Red Hat Reference Architecture Series

Guidelines and Considerations for Performance and Scaling your Red Hat Enterprise Linux OpenStack Platform 7 Cloud

Roger Lopez, Joe Talerico

Version 1.0, 2016-02-12

Table of Contents

Comments and Feedback . . . 2
Staying In Touch . . . 2
Like us on Facebook . . . 2
Follow us on Twitter . . . 2
Plus us on Google+ . . . 2
Executive Summary . . . 3
Reference Architecture Environment . . . 4
Reference Architecture Overview . . . 4
Server Roles . . . 5
Network Topology . . . 6
Best Practices Benchmarking RHEL-OSP 7 Environments . . . 8
Rally . . . 9
Before you Begin . . . 9
Enabling the Required Channels for the Rally VM . . . 10
Rally Installation . . . 11
Rally Configuration . . . 12
Rally Extending Tenant Network (Optional) . . . 13
Benchmarking with Rally . . . 18
Analyzing RHEL-OSP 7 Benchmark Results with Rally . . . 22
Initial boot-storm Rally Results . . . 27
Rally Max Guest Launch . . . 33
Tuning the RHEL-OSP 7 Environment using Browbeat . . . 39
Installation of Browbeat . . . 39
Analyzing Performance Issues generated by the Ansible Playbook . . . 40
Conclusion . . . 49
Appendix A: Contributors . . . 51
Appendix B: References . . . 52
Appendix C: Hardware specifications . . . 53
Appendix D: Rally Validate Nova JSON File . . . 54
Appendix E: Rally Boot and List Server JSON File . . . 55

100 East Davie Street
Raleigh NC 27601 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

The GPG fingerprint of the security@redhat.com key is: CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Send feedback to refarch-feedback@redhat.com

Comments and Feedback

In the spirit of open source, we invite anyone to provide feedback and comments on any of the reference architectures. Although we review our papers internally, sometimes issues or typographical errors are encountered. Feedback allows us to not only improve the quality of the papers we produce, but allows the reader to provide their thoughts on potential improvements and topic expansion to the papers. Feedback on the papers can be provided by emailing refarch-feedback@redhat.com. Please refer to the title within the email.

Staying In Touch

Join us on some of the popular social media sites where we keep our audience informed on new reference architectures as well as offer related information on things we find interesting.

Like us on Facebook

https://www.facebook.com/rhrefarch

Follow us on Twitter

https://twitter.com/RedHatRefArch

Plus us on Google+

https://plus.google.com/u/0/b/114152126783830728030/

Executive Summary

The purpose of this reference architecture is to provide a performance and scaling methodology comprised of utilizing common benchmark workloads. With this methodology, end users can determine bottlenecks that may arise when scaling an OpenStack environment, learn to create benchmarking scenarios to optimize an OpenStack private cloud environment, and analyze the underlying results of those scenarios. This reference environment sets its foundation around the "Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1" reference architecture. For any information relating to the best practices in deploying Red Hat Enterprise Linux OpenStack Platform 7 (RHEL-OSP7), please visit: http://red.ht/1PDxLxp

This reference architecture is best suited for system, storage, and OpenStack administrators deploying RHEL-OSP 7 with the intent of using open source tools to assist in scaling their private cloud environment. The topics that will be covered consist of:

- Benchmarking using Rally [1: https://wiki.openstack.org/wiki/Rally] scenarios
- Analyzing the different Rally scenario benchmarking results
- Using the open source project *Browbeat* [2: https://github.com/jtaleric/browbeat] to assist in determining potential performance issues
- Detailing the best practices to optimize a Red Hat Enterprise Linux OpenStack Platform 7 environment

Reference Architecture Environment

This section focuses on the components for the deployment and scaling of Red Hat Enterprise Linux OpenStack Platform 7 on Red Hat Enterprise Linux 7.

Reference Architecture Overview

This Reference Architecture Diagram is the same configuration used in *Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1* by Jacob Liberman [3: http://red.ht/1PDxLxp] with an additional server added for hosting the Rally VM.

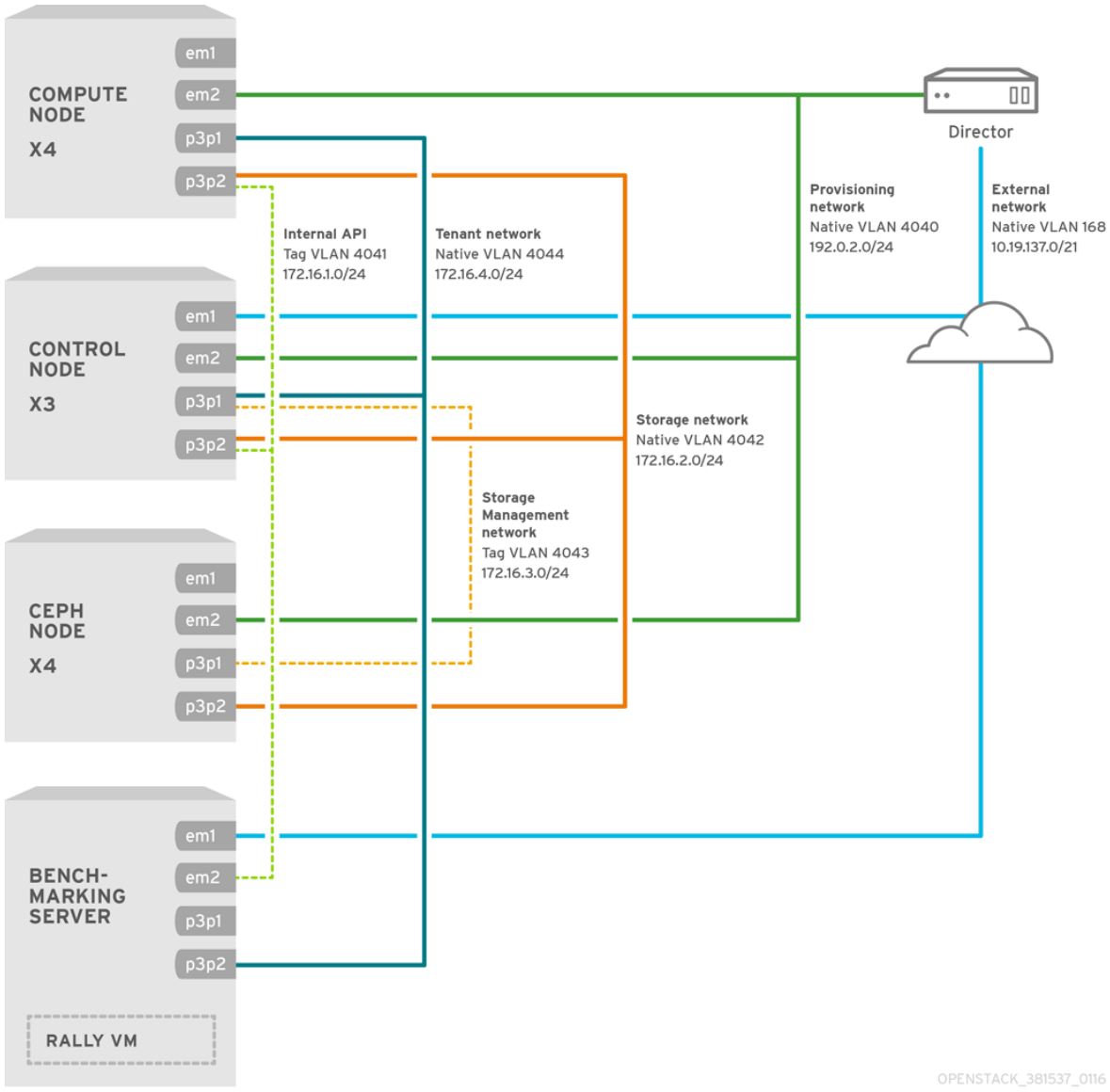

Figure 2.1: Reference Architecture Diagram

The Network Topology section of this paper describes the networking components in detail.

The following Server Roles and Network Topology detail the environment as described in *Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1*. While these two sections can be found in the original reference architecture paper, it has been included in this reference architecture for easy access to the reader.

Server Roles

As depicted in Figure 2.1 Reference Architecture Diagram, the use case requires 13 bare metal servers deployed with the following roles:

- 1 undercloud server
- 3 cloud controllers
- 4 cloud compute nodes
- 4 Ceph storage servers
- 1 benchmarking server hosting the Rally VM

Servers are assigned to roles based on their hardware characteristics.

Table 1. Server hardware by role

Role	Count	Model
Undercloud	1	Dell PowerEdge R720xd
Cloud controller	3	Dell PowerEdge M520
Compute node	4	Dell PowerEdge M520
Ceph storage server	4	Dell PowerEdge R510
Benchmarking Server	1	Dell PowerEdge M520

Appendix C Hardware specifications lists hardware specifics for each server model.

Network Topology

Figure 2.1 Reference Architecture Diagram details and describes the network topology of this reference architecture.

Each server has two Gigabit interfaces (nic1:2) and two 10-Gigabit interfaces (nic3:4) for network isolation to segment OpenStack communication by type.

The following network traffic types are isolated:

- Provisioning
- Internal API
- Storage
- Storage Management
- Tenant
- External

There are six isolated networks but only four physical interfaces. Two networks are isolated on each physical 10 Gb interface using a combination of tagged and native VLANs.

