate_test_list (**kwargs)
    Generate test list based on the test mode.
```

static get_verifier_deployment_dir (*verifier_id, deployment_id*)

Returns Rally deployment directory for current verifier

static get_verifier_id ()

Returns verifier id for current Tempest

static get_verifier_repo_dir (*verifier_id*)

Returns installed verifier repo directory for Tempest

static get_verifier_result (*verif_id*)

Retrieve verification results.

is_successful ()

The overall result of the test.

parse_verifier_result ()

Parse and save test results.

static read_file (*filename*)

Read file and return content as a stripped list.

run (***kwargs*)

Boot the new VM

Here are the main actions: - publish the image - create the flavor

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

run_verifier_tests (***kwargs*)

Execute tempest test cases.

shared_network = True

tempest_blacklist = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/

tempest_conf_yaml = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/

tempest_custom = '/home/docs/checkouts/readthedocs.org/user_builds/functest-api/envs/

tempest_public_blacklist = '/home/docs/checkouts/readthedocs.org/user_builds/functest-

update_auth_section ()

Update auth section in tempest.conf

update_compute_section ()

Update compute section in tempest.conf

update_default_role (*rally_conf='/etc/rally/rally.conf'*)

Detect and update the default role if required

update_network_section ()

Update network section in tempest.conf

update_rally_regex (*rally_conf='/etc/rally/rally.conf'*)

Set image name as tempest img_name_regex

update_scenario_section ()

Update scenario section in tempest.conf

static update_tempest_conf_file (*conf_file, rconfig*)

Update defined paramters into tempest config file

update_validation_section ()

Update validation section in tempest.conf

visibility = 'public'

```
class functest.opnfv_tests.openstack.tempest.tempest.TempestHeat (**kwargs)
    Bases: functest.opnfv_tests.openstack.tempest.tempest.TempestCommon

    Tempest Heat testcase implementation class.

    clean ()
        Cleanup all OpenStack objects. Should be called on completion.

    configure (**kwargs)
        Create all openstack resources for tempest-based testcases and write tempest.conf.

    filename_alt = '/home/opnfv/functest/images/Fedora-Cloud-Base-30-1.2.x86_64.qcow2'

    flavor_alt_disk = 4

    flavor_alt_ram = 512

    flavor_alt_vcpus = 1

class functest.opnfv_tests.openstack.tempest.tempest.TempestHorizon (**kwargs)
    Bases: functest.opnfv_tests.openstack.tempest.tempest.TempestCommon

    Tempest Horizon testcase implementation class.

    configure (**kwargs)
        Create all openstack resources for tempest-based testcases and write tempest.conf.
```

Module contents

functest.opnfv_tests.openstack.vmtip package

Submodules

functest.opnfv_tests.openstack.vmtip.vmtip module

VMTP is a small python application that will automatically perform ping connectivity, round trip time measurement (latency) and TCP/UDP throughput measurement for the following East/West flows on any OpenStack deployment:

- VM to VM same network (private fixed IP, flow #1)
- VM to VM different network using fixed IP (same as intra-tenant L3 fixed IP, flow #2)
- VM to VM different network using floating IP and NAT (same as floating IP inter-tenant L3, flow #3)

```
class functest.opnfv_tests.openstack.vmtip.vmtip.Vmtip (**kwargs)
    Bases: functest.core.singlevm.VmReady2

    Class to run Vmtip as an OPNFV Functest testcase

    check_requirements ()
        Check the requirements of the test case.

        It can be overridden on purpose.

    clean ()
        Clean the resources.

        It can be overridden if resources must be deleted after running the test case.

    create_network_resources ()
        Create router
```

It creates a router which gateway is the external network detected.

Raises: exception on error

create_server_timeout = 300

filename = '/home/opnfv/functest/images/ubuntu-14.04-server-cloudimg-amd64-disk1.img'

flavor_disk = 0

flavor_ram = 2048

flavor_vcpus = 1

generate_keys ()

Generate Keys

Raises: Exception on error

run (**kwargs)

Boot the new VM

Here are the main actions: - publish the image - create the flavor

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

run_vmtop ()

Run Vmtop and generate charts

Raises: Exception on error

ssh_retry_timeout = 240

write_config ()

Write vmtop.conf

Raises: Exception on error

Module contents

functest.opnfv_tests.openstack.vping package

Submodules

functest.opnfv_tests.openstack.vping.vping_ssh module

vPingSSH testcase.

class `functest.opnfv_tests.openstack.vping.vping_ssh.VPingSSH` (**kwargs)

Bases: `functest.core.singlevm.SingleVm2`

VPingSSH testcase implementation.

Class to execute the vPing test using a Floating IP to connect to one VM to issue the ping command to the second

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

execute ()

Ping the second VM

Returns: ping exit codes

prepare ()

Create the security group and the keypair

It can be overridden to set other rules according to the services running in the VM

Raises: Exception on error

functest.opnfv_tests.openstack.vping.vping_userdata module

vping_userdata testcase.

class `functest.opnfv_tests.openstack.vping.vping_userdata.VPingUserData (**kwargs)`

Bases: `functest.core.singlevm.VmReady2`

Class to execute the vPing test using userdata and the VM's console

clean ()

Clean the resources.

It can be overridden if resources must be deleted after running the test case.

run (kwargs)**

Sets up the OpenStack VM instance objects then executes the ping and validates. :return: the exit code from the super.execute() method

Module contents

Module contents

functest.opnfv_tests.sdn package

Subpackages

functest.opnfv_tests.sdn.odl package

Submodules

functest.opnfv_tests.sdn.odl.odl module

Define classes required to run ODL suites.

It has been designed for any context. But helpers are given for running test suites in OPNFV environment.

Example: `$ python odl.py`

class `functest.opnfv_tests.sdn.odl.odl.ODLParser`

Bases: `object`

Parser to run ODL test suites.

parse_args (*argv=None*)

Parse arguments.

It can call `sys.exit` if arguments are incorrect.

Returns: the arguments from cmdline

class `functest.opnfv_tests.sdn.odl.odl.ODLTests` (**kwargs)

Bases: `xtesting.core.robotframework.RobotFramework`

ODL test runner.

basic_suite_dir = `'/src/odl_test/csit/suites/integration/basic'`

default_suites = `['/src/odl_test/csit/suites/integration/basic', '/src/odl_test/csit/s`

neutron_suite_dir = `'/src/odl_test/csit/suites/openstack/neutron'`

odl_test_repo = `'/src/odl_test'`

odl_variables_file = `'/src/odl_test/csit/variables/Variables.robot'`

run (**kwargs)

Run suites in OPNFV environment

It basically checks env vars to call `main()` with the keywords required.

Args: kwargs: Arbitrary keyword arguments.

Returns: EX_OK if all suites ran well. EX_RUN_ERROR otherwise.

run_suites (*suites=None*, **kwargs)

Run the test suites

It has been designed to be called in any context. It requires the following keyword arguments:

- odlusername,
- odlpassword,
- osauthurl,
- neutronurl,
- osusername,
- osprojectname,
- ospassword,
- odlip,
- odlwebport,
- odlrestconfport.

Here are the steps:

- set all `RobotFramework_variables`,
- create the output directories if required,
- get the results in `output.xml`,
- delete temporary files.

Args: kwargs: Arbitrary keyword arguments.

Returns: EX_OK if all suites ran well. EX_RUN_ERROR otherwise.

