Yardstick Framework Code Documentation

Release draft (dc64b51)

OPNFV

February 12, 2016

CONTENTS

1 yardstick **1**
1.1 yardstick package 1

Index **35**

CHAPTER

ONE

YARDSTICK

1.1 yardstick package

1.1.1 Subpackages

yardstick.benchmark package

Subpackages

yardstick.benchmark.contexts package

Submodules

yardstick.benchmark.contexts.base module

class yardstick.benchmark.contexts.base.**Context**

Bases: object

Class that represents a context in the logical model

deploy()

Deploy context.

static get(*context_type*)

Returns instance of a context for context type.

static get_cls(*context_type*)

Return class of specified type.

static get_server(*attr_name*)

lookup server info by name from context attr_name: either a name for a server created by yardstick or a dict with attribute name mapping when using external heat templates

init(*attrs*)

Initiate context.

list = []

undeploy()

Undeploy context.

yardstick.benchmark.contexts.dummy module

class yardstick.benchmark.contexts.dummy.**DummyContext**

Bases: *yardstick.benchmark.contexts.base.Context*

Class that handle dummy info

deploy()
don't need to deploy

init(*attrs*)

undeploy()
don't need to undeploy

yardstick.benchmark.contexts.heat module

class yardstick.benchmark.contexts.heat.**HeatContext**

Bases: *yardstick.benchmark.contexts.base.Context*

Class that represents a context in the logical model

deploy()
deploys template into a stack using cloud

flavor
returns application's default flavor name

image
returns application's default image name

init(*attrs*)
initializes itself from the supplied arguments

undeploy()
undeploys stack from cloud

user
return login user name corresponding to image

yardstick.benchmark.contexts.model module Logical model

class yardstick.benchmark.contexts.model.**Network**(*name*, *context*, *attrs*)

Bases: *yardstick.benchmark.contexts.model.Object*

Class that represents a network in the logical model

static find_by_route_to(*external_network*)
finds a network that has a route to the specified network

static find_external_network()
return the name of an external network some network in this context has a route to

has_route_to(*network_name*)
determines if this network has a route to the named network

list = []

class yardstick.benchmark.contexts.model.**Object**(*name*, *context*)

Bases: object

Base class for classes in the logical model Contains common attributes and methods

dn
returns distinguished name for object

class yardstick.benchmark.contexts.model.**PlacementGroup**(*name*, *context*, *policy*)
Bases: *yardstick.benchmark.contexts.model.Object*

Class that represents a placement group in the logical model Concept comes from the OVF specification. Policy should be one of "availability" or "affinity (there are more but they are not supported)"

add_member(*name*)

static get(*name*)

map = {}

class yardstick.benchmark.contexts.model.**Router**(*name*, *network_name*, *context*, *external_gateway_info*)
Bases: *yardstick.benchmark.contexts.model.Object*

Class that represents a router in the logical model

class yardstick.benchmark.contexts.model.**Server**(*name*, *context*, *attrs*)
Bases: *yardstick.benchmark.contexts.model.Object*

Class that represents a server in the logical model

add_to_template(*template*, *networks*, *scheduler_hints=None*)
adds to the template one or more servers (instances)

flavor
returns a server's flavor name

image
returns a server's image name

list = []

yardstick.benchmark.contexts.model.**update_scheduler_hints**(*scheduler_hints*, *added_servers*, *placement_group*)
update scheduler hints from server's placement configuration TODO: this code is openstack specific and should move somewhere else

yardstick.benchmark.contexts.node module

class yardstick.benchmark.contexts.node.**NodeContext**
Bases: *yardstick.benchmark.contexts.base.Context*

Class that handle nodes info

deploy()
don't need to deploy

init(*attrs*)
initializes itself from the supplied arguments

undeploy()
don't need to undeploy

Module contents

yardstick.benchmark.runners package

Submodules

yardstick.benchmark.runners.arithmetic module A runner that every run arithmetically steps specified input value(s) to the scenario. This just means step value(s) is added to the previous value(s). It is possible to combine several named input values and run with those either as nested for loops or combine each i:th index of each "input value list" until the end of the shortest list is reached (optimally all lists should be defined with the same number of values when using such iter_type).

class yardstick.benchmark.runners.arithmetic.**ArithmeticRunner**(*config*, *queue*)
Bases: *yardstick.benchmark.runners.base.Runner*

Run a scenario arithmetically stepping input value(s)

Parameters

- **– time to wait between each scenario invocation** (*interval*) – type: int unit: seconds default: 1 sec
- **iter_type** –

  – Iteration type of input parameter: nested_for_loops or tuple_loops

  type: string unit: na default: nested_for_loops
- **–** –

  name - name of scenario option that will be increased for each invocation type: string unit: na default: na

  start - value to use in first invocation of scenario type: int unit: na default: none

  stop - value indicating end of invocation. Can be set to same

  value as start for one single value.

  type: int unit: na default: none

  step - value added to start value in next invocation of scenario.

  Must not be set to zero. Can be set negative if start > stop

  type: int unit: na default: none
- **–** – name - and so on.....

yardstick.benchmark.runners.base module

class yardstick.benchmark.runners.base.**Runner**(*config*, *queue*)
Bases: object

abort()
Abort the execution of a scenario

dump_process = None

static get(*config*)
Returns instance of a scenario runner for execution type.

static get_cls(*runner_type*)
return class of specified type

static get_types()
return a list of known runner type (class) names

join(*timeout=None*)

queue = None

static release(*runner*)
Release the runner

static release_dump_process()
Release the dumper process

run(*scenario_cfg*, *context_cfg*)

run_post_stop_action()
run a potentially configured post-stop action

runners = []

static terminate(*runner*)
Terminate the runner

static terminate_all()
Terminate all runners (subprocesses)

yardstick.benchmark.runners.duration module A runner that runs a specific time before it returns

class yardstick.benchmark.runners.duration.**DurationRunner**(*config*, *queue*)
Bases: *yardstick.benchmark.runners.base.Runner*

Run a scenario for a certain amount of time

If the scenario ends before the time has elapsed, it will be started again.

Parameters

duration - amount of time the scenario will be run for type: int unit: seconds default: 1 sec

interval - time to wait between each scenario invocation type: int unit: seconds default: 1 sec

yardstick.benchmark.runners.iteration module A runner that runs a configurable number of times before it returns

class yardstick.benchmark.runners.iteration.**IterationRunner**(*config*, *queue*)
Bases: *yardstick.benchmark.runners.base.Runner*

Run a scenario for a configurable number of times

If the scenario ends before the time has elapsed, it will be started again.

Parameters

iterations - amount of times the scenario will be run for type: int unit: na default: 1

interval - time to wait between each scenario invocation type: int unit: seconds default: 1 sec

yardstick.benchmark.runners.sequence module A runner that every run changes a specified input value to the scenario. The input value in the sequence is specified in a list in the input file.

class yardstick.benchmark.runners.sequence.**SequenceRunner**(*config*, *queue*)
Bases: *yardstick.benchmark.runners.base.Runner*

Run a scenario by changing an input value defined in a list

Parameters

- **- time to wait between each scenario invocation** (*interval*) – type: int unit: seconds default: 1 sec
- **- name of the option that is increased each invocation** (*scenario_option_name*) – type: string unit: na default: none
- **- list of values which are executed in their respective scenarios** (sequence) – type: [int] unit: na default: none

Module contents

yardstick.benchmark.scenarios package

Subpackages

yardstick.benchmark.scenarios.availability package

Subpackages

yardstick.benchmark.scenarios.availability.attacker package

Submodules

yardstick.benchmark.scenarios.availability.attacker.attacker_baremetal module

class yardstick.benchmark.scenarios.availability.attacker.attacker_baremetal.**BaremetalAttacke**

Bases: *yardstick.benchmark.scenarios.availability.attacker.baseattacker.BaseAttacker*

check()

inject_fault()

recover()

setup()

yardstick.benchmark.scenarios.availability.attacker.attacker_process module

class yardstick.benchmark.scenarios.availability.attacker.attacker_process.**ProcessAttacker**(*co*
co
tex

Bases: *yardstick.benchmark.scenarios.availability.attacker.baseattacker.BaseAttacker*

check()

inject_fault()

recover()

setup()

yardstick.benchmark.scenarios.availability.attacker.baseattacker module

class yardstick.benchmark.scenarios.availability.attacker.baseattacker.**BaseAttacker**(*config*, *context*)

Bases: object

attacker_cfgs = {}

static get_attacker_cls(*attacker_cfg*)
return attacker instance of specified type

get_script_fullpath(*path*)

Module contents

yardstick.benchmark.scenarios.availability.monitor package

Submodules

yardstick.benchmark.scenarios.availability.monitor.basemonitor module

class yardstick.benchmark.scenarios.availability.monitor.basemonitor.**BaseMonitor**(*config*, *context*)

Bases: multiprocessing.process.Process

docstring for BaseMonitor

static get_monitor_cls(*monitor_type*)
return monitor class of specified type

get_script_fullpath(*path*)

monitor_func()

run()

setup()

start_monitor()

verify_SLA()

wait_monitor()

class yardstick.benchmark.scenarios.availability.monitor.basemonitor.**MonitorMgr**

Bases: object

docstring for MonitorMgr

init_monitors(*monitor_cfgs*, *context*)

start_monitors()

verify_SLA()

wait_monitors()

yardstick.benchmark.scenarios.availability.monitor.monitor_command module

class yardstick.benchmark.scenarios.availability.monitor.monitor_command.**MonitorOpenstackCmd**(

Bases: *yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor*

docstring for MonitorApi

monitor_func()

setup()

verify_SLA()

yardstick.benchmark.scenarios.availability.monitor.monitor_process module

class yardstick.benchmark.scenarios.availability.monitor.monitor_process.**MonitorProcess**(*config*, *context*)

Bases: *yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor*

docstring for MonitorApi

monitor_func()

setup()

verify_SLA()

Module contents

Submodules

yardstick.benchmark.scenarios.availability.serviceha module

class yardstick.benchmark.scenarios.availability.serviceha.**ServiceHA**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

TODO: docstring of ServiceHA

run(*result*)
execute the benchmark

setup()
scenario setup

teardown()
scenario teardown

Module contents

yardstick.benchmark.scenarios.compute package

Submodules

yardstick.benchmark.scenarios.compute.cpuload module Processor statistics and system load.

class yardstick.benchmark.scenarios.compute.cpuload.**CPULoad**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Collect processor statistics and system load.