Table 2. Network Isolation

Role	Interface	VLAN ID	Network	VLAN Type	CIDR
Undercloud	nic1	168	External	Native	10.19.137.0/21
	nic2	4040	Provisioning	Native	192.0.2.0/24
Control	nic1	168	External	Native	10.19.137.0/21
	nic2	4040	Provisioning	Native	192.0.2.0/24
	nic3	4043	Storage Mgmt	Tagged	172.16.3.0/24
	nic3	4044	Tenant	Native	172.16.4.0/24
	nic4	4041	Internal API	Tagged	172.16.1.0/24
	nic4	4042	Storage	Native	172.16.2.0/24
Compute	nic2	4040	Provisioning	Native	192.0.2.0/24
	nic3	4044	Tenant	Native	172.16.4.0/24
	nic4	4041	Internal API	Tagged	172.16.1.0/24
	nic4	4042	Storage	Native	172.16.2.0/24

Role	Interface	VLAN ID	Network	VLAN Type	CIDR
Ceph storage	nic2	4040	Provisioning	Native	192.0.2.0/24
	nic3	4043	Storage Mgmt	Tagged	172.16.3.0/24
	nic4	4042	Storage	Native	172.16.2.0/24
Benchmarking Server	nic1	168	External	Native	10.19.137.0/21
	nic2	4041	Internal API	Tagged	172.16.1.0/24
	nic3	4044	Tenant	Native	172.16.4.0/24
Rally VM	nic1	168	External	Native	10.19.137.0/21
	nic2	4041	Internal API	Tagged	172.16.1.0/24
	nic3	-	demo_net*	-	172.16.5.0/24

All switch ports must be added to their respective VLANs prior to deploying the overcloud.

`demo_net` relates to the traffic that is flowing through the Tenant network when VMs get an IP address.

Best Practices Benchmarking RHEL-OSP 7 Environments

With a successful deployment completed as described in *Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1*, the next steps involve benchmarking the RHEL-OSP environment. Step one involves using Rally as a benchmarking tool to capture initial baseline results. Baseline results are critical because they provide a foundation on how the RHEL-OSP 7 environment is performing. Without it, modifications that are made to the RHEL-OSP environment cannot be compared in order validate whether future tuning changes effected an environment positively or negatively. Once the baseline results for Rally have been captured, step two involves the use of Tuning the RHEL-OSP 7 Environment using Browbeat. Browbeat was originally developed to investigate the number of OpenStack Service workers needed and database connections utilized. It has since developed new features which allows us to check RHEL-OSP Deployments for common mistakes and run a variety of Rally scenarios in a automated fashion.

This paper focuses on using Browbeat specifically for its `ansible` playbooks that perform performance checks across the overcloud environment. This reference environment uses the performance checks to determine what is to be tuned within the RHEL-OSP 7 environment.

The following is a breakdown list of how to address benchmarking and tuning of a RHEL-OSP environment.

- Benchmark using Rally to capture baseline results.
- Run performance checks on the RHEL-OSP environment using Browbeat
- Re-run the Rally scenarios to capture latest results with the tuning changes.

The subsequent chapters provide the details in running the Rally benchmark and performance checking with Browbeat.

Rally

Rally [1: https://wiki.openstack.org/wiki/Rally] is a benchmarking tool created to answer the underlying question of *"How does OpenStack work at scale?"*. Rally is able to achieve the answer to this question by automating the processes that entails the OpenStack deployment, cloud verification, benchmarking, and profiling. While Rally has the capabilities to offer an assortment of Rally actions to test and validate the OpenStack cloud, this reference environment focuses specifically on using Rally as a benchmarking tool to test specific scenarios using an existing RHEL-OSP cloud and generate HTML reports based upon the captured results.

At the time of this writing, Rally is **not supported** with RHEL-OSP 7, however, it is a key component to benchmark the RHEL-OSP 7 environment.

Before you Begin

Ensure the following prerequisites are met:

- Successful deployment of RHEL-OSP 7 as described in *Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1*
- Deployed virtual machine where the Rally binaries and benchmarking results are to reside
- Rally virtual machine requires access to the Internal API network, Tenant network, and Provisioning network

Please refer to Creating Guests with Virt Manager for more information on creating a virtual machine.

Enabling the Required Channels for the Rally VM

The following repositories are required for a successful installation of the Rally environment.

Table 3. Required Channels - Rally VM

Channel	Repository Name
Extra Packages for Enterprise Linux 7 - x86_64	epel
Red Hat Enterprise Linux 7 Server (RPMS)	rhel-7-server-rpms
Red Hat Enterprise Linux OpenStack Platform 7.0 (RPMS)	rhel-7-server-openstack-7.0-rpms
Red Hat Enterprise Linux High Availability (for RHEL 7 Server) (RPMs)	rhel-ha-for-rhel-7-server-rpms

The following steps are required to subscribe to the appropriate channels listed above.

1. Register the Rally VM with the Content Delivery Network (CDN), using your Customer Portal user name and password credentials.

```
$ sudo subscription-manager register
```

2. Register to the appropriate entitlement pool for the Red Hat Enterprise Linux OpenStack Platform director.

```
$ sudo subscription-manager list --available --all
```

3. Use the pool ID located to attach to the Red Hat Enterprise Linux OpenStack Platform 7 entitlements.

```
$ sudo subscription-manager attach --pool=<ID>
```

4. Disable all default repositories.

```
$ sudo subscription-manager repos --disable=*
```

5. Enable only the required repositories.

```
$ sudo subscription-manager repos --enable=rhel-7-server-rpms --enable=rhel-7-server-openstack-7.0-rpms
--enable=rhel-ha-for-rhel-7-server-rpms
```

6. Install the `epel` package to enable the repository.

```
$ yum install https://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-5.noarch.rpm
```

The `epel` repository is **only** required within the Rally VM.

Rally Installation

With the Rally VM setup with the appropriate repositories, the next step involves the installation of Rally. The steps below provide a step-by-step on the installation process.

Within the Rally virtual machine, as the `root` user,

1. Create a working directory to clone the rally git repository.

```
# mkdir /path/to/myrally
```

2. If not already installed, install `git` using the `yum` command.

```
# yum install git
```

3. Access the working directory.

```
# cd /path/to/myrally/rally
```

4. Clone the Rally repository using `git`.

```
# git clone https://github.com/openstack/rally.git
```

5. Run the `install_rally.sh` script and follow the prompts to install the required dependencies.

```
# /path/to/myrally/rally/install_rally.sh
[ ... Output Abbreviated ...]
Installing rally-manage script to /usr/bin
Installing rally script to /usr/bin
=======================================================================
Information about your Rally installation:
 * Method: system
 * Database at: /var/lib/rally/database
 * Configuration file at: /etc/rally
=======================================================================
```

Rally Configuration

With a successful Rally install, the next steps are to configure the Rally environment by providing Rally access to the overcloud environment. A step-by-step on the configuration process can be found below.

1. Export the RHEL-OSP 7 environment variables with the proper credentials. The credentials for this reference environment are as follows:

```
# export OS_USERNAME=admin
# export OS_TENANT_NAME=admin
# export OS_PASSWORD=
# export OS_AUTH_URL=http://20.0.0.29:35357/v2.0/*
```

The OS_PASSWORD is omitted.

2. Add the existing RHEL-OSP 7 deployment to the Rally database using `--fromenv`.

```
# rally deployment create --fromenv --name=<name>
```

Rally Extending Tenant Network (Optional)

The following steps are optional and only required if the Rally benchmark scenario being run requires direct access to the guest. Direct access refers to `ssh` capability using the Tenant network instead of floating IPs. When running the ***NovaServers.boot_server*** scenario, extending the *Tenant* network is not required as this specific scenario does not `ssh` into the guests but simply launches the guest instances.

In order to enable direct access to the launched guests, please follow the instructions below. The following steps all reside within the benchmarking server. The benchmarking server is hosting the Rally VM.

1. If the current RHEL-OSP 7 deployment does not have any floating IPs available, extend the *Tenant* network to the benchmarking server. Run the following to install the Neutron Open vSwitch agent package within the benchmarking server.

```
# yum install openstack-neutron-openvswitch
```

2. Install the `openstack-selinux` package if it does not already reside within benchmarking server.

```
# yum install openstack-selinux
```

3. Copy the Neutron configuration files from a RHEL-OSP 7 compute node that has been deployed by the RHEL-OSP 7 undercloud node to the benchmarking server.

```
# scp <Compute_Node_IP>:/etc/neutron/* /etc/neutron/
```

4. Ensure the permissions for the copied `/etc/neutron` directory are of owner `root` and group `neutron`.

```
# chown -R root.neutron /etc/neutron
```

5. Set an available IP address to the interface that would have been associated with the *Tenant* network had it been part of the RHEL-OSP 7 cluster. Within this reference environment, nic3 is the network interface used as noted in the Network Isolation table. Edit the */etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini* file and change `local_ip` to an IP address that resides on the *Tenant* network. This reference environment's *Tenant* network resides in the 172.16.4.x subnet.

```
local_ip =172.16.4.250
```

6. The default *ovs_neutron_plugin.ini* looks for a bridge labeled `br-ex`. If it does not already exist, create a bridge labeled `br-ex` that resides within the benchmarking server.

```
# brctl addbr br-ex
```

7. Restart the `openvswitch` and `neutron-openvswitch-agent` services.

```
# systemctl restart openvswitch
# systemctl restart neutron-openvswitch-agent
```

The following steps should be performed on the `undercloud` server.