```
classmethod set_robotframework_vars (odlusername='admin', odlpassword='admin')  
    Set credentials in csit/variables/Variables.robot.
```

Returns: True if credentials are set. False otherwise.

```
functest.opnfv_tests.sdn.odl.odl.main()  
    Entry point
```

Module contents

Module contents

functest.opnfv_tests.vnf package

Subpackages

functest.opnfv_tests.vnf.epc package

Submodules

functest.opnfv_tests.vnf.epc.juju_epc module

Juju testcase implementation.

```
class functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc (**kwargs)
```

Bases: *functest.core.singlevm.SingleVm2*

Abot EPC deployed with JUJU Orchestrator Case

```
check_app (name='abot-epc-basic', status='active')  
    Check application status.
```

```
cidr = '192.168.120.0/24'
```

```
clean ()  
    Clean created objects/functions.
```

```
deploy_orchestrator ()  
    Create network, subnet, router
```

Bootstrap juju

```
deploy_vnf ()  
    Deploy ABOT-OAI-EPC.
```

```
execute ()  
    Prepare testcase (Additional pre-configuration steps).
```

```
filename = '/home/opnfv/functest/images/ubuntu-16.04-server-cloudimg-amd64-disk1.img'
```

```
filename_alt = '/home/opnfv/functest/images/ubuntu-14.04-server-cloudimg-amd64-disk1.i
```

```
flavor_alt_disk = 10
```

```
flavor_alt_ram = 4096
```

```
flavor_alt_vcpus = 1
```

```
flavor_disk = 10
```

```
flavor_ram = 2048
```

```
flavor_vcpus = 1
```

```
juju_timeout = '4800'
```

```
publish_image_alt (name=None)
```

Publish alternative image

It allows publishing multiple images for the child testcases. It forces the same configuration for all sub-testcases.

Returns: image

Raises: exception on error

```
test_vnf ()
```

Run test on ABoT.

```
username = 'ubuntu'
```

```
functest.opnfv_tests.vnf.epc.juju_epc.process_abot_test_result (file_path)
```

Process ABoT Result

```
functest.opnfv_tests.vnf.epc.juju_epc.sig_test_format (sig_test)
```

Process the signaling result to have a short result

```
functest.opnfv_tests.vnf.epc.juju_epc.update_data (obj)
```

Update Result data

Module contents

functest.opnfv_tests.vnf.ims package

Submodules

functest.opnfv_tests.vnf.ims.clearwater module

Ease testing any Clearwater deployment

```
class functest.opnfv_tests.vnf.ims.clearwater.ClearwaterTesting (case_name,  
                                                             bono_ip,  
                                                             ellis_ip)
```

Bases: object

vIMS clearwater base usable by several orchestrators

```
availability_check (signup_code='secret', two_numbers=False)
```

Create one or two numbers

```
run_clearwater_live_test (public_domain, signup_code='secret')
```

Run the Clearwater live tests

It first runs dnsmasq to reach clearwater services by FQDN and then the Clearwater live tests. All results are saved in `ims_test_output.txt`.

Returns:

- a dict containing the overall results
- None on error

functest.opnfv_tests.vnf.ims.cloudify_ims module

functest.opnfv_tests.vnf.ims.heat_ims module

HeatIms testcase implementation.

class `functest.opnfv_tests.vnf.ims.heat_ims.HeatIms (**kwargs)`

Bases: `functest.core.singlevm.VmReady2`

Clearwater vIMS deployed with Heat Orchestrator Case.

clean ()

Clean created objects/functions.

create_network_resources ()

Create all tenant network resources

It creates a router which gateway is the external network detected. The new subnet is attached to that router.

Raises: exception on error

deploy_vnf ()

Deploy Clearwater IMS.

execute ()

Prepare Tenant/User

network, security group, fip, VM creation

`filename = '/home/opnfv/functest/images/ubuntu-14.04-server-cloudimg-amd64-disk1.img'`

`flavor_disk = 3`

`flavor_ram = 1024`

`flavor_vcpus = 1`

`parameters = {'private_mgmt_net_cidr': '192.168.100.0/24', 'private_mgmt_net_gateway'`

`quota_port = 50`

`quota_security_group = 20`

`quota_security_group_rule = 100`

run (**kwargs)

Deploy and test clearwater

Here are the main actions: - deploy clearwater stack via heat - test the vnf instance

Returns: - TestCase.EX_OK - TestCase.EX_RUN_ERROR on error

test_vnf ()

Run test on clearwater ims instance.

Module contents

functest.opnfv_tests.vnf.router package

Subpackages

functest.opnfv_tests.vnf.router.test_controller package

Submodules

functest.opnfv_tests.vnf.router.test_controller.function_test_exec module

vrouter function test execution module