This scenario reads system load averages and CPU usage statistics on a Linux host.

CPU usage statistics are read using the utility 'mpstat'.

If 'mpstat' is not installed on the host usage statistics are instead read directly from '/proc/stat'.

Load averages are read from the file '/proc/loadavg' on the Linux host.

Parameters

- **- Time interval to measure CPU usage. A value of 0** (*interval*) – indicates that processors statistics are to be reported for the time since system startup (boot)
- **type** – [int]
- **unit** – seconds
- **default** – 0

MPSTAT_FIELD_SIZE = 10

run(*result*)
Read processor statistics.

setup()
Scenario setup.

yardstick.benchmark.scenarios.compute.cyclictest module

class yardstick.benchmark.scenarios.compute.cyclictest.**Cyclictest**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute cyclictest benchmark on guest vm

Parameters

- **- run thread #N on processor #N, if possible** (*affinity*) – type: int unit: na default: 1
- **- base interval of thread** (*interval*) – type: int unit: us default: 1000
- **- number of loops, 0 for endless** (*loops*) – type: int unit: na default: 1000
- **- priority of highest prio thread** (*priority*) – type: int unit: na default: 99
- **- number of threads** (*threads*) – type: int unit: na default: 1
- **- dump a latency histogram to stdout after the run** (*histogram*) – here set the max time to be tracked

  type: int unit: ms default: 90
- **link below for more fio args description** (*Read*) – https://rt.wiki.kernel.org/index.php/Cyclictest

REBOOT_CMD_PATTERN = ';\\s*reboot\\b'

TARGET_SCRIPT = 'cyclictest_benchmark.bash'

WORKSPACE = '/root/workspace/'

run(*result*)
execute the benchmark

setup()
scenario setup

yardstick.benchmark.scenarios.compute.lmbench module

class yardstick.benchmark.scenarios.compute.lmbench.**Lmbench**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute lmbench memory read latency or memory bandwidth benchmark in a host

Parameters **- specifies whether to measure memory latency or bandwidth** (*test_type*) – type: string unit: na default: "latency"

Parameters for memory read latency benchmark

stride - number of locations in memory between starts of array elements type: int unit: bytes default: 128

stop_size - maximum array size to test (minimum value is 0.000512) type: float unit: megabytes default: 16.0

Results are accurate to the ~2-5 nanosecond range.

Parameters for memory bandwidth benchmark

size - the amount of memory to test type: int unit: kilobyte default: 128

benchmark - the name of the memory bandwidth benchmark test to execute. Valid test names are rd, wr, rdwr, cp, frd, fwr, fcp, bzero, bcopy

type: string unit: na default: "rd"

warmup - the number of repetitons to perform before taking measurements type: int unit: na default: 0

more info http://manpages.ubuntu.com/manpages/trusty/lmbench.8.html

BANDWIDTH_BENCHMARK_SCRIPT = 'lmbench_bandwidth_benchmark.bash'

LATENCY_BENCHMARK_SCRIPT = 'lmbench_latency_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

yardstick.benchmark.scenarios.compute.perf module

class yardstick.benchmark.scenarios.compute.perf.**Perf**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute perf benchmark in a host

Parameters

- **- perf tool software, hardware or tracepoint events** (*events*) – type: [str] unit: na default: ['task-clock']

- **- simulate load on the host by doing IO operations** (*load*) – type: bool unit: na default: false
- **more info about perf and perf events see https** (*For*) – //perf.wiki.kernel.org

TARGET_SCRIPT = 'perf_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

yardstick.benchmark.scenarios.compute.unixbench module

class yardstick.benchmark.scenarios.compute.unixbench.**Unixbench**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Unixbench cpu benchmark in a host The Run script takes a number of options which you can use to customise a test, and you can specify the names of the tests to run. The full usage is:

Run [-q | -v] [-i <n>] [-c <n> [-c <n> ...]] [test ...]

-i <count>	Run <count> iterations for each test – slower tests use <count> / 3, but at least 1. Defaults to 10 (3 for slow tests).
-c <n>	Run <n> copies of each test in parallel.

Parameters for setting unixbench

run_mode - Run in quiet mode or verbose mode type: string unit: None default: None

test_type - The available tests are organised into categories; type: string unit: None default: None

iterations - Run <count> iterations for each test – slower tests use <count> / 3, but at least 1. Defaults to 10 (3 for slow tests).

type: int unit: None default: None

copies - Run <n> copies of each test in parallel. type: int unit: None default: None

more info https://github.com/kdlucas/byte-unixbench/blob/master/UnixBench

TARGET_SCRIPT = 'unixbench_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

Module contents

yardstick.benchmark.scenarios.dummy package

Submodules

yardstick.benchmark.scenarios.dummy.dummy module

class yardstick.benchmark.scenarios.dummy.dummy.**Dummy**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Dummy echo

run(*result*)

execute the benchmark

setup()

scenario setup

Module contents

yardstick.benchmark.scenarios.networking package

Submodules

yardstick.benchmark.scenarios.networking.iperf3 module

class yardstick.benchmark.scenarios.networking.iperf3.**Iperf**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute iperf3 between two hosts

By default TCP is used but UDP can also be configured. For more info see http://software.es.net/iperf

Parameters

bytes - number of bytes to transmit

only valid with a non duration runner, mutually exclusive with blockcount type: int unit: bytes default: 56

udp - use UDP rather than TCP type: bool unit: na default: false

nodelay - set TCP no delay, disabling Nagle's Algorithm type: bool unit: na default: false

blockcount - number of blocks (packets) to transmit,

only valid with a non duration runner, mutually exclusive with bytes type: int unit: bytes default: -

run(*result*)

execute the benchmark

setup()

teardown()

yardstick.benchmark.scenarios.networking.netperf module

class yardstick.benchmark.scenarios.networking.netperf.**Netperf**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute netperf between two hosts

Parameters

- **- to specify the test you wish to perform.** (*testname*) –

- **valid testnames are TCP_STREAM, TCP_RR, UDP_STREAM, UDP_RR** (*the*) – type: string unit: na default: TCP_STREAM
- **- value set the local send size to value bytes.** (*send_msg_size*) – type: int unit: bytes default: na
- **- setting the receive size for the remote system.** (*recv_msg_size*) – type: int unit: bytes default: na
- **- set the request and/or response sizes based on sizespec.** (*req_rsp_size*) – type: string unit: na default: na
- **- duration of the test** (duration) – type: int unit: seconds default: 20
- **link below for more netperf args description** (*read*) –
- **http** – //www.netperf.org/netperf/training/Netperf.html

TARGET_SCRIPT = 'netperf_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

yardstick.benchmark.scenarios.networking.ping module

class yardstick.benchmark.scenarios.networking.ping.**Ping**(*scenario_cfg*, *context_cfg*)
Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute ping between two hosts

Parameters **- number of data bytes to send** (*packetsize*) – type: int unit: bytes default: 56

TARGET_SCRIPT = 'ping_benchmark.bash'

run(*result*)
execute the benchmark

yardstick.benchmark.scenarios.networking.ping6 module

class yardstick.benchmark.scenarios.networking.ping6.**Ping6**(*scenario_cfg*, *context_cfg*)
Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute ping6 between two hosts

read link below for more ipv6 info description: http://wiki.opnfv.org/ipv6_opnfv_project

FIND_HOST_SCRIPT = 'ping6_find_host.bash'

METADATA_SCRIPT = 'ping6_metadata.txt'

POST_TEARDOWN_SCRIPT = 'ping6_post_teardown.bash'

PRE_SETUP_SCRIPT = 'ping6_pre_setup.bash'

RADVD_SCRIPT = 'ping6_radvd.conf'

SETUP_ODL_SCRIPT = 'ping6_setup_with_odl.bash'

SETUP_SCRIPT = 'ping6_setup.bash'

TARGET_SCRIPT = 'ping6_benchmark.bash'

TEARDOWN_SCRIPT = 'ping6_teardown.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

teardown()
teardown the benchmark

yardstick.benchmark.scenarios.networking.pktgen module

class yardstick.benchmark.scenarios.networking.pktgen.**Pktgen**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute pktgen between two hosts

Parameters

- **- packet size in bytes without the CRC** (*packetsize*) – type: int unit: bytes default: 60
- **- number of UDP ports to test** (*number_of_ports*) – type: int unit: na default: 10
- **- duration of the test** (duration) – type: int unit: seconds default: 20

TARGET_SCRIPT = 'pktgen_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

yardstick.benchmark.scenarios.networking.sfc module

class yardstick.benchmark.scenarios.networking.sfc.**Sfc**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

SFC scenario class

PRE_SETUP_SCRIPT = 'sfc_pre_setup.bash'

SERVER_SCRIPT = 'sfc_server.bash'

TACKER_SCRIPT = 'sfc_tacker.bash'

TEARDOWN_SCRIPT = 'sfc_teardown.bash'

run(*result*)
Creating client and server VMs to perform the test

setup()
scenario setup

teardown()
for scenario teardown remove tacker VNFs, chains and classifiers

yardstick.benchmark.scenarios.networking.vtc_instantiation_validation module

class yardstick.benchmark.scenarios.networking.vtc_instantiation_validation.**VtcInstantiationV**