1. As the `stack` user, source the *overcloudrc* file to set environment variables for the overcloud.

```
# su - stack
# source overcloudrc
```

2. List the Neutron networks.

```
# neutron net-list
+--------------------------------------+------------------------------------------------------
+-------------------------------------------------------+
| id                                   | name                                                 | subnets
|
+--------------------------------------+------------------------------------------------------
+-------------------------------------------------------+
| 0fd1b597-7ed0-45cf-b9e2-a5dfbee80377 | demo_net                                             | f5a4fbf8-
1d86-4d64-ad9e-4d012a5fd1b7 172.16.5.0/24    |
| cee2c56b-2bb1-476d-9905-567af6e86978 | HA network tenant cc5e33d027b847d480957f5e30d04620 | 6dcf6e82-
0f13-4418-bca9-9791b11da05a 169.254.192.0/18 |
| 330c5120-9e36-4049-a036-6997733af443 | ext-net                                              | d9200dac-
a99b-41ad-88be-f79bb2f06676 10.19.136.0/21   |
+--------------------------------------+------------------------------------------------------
+-------------------------------------------------------+
```

3. Capture the `demo_net` uuid and export the variable `netid` with the specified value

```
# export netid=0fd1b597-7ed0-45cf-b9e2-a5dfbee80377
```

4. Verify the `netid` is the value expected.

```
# echo $netid
0fd1b597-7ed0-45cf-b9e2-a5dfbee80377
```

5. Export a variable labeled `hostid` with the `hostname` value.

```
# export hostid=iaas-vms.cloud.lab.eng.bos.redhat.com
```

The benchmarking servers hostname is `iaas-vms.cloud.lab.eng.bos.redhat.com` for this reference environment. This server hosts the Rally VM.

6. Verify the `hostid` value.

```
# echo $hostid
iaas-vms.cloud.lab.eng.bos.redhat.com
```

7. Create a `neutron` port labeled `rally-port` that binds to the `host_id` and creates the port within the network of the associated `netid`.

```
# neutron port-create --name rally-port --binding:host_id=$hostid $netid
Created a new port:
+-----------------------
+-------------------------------------------------------------------------------+
| Field                 | Value
|
+-----------------------
+-------------------------------------------------------------------------------+
| admin_state_up        | True
|
| allowed_address_pairs |
|
| binding:host_id       | iaas-vms.cloud.lab.eng.bos.redhat.com
|
| binding:profile       | {}
|
| binding:vif_details   | {"port_filter": true, "ovs_hybrid_plug": true}
|
| binding:vif_type      | ovs
|
| binding:vnic_type     | normal
|
| device_id             |
|
| device_owner          |
|
| extra_dhcp_opts       |
|
| fixed_ips             | {"subnet_id": "f5a4fbf8-1d86-4d64-ad9e-4d012a5fd1b7", "ip_address":
"172.16.5.9"} |
| id                    | db6cfb2f-ad04-402c-b5d9-216b8a716841
|
| mac_address           | fa:16:3e:f2:72:9b
|
| name                  | rally-port
|
| network_id            | 0fd1b597-7ed0-45cf-b9e2-a5dfbee80377
|
| security_groups       | 38807133-c370-47ae-9b04-16a882de1212
|
| status                | DOWN
|
| tenant_id             | 67b93212a2a34ccfb2014fdc34f4275e
|
+-----------------------
+-------------------------------------------------------------------------------+
```

8. Within the benchmarking server, modify the Rally VM XML with the following:

```
# virsh edit rally
...
    <interface type='bridge'>
      <mac address='fa:16:3e:f2:72:9b'/>
      <source bridge='br-int'/>
      <virtualport type='openvswitch'>
        <parameters interfaceid='neutron-port-id'/>
      </virtualport>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0f' function='0x0'/>
    </interface>
...
```

Ensure to update the value `neutron-port-id` with the `id` and parameter `mac address` with the value of `mac_address` located in the previous step when creating the `rally-port`.

9. Once the XML file changes have been applied to the Rally guest, shutdown the guest and start the guest back up.

Simply doing a reboot will not apply the changes.

10. Once all the above steps are completed, login to the Rally VM and run a sample scenario to test the environment.

```
# rally task start /path/to/rally/samples/tasks/scenarios/keystone/create-user.json
```

Benchmarking with Rally

The following section describes the different test case scenarios used to benchmark the existing RHEL-OSP environment. The results captured by these tests are analyzed in section Analyzing RHEL-OSP 7 Benchmark Results with Rally

Rally runs different types of scenarios based on the information provided by a user defined `.json` file. While Rally consists of many scenarios, this reference environment consists of showing the following scenarios that focus on end user usability of the RHEL-OSP cloud.

- Keystone.create-user (setup validation)
- Authenticate.validate_nova
- NovaServers.boot_and_list_server

The user defined `.json` files associated with these Rally scenarios are provided starting with the Appendix D: Rally Validate Nova JSON File

In order to properly create the user defined `.json` files, understanding how to assign parameter values is critical. The following example breaks down an existing `.json` file that runs the **NovaServers.boot_and_list_server** scenario.

```
{% set flavor_name = flavor_name or "m1.small" %}
{
    "NovaServers.boot_and_list_server": [
        {
            "args": {
                "flavor": {
                    "name": "{{flavor_name}}"
                },
                "nics": [{
                    "net-id": "0fd1b597-7ed0-45cf-b9e2-a5dfbee80377"
                }],
                "image": {
                    "name": "rhel-server7"
                },
                "detailed": true
            },
            "runner": {
                "concurrency": 1,
                "times": 1,
                "type": "constant"
            },
            "context": {
                "users": {
                    "tenants": 1,
                    "users_per_tenant": 1
                },
                "quotas": {
                    "neutron": {
                        "network": -1,
                        "port": -1
                    },
                    "nova": {
                        "instances": -1,
                        "cores": -1,
                        "ram": -1
                    }
                }
            }
        }
    ]
}
```

A `.json` file consists of the following:

- A curly bracket `{`, followed by the name of the Rally scenario, e.g. "NovaServers.boot_server", followed by a colon `:` and bracket `[`. The syntax is critical when creating a `.json` file otherwise the

Rally task fails. Each value assigned requires a comma `,` unless it is the final argument in a section.

- The next piece of the syntax are the arguments, `args`.
  - `args` consists of parameters that are assigned user defined values. The most notable parameters include:
    - `nics` - The UUID of the shared network to use in order to boot and delete instances.
    - `flavor` - The size of the guest instances to be created, e.g. `m1.small`.
    - `image` - The name of the image file used for creating guest instances.
    - `quotas` - Specification of quotas for the CPU cores, instances, and memory (RAM). Setting a value of `-1` for `cores`, `instances`, and `ram` allows for use of all the resources available within the RHEL-OSP 7 cloud.
    - `tenants` - amount of total tenants to be created.
    - `users_per_tenant` - amount of users to be created within each tenant.
    - `concurrency` - amount of guest instances to run on each iteration.
    - `times` - amount of iterations to perform.
- The closing syntax of a `json` file are the ending bracket `]` and curly bracket bracket `}`.

Once defining a `json` file is complete, the next step into properly benchmarking with Rally is to run a scenario. Using the provided `json` file, a user can ensure that their RHEL-OSP 7 environment can properly create guest instances.

```
# rally task start simple-boot-server.json
[ ... Output Abbreviated ... ]
+-------------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                     |
+------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| action           | min    | median | 90%ile | 95%ile | max    | avg    | success | count |
+------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| nova.boot_server | 53.182 | 53.182 | 53.182 | 53.182 | 53.182 | 53.182 | 100.0%  | 1     |
| total            | 53.182 | 53.182 | 53.182 | 53.182 | 53.182 | 53.182 | 100.0%  | 1     |
+------------------+--------+--------+--------+--------+--------+--------+---------+-------+
Load duration: 53.2088990211
Full duration: 78.8793258667

HINTS:
* To plot HTML graphics with this data, run:
        rally task report cd34b115-51e2-4e02-83ee-d5780ecdded4 --out output.html

* To generate a JUnit report, run:
        rally task report cd34b115-51e2-4e02-83ee-d5780ecdded4 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results cd34b115-51e2-4e02-83ee-d5780ecdded4
```

In Analyzing RHEL-OSP 7 Benchmark Results with Rally, the focal point is on running different Rally scenarios and analyzing those results.

For more information regarding Rally, please visit http://rally.readthedocs.org/en/latest/overview.html

Analyzing RHEL-OSP 7 Benchmark Results with Rally

With the basic fundamentals in creating a `.json` file as seen in the section Benchmarking with Rally, this section changes its focus to analyzing the different pre-defined benchmarking Rally scenarios using Rally's HTML Reporting to analyze the captured results. The Rally scenarios to be analyzed are:

- KeystoneBasic.create-user (setup validation)
- Authenticate.validate_nova
- NovaServers.boot_and_list_server

The Keystone.create-user rally scenario looks like the following:

```
{
    "KeystoneBasic.create_user": [
        {
            "args": {},
            "runner": {
                "type": "constant",
                "times": 100,
                "concurrency": 10
            }
        }
    ]
}
```

The KeystoneBasic.create_user scenario reads as attempting to create 10 users in parallel (concurrency) until it reaches the maximum value (times) of 100.

As with all scenarios that are discussed in this reference architecture, the key parameters `concurrency` and `times` control the amount of load to be placed on a RHEL-OSP environment. This is an important part of analysing the results as it allows for diagnosing a RHEL-OSP environment at different workload levels.