```

class functest.opnfv_tests.vnf.router.test_controller.function_test_exec.FunctionTestExec (
    Bases: object
    vrouter function test execution class
    config_reference_vnf (target_vnf, reference_vnf, test_kind)
    config_target_vnf (target_vnf, reference_vnf, test_kind)
    logger = <Logger functest.opnfv_tests.vnf.router.test_controller.function_test_exec (WARNING)>
    result_check (target_vnf, reference_vnf, test_kind, test_list)
    run (target_vnf, reference_vnf_list, test_info, test_list)

```

Module contents

functest.opnfv_tests.vnf.router.vnf_controller package

Submodules

functest.opnfv_tests.vnf.router.vnf_controller.checker module

vrouter test result check module

```

class functest.opnfv_tests.vnf.router.vnf_controller.checker.Checker
    Bases: object
    vrouter test result check class
    static load_check_rule (rule_file_dir, rule_file_name, parameter)
    logger = <Logger functest.opnfv_tests.vnf.router.vnf_controller.checker (WARNING)>
    static regexp_information (response, rules)

```

functest.opnfv_tests.vnf.router.vnf_controller.command_generator module

command generator module for vrouter testing

```

class functest.opnfv_tests.vnf.router.vnf_controller.command_generator.CommandGenerator
    Bases: object
    command generator class for vrouter testing
    static command_create (template, parameter)
    static load_template (template_dir, template)
    logger = <Logger functest.opnfv_tests.vnf.router.vnf_controller.command_generator (WARNING)>

```

functest.opnfv_tests.vnf.router.vnf_controller.ssh_client module

ssh client module for vrouter testing

```
class functest.opnfv_tests.vnf.router.vnf_controller.ssh_client.SshClient (ip_address,  
user,  
pass-  
word=None,  
key_filename=None)  
  
Bases: object  
ssh client class for vrouter testing  
close ()  
connect (time_out=10, retrycount=10)  
static error_check (response, err_strs=None)  
logger = <Logger functest.opnfv_tests.vnf.router.vnf_controller.ssh_client (WARNING)>  
send (cmd, prompt, timeout=10)
```

functest.opnfv_tests.vnf.router.vnf_controller.vm_controller module

vm controll module

```
class functest.opnfv_tests.vnf.router.vnf_controller.vm_controller.VmController (util_info)  
Bases: object  
vm controll class  
command_create_and_execute (ssh, test_cmd_file_path, cmd_input_param, prompt_file_path)  
command_execute (ssh, command, prompt)  
command_gen_from_template (command_file_path, cmd_input_param)  
command_list_execute (ssh, command_list, prompt)  
config_vm (vm_info, test_cmd_file_path, cmd_input_param, prompt_file_path)  
connect_ssh_and_config_vm (vm_info, test_cmd_file_path, cmd_input_param,  
prompt_file_path)  
logger = <Logger functest.opnfv_tests.vnf.router.vnf_controller.vm_controller (WARNING)>
```

functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller module

vrouter controll module

```
class functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController (util_info)  
Bases: object  
vrouter controll class  
config_vnf (source_vnf, destination_vnf, test_cmd_file_path, parameter_file_path, prompt_file_path)  
logger = <Logger functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller (WARNING)>  
output_check_result_detail_data (res_data_list)
```

```
result_check (target_vnf, reference_vnf, check_rule_file_path_list, parameter_file_path,  
              prompt_file_path)
```

Module contents

Submodules

functest.opnfv_tests.vnf.router.cloudify_vrouter module

functest.opnfv_tests.vnf.router.utilvnf module

Utility module of vrouter testcase

```
class functest.opnfv_tests.vnf.router.utilvnf.Utilvnf  
    Bases: object  
    Utility class of vrouter testcase  
  
    static convert_functional_test_result (result_data_list)  
get_address (server_name, network_name)  
get_blueprint_outputs (cfy_manager_ip, deployment_name)  
get_blueprint_outputs_networks (cfy_manager_ip, deployment_name)  
get_blueprint_outputs_vnfs (cfy_manager_ip, deployment_name)  
get_mac_address (server_name, network_name)  
static get_reference_vnf_list (vnf_info_list)  
static get_target_vnf (vnf_info_list)  
static get_test_scenario (file_path)  
static get_vnf_info (vnf_info_list, vnf_name)  
get_vnf_info_list (cfy_manager_ip, topology_deploy_name, target_vnf_name)  
logger = <Logger functest.opnfv_tests.vnf.router.utilvnf (WARNING)>  
output_test_result_json ()  
request_vm_delete (vnf_info_list)  
set_credentials (cloud)  
write_result_data (result_data)
```

functest.opnfv_tests.vnf.router.vrouter_base module

vrouter testing base class module

```
class functest.opnfv_tests.vnf.router.vrouter_base.VrouterOnBoardingBase (util,  
                                                                    util_info)  
    Bases: object  
    vrouter testing base class  
  
    function_test_vrouter (target_vnf_name, test_info)  
        function test execution
```