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Instantiation Validation TC on the vTC

run(*result*)
execute test

setup()
scenario setup

yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy module

class yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy.**VtcInstanti**

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Instantiation Validation TC on the vTC

run(*result*)
execute test

setup()
scenario setup

yardstick.benchmark.scenarios.networking.vtc_throughput module

class yardstick.benchmark.scenarios.networking.vtc_throughput.**VtcThroughput**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Instantiation Validation TC on the vTC

run(*result*)
execute test

setup()
scenario setup

yardstick.benchmark.scenarios.networking.vtc_throughput_noisy module

class yardstick.benchmark.scenarios.networking.vtc_throughput_noisy.**VtcThroughputNoisy**(*scenario*
context_cfg)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute Instantiation Validation TC on the vTC

run(*result*)
execute test

setup()
scenario setup

Module contents

yardstick.benchmark.scenarios.parser package

Submodules

yardstick.benchmark.scenarios.parser.parser module

class yardstick.benchmark.scenarios.parser.parser.**Parser**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

running Parser Yang-to-Tosca module as a tool validating output against expected outcome

more info https://wiki.opnfv.org/parser

PARSER_SCRIPT = 'parser.sh'

SETUP_SCRIPT = 'parser_setup.sh'

TEARDOWN_SCRIPT = 'parser_teardown.sh'

run(*result*)

execute the translation

setup()

scenario setup

teardown()

for scenario teardown remove parser and pyang

Module contents

yardstick.benchmark.scenarios.storage package

Submodules

yardstick.benchmark.scenarios.storage.fio module

class yardstick.benchmark.scenarios.storage.fio.**Fio**(*scenario_cfg*, *context_cfg*)

Bases: *yardstick.benchmark.scenarios.base.Scenario*

Execute fio benchmark in a host

Parameters

- **- file name for fio workload** (*filename*) – type: string unit: na default: /home/ubuntu/data.raw
- **- block size used for the io units** (*bs*) – type: int unit: bytes default: 4k
- **- number of iobuffers to keep in flight** (*iodepth*) – type: int unit: na default: 1
- **- type of io pattern [read, write, randwrite, randread, rw, randrw]** (*rw*) – type: string unit: na default: write
- **- run time before logging any performance** (*ramp_time*) – type: int unit: seconds default: 20
- **link below for more fio args description** (*Read*) – http://www.bluestop.org/fio/HOWTO.txt

TARGET_SCRIPT = 'fio_benchmark.bash'

run(*result*)
execute the benchmark

setup()
scenario setup

Module contents

Submodules

yardstick.benchmark.scenarios.base module Scenario base class

class yardstick.benchmark.scenarios.base.**Scenario**
Bases: object

static get(*scenario_type*)
Returns instance of a scenario runner for execution type.

static get_cls(*scenario_type*)
return class of specified type

static get_types()
return a list of known runner type (class) names

run(*args*)
catcher for not implemented run methods in subclasses

setup()
default impl for scenario setup

teardown()
default impl for scenario teardown

Module contents

Module contents

yardstick.cmd package

Subpackages

yardstick.cmd.commands package

Submodules

yardstick.cmd.commands.runner module Handler for yardstick command 'runner'

class yardstick.cmd.commands.runner.**RunnerCommands**
Bases: object

Runner commands.

Set of commands to discover and display runner types.

do_list(*args*)
List existing runner types

do_show(*args*)
Show details of a specific runner type

yardstick.cmd.commands.scenario module Handler for yardstick command 'scenario'

class yardstick.cmd.commands.scenario.**ScenarioCommands**
Bases: object

Scenario commands.

Set of commands to discover and display scenario types.

do_list(*args*)
List existing scenario types

do_show(*args*)
Show details of a specific scenario type

yardstick.cmd.commands.task module Handler for yardstick command 'task'

class yardstick.cmd.commands.task.**TaskCommands**
Bases: object

Task commands.

Set of commands to manage benchmark tasks.

do_start(*args*)
Start a benchmark scenario.

class yardstick.cmd.commands.task.**TaskParser**(*path*)
Bases: object

Parser for task config files in yaml format

parse_suite()
parse the suite file and return a list of task config file paths and lists of optional parameters if present

parse_task(*task_name*, *task_args=None*, *task_args_file=None*)
parses the task file and return an context and scenario instances

yardstick.cmd.commands.task.**atexit_handler**()
handler for process termination

yardstick.cmd.commands.task.**is_ip_addr**(*addr*)
check if string addr is an IP address

yardstick.cmd.commands.task.**parse_nodes_with_context**(*scenario_cfg*)
paras the 'nodes' fields in scenario

yardstick.cmd.commands.task.**parse_task_args**(*src_name*, *args*)

yardstick.cmd.commands.task.**print_invalid_header**(*source_name*, *args*)

yardstick.cmd.commands.task.**run_one_scenario**(*scenario_cfg*, *output_file*)
run one scenario using context

yardstick.cmd.commands.task.**runner_join**(*runner*)
join (wait for) a runner, exit process at runner failure

Module contents

Submodules

yardstick.cmd.cli module

Command-line interface to yardstick

class yardstick.cmd.cli.**YardstickCLI**
Command-line interface to yardstick

categories = {'runner': <class 'yardstick.cmd.commands.runner.RunnerCommands'>, 'task': <class 'yardstick.cmd.c

main(*argv*)
run the command line interface

yardstick.cmd.cli.**find_config_files**(*path_list*)

Module contents

yardstick.cmd.**print_hbar**(*barlen*)
print to stdout a horizontal bar

yardstick.common package

Submodules

yardstick.common.task_template module

class yardstick.common.task_template.**TaskTemplate**
Bases: object

classmethod render(*task_template*, ***kwargs*)
Render jinja2 task template to Yardstick input task.

Parameters

- **task_template** – string that contains template
- **kwargs** – Dict with template arguments

:returns:rendered template str

yardstick.common.task_template.**is_really_missing**(*mis*, *task_template*)

yardstick.common.template_format module

yardstick.common.template_format.**parse**(*tmpl_str*)
Takes a string and returns a dict containing the parsed structure.

This includes determination of whether the string is using the JSON or YAML format.

yardstick.common.utils module

yardstick.common.utils.**cliargs**(**args*, ***kwargs*)

yardstick.common.utils.**import_modules_from_package**(*package*)
Import modules from package and append into sys.modules

Param package - Full package name. For example: rally.deploy.engines

yardstick.common.utils.**itersubclasses**(*cls*, *_seen=None*)
Generator over all subclasses of a given class in depth first order.

yardstick.common.utils.**try_append_module**(*name*, *modules*)

Module contents

yardstick.dispatcher package

Submodules

yardstick.dispatcher.base module

class yardstick.dispatcher.base.**Base**(*conf*)
Bases: object

flush_result_data()
Flush result data into permanent storage media interface.

static get(*config*)
Returns instance of a dispatcher for dispatcher type.

static get_cls(*dispatcher_type*)
Return class of specified type.

record_result_data(*data*)
Recording result data interface.

yardstick.dispatcher.file module

class yardstick.dispatcher.file.**FileDispatcher**(*conf*)
Bases: *yardstick.dispatcher.base.Base*

Dispatcher class for recording data to a file.

flush_result_data()

record_result_data(*data*)

yardstick.dispatcher.http module

class yardstick.dispatcher.http.**HttpDispatcher**(*conf*)
Bases: *yardstick.dispatcher.base.Base*

Dispatcher class for posting data into a http target.

flush_result_data()

record_result_data(*data*)

yardstick.dispatcher.influxdb module

class yardstick.dispatcher.influxdb.**InfluxdbDispatcher**(*conf*)
Bases: *yardstick.dispatcher.base.Base*

Dispatcher class for posting data into an influxdb target.

flush_result_data()

record_result_data(*data*)

yardstick.dispatcher.influxdb_line_protocol module

yardstick.dispatcher.influxdb_line_protocol.**make_lines**(*data*)
Extracts the points from the given dict and returns a Unicode string matching the line protocol introduced in InfluxDB 0.9.0.

line protocol format: <measurement>[,<tag-key>=<tag-value>...] <field-key>=<field-value> [,<field2-key>=<field2-value>...] [unix-nano-timestamp]

Ref: https://influxdb.com/docs/v0.9/write_protocols/write_syntax.html https://influxdb.com/docs/v0.9/write_protocols/line.html

Module contents

yardstick.orchestrator package

Submodules

yardstick.orchestrator.heat module

Heat template and stack management

class yardstick.orchestrator.heat.**HeatObject**
Bases: object

base class for template and stack

status()
returns stack state as a string

class yardstick.orchestrator.heat.**HeatStack**(*name*)
Bases: *yardstick.orchestrator.heat.HeatObject*

Represents a Heat stack (deployed template)

delete(*block=True*, *retries=3*)
deletes a stack in the target cloud using heat (with retry) Sometimes delete fail with "InternalServerError" and the next attempt succeeds. So it is worthwhile to test a couple of times.

static delete_all()

stacks = []

static stacks_exist()
check if any stack has been deployed

update()
update a stack

class yardstick.orchestrator.heat.**HeatTemplate**(*name*, *template_file=None*, *heat_parameters=None*)
Bases: *yardstick.orchestrator.heat.HeatObject*

Describes a Heat template and a method to deploy template to a stack

add_floating_ip(*name*, *network_name*, *port_name*, *router_if_name*, *secgroup_name=None*)
add to the template a Neutron FloatingIP resource see: https://bugs.launchpad.net/heat/+bug/1299259

add_keypair(*name*)
add to the template a Nova KeyPair

add_network(*name*)
add to the template a Neutron Net

add_port(*name*, *network_name*, *subnet_name*, *sec_group_id=None*)
add to the template a named Neutron Port

add_router(*name*, *ext_gw_net*, *subnet_name*)
add to the template a Neutron Router and interface

add_router_interface(*name*, *router_name*, *subnet_name*)
add to the template a Neutron RouterInterface and interface

add_security_group(*name*)
add to the template a Neutron SecurityGroup

add_server(*name*, *image*, *flavor*, *ports=None*, *networks=None*, *scheduler_hints=None*, *user=None*, *key_name=None*, *user_data=None*, *metadata=None*, *additional_properties=None*)
add to the template a Nova Server

add_servergroup(*name*, *policy*)
add to the template a Nova ServerGroup

add_subnet(*name*, *network*, *cidr*)
add to the template a Neutron Subnet

create(*block=True*)
creates a template in the target cloud using heat returns a dict with the requested output values from the template