When creating a `.json` file, the value of `concurrency` and `times` are static values that dicatate the maximum number of users to launch for a specified scenario. To overcome this limitation, a script labeled `rally-wrapper.sh` that increments the maximum number of users to launch until the success rate is no longer satisified. The `rally-wrapper.sh` script increments the values of `concurrency` and `times` by a value of 10 as long as the success rate is met. The variable that controls the success rate within the `rally-wrapper.sh` script is labeled `EXPECTED_SUCCESS`.

The `rally-wrapper.sh` script is the perfect complement to all the rally scenarios discussed as the ability to increase the maximum number of users or guests (depending on the scenario) allows for diagnosis

of any errors as quickly as possible.

The contents of the `rally-wrapper.sh` script:

```
# cat rally-wrapper.sh
#
# This code will increment by 10
# @author Joe Talerico <jtaleric@redhat.com>
#
RALLY_JSON="ra-scaleboot-nonetworking.json"
EXPECTED_SUCCESS="100"
REPEAT=1
INCREMENT=10
TIMESTAMP=$(date +%s)
mkdir -p run-${REPEAT}

while [[ $REPEAT -gt 0 ]] ; do
 RUN=true
 while $RUN ; do
  CONCURRENCY=`cat ${RALLY_JSON} | grep concurrency | awk '{print $2}'`
  echo "Current number of guests launching : ${CONCURRENCY}"
  RALLY_RESULT=$(rally task start ${RALLY_JSON})
  TASK=$(echo "${RALLY_RESULT}" | grep Task | grep finished | awk '{print substr($2,0,length($2)-1)}')
  RUN_RESULT=$(echo "${RALLY_RESULT}" | grep total | awk '{print $16}')
  echo "     Task : ${TASK}"
  echo "     Result : ${RUN_RESULT}"
  rally task report ${TASK} --out run-${REPEAT}/${TASK}.html
  rally task results ${TASK} > run-${REPEAT}/${TASK}.json

  SUCCESS_RATE=$(echo "${RUN_RESULT}" | awk -F. '{ print $1 }')

  if [ "${SUCCESS_RATE}" -ge "${EXPECTED_SUCCESS}" ] ; then
   NEW_CON=$(echo "`cat ${RALLY_JSON} | grep concurrency | awk '{print $2}'`+${INCREMENT}" | bc)
   sed -i "s/\"times\"\:.*$/\"times\"\: ${NEW_CON},/g" ${RALLY_JSON}
   sed -i "s/\"concurrency\"\:.*$/\"concurrency\"\: ${NEW_CON}/g" ${RALLY_JSON}
  else
   RUN=false
   sed -i "s/\"times\"\:.*$/\"times\"\: 10,/g" ${RALLY_JSON}
   sed -i "s/\"concurrency\"\:.*$/\"concurrency\"\: 10/g" ${RALLY_JSON}
  fi
  sleep 60
 done
 let REPEAT-=1
done
```

The following rally scenario Authenticate.validate_nova attempts to create 3 tenants with 5 users in each then attempts to authenticate to nova 10 times performing 5 authentications per second.

```
{
    "Authenticate.validate_nova": [
        {
            "args": {
                "repetitions": 2
            },
            "runner": {
                "type": "constant",
                "times": 10,
                "concurrency": 5
            },
            "context": {
                "users": {
                    "tenants": 3,
                    "users_per_tenant": 5
                }
            }
        }
    ]
}
```

The Rally scenario NovaServers.boot_and_list_server scenario attempts to launch a `m1.small` guest with 1 vCPU, 2GB of RAM, and 20GB of storage disk (default values) and list all the booted guest instances. In order to determine the maximum amount of deployable guest instances within the OpenStack cloud, the theoretical limits of the RHEL-OSP 7 environment must be calculated. Within the reference environment, the theoretical limit of 383GB of Total RAM available to deploy guest instances constitutes to 187 guest instances available for deployment with the `m1.small` flavor. The theoretical value to determine maximum number of deployable guest instances is determined by calculating ((total ram - reserved memory)* memory overcommit) / RAM of each instance.

The value of 1 is used to represent the default memory overcommit ratio (1:1) within the RHEL-OSP 7 environment. The overcommit ratio, found within the */etc/nova/nova.conf* file, plays an important role in scalability and performance. As the value of overcommit is increased, scalability increases while performance decreases. Performance suffers due to more demand added to the OpenStack cloud for additional guests using a fixed amount of resources. However, as the value of overcommit is decreased, performance increases as less resources are shared across the different guests but scalability suffers due to the ratio 1:1 mapping to the same value of physical hardware.

Using the `nova hypervisor-stats` command, a user can learn a lot about the current environment. For example, this is the hypervisor stats of the existing reference environment configuration when only one compute node is available.

```
$ nova hypervisor-stats
+----------------------+-------+
| Property             | Value |
+----------------------+-------+
| count                | 1     |
| current_workload     | 0     |
| disk_available_least | 36949 |
| free_disk_gb         | 37001 |
| free_ram_mb          | 93912 |
| local_gb             | 37021 |
| local_gb_used        | 20    |
| memory_mb            | 96472 |
| memory_mb_used       | 2560  |
| running_vms          | 1     |
| vcpus                | 32    |
| vcpus_used           | 1     |
+----------------------+-------+
```

Taking a closer look at these stats, the key values to further investigate include: count, free_ram_mb, memory_mb, memory_mb_used, and vcpus.

The breakdown of these is as follows:

- count - represents the number of compute nodes available
- free_ram_mb - amount of RAM available to launch instances shown in megabytes
- memory_mb - total amount of RAM in megabytes
- memory_mb_used - total memory used in the environment (includes reserved_memory and memory of existing instances)
- vpus - total available virtual CPUs.

With these statistics, to calculate the amount of guest instances that can be launched take free_ram_mb and divide by the amount of memory consumed by the particular flavor that is to be deployed. In this reference environment, the `m1.small` flavor consumes 2GB per instance. This consistutes to 93912 MB / 2048 MB = 45 guest instances (rounded down).

The running_vms value of 1 is due to the creation of an already existing VM.

Once the theoretical maximum amount of deployable guest instances within an existing RHEL-OSP 7 environment is identified, the next step is to create a `.json` file that attempts to reach the theoretical upper boundaries calculated. The `.json` file to capture the maximum number of guests that can be launched by a RHEL-OSP 7 environment is as follows.

```
{% set flavor_name = flavor_name or "m1.small" %}
{
    "NovaServers.boot_and_list_server": [
        {
            "args": {
                "flavor": {
                    "name": "{{flavor_name}}"
                },
                "nics": [{
                    "net-id": "0fd1b597-7ed0-45cf-b9e2-a5dfbee80377"
                }],
                "image": {
                    "name": "rhel-server7"
                },
                "detailed": true
            },
            "runner": {
                "concurrency": 1,
                "times": 1,
                "type": "constant"
            },
            "context": {
                "users": {
                    "tenants": 1,
                    "users_per_tenant": 1
                },
                "quotas": {
                    "neutron": {
                        "network": -1,
                        "port": -1
                    },
                    "nova": {
                        "instances": -1,
                        "cores": -1,
                        "ram": -1
                    }
                }
            }
        }
    ]
}
```

The initial objective is use the `rally-wrapper.sh` script while using small `times` and `concurrency` values that increment slowly in order to diagnose any errors as quickly as possible.

Initial boot-storm Rally Results

The boot-storm tests in rally attempt to launch as many guests as the RHEL-OSP environment can handle simultaneously. The initial tests consist of boot-storm tests with 1, 2, 3 and 4 compute nodes.

When attempting to launch as many guests as the RHEL-OSP environment can handle simultaneously, one must first calculate the amount of free RAM available to launch instances. As shown in Analyzing RHEL-OSP 7 Benchmark Results with Rally, the `nova hypervisor-stats` is a critical component to this success. With 1 compute node, this existing reference environment is able to launch a maximum of 45 guests (92376 MB / 2048 MB) if each guest takes 2 GB of RAM. With this information, one can then use the `rally-wrapper.sh` script and set the `concurrency` value to 45, and the `times` value to 45. This strategy then attempts to launch as many guests as the environment can handle simultaneously.

A snippet of the initial results for one compute node are shown below.

Listing 1. 1-compute-node-45times-45concurrency-vif-plugging-timeout-30

```
+-------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                         |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action            | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server  | 68.79  | 183.686 | 279.841 | 296.122 | 304.14  | 178.048 | 100.0%  | 45    |
| nova.list_servers | 0.849  | 0.993   | 1.246   | 1.327   | 1.402   | 1.043   | 100.0%  | 45    |
| total             | 69.933 | 184.674 | 280.906 | 297.124 | 305.223 | 179.092 | 100.0%  | 45    |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 305.434197903
Full duration: 445.057275057

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 80493bb8-8229-4ec2-ba7b-bdca0954dc73 --out output.html

* To generate a JUnit report, run:
        rally task report 80493bb8-8229-4ec2-ba7b-bdca0954dc73 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 80493bb8-8229-4ec2-ba7b-bdca0954dc73
```

Taking a closer look at the initial results, results show launching all 45 guests simultaneously achieves a success rate of 100%. While this is good news, the actual response times to boot those instances are quite high. The minimum time to boot was listed at 68.79 seconds while the average time was 178.048 seconds.

The first question one might ask is *Why are the boot time values so high?* The answer lies within the */etc/nova/nova.conf* file of the compute node. The parameter `vif_plugging_timeout` determines the amount of time the nova service waits for neutron to report back that the port setup process is

complete prior to nova continuing with the booting of an instance. The parameter `vif_plugging_is_fatal` determines what nova should do with that instance upon it exceeding the assigned timeout value.