```
get_vnf_info_list (target_vnf_name)  
test_vnf ()  
    vrouter test execution
```

Module contents

Module contents

Module contents

functest.utils package

Submodules

functest.utils.config module

```
class functest.utils.config.Config  
    Bases: object  
  
    fill ()  
  
    patch_file (patch_file_path)
```

functest.utils.constants module

functest.utils.env module

```
functest.utils.env.get (env_var)  
functest.utils.env.string ()
```

functest.utils.functest_utils module

```
functest.utils.functest_utils.convert_dict_to_ini (value)  
    Convert dict to oslo.conf input  
  
functest.utils.functest_utils.convert_ini_to_dict (value)  
    Convert oslo.conf input to dict  
  
functest.utils.functest_utils.convert_ini_to_list (value)  
    Convert list to oslo.conf input  
  
functest.utils.functest_utils.convert_list_to_ini (value)  
    Convert list to oslo.conf input  
  
functest.utils.functest_utils.execute_command (cmd, info=False, error_msg="", verbose=True, output_file=None)  
  
functest.utils.functest_utils.execute_command_raise (cmd, info=False, error_msg="", verbose=True, output_file=None)
```

`functest.utils.functest_utils.get_nova_version (cloud)`

Get Nova API microversion

Returns:

- Nova API microversion
- None on operation error

`functest.utils.functest_utils.get_openstack_version (cloud)`

Detect OpenStack version via Nova API microversion

It follows [MicroversionHistory](#).

Returns:

- OpenStack release
- Unknown on operation error

`functest.utils.functest_utils.get_parameter_from_yaml (parameter, yfile)`

Returns the value of a given parameter in file.yaml parameter must be given in string format with dots Example: `general.openstack.image_name`

`functest.utils.functest_utils.list_services (cloud)`

Search Keystone services via `$OS_INTERFACE`.

It mainly conforms with [Shade](#) but allows testing vs public endpoints. It's worth mentioning that it doesn't support keystone v2.

Returns a list of `munch.Munch` containing the services description

Raises `OpenStackCloudException` if something goes wrong during the openstack API call.

`functest.utils.functest_utils.search_services (cloud, name_or_id=None, filters=None)`

Search Keystone services via `$OS_INTERFACE`.

It mainly conforms with [Shade](#) but allows testing vs public endpoints. It's worth mentioning that it doesn't support keystone v2.

Parameters

- **name_or_id** – Name or id of the desired service.
- **filters** – a dict containing additional filters to use. e.g. `{'type': 'network'}`.

Returns a list of `munch.Munch` containing the services description

Raises `OpenStackCloudException` if something goes wrong during the openstack API call.

Module contents

1.1.2 Module contents

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

Python Module Index

f

functest, 27

functest.core, 8

functest.core.singlevm, 3

functest.core.tenantnetwork, 6

functest.opnfv_tests, 26

functest.opnfv_tests.openstack, 18

functest.opnfv_tests.openstack.api, 8

functest.opnfv_tests.openstack.api.connection_check, 8

functest.opnfv_tests.openstack.cinder, 9

functest.opnfv_tests.openstack.cinder.cinder_test, 9

functest.opnfv_tests.openstack.patrole, 9

functest.opnfv_tests.openstack.patrole.patrole, 9

functest.opnfv_tests.openstack.rally, 12

functest.opnfv_tests.openstack.rally.rally, 10

functest.opnfv_tests.openstack.refstack, 13

functest.opnfv_tests.openstack.refstack.refstack, 12

functest.opnfv_tests.openstack.shaker, 14

functest.opnfv_tests.openstack.shaker.shaker, 13

functest.opnfv_tests.openstack.tempest, 16

functest.opnfv_tests.openstack.tempest.tempest, 14

functest.opnfv_tests.openstack.vmtop, 17

functest.opnfv_tests.openstack.vmtop.vmtop, 16

functest.opnfv_tests.openstack.vping, 18

functest.opnfv_tests.openstack.vping.vping_ssh, 17

functest.opnfv_tests.openstack.vping.vping_userdata, 18

functest.opnfv_tests.sdn, 20

functest.opnfv_tests.sdn.odl, 20

functest.opnfv_tests.sdn.odl.odl, 18

functest.opnfv_tests.vnf, 26

functest.opnfv_tests.vnf.epc, 21

functest.opnfv_tests.vnf.epc.juju_epc, 20

functest.opnfv_tests.vnf.ims, 22

functest.opnfv_tests.vnf.ims.clearwater, 21

functest.opnfv_tests.vnf.ims.heat_ims, 22

functest.opnfv_tests.vnf.router, 26

functest.opnfv_tests.vnf.router.test_controller, 23

functest.opnfv_tests.vnf.router.test_controller.functest, 23

functest.opnfv_tests.vnf.router.utilvnf, 25

functest.opnfv_tests.vnf.router.vnf_controller, 25

functest.opnfv_tests.vnf.router.vnf_controller.check, 23

functest.opnfv_tests.vnf.router.vnf_controller.com, 23

functest.opnfv_tests.vnf.router.vnf_controller.ssh, 24

functest.opnfv_tests.vnf.router.vnf_controller.vmtop, 24

functest.opnfv_tests.vnf.router.vnf_controller.vnf, 24

functest.opnfv_tests.vnf.router.vrouter_base, 25

functest.utils, 27

functest.utils.config, 26

functest.utils.constants, 26

`functest.utils.env`, [26](#)

`functest.utils.functest_utils`, [26](#)

A

apply_blacklist() (functest.opnfv_tests.openstack.rally.rally.RallyBase method), 10
 apply_blacklist() (functest.opnfv_tests.openstack.rally.rally.RallyJobs method), 12
 apply_tempest_blacklist() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 14
 availability_check() (functest.opnfv_tests.vnf.ims.clearwater.ClearwaterTesting method), 21
 check_console_regex (functest.core.singlevm.SingleVm1 attribute), 3
 check_extensions() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 14
 check_regex_in_console() (functest.core.singlevm.VmReady1 method), 5
 check_requirements() (functest.opnfv_tests.openstack.shaker.shaker.Shaker method), 13
 check_requirements() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 14

B

backup_tempest_config() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 14