Module contents

yardstick.plot package

Submodules

yardstick.plot.plotter module

Module contents

yardstick.resources package

Module contents

yardstick.vTC package

Subpackages

yardstick.vTC.apexlake package

Subpackages

yardstick.vTC.apexlake.experimental_framework package

Subpackages

yardstick.vTC.apexlake.experimental_framework.benchmarks package

Submodules

yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmark_base_class module

class yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmark_base_class.**Benchmark**

Bases: object

This class represents a Benchmark that we want to run on the platform. One of them will be the calculation of the throughput changing the configuration parameters

finalize()
Finalizes the benchmark :return:

get_features()

get_name()

get_params()

init()
Initializes the benchmark :return:

run()
This method executes the specific benchmark on the VNF already instantiated :return: list of dictionaries (every dictionary contains the results of a data point

yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_benchmark module

class yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_bench

Bases: experimental_framework.benchmarks.benchmark_base_class.BenchmarkBaseClass

finalize()
Finalizes the benchmark :return: None

get_features()

init()
Initialize the benchmark :return: None

run()

yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_noisy_neighbors_benchmark module

class yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_noisy

Bases: *yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_*

finalize()

get_features()

init()

yardstick.vTC.apexlake.experimental_framework.benchmarks.multi_tenancy_throughput_benchmark module

class yardstick.vTC.apexlake.experimental_framework.benchmarks.multi_tenancy_throughput_bench

Bases: experimental_framework.benchmarks.rfc2544_throughput_benchmark.RFC2544ThroughputBe

finalize()
Finalizes the benchmark return: None

get_features()

init()
Initialize the benchmark return: None

yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544_throughput_benchmark module

class yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544_throughput_benchmark.R

Bases: experimental_framework.benchmarks.benchmark_base_class.BenchmarkBaseClass

Calculates the throughput of the VNF under test according to the RFC2544.

finalize()

Returns None

get_features()
Returns the features associated to the benchmark :return:

init()
Initialize the benchmark :return: None

run()
Sends and receive traffic according to the RFC methodology in order to measure the throughput of the workload :return: Results of the testcase (type: dict)

yardstick.vTC.apexlake.experimental_framework.benchmarks.test_benchmark module
class yardstick.vTC.apexlake.experimental_framework.benchmarks.test_benchmark.**TestBenchmark**(*n*
p

Bases: experimental_framework.benchmarks.benchmark_base_class.BenchmarkBaseClass

finalize()

get_features()

init()

run()

Module contents Benchmarks to be executed within the framework

yardstick.vTC.apexlake.experimental_framework.constants package

Submodules

yardstick.vTC.apexlake.experimental_framework.constants.conf_file_sections module
yardstick.vTC.apexlake.experimental_framework.constants.conf_file_sections.**get_sections**()
yardstick.vTC.apexlake.experimental_framework.constants.conf_file_sections.**get_sections_ap**

yardstick.vTC.apexlake.experimental_framework.constants.framework_parameters module
yardstick.vTC.apexlake.experimental_framework.constants.framework_parameters.**get_supported_**

Module contents Constants

yardstick.vTC.apexlake.experimental_framework.libraries package

Module contents Libraries to be used by the framework.

yardstick.vTC.apexlake.experimental_framework.packet_generators package

Submodules

yardstick.vTC.apexlake.experimental_framework.packet_generators.base_packet_generator module

class yardstick.vTC.apexlake.experimental_framework.packet_generators.base_packet_generator.**B**

send_traffic()

Starts the traffic generation. According to the specific packet generator it requires prior initialization :return: None

yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_packet_generator module

class yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_packet_generator.**D**

Bases: *yardstick.vTC.apexlake.experimental_framework.packet_generators.base_packet_genera*

init_dpdk_pktgen(*dpdk_interfaces*, *lua_script='generic_test.lua'*, *pcap_file_0=''*, *pcap_file_1=''*, *vlan_0=''*, *vlan_1=''*)

Initializes internal parameters and configuration of the module. Needs to be called before the send_traffic :param dpdk_interfaces: Number of interfaces to be used (type: int) :param lua_script: Full path of the Lua script to be used (type: str) :param pcap_file_0: Full path of the Pcap file to be used for port 0

(type: str)

Parameters

- **pcap_file_1** – Full path of the Pcap file to be used for port 1 (type: str)
- **vlan_0** – VLAN tag to be used for port 0 (type: str)
- **vlan_1** – VLAN tag to be used for port 1 (type: str)

Returns

send_traffic()

Calls the packet generator and starts to send traffic Blocking call

Module contents Packet generators

Submodules

yardstick.vTC.apexlake.experimental_framework.api module

class yardstick.vTC.apexlake.experimental_framework.api.**FrameworkApi**

Bases: object

static **execute_framework**(*test_cases*, *iterations*, *heat_template*, *heat_template_parameters*, *deployment_configuration*, *openstack_credentials*)

Executes the framework according the inputs

Parameters

- **test_cases** – Test cases to be ran on the workload (dict() of dict())

  Example: test_case = dict() test_case['name'] = 'module.Class' test_case['params'] = dict() test_case['params']['throughput'] = '1' test_case['params']['vlan_sender'] = '1007' test_case['params']['vlan_receiver'] = '1006' test_cases = [test_case]

- **iterations** – Number of cycles to be executed (int)

- **heat_template** – (string) File name of the heat template of the workload to be deployed. It contains the parameters to be evaluated in the form of #parameter_name. (See heat_templates/vTC.yaml as example).
- **heat_template_parameters** – (dict) Parameters to be provided as input to the heat template. See http://docs.openstack.org/developer/heat/ template_guide/hot_guide.html - section "Template input parameters" for further info.
- **deployment_configuration** – (dict[string] = list(strings))) Dictionary of parameters representing the deployment configuration of the workload The key is a string corresponding to the name of the parameter, the value is a list of strings representing the value to be assumed by a specific param. The parameters are user defined: they have to correspond to the place holders (#parameter_name) specified in the heat template.

Returns dict() Containing results

static get_test_case_features(*test_case*)
Returns a list of features (description, requested parameters, allowed values, etc.) for a specified test case.

Parameters **test_case** – name of the test case (string) The string represents the test case and can be obtained calling "get_available_test_cases()" method.

Returns dict() containing the features of the test case

static init()
Initializes the Framework

Returns None

yardstick.vTC.apexlake.experimental_framework.benchmarking_unit module The Benchmarking Unit manages the Benchmarking of VNFs orchestrating the initialization, execution and finalization

class yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.**BenchmarkingUnit**(*heat_ten*
open-
stack_cr
heat_ten
it-
er-
a-
tions,
bench-
marks)
Management of the overall Benchmarking process

static extract_experiment_name(*template_file_name*)
Generates a unique experiment name for a given template.

Parameters **template_file_name** – (str) File name of the template used during the experiment string

Returns (str) Experiment Name

finalize()
Finalizes the Benchmarking Unit Destroys all the stacks deployed by the framework and save results on csv file.

Returns None

static get_benchmark_class(*complete_module_name*)
Returns the classes included in a given module.

Parameters **complete_module_name** – (str) Complete name of the module as returned by get_available_test_cases.

Returns Class related to the benchmark/TC present in the requested module.

get_benchmark_name(*name*, *instance=0*)
Returns the name to be used for the benchmark/test case (TC). This is required since each benchmark/TC could be run more than once within the same cycle, with different initialization parameters. In order to distinguish between them, a unique name is generated.

Parameters

- **name** – (str) original name of the benchmark/TC
- **instance** – (int) number of instance already in the queue for this type of benchmark/TC.

Returns (str) name to be assigned to the benchmark/TC

get_experiment_configuration(*template_file_name*)
Reads and returns the configuration for the specific experiment (heat template)

Parameters **template_file_name** – (str) Name of the file for the heat template for which it is requested the configuration

Returns dict() Configuration parameters and values

static get_required_benchmarks(*required_benchmarks*)
Returns instances of required test cases.

Parameters **required_benchmarks** – (list() of strings) Benchmarks to be executed by the experimental framework.

Returns list() of BenchmarkBaseClass

initialize()
Initialize the environment in order to run the benchmarking

Returns None

run_benchmarks()
Runs all the requested benchmarks and collect the results.

Returns None

yardstick.vTC.apexlake.experimental_framework.common module

class yardstick.vTC.apexlake.experimental_framework.common.**ConfigurationFile**(*sections*, *config_file='conf.cfg'*)
Used to extract data from the configuration file

get_variable(*section*, *variable_name*)
Returns the value correspondent to a variable

Parameters

- **section** – section to be loaded (string)
- **variable_name** – name of the variable (string)

Returns string

get_variable_list(*section*)
Returns the list of the available variables in a section :param section: section to be loaded (string) :return: list

class yardstick.vTC.apexlake.experimental_framework.common.**InputValidation**
Bases: object

static validate_boolean(*boolean*, *message*)

static validate_configuration_file_parameter(*section*, *parameter*, *message*)

static validate_configuration_file_section(*section*, *message*)

static validate_dictionary(*param*, *message*)

static validate_directory_exist_and_format(*directory*, *message*)

static validate_file_exist(*file_name*, *message*)

static validate_integer(*param*, *message*)

static validate_os_credentials(*credentials*)

static validate_string(*param*, *message*)

yardstick.vTC.apexlake.experimental_framework.common.**get_base_dir**()

yardstick.vTC.apexlake.experimental_framework.common.**get_benchmarks_from_conf_file**()

yardstick.vTC.apexlake.experimental_framework.common.**get_credentials**()
Returns the credentials for OpenStack access from the configuration file :return: dictionary

yardstick.vTC.apexlake.experimental_framework.common.**get_deployment_configuration_variables**

yardstick.vTC.apexlake.experimental_framework.common.**get_dpdk_pktgen_vars**()

yardstick.vTC.apexlake.experimental_framework.common.**get_file_first_line**(*file_name*)
Returns the first line of a file

Parameters **file_name** – name of the file to be read (str)

Returns str

yardstick.vTC.apexlake.experimental_framework.common.**get_heat_template_params**()
Returns the list of deployment parameters from the configuration file for the heat template