RHEL-OSP 7 ships with `vif_plugging_is_fatal` set to False and `vif_plugging_timeout` set to 30, which makes nova wait for 30 seconds. To avoid the unncessary waiting, it is recommended to keep `vif_plugging_is_fatal` set to false and `vif_plugging_timeout` set to zero to avoid the wait.

With the modification of the vif parameters, rerunning the initial bootstorm test on 1 compute node resulted in the following:

Listing 2. 1-compute-node-45times-45concurrency-vif-plugging-timeout-0

```
+------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                        |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action            | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server  | 33.18  | 111.991 | 163.766 | 178.831 | 180.093 | 106.784 | 100.0%  | 45    |
| nova.list_servers | 0.843  | 1.039   | 1.262   | 1.296   | 1.41    | 1.065   | 100.0%  | 45    |
| total             | 34.023 | 112.924 | 164.718 | 179.76  | 181.259 | 107.849 | 100.0%  | 45    |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 181.42532897
Full duration: 323.352867126

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 23529f43-798d-44f1-9c7a-7488bfe86585 --out output.html

* To generate a JUnit report, run:
        rally task report 23529f43-798d-44f1-9c7a-7488bfe86585 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 23529f43-798d-44f1-9c7a-7488bfe86585
```

The changing of these values drastically improves the response times. Referencing just the minimum and average values, the minimum boot time is decreased 51.7% percent and the average boot time is decreased by 40% percent.

With the understanding of how `vif_plugging_is_fatal` and `vif_plugging_timeout` effect the bootstorm performance, the next step is to continue on with the bootstorm testing for 2, 3 and 4 compute nodes.

When adding an additional compute node to launch instances, the environment has 186800 MB of RAM available verified by `nova hypervisor-stats`. Each guest instance takes up 2048 MB of RAM, thus allows for the max concurrency launch of 91 instances (186800 MB / 2048 MB). Below is the response time results of that run.

Listing 3. 2-compute-node-91times-91concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                         |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action            | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server  | 43.502 | 171.631 | 295.4   | 309.489 | 314.713 | 168.553 | 100.0%  | 91    |
| nova.list_servers | 1.355  | 1.961   | 2.522   | 2.565   | 3.024   | 2.031   | 94.5%   | 91    |
| total             | 45.928 | 167.625 | 274.446 | 291.547 | 307.154 | 162.119 | 94.5%   | 91    |
+-------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 315.15325284
Full duration: 499.308477163

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 04d9e8aa-da94-4724-a904-37abca471543 --out output.html

* To generate a JUnit report, run:
        rally task report 04d9e8aa-da94-4724-a904-37abca471543 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 04d9e8aa-da94-4724-a904-37abca471543
```

As a compute node is added, the RHEL-OSP environment is able to achieve a 100% success rate when booting 91 guest instances concurrently. As additional guest instances are launched simultaneously, the minimum boot time and average boot time increases. The minimum boot time increased by 23.7% when doubling the amount of compute nodes and instances launched, while the average boot time inreased by 36.6%.

When taking a closer look at the Rally results, one piece of information stands out. *Why is nova able to boot with a 100% success rate, but does not achieve 100% success rate when listing the servers?* The reason for this is due to a bug found within Rally: https://bugs.launchpad.net/rally/+bug/1510175, where it is possible to see discrepancies between the Rally HTML report and the results found within the Rally log file. To confirm if the success rate drop is related to the `nova.list_servers` and `nova.boot_servers` refer to the HTML reports Failures tab.

In order to launch the Rally HTML report, run via command line `rally task report <id_of_report> --out <name>.html`. An example from the above listing shows the following command:

```
rally task report 04d9e8aa-da94-4724-a904-37abca471543 --out output.html
```

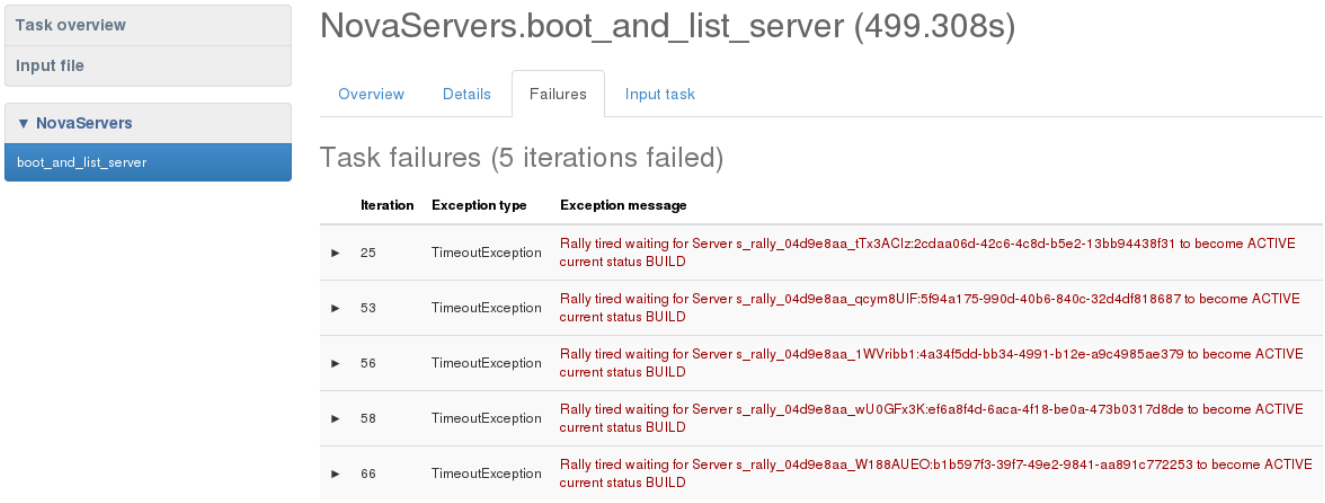

Figure 4.1. Rally Task Failures with 2 Compute Nodes with 91 Concurrency and Times

Within the Failures tab, 5 tasks fail all with timeout exceptions. Each exception fails with Rally unable to change the status of a booting instance from its BUILD state to its ACTIVE state. When a guest instance is first launched, it is placed in a BUILD state. A boot failure occurs when a guest instance is unable to achieve an ACTIVE state. When reading the above error, it explicitly says that it timed out waiting for the guest instance to change states. Due to this, it is clear that the failures are not caused by `nova.list_servers` but instead by nova unable to successfully boot up all the instances to an ACTIVE state.

With knowing what the failures are, the next question to answer is *Why are these instances failing?* To answer this question, one must turn to the Ceph nodes. Within the Ceph nodes verify whether there is any unusually high amount of I/O wait. By capturing the results of `top` while an existing Rally task is running, the CPU wait times can be further reviewed. The image below displays the `top` event when attempting to launch 91 guest instances concurrently with 2 compute nodes.

```
top - 15:28:25 up 62 days, 21:58,  1 user,  load average: 5.39, 2.00, 1.02
Tasks: 393 total,   1 running, 392 sleeping,   0 stopped,   0 zombie
%Cpu(s): 10.8 us,  5.0 sy,  0.0 ni, 68.5 id, 15.0 wa,  0.0 hi,  0.6 si,  0.0 st
KiB Mem : 49281020 total, 24581764 free,  4795000 used, 19904256 buff/cache
KiB Swap:        0 total,        0 free,        0 used. 43921080 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
22770 root      20   0 1865060 579952  14200 S  48.0  1.2   1297:29 ceph-osd
24114 root      20   0 1724576 552188  13672 S  48.0  1.1   1374:30 ceph-osd
20232 root      20   0 1612940 605108  13452 S  46.4  1.2   1272:53 ceph-osd
28293 root      20   0 1401960 399440  13416 S  42.7  0.8   1270:05 ceph-osd
18426 root      20   0 1440820 419068  13880 S  40.7  0.9   1323:25 ceph-osd
30310 root      20   0 1370728 381568  13740 S  40.4  0.8   1227:09 ceph-osd
21353 root      20   0 1309944 302912  13460 S  38.4  0.6   1157:25 ceph-osd
25847 root      20   0 1333224 327920  13364 S  37.1  0.7   1264:57 ceph-osd
31784 root      20   0 1290596 248652  13736 S  31.5  0.5   1066:18 ceph-osd
19247 root      20   0 1249268 246552  13828 S  20.9  0.5   1113:00 ceph-osd
31379 root      20   0       0      0      0 S   2.6  0.0   0:09.70 kworker/21:2
```

Figure 4.2. High CPU Wait during 2 Compute Node Run with 91 Concurrency and 91 Times

As depicted in the image above, the Ceph nodes cannot keep up with the amount of I/O requests coming into the environment as a large amount of spawning guest instance requests come in from the Rally task. Due to this, failures such as instances not achieving an ACTIVE state from their current

BUILD state occurs. Another side effect of waiting for I/O to complete, is the higher boot times as compute nodes are added to support additional guest instances. In order to alleviate this bottleneck, changing the Ceph environment to use cached journals or including more Ceph nodes are potential solutions to alleviate the bottleneck. However, hardware restrictions in the System Engineering lab, prevented us from making these changes.

Continuing with findings, the figures below show as increasing the guest instances concurrently causes higher boot times and a lower success rate due to the I/O wait bottleneck by the Ceph nodes.

Listing 4. 3-compute-node-137times-137concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                         |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action           | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server | 61.482 | 262.091 | 324.636 | 324.899 | 326.21  | 233.9   | 100.0%  | 137   |
| nova.list_servers | 1.9    | 3.086   | 3.86    | 3.903   | 4.259   | 3.086   | 71.5%   | 137   |
| total            | 64.951 | 214.873 | 308.903 | 313.681 | 320.596 | 201.339 | 71.5%   | 137   |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 327.246902943
Full duration: 641.576355934

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 5580c462-fe7c-4919-b549-776b2a222ec5 --out output.html

* To generate a JUnit report, run:
        rally task report 5580c462-fe7c-4919-b549-776b2a222ec5 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 5580c462-fe7c-4919-b549-776b2a222ec5
```

When running the same test with 3 compute nodes, the RHEL-OSP environment is able to achieve a 71.5% success rate when booting 137 guest instances concurrently. As additional guest instances are launched simultaneously, the minimum boot time increases by 29.2%, while the average boot time increases by 27.9% from the previous test of 2 compute node with 91 guest instances launched.