 basic_suite_dir (functest.opnfv_tests.sdn.odl.odl.ODLTests attribute), 19
 blacklist_file (functest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 10
 boot_vm() (functest.core.singlevm.VmReady1 method), 4
 build_task_args() (functest.opnfv_tests.openstack.rally.rally.RallyBase method), 10
 build_task_args() (functest.opnfv_tests.openstack.rally.rally.RallyJobs method), 12
 check_services() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 14
 Checker (class in functest.opnfv_tests.vnf.router.vnf_controller.checker), 23
 cidr (functest.core.tenantnetwork.TenantNetwork1 attribute), 7
 cidr (functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc attribute), 20
 CinderCheck (class in functest.opnfv_tests.openstack.cinder.cinder_test), 9
 clean() (functest.core.singlevm.SingleVm1 method), 3
 clean() (functest.core.singlevm.SingleVm2 method), 4
 clean() (functest.core.singlevm.VmReady1 method), 5
 clean() (functest.core.singlevm.VmReady2 method), 6
 clean() (functest.core.tenantnetwork.NewProject method), 7

C

check_app() (functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc method), 20
 check_console_loop (functest.core.singlevm.SingleVm1 attribute), 3
 check_console_loop (functest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13
 clean() (functest.core.tenantnetwork.TenantNetwork1 method), 7
 clean() (functest.core.tenantnetwork.TenantNetwork2 method), 8
 clean() (functest.opnfv_tests.openstack.cinder.cinder_test.CinderCheck method), 9
 clean() (functest.opnfv_tests.openstack.rally.rally.RallyBase method), 10

method), 10
 clean() (functest.opnfv_tests.openstack.shaker.shaker.Shaker method), 23
 clean() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 13
 clean() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 14
 clean() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 16
 clean() (functest.opnfv_tests.openstack.vmt.vmt.Vmt config_vnf() (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController method), 16
 clean() (functest.opnfv_tests.openstack.vping.vping_ssh.VPingSSH config_ssh() (functest.opnfv_tests.openstack.patrole.patrole.Patrole method), 17
 clean() (functest.opnfv_tests.openstack.vping.vping_userdata.VPingUserData (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 18
 clean() (functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc configure() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 20
 clean() (functest.opnfv_tests.vnf.ims.heat_ims.HeatIms configure() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 22
 clean_orphan_security_groups() (functest.core.singlevm.VmReady1 method), 5
 clean_rally_conf() (functest.opnfv_tests.openstack.rally.rally.RallyBase static method), 10
 clean_rally_conf() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 14
 clean_rally_logs() (functest.opnfv_tests.openstack.rally.rally.RallyBase static method), 10
 ClearwaterTesting (class in functest.opnfv_tests.vnf.ims.clearwater), 21
 close() (functest.opnfv_tests.vnf.router.vnf_controller.ssh_client.SshClient method), 24
 command_create() (functest.opnfv_tests.vnf.router.vnf_controller.command_generator.CommandGenerator static method), 23
 command_create_and_execute() (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController method), 24
 command_execute() (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController method), 24
 command_gen_from_template() (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController method), 24
 command_list_execute() (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController method), 24
 CommandGenerator (class in functest.opnfv_tests.vnf.router.vnf_controller.command_generator), 23
 concurrency (functest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 10
 Config (class in functest.utils.config), 26
 config_reference_vnf() (functest.core.singlevm.VmReady1 method), 5

create_floating_ip_timeout (*functest.core.singlevm.SingleVm1* attribute), 4
 create_network_resources (*functest.core.tenantnetwork.TenantNetwork1* method), 7
 create_network_resources (*functest.opnfv_tests.openstack.vmtplib.Vmtplib* method), 16
 create_network_resources (*functest.opnfv_tests.vnf.ims.heat_ims.HeatIms* method), 22
 create_rally_deployment (*functest.opnfv_tests.openstack.rally.rally.RallyBase* static method), 10
 create_server_timeout (*functest.core.singlevm.VmReady1* attribute), 5
 create_server_timeout (*functest.opnfv_tests.openstack.shaker.shaker.Shaker* attribute), 13
 create_server_timeout (*functest.opnfv_tests.openstack.vmtplib.Vmtplib* attribute), 17
 create_verifier (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* static method), 14

D

default_suites (*functest.opnfv_tests.sdn.odl.odl.ODLTests* attribute), 19
 deploy_orchestrator (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* method), 20
 deploy_vnf (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* method), 20
 deploy_vnf (*functest.opnfv_tests.vnf.ims.heat_ims.HeatIms* method), 22

E

error_check (*functest.opnfv_tests.vnf.router.vnf_controller.ssh_client.SshClient* static method), 24
 excl_func (*functest.opnfv_tests.openstack.rally.rally.RallyBase* method), 10
 excl_scenario (*functest.opnfv_tests.openstack.rally.rally.RallyBase* static method), 10
 execute (*functest.core.singlevm.SingleVm1* method), 4
 execute (*functest.opnfv_tests.openstack.cinder.cinder_test.CinderClient* method), 9
 execute (*functest.opnfv_tests.openstack.shaker.shaker.Shaker* method), 13
 execute (*functest.opnfv_tests.openstack.vping.vping_ssh.VPingSSH* method), 17
 execute (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* method), 20
 execute (*functest.opnfv_tests.vnf.ims.heat_ims.HeatIms* method), 22