Returns dict

yardstick.vTC.apexlake.experimental_framework.common.**get_result_dir**()

yardstick.vTC.apexlake.experimental_framework.common.**get_template_dir**()

yardstick.vTC.apexlake.experimental_framework.common.**get_testcase_params**()
Returns the list of testcase parameters from the configuration file

Returns dict

yardstick.vTC.apexlake.experimental_framework.common.**init**(*api=False*)

yardstick.vTC.apexlake.experimental_framework.common.**init_conf_file**(*api=False*)

yardstick.vTC.apexlake.experimental_framework.common.**init_general_vars**(*api=False*)

yardstick.vTC.apexlake.experimental_framework.common.**init_influxdb**()

yardstick.vTC.apexlake.experimental_framework.common.**init_log**()

yardstick.vTC.apexlake.experimental_framework.common.**init_pktgen**()

yardstick.vTC.apexlake.experimental_framework.common.**push_data_influxdb**(*data*)

yardstick.vTC.apexlake.experimental_framework.common.**replace_in_file**(*file*, *text_to_search*, *text_to_replace*)

Replaces a string within a file

Parameters

- **file** – name of the file (str)
- **text_to_search** – text to be replaced
- **text_to_replace** – new text that will replace the previous

Returns None

yardstick.vTC.apexlake.experimental_framework.common.**run_command**(*command*)

yardstick.vTC.apexlake.experimental_framework.deployment_unit module

class yardstick.vTC.apexlake.experimental_framework.deployment_unit.**DeploymentUnit**(*openstack_cre*

This unit is in charge to manage the deployment of the workloads under test and any other workloads necessary to the benchmark

deploy_heat_template(*template_file*, *stack_name*, *parameters*, *attempt=0*)

Deploys a heat template and in case of failure retries 3 times :param template_file: full path file name of the heat template :param stack_name: name of the stack to deploy :param parameters: parameters to be given to the heat template :param attempt: number of current attempt :return: returns True in case the creation is completed

returns False in case the creation is failed

destroy_all_deployed_stacks()

Destroys all the stacks currently deployed :return: None

destroy_heat_template(*stack_name*)

Destroys a stack :param stack_name: Stack of the name to be destroyed (sting) :return: None

yardstick.vTC.apexlake.experimental_framework.heat_manager module

class yardstick.vTC.apexlake.experimental_framework.heat_manager.**HeatManager**(*credentials*)

check_stack_status(*stack_name*)

Returns a string representing the status of a stack from Heat perspective :param stack_name: Name of the stack to be checked (type: str) :return: (type: str)

create_stack(*template_file*, *stack_name*, *parameters*)

delete_stack(*stack_name*)

init_heat()

is_stack_deployed(*stack_name*)

print_stacks(*name=None*)

validate_heat_template(*heat_template_file*)

yardstick.vTC.apexlake.experimental_framework.heat_template_generation module Generation of the heat templates from the base template

class yardstick.vTC.apexlake.experimental_framework.heat_template_generation.**TreeNode**
This class represent the node of the configuration tree. Each node represents a single configuration value for a single configuration parameter.

add_child(*node*)
Adds a node as a child for the current node :param node: node to be added as a child (type: TreeNode) :return: None

get_children()
Returns the children of the current node :return type: list of TreeNode

static get_leaves(*node*)
Returns all the leaves of a tree. :param node: root of the tree (TreeNode) :return type: list

get_parent()
Returns the parent node of the current one :return type: TreeNode

get_path()
Returns all the path from the current node to the root of the tree. :return type: list of TreeNode

get_variable_name()
Returns the name of the variable correspondent to the current node :return type: str

get_variable_value()
Returns the value of the variable correspondent to the current node :return type: str or int

set_variable_name(*name*)
Sets the name of the variable for the current node :param name: Name of the variable (type: str) :return None

set_variable_value(*value*)
Sets the value of the variable for the current node :param value: value of the variable (type: str) :return None

yardstick.vTC.apexlake.experimental_framework.heat_template_generation.**generates_templates**

Generates the heat templates for the experiments :return: None

yardstick.vTC.apexlake.experimental_framework.heat_template_generation.**get_all_heat_templat**

Loads and returns all the generated heat templates :param template_dir: directory to search in (type: str) :param template_file_extension: extension of the file for templates

(type: str)

Returns type: list

Module contents Experimental Framework

Submodules

yardstick.vTC.apexlake.setup module

Module contents Benchmarking Framework

Module contents

1.1.2 Submodules

1.1.3 yardstick.main module

yardstick - command line tool for managing benchmarks

Example invocation: $ yardstick task start samples/ping.yaml

Servers are the same as VMs (Nova calls them servers in the API)

Many tests use a client/server architecture. A test client is configured to use a specific test server e.g. using an IP address. This is true for example iperf. In some cases the test server is included in the kernel (ping, pktgen) and no additional software is needed on the server. In other cases (iperf) a server process needs to be installed and started.

One server is required to host the test client program (such as ping or iperf). In the task file this server is called host.

A server can be the _target_ of a test client (think ping destination argument). A target server is optional but needed in most test scenarios. In the task file this server is called target. This is probably the same as DUT in existing terminology.

Existing terminology: https://www.ietf.org/rfc/rfc1242.txt (throughput/latency) https://www.ietf.org/rfc/rfc2285.txt (DUT/SUT)

New terminology: NFV TST

`yardstick.main.main()`
yardstick main

1.1.4 yardstick.ssh module

High level ssh library.

Usage examples:

Execute command and get output:

ssh = sshclient.SSH("root", "example.com", port=33) status, stdout, stderr = ssh.execute("ps ax") if status:

raise Exception("Command failed with non-zero status.")

print stdout.splitlines()

Execute command with huge output:

class PseudoFile(object):

def write(chunk):

if "error" in chunk: email_admin(chunk)

ssh = sshclient.SSH("root", "example.com") ssh.run("tail -f /var/log/syslog", stdout=PseudoFile(), timeout=False)

Execute local script on remote side:

ssh = sshclient.SSH("user", "example.com") status, out, err = ssh.execute("/bin/sh -s arg1 arg2",

stdin=open("~/myscript.sh", "r"))

Upload file:

ssh = sshclient.SSH("user", "example.com") ssh.run("cat > ~/upload/file.gz", stdin=open("/store/file.gz", "rb"))

Eventlet:

eventlet.monkey_patch(select=True, time=True) or eventlet.monkey_patch() or sshclient = eventlet.import_patched("opentstack.common.sshclient")

class yardstick.ssh.**SSH**(*user*, *host*, *port=22*, *pkey=None*, *key_filename=None*, *password=None*)
Bases: object

Represent ssh connection.

close()

execute(*cmd*, *stdin=None*, *timeout=3600*)
Execute the specified command on the server.

Parameters

- **cmd** – Command to be executed.
- **stdin** – Open file to be sent on process stdin.
- **timeout** – Timeout for execution of the command.

Returns tuple (exit_status, stdout, stderr)

put(*files*, *remote_path='.'*, *recursive=False*)

run(*cmd*, *stdin=None*, *stdout=None*, *stderr=None*, *raise_on_error=True*, *timeout=3600*)
Execute specified command on the server.

Parameters

- **cmd** – Command to be executed.
- **stdin** – Open file or string to pass to stdin.
- **stdout** – Open file to connect to stdout.
- **stderr** – Open file to connect to stderr.
- **raise_on_error** – If False then exit code will be return. If True then exception will be raized if non-zero code.
- **timeout** – Timeout in seconds for command execution. Default 1 hour. No timeout if set to 0.

wait(*timeout=120*, *interval=1*)
Wait for the host will be available via ssh.

exception yardstick.ssh.**SSHError**
Bases: exceptions.Exception

exception yardstick.ssh.**SSHTimeout**
Bases: *yardstick.ssh.SSHError*

1.1.5 Module contents

A

abort() (yardstick.benchmark.runners.base.Runner method), 4
add_child() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31
add_floating_ip() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_keypair() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_member() (yardstick.benchmark.contexts.model.PlacementGroup method), 3
add_network() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_port() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_router() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_router_interface() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_security_group() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_server() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_servergroup() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_subnet() (yardstick.orchestrator.heat.HeatTemplate method), 22
add_to_template() (yardstick.benchmark.contexts.model.Server method), 3
ArithmeticRunner (class in yardstick.benchmark.runners.arithmetic), 4
atexit_handler() (in module yardstick.cmd.commands.task), 18
attacker_cfgs (yardstick.benchmark.scenarios.availability.attacker.baseattacker.BaseAttacker attribute), 7

B

BANDWIDTH_BENCHMARK_SCRIPT (yardstick.benchmark.scenarios.compute.lmbench.Lmbench attribute), 10
BaremetalAttacker (class in yardstick.benchmark.scenarios.availability.attacker.attacker_baremetal), 6
Base (class in yardstick.dispatcher.base), 20
BaseAttacker (class in yardstick.benchmark.scenarios.availability.attacker.baseattacker), 7
BaseMonitor (class in yardstick.benchmark.scenarios.availability.monitor.basemonitor), 7
BasePacketGenerator (class in yardstick.vTC.apexlake.experimental_framework.packet_generators.b 26
BenchmarkBaseClass (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.benchm 23
BenchmarkingUnit (class in yardstick.vTC.apexlake.experimental_framework.benchmarking_unit) 27

C

categories (yardstick.cmd.cli.YardstickCLI attribute), 19
check() (yardstick.benchmark.scenarios.availability.attacker.attacker_barem method), 6
check() (yardstick.benchmark.scenarios.availability.attacker.attacker_proces method), 6
check_stack_status() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatM method), 30
cliargs() (in module yardstick.common.utils), 20
close() (yardstick.ssh.SSH method), 33
ConfigurationFile (class in yardstick.vTC.apexlake.experimental_framework.common), 28
Context (class in yardstick.benchmark.contexts.base), 1
CPULoad (class in yardstick.benchmark.scenarios.compute.cpuload), 9

create() (yardstick.orchestrator.heat.HeatTemplate method), 22
create_stack() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
Cyclictest (class in yardstick.benchmark.scenarios.compute.cyclictest), 9