71.5% success rate is for the nova.boot_server not the nova.list_server due to BZ: https://bugs.launchpad.net/rally/+bug/1510175

Listing 5. 4-compute-node-183times-183concurrency-vif-plugging-timeout-0

```
+-----------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                       |
+--------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action             | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+--------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server   | 86.67  | 328.305 | 337.028 | 338.162 | 339.256 | 272.641 | 100.0%  | 183   |
| nova.list_servers | 3.337  | 4.649   | 5.227   | 5.506   | 6.123   | 4.659   | 47.5%   | 183   |
| total              | 91.243 | 208.544 | 291.381 | 326.766 | 334.988 | 209.405 | 47.5%   | 183   |
+--------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 339.872634888
Full duration: 1070.80222201

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 3812352f-6c0f-4b37-8544-cd40b51f21ef --out output.html

* To generate a JUnit report, run:
        rally task report 3812352f-6c0f-4b37-8544-cd40b51f21ef --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 3812352f-6c0f-4b37-8544-cd40b51f21ef
```

When running the same test with 4 compute nodes, the RHEL-OSP environment is able to achieve a 47.5% success rate when booting 183 guest instances concurrently. As additional guest instances are launched simultaneously, the minimum boot time increases from by 29.2%, while the average boot time increases by 27.9% from the previous test of 3 compute nodes with 137 guest instances launched.

47.5% success rate is for the nova.boot_server not the nova.list_server due to BZ: https://bugs.launchpad.net/rally/+bug/1510175

Rally Max Guest Launch

While the boot-storm tests attempt to launch as many guests as the RHEL-OSP environment can handle simultaneously, the max guest test tries to see whether the reference environment can achieve the theoretical max of `m1.small` guests. The initial calculation is to take free_ram_mb / amount of RAM of flavor to find max guests that can be launched using the `nova hypervisor-stats` command. The `rally-wrapper.sh` script does the max guest launch test by concurrently launching 8 guest instances until reaching the maximum amount of times for 1, 2, 3 and 4 compute nodes.

Listing 6. nova.boot_and_list_server_one_compute

```
{
    "NovaServers.boot_and_list_server": [
        {
            "args": {
                "flavor": {
                    "name": "m1.small"
                },
                "nics": [{
                    "net-id": "0fd1b597-7ed0-45cf-b9e2-a5dfbee80377"
                }],
                "image": {
                    "name": "rhel-server7"
                },
                "detailed": true
            },
            "runner": {
                "concurrency": 8,
                "times": 45,
                "type": "constant"
            },
            "context": {
                "users": {
                    "tenants": 1,
                    "users_per_tenant": 1
                },
                "quotas": {
                    "neutron": {
                        "network": -1,
                        "port": -1
                    },
                    "nova": {
                        "instances": -1,
                        "cores": -1,
                        "ram": -1
                    }
                }
            }
        }
    ]
}
```

Listing 7. 1-compute-node-45times-8concurrency-vif-plugging-timeout-0

```
+-----------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                   |
+--------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| action             | min    | median | 90%ile | 95%ile | max    | avg    | success | count |
+--------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| nova.boot_server   | 27.035 | 34.348 | 39.161 | 42.1   | 42.87  | 34.042 | 100.0%  | 45    |
| nova.list_servers  | 0.432  | 0.852  | 1.142  | 1.158  | 1.312  | 0.824  | 100.0%  | 45    |
| total              | 28.194 | 35.25  | 39.694 | 42.576 | 43.348 | 34.866 | 100.0%  | 45    |
+--------------------+--------+--------+--------+--------+--------+--------+---------+-------+
Load duration: 210.317100048
Full duration: 347.372730017
```

The results above show that a success rate 100% for the max amount of guest instances of 45 when running with 1 compute node. The value of 45 guest instances is calculated using `nova hypervisor-stats` to correctly calculate the supported guest instance value.

When running with 2 compute nodes, 91 guest instances are supported. Below is the `.json` file used to achieve the 91 max guest instances.

Listing 8. nova.boot_and_list_server_two_compute

```
{
    "NovaServers.boot_and_list_server": [
        {
            "args": {
                "flavor": {
                    "name": "m1.small"
                },
                "nics": [{
                    "net-id": "0fd1b597-7ed0-45cf-b9e2-a5dfbee80377"
                }],
                "image": {
                    "name": "rhel-server7"
                },
                "detailed": true
            },
            "runner": {
                "concurrency": 8,
                "times": 91,
                "type": "constant"
            },
            "context": {
                "users": {
                    "tenants": 1,
                    "users_per_tenant": 1
                },
                "quotas": {
                    "neutron": {
                        "network": -1,
                        "port": -1
                    },
                    "nova": {
                        "instances": -1,
                        "cores": -1,
                        "ram": -1
                    }
                }
            }
        }
    ]
}
```

Listing 9. 2-compute-node-91times-8concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                     |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
| action            | min    | median | 90%ile | 95%ile  | max     | avg    | success | count |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
| nova.boot_server  | 24.637 | 32.123 | 45.629 | 117.858 | 152.226 | 41.814 | 100.0%  | 91    |
| nova.list_servers | 0.412  | 1.23   | 1.741  | 1.84    | 2.632   | 1.2    | 100.0%  | 91    |
| total             | 26.317 | 33.458 | 46.451 | 118.356 | 152.743 | 43.015 | 100.0%  | 91    |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
Load duration: 500.732126951
Full duration: 652.159901142
```

The results above show that a success rate 100% for the max amount of guest instances of 91 when running with 2 compute node. The value of 91 guest instances is calculated using `nova hypervisor-stats` to correctly calculate the supported guest instance value. Further investigating the results, it is easy to see that not only is the 100% success rate achieved, but the boot times are significantly lower than the boot-storm results as launching a lower amount of guest instances simultaneously decreases the amount of I/O stress put on the Ceph nodes.

The same applies with the results of 3 compute nodes and 4 compute nodes. Both are able to achieve 100% success rate in achieving the max guests with a lower boot time for each guest instance due to launching a much smaller amount of guest instances simultaneously.

Listing 10. 3-compute-node-137times-8concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                     |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
| action            | min    | median | 90%ile | 95%ile | max     | avg    | success | count |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
| nova.boot_server  | 26.235 | 31.75  | 43.451 | 48.708 | 132.845 | 36.095 | 100.0%  | 137   |
| nova.list_servers | 0.391  | 1.598  | 2.481  | 2.664  | 2.826   | 1.56   | 100.0%  | 137   |
| total             | 27.269 | 33.148 | 44.328 | 49.962 | 133.556 | 37.655 | 100.0%  | 137   |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
Load duration: 659.672354221
Full duration: 824.834990025
```

Listing 11. 4-compute-node-183times-8concurrency-vif-plugging-timeout-0

```
+-----------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                   |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
| action            | min    | median | 90%ile | 95%ile | max     | avg    | success | count |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
| nova.boot_server  | 25.672 | 32.315 | 46.114 | 49.741 | 145.58  | 35.707 | 100.0%  | 183   |
| nova.list_servers | 0.388  | 1.884  | 3.082  | 3.236  | 5.096   | 1.933  | 100.0%  | 183   |
| total             | 27.055 | 34.53  | 47.828 | 51.143 | 146.348 | 37.642 | 100.0%  | 183   |
+-------------------+--------+--------+--------+--------+---------+--------+---------+-------+
Load duration: 870.662543058
Full duration: 1055.67080402
```

Tuning the RHEL-OSP 7 Environment using Browbeat

Once the initial baseline benchmarking results using Rally are captured, the focus shifts to tuning the RHEL-OSP 7 environment. Because of the complexities involved in tuning a RHEL-OSP environment, the Red Hat Performance team has created an opensource project named Browbeat. Browbeat is set a scripts and Ansible playbooks to help determine different performance characteristics of RHEL-OSP. One of the key features of Browbeat is its Ansible playbooks that perform overcloud checks across the environment. This reference environment uses the overcloud checks to determine if the different OpenStack services are running optimally.

Browbeat is **not** supported and is provided as helper tool to identify common performance issues.