 execute_command (*functest.utils.functest_utils*), 26
 execute_command_raise (*functest.utils.functest_utils*), 26
 export_task (*functest.opnfv_tests.openstack.rally.rally.RallyBase* static method), 10
 extra_alt_properties (*functest.core.singlevm.VmReady1* attribute), 5
 extra_properties (*functest.core.singlevm.VmReady1* attribute), 5

F

file_is_empty (*functest.opnfv_tests.openstack.rally.rally.RallyBase* static method), 10
 filename (*functest.core.singlevm.VmReady1* attribute), 5
 filename (*functest.opnfv_tests.openstack.shaker.shaker.Shaker* attribute), 13
 filename (*functest.opnfv_tests.openstack.vmtplib.Vmtplib* attribute), 17
 filename (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* attribute), 20
 filename (*functest.opnfv_tests.vnf.ims.heat_ims.HeatIms* attribute), 22
 filename_alt (*functest.core.singlevm.VmReady1* attribute), 5
 filename_alt (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* attribute), 14
 filename_alt (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* attribute), 16
 filename_alt (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* attribute), 20
 filename_alt (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* attribute), 20
 flavor_alt_disk (*functest.core.singlevm.VmReady1* attribute), 5
 flavor_alt_disk (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* attribute), 16
 flavor_alt_disk (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* attribute), 20
 flavor_alt_extra_specs (*functest.core.singlevm.VmReady1* attribute), 5
 flavor_alt_ram (*functest.core.singlevm.VmReady1* attribute), 5
 flavor_alt_ram (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* attribute), 16
 flavor_alt_ram (*functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc* attribute), 20
 flavor_alt_vcpus (*functest.core.singlevm.VmReady1* attribute), 5
 flavor_alt_vcpus (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon* attribute), 16

functest.opnfv_tests.vnf.router.vnf_controller.openstack.version() (in module
 (module), 24 *functest.utils.functest_utils*), 27
 functest.opnfv_tests.vnf.router.vnf_controller.parameters_from_yaml() (in module
 (module), 24 *functest.utils.functest_utils*), 27
 functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller
 (module), 24 (*functest.core.tenantnetwork.TenantNetwork1*
 functest.opnfv_tests.vnf.router.vrouter_base static method), 7
 (module), 25 get_reference_vnf_list()
 functest.utils (module), 27 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
 functest.utils.config (module), 26 static method), 25
 functest.utils.constants (module), 26 get_target_vnf() (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
 functest.utils.env (module), 26 static method), 25
 functest.utils.functest_utils (module), 26 get_task_id() (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
 function_test_vrouter() static method), 10
 (*functest.opnfv_tests.vnf.router.vrouter_base.VrouterOnBoardingBase*
 method), 25 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
 FunctionTestExec (class in static method), 25
functest.opnfv_tests.vnf.router.test_controller.function_test_exec)
 23 (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*
 static method), 14
G get_verifier_deployment_id()
 generate_keys() (*functest.opnfv_tests.openstack.vmt.vmt.Vmt*) (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
 method), 17 static method), 10
 generate_test_list() get_verifier_id()
 (*functest.opnfv_tests.openstack.refstack.refstack.Refstack*) (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*
 method), 12 static method), 15
 generate_test_list() get_verifier_repo_dir()
 (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*) (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*
 method), 14 static method), 15
 get() (in module *functest.utils.env*), 26 get_verifier_result()
 get_address() (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*) (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*
 method), 25 static method), 15
 get_blueprint_outputs() get_vnf_info() (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf* static method), 25
 method), 25 get_vnf_info_list()
 get_blueprint_outputs_networks() (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf* method), 25
 method), 25 get_vnf_info_list()
 get_blueprint_outputs_vnfs() (*functest.opnfv_tests.vnf.router.vrouter_base.VrouterOnBoardingBase*
 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf* method), 25
 method), 25
 get_default_role() **H**
 (*functest.core.tenantnetwork.TenantNetwork1* Heat Ims (class in *functest.opnfv_tests.vnf.ims.heat_ims*),
 static method), 7 22
 get_environ() (*functest.core.tenantnetwork.NewProject*
 method), 7
 get_external_network() image_alt_format (*functest.core.singlevm.VmReady1*
 (*functest.core.tenantnetwork.TenantNetwork1* attribute), 6
 static method), 7 image_format (*functest.core.singlevm.VmReady1* at-
 attribute), 6
 get_mac_address() in_iterable_re() (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
 (*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf* static method), 10
 method), 25
 get_nova_version() (in module is_successful() (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
functest.utils.functest_utils), 26 method), 11

is_successful() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon_detail_data() method), 15

iterations_amount (functest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11

J

juju_timeout (functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc attribute), 21

JujuEpc (class in functest.opnfv_tests.vnf.epc.juju_epc), 20

L

list_services() (in module functest.utils.functest_utils), 27

load_check_rule() (functest.opnfv_tests.vnf.router.vnf_controller.checker.Checker static method), 23