D

delete() (yardstick.orchestrator.heat.HeatStack method), 21
delete_all() (yardstick.orchestrator.heat.HeatStack static method), 21
delete_stack() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
deploy() (yardstick.benchmark.contexts.base.Context method), 1
deploy() (yardstick.benchmark.contexts.dummy.DummyContext method), 2
deploy() (yardstick.benchmark.contexts.heat.HeatContext method), 2
deploy() (yardstick.benchmark.contexts.node.NodeContext method), 3
deploy_heat_template() (yardstick.vTC.apexlake.experimental_framework.deployment_unit.DeploymentUnit method), 30
DeploymentUnit (class in yardstick.vTC.apexlake.experimental_framework.deployment_unit), 30
destroy_all_deployed_stacks() (yardstick.vTC.apexlake.experimental_framework.deployment_unit.DeploymentUnit method), 30
destroy_heat_template() (yardstick.vTC.apexlake.experimental_framework.deployment_unit.DeploymentUnit method), 30
dn (yardstick.benchmark.contexts.model.Object attribute), 2
do_list() (yardstick.cmd.commands.runner.RunnerCommands method), 17
do_list() (yardstick.cmd.commands.scenario.ScenarioCommands method), 18
do_show() (yardstick.cmd.commands.runner.RunnerCommands method), 18
do_show() (yardstick.cmd.commands.scenario.ScenarioCommands method), 18
do_start() (yardstick.cmd.commands.task.TaskCommands method), 18
DpdkPacketGenerator (class in yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_packet_generator), 26
Dummy (class in yardstick.benchmark.scenarios.dummy.dummy), 12
DummyContext (class in yardstick.benchmark.contexts.dummy), 2
dump_part_results (yardstick.benchmark.runners.base.Runner attribute), 4
DurationRunner (class in yardstick.benchmark.runners.duration), 5

E

execute() (yardstick.ssh.SSH method), 33
execute_framework() (yardstick.vTC.apexlake.experimental_framework.api.FrameworkApi static method), 26
extract_experiment_name() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit. static method), 27

F

FileDispatcher (class in yardstick.dispatcher.file), 20
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarking_ method), 27
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.be method), 23
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.ins method), 24
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.ins method), 24
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.mu method), 24
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc method), 24
finalize() (yardstick.vTC.apexlake.experimental_framework.benchmarks.tes method), 25
find_by_route_to() (yardstick.benchmark.contexts.model.Network static method), 2
find_config_files() (in module yardstick.cmd.cli), 19
find_external_network() (yardstick.benchmark.contexts.model.Network static method), 2
FIND_HOST_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
Fio (class in yardstick.benchmark.scenarios.storage.fio), 16
flavor (yardstick.benchmark.contexts.heat.HeatContext attribute), 2
flavor (yardstick.benchmark.contexts.model.Server attribute), 3
flush_result_data() (yardstick.dispatcher.base.Base method), 20
flush_result_data() (yardstick.dispatcher.file.FileDispatcher method), 20

flush_result_data() (yardstick.dispatcher.http.HttpDispatcher method), 20
flush_result_data() (yardstick.dispatcher.influxdb.InfluxdbDispatcher method), 21
FrameworkApi (class in yardstick.vTC.apexlake.experimental_framework.api), 26

G

generates_templates() (in module yardstick.vTC.apexlake.experimental_framework.heat_template_generation), 31
get() (yardstick.benchmark.contexts.base.Context static method), 1
get() (yardstick.benchmark.contexts.model.PlacementGroup static method), 3
get() (yardstick.benchmark.runners.base.Runner static method), 4
get() (yardstick.benchmark.scenarios.base.Scenario static method), 17
get() (yardstick.dispatcher.base.Base static method), 20
get_all_heat_templates() (in module yardstick.vTC.apexlake.experimental_framework.heat_template_generation), 31
get_attacker_cls() (yardstick.benchmark.scenarios.availability.attacker.baseattacker.BaseAttacker static method), 7
get_base_dir() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_benchmark_class() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit static method), 27
get_benchmark_name() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 28
get_benchmarks_from_conf_file() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_children() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31
get_cls() (yardstick.benchmark.contexts.base.Context static method), 1
get_cls() (yardstick.benchmark.runners.base.Runner static method), 4
get_cls() (yardstick.benchmark.scenarios.base.Scenario static method), 17
get_cls() (yardstick.dispatcher.base.Base static method), 20
get_credentials() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_deployment_configuration_variables_from_conf_file() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_dpdk_pktgen_vars() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_experiment_configuration() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 28
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 23
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 24
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 24
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 24
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 24
get_features() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 25
get_file_first_line() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_heat_template_params() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_leaves() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode static method), 31
get_monitor_cls() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor static method), 7
get_name() (yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmarking_unit.BenchmarkingUnit method), 23
get_params() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 28
get_parent() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31
get_path() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31
get_required_benchmarks() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit static method), 28
get_result_dir() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_script_fullpath() (yardstick.benchmark.scenarios.availability.attacker.baseattacker.BaseAttacker method), 7
get_script_fullpath() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
get_sections() (in module yard-

stick.vTC.apexlake.experimental_framework.constants.conf_file_sections), 25
get_sections_api() (in module yardstick.vTC.apexlake.experimental_framework.constants.conf_file_sections), 25
get_server() (yardstick.benchmark.contexts.base.Context static method), 1
get_supported_packet_generators() (in module yardstick.vTC.apexlake.experimental_framework.constants.framework_constants), 25
get_template_dir() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_test_case_features() (yardstick.vTC.apexlake.experimental_framework.api.FrameworkApi static method), 27
get_testcase_params() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
get_types() (yardstick.benchmark.runners.base.Runner static method), 4
get_types() (yardstick.benchmark.scenarios.base.Scenario static method), 17
get_variable() (yardstick.vTC.apexlake.experimental_framework.common.ConfigurationFile method), 28
get_variable_list() (yardstick.vTC.apexlake.experimental_framework.common.ConfigurationFile method), 28
get_variable_name() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31
get_variable_value() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 31

H

has_route_to() (yardstick.benchmark.contexts.model.Network method), 2
HeatContext (class in yardstick.benchmark.contexts.heat), 2
HeatManager (class in yardstick.vTC.apexlake.experimental_framework.heat_manager), 30
HeatObject (class in yardstick.orchestrator.heat), 21
HeatStack (class in yardstick.orchestrator.heat), 21
HeatTemplate (class in yardstick.orchestrator.heat), 22
HttpDispatcher (class in yardstick.dispatcher.http), 20

I

image (yardstick.benchmark.contexts.heat.HeatContext attribute), 2
image (yardstick.benchmark.contexts.model.Server attribute), 3
import_modules_from_package() (in module yardstick.common.utils), 20
InfluxdbDispatcher (class in yardstick.dispatcher.influxdb), 21
init() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
init() (yardstick.benchmark.contexts.base.Context method), 1
init() (yardstick.benchmark.contexts.dummy.DummyContext method), 2
init() (yardstick.benchmark.contexts.heat.HeatContext method), 2
init() (yardstick.benchmark.contexts.node.NodeContext method), 3
init() (yardstick.vTC.apexlake.experimental_framework.api.FrameworkApi static method), 27
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmark_base_class method), 23
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_benchmark.Context method), 24
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_benchmark.VNF method), 24
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.multi_tenancy_throughput_benchmark.MultiTenancyThroughputBenchmark method), 24
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544_throughput_benchmark.RFC2544ThroughputBenchmark method), 25
init() (yardstick.vTC.apexlake.experimental_framework.benchmarks.test_benchmark.TestBenchmark method), 25
init() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.TreeNode method), 29
init() (yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_pktgen.TreeNode method), 26
init_general_vars() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
init_heat() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
init_influxdb() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
init_log() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
init_monitors() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
init_pktgen() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
initialize() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 28
inject_fault() (yardstick.benchmark.scenarios.availability.attacker.attacker_baseclass.AttackerBaseClass

method), 6
inject_fault() (yardstick.benchmark.scenarios.availability.attacker.attacker_process.AttackerProcess method), 6
InputValidation (class in yardstick.vTC.apexlake.experimental_framework.common), 29
InstantiationValidationBenchmark (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_benchmark), 24
InstantiationValidationNoisyNeighborsBenchmark (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_noisy_neighbors_benchmark), 24
Iperf (class in yardstick.benchmark.scenarios.networking.iperf3), 12
is_ip_addr() (in module yardstick.cmd.commands.task), 18
is_really_missing() (in module yardstick.common.task_template), 19
is_stack_deployed() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
IterationRunner (class in yardstick.benchmark.runners.iteration), 5
itersubclasses() (in module yardstick.common.utils), 20

J

join() (yardstick.benchmark.runners.base.Runner method), 4

L

LATENCY_BENCHMARK_SCRIPT (yardstick.benchmark.scenarios.compute.lmbench.Lmbench attribute), 10
list (yardstick.benchmark.contexts.base.Context attribute), 1
list (yardstick.benchmark.contexts.model.Network attribute), 2
list (yardstick.benchmark.contexts.model.Server attribute), 3
Lmbench (class in yardstick.benchmark.scenarios.compute.lmbench), 10

M

main() (in module yardstick.main), 32
main() (yardstick.cmd.cli.YardstickCLI method), 19
make_lines() (in module yardstick.dispatcher.influxdb_line_protocol), 21
map (yardstick.benchmark.contexts.model.PlacementGroup attribute), 3
METADATA_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
monitor_func() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
monitor_func() (yardstick.benchmark.scenarios.availability.monitor.monitor_command.MonitorOpenstackCmd method), 8
monitor_func() (yardstick.benchmark.scenarios.availability.monitor.monitor_process.MonitorProcess method), 8
MonitorMgr (class in yardstick.benchmark.scenarios.availability.monitor.basemonitor), 7
MonitorOpenstackCmd (class in yardstick.benchmark.scenarios.availability.monitor.monitor_command), 8
MonitorProcess (class in yardstick.benchmark.scenarios.availability.monitor.monitor_process), 8
MPSTAT_FIELD_SIZE (yardstick.benchmark.scenarios.compute.cpuload.CPULoad attribute), 9
MultiTenancyThroughputBenchmark (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.multi_tenancy_throughput_benchmark), 24