Installation of Browbeat

Within the `rally` VM, as the `stack` user,

1. If not already installed, install `git` via the `yum` command:

```
$ yum install git
```

2. Install the `ansible` package via `yum`

```
$ yum install ansible
```

3. Change into a working directory to `git clone` the browbeat repository

```
$ mkdir /path/to/mybrowbeat
$ cd /path/to/mybrowbeat
$ git clone https://github.com/jtaleric/browbeat.git
```

4. Install the public key into the `stack` users *authorized_keys* file

```
$ ssh-copy-id stack@<undercloud-ip>
```

5. Run the script labeled *gen_hosts.sh* to generate an overcloud *hosts* file for Ansible and generate a *jumpbox* configuration file. This example calls the host file 'env_machines' and can reside anywhere within the Rally VM.

```
$ ./gen_hostfile.sh <undercloud-ip> /path/to/env_machines
```

6. Export the *env_machines* file as an Ansible `ssh` argument:

```
$ export ANSIBLE_SSH_ARGS='-F ./path/to/env_machines'
```

7. Run the check playbook to identify common performance issues:

```
$ ansible-playbook -i hosts check/site.yml
```

Analyzing Performance Issues generated by the Ansible Playbook

Browbeat's check playbook verifies if certain settings are properly set within the RHEL-OSP 7 environment. It is important to note that the following verifications will continue to increase as the project develops. However, at the time of this writing the current verifications are specified below.

Browbeat's verification checks are separated by `roles`. The `roles` are broken up as follows:

- `common` - verification checks that are common to RHEL-OSP 7 Director, Controller, Compute, and Ceph nodes.
- `controller` - verification checks that are specific to the RHEL-OSP 7 Controller nodes
- `compute` - verification checks that are specific to the RHEL-OSP 7 Compute nodes
- `ceph` - verification checks that are specific to the RHEL-OSP 7 Ceph nodes
- `keystone` - verification checks that are specific to the RHEL-OSP 7 Keystone service

A further breakdown of the verification checks for each `role` consists of the following:

Browbeat's check playbook for RHEL-OSP 7 common nodes:

- Verify SELinux status for each host
- Verify if the `tuned` daemon is running on each host
- Verify the current running `tuned` profile for each host

Browbeat's check playbook for RHEL-OSP 7 Keystone service:

- Verify if a `cron` job exists that will remove expired tokens from the MariaDB database as an accumlation of tokens can cause performance degration.
- Check the `keystone` token provider, which by default is UUID.

Browbeat's check playbook for RHEL-OSP 7 controller nodes:

- Verify the maximum allowed connections for the MariaDB database is sufficient.
- Verify the amount of file descriptors assigned for RabbitMQ is set to 16384.
- Verify `nova` and `neutron` are communicating about VIF Plugging
- Verify HAProxy default max connections
- Verify netns tuning
- Verify if RabbitMQ contains partitions

The current verification checks for RHEL-OSP 7 compute nodes and Ceph nodes are currently limited. However, with the rapid development of this project, this should change in the near future.

An example output of the overcloud checks from Browbeat.

```
# Browbeat generated bug report
----------------------------------------
| Issues for host : overcloud-controller-0
----------------------------------------
Bug: bz1095811
Name: Network connectivity issues after 1000 netns
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1095811

Bug: nova_vif_timeout_result
Name: Nova VIF timeout should be >= 300
URL: none

Bug: bz1264740
Name: RHEL OSP Director must be configure with nova-event-callback by default
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1264740

----------------------------------------
| Issues for host : overcloud-controller-1
----------------------------------------
Bug: bz1095811
Name: Network connectivity issues after 1000 netns
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1095811

Bug: nova_vif_timeout_result
Name: Nova VIF timeout should be >= 300
URL: none

Bug: bz1264740
Name: RHEL OSP Director must be configure with nova-event-callback by default
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1264740
```

```
---------------------------------------
| Issues for host : overcloud-controller-2
---------------------------------------
Bug: bz1095811
Name: Network connectivity issues after 1000 netns
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1095811

Bug: nova_vif_timeout_result
Name: Nova VIF timeout should be >= 300
URL: none

Bug: bz1264740
Name: RHEL OSP Director must be configure with nova-event-callback by default
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1264740



---------------------------------------
| Issues for host : overcloud-novacompute-0
---------------------------------------
Bug: bz1282644
Name: increase reserved_host_memory_mb
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1282644

Bug: bz1264740
Name: RHEL OSP Director must be configure with nova-event-callback by default
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1264740

Bug: tuned_profile_result
Name: Ensure Tuned Profile is set to virtual-host
URL: none

Bug: bz1245714
Name: No Swap Space allocated
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1245714

Bug: nova_vif_timeout_result
Name: Nova VIF timeout should be >= 300
URL: none

---------------------------------------
| Issues for host : overcloud-novacompute-1
---------------------------------------
Bug: bz1282644
Name: increase reserved_host_memory_mb
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1282644

Bug: bz1264740
Name: RHEL OSP Director must be configure with nova-event-callback by default
```

```
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1264740

Bug: tuned_profile_result
Name: Ensure Tuned Profile is set to virtual-host
URL: none

Bug: bz1245714
Name: No Swap Space allocated
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1245714

Bug: nova_vif_timeout_result
Name: Nova VIF timeout should be >= 300
URL: none
```

With the output generated by Browbeat, users can tweak their existing RHEL-OSP environment to fix issues that might ultimately provide better performance and scalability. For demonstration purposes, this reference environment takes a closer look at the following bugs reported by Browbeat:

```
----------------------------------------
| Issues for host : overcloud-novacompute-1
----------------------------------------
Bug: bz1282644
Name: increase reserved_host_memory_mb
URL: https://bugzilla.redhat.com/show_bug.cgi?id=1282644

AND

Bug: tuned_profile_result
Name: Ensure Tuned Profile is set to virtual-host
URL: none
```

The first bug BZ11282644, references the parameter `reserved_host_memory_mb` that is found within the compute node's */etc/nova/nova.conf* file. The parameter `reserved_host_memory_mb` ensures that X amount of megabytes are reserved for the hypervisor and cannot be used as additional RAM for launching guest instances. The default value of `reserved_host_memory_mb` is currently 512 MB of RAM. However, as guest instances are launched and the hypervisor reaches its 512 MB threshold, this leads to a potential out of memory (OOM) situation were guest instances and/or processes can be killed within the RHEL-OSP environment as 512 MB of RAM for the hypervisor is not enough memory run the environment. In order to alleviate potential scenarios with OOM, it is recommended to provide each compute node within the RHEL-OSP 7 environment at least 2048 MB of reserved host memory.

Besides increasing the reserved host memory for the hypervisor, *how does the* `reserved_host_memory_mb` *parameter effect performance and/or scalability?*

By increasing the amount of RAM the hypervisor is reserving, it limits the amount of launchable guest

instances. This can be clearly seen when running the `nova hypervisor-stats` command.

```
$ nova hypervisor-stats
+----------------------+-------+
| Property             | Value |
+----------------------+-------+
| count                | 1     |
| current_workload     | 0     |
| disk_available_least | 36949 |
| free_disk_gb         | 37001 |
| free_ram_mb          | 93912 |
| local_gb             | 37021 |
| local_gb_used        | 20    |
| memory_mb            | 96472 |
| memory_mb_used       | 2560  |
| running_vms          | 1     |
| vcpus                | 32    |
| vcpus_used           | 1     |
+----------------------+-------+
```

In the example above, memory_mb_used is 2560 MB. With one virtual machine currently running taking 2048MB, this leaves exactly 512 MB of RAM being used which is reserved by the hypervisor which totals 2560. To calculate amount of launchable guests: ((total_ram_mb - reserved_host_memory_mb) * memory ovecommit) / RAM of each instance

In this scenario, ((96472 - 512) * 1)/ 2048 = 46 guests

Default memory overcommit is 1:1 within RHEL-OSP 7

Once the reserved memory for the hypervisor is increased from 2048 mb from 512 mb, `nova hypervisor-stats` reveals:

```
$ nova hypervisor-stats
+----------------------+-------+
| Property             | Value |
+----------------------+-------+
| count                | 1     |
| current_workload     | 0     |
| disk_available_least | 36949 |
| free_disk_gb         | 37001 |
| free_ram_mb          | 92376 |
| local_gb             | 37021 |
| local_gb_used        | 20    |
| memory_mb            | 96472 |
| memory_mb_used       | 4096  |
| running_vms          | 1     |
| vcpus                | 32    |
| vcpus_used           | 1     |
+----------------------+-------+
```

In this scenario, total memory used is 4096 which includes the 2048 reserved for the hypervisor and the one virtual machine that is currently taking up 2048 for a total of 4096 mb. To calculate the amount of launchable guests using the same formula from above: ((96472 - 2048) * 1)/ 2048 = 45 guests

While the impact in this scenario is small (losing 1 virtual machine when using only 1 compute node), as compute nodes are added and different flavors for guest instances are used, it can have a larger overall impact of instances that can be launched.

The last issue that is to be addressed in this example is the tuned_profile_result. The `tuned` package in Red Hat Enterprise Linux 7 is recommended for automatically tuning the system for common workloads via the use of profiles. Each profile is tailored for different workload scenarios such as: throughput-performance, virtual-host, and virtual-guest.

In the chapter Analyzing RHEL-OSP 7 Benchmark Results with Rally, we discuss how the RHEL-OSP environment performed and scaled with and without the default settings prior to tuning with Browbeat. To summarize, when doing the boot-storm tests, as the concurrency value increased closer to the maximum amount of guest instances that the environment could launch it caused high CPU wait times due to the Ceph nodes not being able to keep up with the amount of guest instances being booted simulatenously. When looking at achieving max guests with a low concurrency, the RHEL-OSP environment is able to achieve reaching the max guests, however, at times the average and max boot times were higher than expected.