load_template() (functest.opnfv_tests.vnf.router.vnf_controller.command_generator.CommandGenerator static method), 23

logger (functest.opnfv_tests.vnf.router.test_controller.function_test_attribute.FunctionTestExec attribute), 23

logger (functest.opnfv_tests.vnf.router.utilvnf.Utilvnf attribute), 25

logger (functest.opnfv_tests.vnf.router.vnf_controller.checker.Checker attribute), 23

logger (functest.opnfv_tests.vnf.router.vnf_controller.command_generator.CommandGenerator attribute), 23

logger (functest.opnfv_tests.vnf.router.vnf_controller.ssh_client.SshClient attribute), 24

logger (functest.opnfv_tests.vnf.router.vnf_controller.vm_controller.VmController attribute), 24

logger (functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController attribute), 24

M

main() (in module functest.opnfv_tests.sdn.odl.odl), 20

N

neutron_suite_dir (functest.opnfv_tests.sdn.odl.odl.ODLTests attribute), 19

NewProject (class in functest.core.tenantnetwork), 7

O

odl_test_repo (functest.opnfv_tests.sdn.odl.odl.ODLTests attribute), 19

odl_variables_file (functest.opnfv_tests.sdn.odl.odl.ODLTests attribute), 19

ODLParser (class in functest.opnfv_tests.sdn.odl.odl), 18

ODLTests (class in functest.opnfv_tests.sdn.odl.odl), 19

output_test_result_json() (functest.opnfv_tests.vnf.router.utilvnf.Utilvnf method), 25

P

parameters (functest.opnfv_tests.vnf.ims.heat_ims.HeatIms attribute), 22

parse_args() (functest.opnfv_tests.sdn.odl.odl.ODLParser method), 18

parse_verifier_result() (functest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

patch_file() (functest.utils.config.Config method), 26

Patrole (class in functest.opnfv_tests.openstack.patrole.patrole), 13

port (functest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13

prepare() (functest.core.singlevm.SingleVm1 method), 4

prepare() (functest.opnfv_tests.openstack.cinder.cinder_test.CinderChecker method), 9

prepare() (functest.opnfv_tests.openstack.shaker.shaker.Shaker method), 13

prepare() (functest.opnfv_tests.openstack.vping.vping_ssh.VPingSSH method), 18

prepare_run() (functest.opnfv_tests.openstack.rally.rally.RallyBase method), 12

prepare_run() (functest.opnfv_tests.openstack.rally.rally.RallyJobs method), 12

prepare_task() (functest.opnfv_tests.openstack.rally.rally.RallyBase method), 11

prepare_task() (functest.opnfv_tests.openstack.rally.rally.RallyJobs method), 12

process_abot_test_result() (in module functest.opnfv_tests.vnf.epc.juju_epc), 21

publish_image() (functest.core.singlevm.VmReady1 method), 6

publish_image_alt() (functest.core.singlevm.VmReady1 method), 6

publish_image_alt() (functest.opnfv_tests.vnf.epc.juju_epc.JujuEpc method), 21

Q

quota_cores (functest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13

quota_instances (functest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13

quota_port (functest.opnfv_tests.vnf.ims.heat_ims.HeatIms attribute), 22

quota_security_group (func`test.opnfv_tests.vnf.ims.heat_ims.HeatIms` attribute), 22

quota_security_group_rule (func`test.opnfv_tests.vnf.ims.heat_ims.HeatIms` attribute), 22

R

rally_aar4_patch_path (func`test.opnfv_tests.openstack.rally.rally.RallyBase` attribute), 11

rally_conf_path (func`test.opnfv_tests.openstack.rally.rally.RallyBase` attribute), 11

rally_dir (func`test.opnfv_tests.openstack.rally.rally.RallyBase` attribute), 11

rally_scenario_dir (func`test.opnfv_tests.openstack.rally.rally.RallyBase` attribute), 11

RallyBase (class in func`test.opnfv_tests.openstack.rally.rally`), 10

RallyFull (class in func`test.opnfv_tests.openstack.rally.rally`), 12

RallyJobs (class in func`test.opnfv_tests.openstack.rally.rally`), 12

RallySanity (class in func`test.opnfv_tests.openstack.rally.rally`), 12

read_file() (func`test.opnfv_tests.openstack.tempest.tempest.TempestCommon` static method), 15

Refstack (class in func`test.opnfv_tests.openstack.refstack.refstack`), 12

regexp_information() (func`test.opnfv_tests.vnf.router.vnf_controller.checker_checker` static method), 23

request_vm_delete() (func`test.opnfv_tests.vnf.router.utilvnf.Utilvnf` method), 25

result_check() (func`test.opnfv_tests.vnf.router.test_controller.function_test_exec.FunctestTaskExec` method), 23

result_check() (func`test.opnfv_tests.vnf.router.vnf_controller.vnf_controller.VnfController` method), 24

run() (func`test.core.singlevm.SingleVm1` method), 4

run() (func`test.core.singlevm.VmReady1` method), 6

run() (func`test.core.tenantnetwork.TenantNetwork1` method), 7

run() (func`test.opnfv_tests.openstack.api.connection_checker.ConnectionChecker` method), 8

run() (func`test.opnfv_tests.openstack.patrole.patrole.Patrole` method), 9

run() (func`test.opnfv_tests.openstack.rally.rally.RallyBase` ssh_connect_loops method), 11

run() (func`test.opnfv_tests.openstack.tempest.tempest.TempestCommon` method), 15

run() (func`test.opnfv_tests.openstack.vmtop.vmtop.Vmtop` method), 17

run() (func`test.opnfv_tests.openstack.vping.vping_userdata.VPingUserData` method), 18

run() (func`test.opnfv_tests.sdn.odl.odl.ODLTests` method), 19

run() (func`test.opnfv_tests.vnf.ims.heat_ims.HeatIms` method), 22

run() (func`test.opnfv_tests.vnf.router.test_controller.function_test_exec.FunctestTaskExec` method), 23

run_clearwater_live_test() (func`test.opnfv_tests.vnf.ims.clearwater.ClearwaterTesting` method), 21

run_suites() (func`test.opnfv_tests.sdn.odl.odl.ODLTests` method), 19

run_task() (func`test.opnfv_tests.openstack.rally.rally.RallyBase` method), 11

run_tests() (func`test.opnfv_tests.openstack.rally.rally.RallyBase` method), 11

run_verifier_tests() (func`test.opnfv_tests.openstack.tempest.tempest.TempestCommon` method), 15