N

Netperf (class in yardstick.benchmark.scenarios.networking.netperf), 12
Network (class in yardstick.benchmark.contexts.model), 2
NodeContext (class in yardstick.benchmark.contexts.node), 3

O

Object (class in yardstick.benchmark.contexts.model), 2

P

parse() (in module yardstick.common.template_format), 19
parse_nodes_with_context() (in module yardstick.cmd.commands.task), 18
parse_suite() (yardstick.cmd.commands.task.TaskParser method), 18
parse_task() (yardstick.cmd.commands.task.TaskParser method), 18
parse_task_args() (in module yardstick.cmd.commands.task), 18
Parser (class in yardstick.benchmark.scenarios.parser.parser), 16
PARSER_SCRIPT (yardstick.benchmark.scenarios.parser.parser.Parser attribute), 16

Perf (class in yardstick.benchmark.scenarios.compute.perf), 10
Ping (class in yardstick.benchmark.scenarios.networking.ping), 13
Ping6 (class in yardstick.benchmark.scenarios.networking.ping6), 13
Pktgen (class in yardstick.benchmark.scenarios.networking.pktgen), 14
PlacementGroup (class in yardstick.benchmark.contexts.model), 3
POST_TEARDOWN_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
PRE_SETUP_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
PRE_SETUP_SCRIPT (yardstick.benchmark.scenarios.networking.sfc.Sfc attribute), 14
print_hbar() (in module yardstick.cmd), 19
print_invalid_header() (in module yardstick.cmd.commands.task), 18
print_stacks() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
ProcessAttacker (class in yardstick.benchmark.scenarios.availability.attacker.attacker_process), 6
push_data_influxdb() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
put() (yardstick.ssh.SSH method), 33

Q

queue (yardstick.benchmark.runners.base.Runner attribute), 4

R

RADVD_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
REBOOT_CMD_PATTERN (yardstick.benchmark.scenarios.compute.cyclictest.Cyclictest attribute), 9
record_result_data() (yardstick.dispatcher.base.Base method), 20
record_result_data() (yardstick.dispatcher.file.FileDispatcher method), 20
record_result_data() (yardstick.dispatcher.http.HttpDispatcher method), 20
record_result_data() (yardstick.dispatcher.influxdb.InfluxdbDispatcher method), 21
recover() (yardstick.benchmark.scenarios.availability.attacker.attacker_barer method), 6
recover() (yardstick.benchmark.scenarios.availability.attacker.attacker_proce method), 6
release() (yardstick.benchmark.runners.base.Runner static method), 4
release_dump_process() (yardstick.benchmark.runners.base.Runner static method), 5
render() (yardstick.common.task_template.TaskTemplate class method), 19
replace_in_file() (in module yardstick.vTC.apexlake.experimental_framework.common), 29
RFC2544ThroughputBenchmark (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544 24
Router (class in yardstick.benchmark.contexts.model), 3
run() (yardstick.benchmark.runners.base.Runner method), 5
run() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.Base method), 7
run() (yardstick.benchmark.scenarios.availability.serviceha.ServiceHA method), 8
run() (yardstick.benchmark.scenarios.base.Scenario method), 17
run() (yardstick.benchmark.scenarios.compute.cpuload.CPULoad method), 9
run() (yardstick.benchmark.scenarios.compute.cyclictest.Cyclictest method), 10
run() (yardstick.benchmark.scenarios.compute.lmbench.Lmbench method), 10
run() (yardstick.benchmark.scenarios.compute.perf.Perf method), 11
run() (yardstick.benchmark.scenarios.compute.unixbench.Unixbench method), 11
run() (yardstick.benchmark.scenarios.dummy.dummy.Dummy method), 12
run() (yardstick.benchmark.scenarios.networking.iperf3.Iperf method), 12
run() (yardstick.benchmark.scenarios.networking.netperf.Netperf method), 13
run() (yardstick.benchmark.scenarios.networking.ping.Ping method), 13
run() (yardstick.benchmark.scenarios.networking.ping6.Ping6 method), 14
run() (yardstick.benchmark.scenarios.networking.pktgen.Pktgen method), 14
run() (yardstick.benchmark.scenarios.networking.sfc.Sfc method), 14
run() (yardstick.benchmark.scenarios.networking.vtc_instantiation_validatio

method), 15
run() (yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy.VtcInstantiationValidationNoisy method), 15
run() (yardstick.benchmark.scenarios.networking.vtc_throughput.VtcThroughput method), 15
run() (yardstick.benchmark.scenarios.networking.vtc_throughput_noisy.VtcThroughputNoisy method), 15
run() (yardstick.benchmark.scenarios.parser.parser.Parser method), 16
run() (yardstick.benchmark.scenarios.storage.fio.Fio method), 17
run() (yardstick.ssh.SSH method), 33
run() (yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmark_base_class.BenchmarkBaseClass method), 23
run() (yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_validation_benchmark.InstantiationValidationBenchmark method), 24
run() (yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544_throughput_benchmark.RFC2544ThroughputBenchmark method), 25
run() (yardstick.vTC.apexlake.experimental_framework.benchmarks.test_benchmark.TestBenchmark method), 25
run_benchmarks() (yardstick.vTC.apexlake.experimental_framework.benchmarking_unit.BenchmarkingUnit method), 28
run_command() (in module yardstick.vTC.apexlake.experimental_framework.common), 30
run_one_scenario() (in module yardstick.cmd.commands.task), 18
run_post_stop_action() (yardstick.benchmark.runners.base.Runner method), 5
Runner (class in yardstick.benchmark.runners.base), 4
runner_join() (in module yardstick.cmd.commands.task), 18
RunnerCommands (class in yardstick.cmd.commands.runner), 17
runners (yardstick.benchmark.runners.base.Runner attribute), 5

S

Scenario (class in yardstick.benchmark.scenarios.base), 17
ScenarioCommands (class in yardstick.cmd.commands.scenario), 18
send_traffic() (yardstick.vTC.apexlake.experimental_framework.packet_generators.base_packet_generator.BasePacketGenerator method), 26
send_traffic() (yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_packet_generator.DpdkPacketGenerator method), 26
SequenceRunner (class in yardstick.benchmark.runners.sequence), 5
Server (class in yardstick.benchmark.contexts.model), 3
SERVER_SCRIPT (yardstick.benchmark.scenarios.networking.sfc.Sfc attribute), 14
ServiceHA (class in yardstick.benchmark.scenarios.availability.serviceha), 8
set_vtc_throughput() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.VTCTemplate method), 31
set_variable_value() (yardstick.vTC.apexlake.experimental_framework.heat_template_generation.VTCTemplate method), 31
setup() (yardstick.benchmark.scenarios.availability.attacker.attacker_baremetal.AttackerBaremetal method), 6
setup() (yardstick.benchmark.scenarios.availability.attacker.attacker_process.AttackerProcess method), 6
setup() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
setup() (yardstick.benchmark.scenarios.availability.monitor.monitor_command.MonitorCommand method), 8
setup() (yardstick.benchmark.scenarios.availability.monitor.monitor_process.MonitorProcess method), 8
setup() (yardstick.benchmark.scenarios.availability.serviceha.ServiceHA method), 8
setup() (yardstick.benchmark.scenarios.base.Scenario method), 17
setup() (yardstick.benchmark.scenarios.compute.cpuload.CPULoad method), 9
setup() (yardstick.benchmark.scenarios.compute.cyclictest.Cyclictest method), 10
setup() (yardstick.benchmark.scenarios.compute.lmbench.Lmbench method), 10
setup() (yardstick.benchmark.scenarios.compute.perf.Perf method), 11
setup() (yardstick.benchmark.scenarios.compute.unixbench.Unixbench method), 11
setup() (yardstick.benchmark.scenarios.dummy.dummy.Dummy method), 12
setup() (yardstick.benchmark.scenarios.networking.iperf3.Iperf method), 12
setup() (yardstick.benchmark.scenarios.networking.netperf.Netperf method), 13
setup() (yardstick.benchmark.scenarios.networking.ping6.Ping6 method), 14
setup() (yardstick.benchmark.scenarios.networking.pktgen.Pktgen method), 14
setup() (yardstick.benchmark.scenarios.networking.sfc.Sfc method), 14
setup() (yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy.VtcInstantiationValidationNoisy method), 15
setup() (yardstick.benchmark.scenarios.networking.vtc_instantiation_validation.VtcInstantiationValidation method), 15
setup() (yardstick.benchmark.scenarios.networking.vtc_throughput.VtcThroughput method), 15
setup() (yardstick.benchmark.scenarios.networking.vtc_throughput_noisy.VtcThroughputNoisy method), 15
setup() (yardstick.benchmark.scenarios.parser.parser.Parser

method), 16
setup() (yardstick.benchmark.scenarios.storage.fio.Fio method), 17
SETUP_ODL_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
SETUP_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
SETUP_SCRIPT (yardstick.benchmark.scenarios.parser.parser.Parser attribute), 16
Sfc (class in yardstick.benchmark.scenarios.networking.sfc), 14
SSH (class in yardstick.ssh), 33
SSHError, 33
SSHTimeout, 33
stacks (yardstick.orchestrator.heat.HeatStack attribute), 21
stacks_exist() (yardstick.orchestrator.heat.HeatStack static method), 21
start_monitor() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
start_monitors() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.MonitorMgr method), 7
status() (yardstick.orchestrator.heat.HeatObject method), 21

T

TACKER_SCRIPT (yardstick.benchmark.scenarios.networking.sfc.Sfc attribute), 14
TARGET_SCRIPT (yardstick.benchmark.scenarios.compute.cyclictest.Cyclictest attribute), 9
TARGET_SCRIPT (yardstick.benchmark.scenarios.compute.perf.Perf attribute), 11
TARGET_SCRIPT (yardstick.benchmark.scenarios.compute.unixbench.Unixbench attribute), 11
TARGET_SCRIPT (yardstick.benchmark.scenarios.networking.netperf.Netperf attribute), 13
TARGET_SCRIPT (yardstick.benchmark.scenarios.networking.ping.Ping attribute), 13
TARGET_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
TARGET_SCRIPT (yardstick.benchmark.scenarios.networking.pktgen.Pktgen attribute), 14
TARGET_SCRIPT (yardstick.benchmark.scenarios.storage.fio.Fio attribute), 16
TaskCommands (class in yardstick.cmd.commands.task), 18
TaskParser (class in yardstick.cmd.commands.task), 18
TaskTemplate (class in yardstick.common.task_template), 19
teardown() (yardstick.benchmark.scenarios.availability.serviceha.ServiceHA method), 8
teardown() (yardstick.benchmark.scenarios.base.Scenario method), 17
teardown() (yardstick.benchmark.scenarios.networking.iperf3.Iperf method), 12
teardown() (yardstick.benchmark.scenarios.networking.ping6.Ping6 method), 14
teardown() (yardstick.benchmark.scenarios.networking.sfc.Sfc method), 14
teardown() (yardstick.benchmark.scenarios.parser.parser.Parser method), 16
TEARDOWN_SCRIPT (yardstick.benchmark.scenarios.networking.ping6.Ping6 attribute), 13
TEARDOWN_SCRIPT (yardstick.benchmark.scenarios.networking.sfc.Sfc attribute), 14
TEARDOWN_SCRIPT (yardstick.benchmark.scenarios.parser.parser.Parser attribute), 16
terminate() (yardstick.benchmark.runners.base.Runner static method), 5
terminate_all() (yardstick.benchmark.runners.base.Runner static method), 5
TestBenchmark (class in yardstick.vTC.apexlake.experimental_framework.benchmarks.test_be 25
TreeNode (class in yardstick.vTC.apexlake.experimental_framework.heat_template_gene 30
try_append_module() (in module yardstick.common.utils), 20

U

undeploy() (yardstick.benchmark.contexts.base.Context method), 1
undeploy() (yardstick.benchmark.contexts.dummy.DummyContext method), 2
undeploy() (yardstick.benchmark.contexts.heat.HeatContext method), 2
undeploy() (yardstick.benchmark.contexts.node.NodeContext method), 3
Unixbench (class in yardstick.benchmark.scenarios.compute.unixbench), 11

update() (yardstick.orchestrator.heat.HeatStack method), 21
update_scheduler_hints() (in module yardstick.benchmark.contexts.model), 3
user (yardstick.benchmark.contexts.heat.HeatContext attribute), 2

V

validate_boolean() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_configuration_file_parameter() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_configuration_file_section() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_dictionary() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_directory_exist_and_format() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_file_exist() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_heat_template() (yardstick.vTC.apexlake.experimental_framework.heat_manager.HeatManager method), 30
validate_integer() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_os_credentials() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
validate_string() (yardstick.vTC.apexlake.experimental_framework.common.InputValidation static method), 29
verify_SLA() (yardstick.benchmark.scenarios.availability.monitor.baseMonitor.BaseMonitor method), 7
verify_SLA() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.MonitorMgr method), 7
verify_SLA() (yardstick.benchmark.scenarios.availability.monitor.monitor_command.MonitorOpenstackCmd method), 8
verify_SLA() (yardstick.benchmark.scenarios.availability.monitor.monitor_process.MonitorProcess method), 8
VtcInstantiationValidation (class in yardstick.benchmark.scenarios.networking.vtc_instantiation_validation), 15
VtcInstantiationValidationNoisy (class in yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy), 15
VtcThroughput (class in yardstick.benchmark.scenarios.networking.vtc_throughput), 15
VtcThroughputNoisy (class in yardstick.benchmark.scenarios.networking.vtc_throughput_noisy), 15

W

wait() (yardstick.ssh.SSH method), 33
wait_monitor() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
wait_monitors() (yardstick.benchmark.scenarios.availability.monitor.basemonitor.BaseMonitor method), 7
WORKSPACE (yardstick.benchmark.scenarios.compute.cyclictest.Cyclictest attribute), 9

Y

yardstick (module), 33
yardstick.benchmark (module), 17
yardstick.benchmark.contexts (module), 3
yardstick.benchmark.contexts.base (module), 1
yardstick.benchmark.contexts.dummy (module), 2
yardstick.benchmark.contexts.heat (module), 2
yardstick.benchmark.contexts.model (module), 2
yardstick.benchmark.contexts.node (module), 3
yardstick.benchmark.runners (module), 6
yardstick.benchmark.runners.arithmetic (module), 4
yardstick.benchmark.runners.base (module), 4
yardstick.benchmark.runners.duration (module), 5
yardstick.benchmark.runners.iteration (module), 5
yardstick.benchmark.runners.sequence (module), 5
yardstick.benchmark.scenarios (module), 17
yardstick.benchmark.scenarios.availability (module), 8
yardstick.benchmark.scenarios.availability.attacker (module), 7
yardstick.benchmark.scenarios.availability.attacker.attacker_baremetal (module), 6
yardstick.benchmark.scenarios.availability.attacker.attacker_process (module), 6
yardstick.benchmark.scenarios.availability.attacker.baseattacker (module), 7
yardstick.benchmark.scenarios.availability.monitor (module), 8
yardstick.benchmark.scenarios.availability.monitor.basemonitor (module), 7
yardstick.benchmark.scenarios.availability.monitor.monitor_command (module), 8
yardstick.benchmark.scenarios.availability.monitor.monitor_process (module), 8
yardstick.benchmark.scenarios.availability.serviceha (module), 8
yardstick.benchmark.scenarios.base (module), 17
yardstick.benchmark.scenarios.compute (module), 11
yardstick.benchmark.scenarios.compute.cpuload (module), 9
yardstick.benchmark.scenarios.compute.cyclictest (module), 9

yardstick.benchmark.scenarios.compute.lmbench (module), 10
yardstick.benchmark.scenarios.compute.perf (module), 10
yardstick.benchmark.scenarios.compute.unixbench (module), 11
yardstick.benchmark.scenarios.dummy (module), 12
yardstick.benchmark.scenarios.dummy.dummy (module), 12
yardstick.benchmark.scenarios.networking (module), 15
yardstick.benchmark.scenarios.networking.iperf3 (module), 12
yardstick.benchmark.scenarios.networking.netperf (module), 12
yardstick.benchmark.scenarios.networking.ping (module), 13
yardstick.benchmark.scenarios.networking.ping6 (module), 13
yardstick.benchmark.scenarios.networking.pktgen (module), 14
yardstick.benchmark.scenarios.networking.sfc (module), 14
yardstick.benchmark.scenarios.networking.vtc_instantiation_validation (module), 15
yardstick.benchmark.scenarios.networking.vtc_instantiation_validation_noisy (module), 15
yardstick.benchmark.scenarios.networking.vtc_throughput (module), 15
yardstick.benchmark.scenarios.networking.vtc_throughput_noisy (module), 15
yardstick.benchmark.scenarios.parser (module), 16
yardstick.benchmark.scenarios.parser.parser (module), 16
yardstick.benchmark.scenarios.storage (module), 17
yardstick.benchmark.scenarios.storage.fio (module), 16
yardstick.cmd (module), 19
yardstick.cmd.cli (module), 19
yardstick.cmd.commands (module), 19
yardstick.cmd.commands.runner (module), 17
yardstick.cmd.commands.scenario (module), 18
yardstick.cmd.commands.task (module), 18
yardstick.common (module), 20
yardstick.common.task_template (module), 19
yardstick.common.template_format (module), 19
yardstick.common.utils (module), 20
yardstick.dispatcher (module), 21
yardstick.dispatcher.base (module), 20
yardstick.dispatcher.file (module), 20
yardstick.dispatcher.http (module), 20
yardstick.dispatcher.influxdb (module), 21
yardstick.dispatcher.influxdb_line_protocol (module), 21
yardstick.main (module), 32
yardstick.orchestrator (module), 23
yardstick.orchestrator.heat (module), 21
yardstick.plot (module), 23
yardstick.resources (module), 23
yardstick.ssh (module), 32
yardstick.vTC (module), 32
yardstick.vTC.apexlake (module), 31
yardstick.vTC.apexlake.experimental_framework (module), 31
yardstick.vTC.apexlake.experimental_framework.api (module), 26
yardstick.vTC.apexlake.experimental_framework.benchmarking_unit (module), 27
yardstick.vTC.apexlake.experimental_framework.benchmarks (module), 25
yardstick.vTC.apexlake.experimental_framework.benchmarks.benchmark_l (module), 23
yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_ (module), 24
yardstick.vTC.apexlake.experimental_framework.benchmarks.instantiation_ (module), 24
yardstick.vTC.apexlake.experimental_framework.benchmarks.multi_tenanc (module), 24
yardstick.vTC.apexlake.experimental_framework.benchmarks.rfc2544_thro (module), 24
yardstick.vTC.apexlake.experimental_framework.benchmarks.test_benchm (module), 25
yardstick.vTC.apexlake.experimental_framework.common (module), 28
yardstick.vTC.apexlake.experimental_framework.constants (module), 25
yardstick.vTC.apexlake.experimental_framework.constants.conf_file_sectio (module), 25
yardstick.vTC.apexlake.experimental_framework.constants.framework_para (module), 25
yardstick.vTC.apexlake.experimental_framework.deployment_unit (module), 30
yardstick.vTC.apexlake.experimental_framework.heat_manager (module), 30
yardstick.vTC.apexlake.experimental_framework.heat_template_generation (module), 30
yardstick.vTC.apexlake.experimental_framework.libraries (module), 25
yardstick.vTC.apexlake.experimental_framework.packet_generators (module), 26
yardstick.vTC.apexlake.experimental_framework.packet_generators.base_p (module), 26
yardstick.vTC.apexlake.experimental_framework.packet_generators.dpdk_p (module), 26
YardstickCLI (class in yardstick.cmd.cli), 19