run_vmtop() (func`test.opnfv_tests.openstack.vmtop.vmtop.Vmtop` method), 17

S

search_services() (in module func`test.utils.functest_utils`), 27

set_credentials() (func`test.opnfv_tests.vnf.router.vnf_controller.ssh_client.SshClient` method), 24

set_credentials() (func`test.opnfv_tests.vnf.router.utilvnf.Utilvnf` method), 25

set_framework_vars() (func`test.opnfv_tests.sdn.odl.odl.ODLTests` class method), 19

Shaker (class in func`test.opnfv_tests.openstack.shaker.shaker`), 13

singlevm_controller (func`test.opnfv_tests.openstack.shaker.shaker.Shaker` attribute), 13

singlevm_controller (func`test.opnfv_tests.openstack.tenantnetwork.TenantNetwork1` attribute), 8

shared_network (func`test.opnfv_tests.openstack.rally.rally.RallyBase` attribute), 11

shared_network (func`test.opnfv_tests.openstack.tempest.tempest.TempestCommon` attribute), 15

ssh_connect_loops (in module func`test.opnfv_tests.vnf.epc.juju_epc`), 21

SingleVm1 (class in func`test.core.singlevm`), 3

SingleVm2 (class in func`test.core.singlevm`), 4

ssh_connect_loops (funcstest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13 TenantNetwork1 (class in funcstest.core.tenantnetwork), 7 TenantNetwork2 (class in funcstest.core.tenantnetwork), 8

ssh_connect_timeout (funcstest.core.singlevm.SingleVm1 attribute), 4 tenants_amount (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11

ssh_retry_timeout (funcstest.opnfv_tests.openstack.vmtop.vmtop.Vmtop attribute), 17 test_vnf () (funcstest.opnfv_tests.vnf.epc.juju_epc.JujuEpc method), 21

SshClient (class in funcstest.opnfv_tests.vnf.router.vnf_controller.ssh_client), 24 test_vnf () (funcstest.opnfv_tests.vnf.ims.heat_ims.HeatIms method), 22 test_vnf () (funcstest.opnfv_tests.vnf.router.vrouter_base.VrouterOnBoarding method), 26

stests (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11

stests (funcstest.opnfv_tests.openstack.rally.rally.RallyJobs attribute), 12

string () (in module funcstest.utils.env), 26 update_auth_section () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

support_dir (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11 update_compute_section () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

T

task_dir (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11 update_data () (in module funcstest.opnfv_tests.vnf.epc.juju_epc), 21

task_succeed () (funcstest.opnfv_tests.openstack.rally.rally.RallyBase static method), 11 default_role () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

task_timeout (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11 update_keystone_default_role () (funcstest.opnfv_tests.openstack.rally.rally.RallyFull static method), 11

task_timeout (funcstest.opnfv_tests.openstack.rally.rally.RallyJobs attribute), 12 network_section () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

temp_dir (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11 update_rally_logs () (funcstest.opnfv_tests.openstack.rally.rally.RallyBase static method), 11

tempest_blacklist (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon attribute), 15 update_rally_regex () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

tempest_conf_yaml (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon attribute), 15 update_scenario_section () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

tempest_custom (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon attribute), 15 update_tempest_conf_file () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon static method), 15

tempest_public_blacklist (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon attribute), 15 update_validation_section () (funcstest.opnfv_tests.openstack.tempest.tempest.TempestCommon method), 15

TempestCommon (class in funcstest.opnfv_tests.openstack.tempest.tempest), 14

TempestHeat (class in funcstest.opnfv_tests.openstack.tempest.tempest), 15 username (funcstest.core.singlevm.SingleVm1 attribute), 4

TempestHorizon (class in funcstest.opnfv_tests.openstack.tempest.tempest), 16 username (funcstest.opnfv_tests.openstack.shaker.shaker.Shaker attribute), 13 username (funcstest.opnfv_tests.vnf.epc.juju_epc.JujuEpc attribute), 21

template_dir (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11 tenants_amount (funcstest.opnfv_tests.openstack.rally.rally.RallyBase attribute), 11

Utilvnf (*class in* `functest.opnfv_tests.vnf.router.utilvnf`),
25

V

verify_report() (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
static method), 11

visibility (*functest.core.singlevm.VmReady1* *at-*
tribute), 6

visibility (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
attribute), 12

visibility (*functest.opnfv_tests.openstack.tempest.tempest.TempestCommon*
attribute), 15

VmController (*class in*
functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller),
24

VmReady1 (*class in* `functest.core.singlevm`), 4

VmReady2 (*class in* `functest.core.singlevm`), 6

Vmtp (*class in* `functest.opnfv_tests.openstack.vmt.vmt`),
16

VnfController (*class in*
functest.opnfv_tests.vnf.router.vnf_controller.vnf_controller),
24

volume_service_type
(*functest.opnfv_tests.openstack.rally.rally.RallyBase*
attribute), 12

volume_timeout (*functest.opnfv_tests.openstack.cinder.cinder_test.CinderCheck*
attribute), 9

volume_version (*functest.opnfv_tests.openstack.rally.rally.RallyBase*
attribute), 12

VPingSSH (*class in* `functest.opnfv_tests.openstack.vping.vping_ssh`),
17

VPingUserdata (*class in*
functest.opnfv_tests.openstack.vping.vping_userdata),
18

VrouterOnBoardingBase (*class in*
functest.opnfv_tests.vnf.router.vrouter_base),
25

W

write_config() (*functest.opnfv_tests.openstack.vmt.vmt.Vmtp*
method), 17

write_result_data()
(*functest.opnfv_tests.vnf.router.utilvnf.Utilvnf*
method), 25