When running the tests within the Analyzing RHEL-OSP 7 Benchmark Results with Rally, all the Controllers, Compute, and Ceph nodes ran with the tuned profile throughput-performance. Below is the example results when running the 2 compute nodes with 8 concurrency 91 times.

Listing 12. 2-compute-node-91times-8concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                     |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
| action            | min    | median | 90%ile | 95%ile  | max     | avg    | success | count |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
| nova.boot_server  | 24.637 | 32.123 | 45.629 | 117.858 | 152.226 | 41.814 | 100.0%  | 91    |
| nova.list_servers | 0.412  | 1.23   | 1.741  | 1.84    | 2.632   | 1.2    | 100.0%  | 91    |
| total             | 26.317 | 33.458 | 46.451 | 118.356 | 152.743 | 43.015 | 100.0%  | 91    |
+-------------------+--------+--------+--------+---------+---------+--------+---------+-------+
Load duration: 500.732126951
Full duration: 652.159901142
```

While a very high success rate is achieved, the question lies *Does the performance and scalability get impacted if we implement Browbeat's recommendation?* While Browbeat's bug report only mentions changing the tuned profile for compute nodes, further research found within this Red Hat Performance Brief: http://bit.ly/20vAp0c shows that the following tuned recommendations can potentially show an increase in performance. They include:

- Controller nodes - throughtput-performance profile
- Compute nodes - virtual-host profile
- Ceph nodes - virtual-host profile

When re-running the max guest launch test with the above changes on each node keeping the 8 concurrency, 91 times, and using 2 compute nodes the results are:

Listing 13. 2-compute-node-91times-8concurrency-vif-plugging-timeout-0-tuned-profile-changes

```
+-------------------------------------------------------------------------------------------+
|                                  Response Times (sec)                                     |
+-------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| action            | min    | median | 90%ile | 95%ile | max    | avg    | success | count |
+-------------------+--------+--------+--------+--------+--------+--------+---------+-------+
| nova.boot_server  | 24.791 | 31.464 | 34.104 | 35.382 | 36.264 | 31.352 | 100.0%  | 91    |
| nova.list_servers | 0.38   | 1.223  | 1.867  | 1.925  | 2.355  | 1.227  | 100.0%  | 91    |
| total             | 27.147 | 32.774 | 34.858 | 36.306 | 37.318 | 32.579 | 100.0%  | 91    |
+-------------------+--------+--------+--------+--------+--------+--------+---------+-------+
Load duration: 388.272264957
Full duration: 504.665802002
```

Referencing just the minimum and average values, the minimum boot time increased less than 1% percent but the key is average boot time decreasing by 25% percent!

The original boot-storm test with the same two compute node configuration with 91 concurrency and 91 times reported:

Listing 14. 2-compute-node-91times-91concurrency-vif-plugging-timeout-0

```
+-------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                         |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action           | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server | 43.502 | 171.631 | 295.4   | 309.489 | 314.713 | 168.553 | 100.0%  | 91    |
| nova.list_servers | 1.355  | 1.961   | 2.522   | 2.565   | 3.024   | 2.031   | 94.5%   | 91    |
| total            | 45.928 | 167.625 | 274.446 | 291.547 | 307.154 | 162.119 | 94.5%   | 91    |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 315.15325284
Full duration: 499.308477163

HINTS:
* To plot HTML graphics with this data, run:
        rally task report 04d9e8aa-da94-4724-a904-37abca471543 --out output.html

* To generate a JUnit report, run:
        rally task report 04d9e8aa-da94-4724-a904-37abca471543 --junit --out output.xml

* To get raw JSON output of task results, run:
        rally task results 04d9e8aa-da94-4724-a904-37abca471543
```

When re-running the same environment but with the `tuned` profile changes across all the nodes, the results show:

Listing 15. 2-compute-node-91times-91concurrency-vif-plugging-timeout-0-tuned-profile-changes

```
+-------------------------------------------------------------------------------------------------+
|                                    Response Times (sec)                                         |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| action           | min    | median  | 90%ile  | 95%ile  | max     | avg     | success | count |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
| nova.boot_server | 43.39  | 152.542 | 259.255 | 270.847 | 275.094 | 166.24  | 100.0%  | 91    |
| nova.list_servers | 1.525  | 1.967   | 2.436   | 2.582   | 3.285   | 2.025   | 100.0%  | 91    |
| total            | 45.294 | 154.223 | 261.232 | 272.594 | 276.72  | 168.266 | 100.0%  | 91    |
+------------------+--------+---------+---------+---------+---------+---------+---------+-------+
Load duration: 276.911462784
Full duration: 398.153054953
```

Referencing just the minimum and average values, the minimum boot time decreased less than 1% percent but the key is average boot time decreasing by 13.7% percent. While the decrease in the average boot time isn't as drastic as the previous example with a lower concurrency rate, this is due to the Ceph storage unable to keep up with the workload. However, it is important to mention that even

with the Ceph storage being the bottleneck, an increase in performance was obtained changing the `tuned` profiles. As these results show, implementing the different recommendations for tuning using Browbeat can have a great impact in overall performance and scalability.

Conclusion

Red Hat solutions involving Red Hat Enterprise Linux OpenStack Platform are created to deliver a production-ready foundation that simplifies the deployment process, shares the latest best practices, and gets the utmost performance and scale for a public or private OpenStack cloud. The steps and procedures covered within this reference architecture provide system, storage, and cloud administrators the blueprint required to using the different open source benchmarking tools to assist in understanding of the performance and scale potential of an existing cloud environment.

Successfully benchmarking and scaling an existing RHEL-OSP environment consists of the following:

- Capturing baseline results using the benchmarking tool Rally
- Running performance checks on the RHEL-OSP environment using Browbeat
- Modify the RHEL-OSP environment based upon the performance check findings from Browbeat
- Re-run the Rally scenarios to capture the latest results with the tuning changes.
- Analyzing and comparing the results

The Rally test results for the reference environment demonstrated that `vif_plugging_timeout` and `vif_plugging_is_fatal` parameter values play a critical role in nova guest instance boot times. When the `vif_plugging_timeout` value is decreased, the RHEL-OSP environment saw a great decrease in guest instance boot times by as much as 51.7%. As performance and scalability were measured when running the different scenarios, the amount of RAM, as well as, the available spindles within the Ceph nodes limited the performance and scalability of the environment. With regards to performance, as the Rally scenarios increased the concurrency value, this lead to high CPU wait times due to the Ceph nodes not being able to keep up with the amount of guest instances being booted simulatenously leading at times to nova guest instance boot failures. To increase performance, besides adding additional spindles, adding higher speed drives (SSD, NVMe) for journals and/or data could also alleviate the Ceph node bottleneck. With regards to scalability, the max amount of guests that could be launched directly depends on the amount of RAM available on each compute node. The max guest instances per compute is easily calculated using the `nova hypervisor-stats` command when launching X amount of guest instances all taking the same amount of RAM resources. When re-running the Rally scenarios with the recommendations of Browbeat, specifically the `tuned` profiles for each node in the RHEL-OSP environment the boot times drastically improved even when the Ceph nodes were the bottleneck. The max-guest tests running with 2 compute nodes using a low concurrency value showed a decrease in average boot times of 25%, while boot-storm tests with a high concurrency value showed a decrease in average boot times of 13.7%. These small changes in the environment show the importance of taking the Browbeat recommendations and implementing them.

The tools described within this reference architecture help the administrator know the physical limits and proper tuning changes of the configuration under test. Monitoring these resources when in production should allow for the administrator to tweak setting as needed, add compute nodes to extend VM capacity, and storage nodes to improve IO/sec.

For any questions or concerns, please email refarch-feedback@redhat.com and ensure to visit the Red Hat Reference Architecture page to find about all of our Red Hat solution offerings.

Appendix A: Contributors

1. John Herr - content review
2. Steven Reichard - content review
3. Douglas Shakshober - content review
4. Dan Allen - Asciidoctor PDF

Appendix B: References

1. *Deploying Red Hat Enterprise Linux OpenStack Platform 7 with RHEL-OSP director 7.1* by Jacob Liberman - http://red.ht/1PDxLxp
2. *Red Hat Enterprise Linux OpenStack Platform on Inktank Ceph Enterprise* by Red Hat Performance Team - http://bit.ly/20vAp0c

Appendix C: Hardware specifications

Table 4. Hardware specifications

Count	Model	Description
9	Dell PowerEdge M520	2x Intel Xeon CPU E5-2450 0 @ 2.10GHz, Broadcom 5720 1Gb Dual Port LOMs, Broadcom 57810S-k Dual Port 10Gb NIC, 6x DDR3 8192 MB @1333 MHZ DIMMs, 2 x 146GB SAS internal disk drives
4	Dell PowerEdge R510	2x Intel® Xeon® CPU X5650 @ 2.67 GHz (6 core), 2 x Broadcom NetXtreme II BCM5709S Gb Ethernet, 2x Emulex Corporation OneConnect 10Gb NIC, 6 x DDR3 8192 MB @1333 MHZ DIMMs, 12x 146GB SAS internal disk drives
1	Dell PowerEdge R720xd	2x Intel® Xeon® CPU X5650 @ 2.67 GHz (6 core), 2 x Broadcom NetXtreme II BCM5709S Gb Ethernet, 2x Emulex Corporation OneConnect 10Gb NIC, 6 x DDR3 8192 MB @1333 MHZ DIMMs, 12x 146GB SAS internal disk drives

Appendix D: Rally Validate Nova JSON File

```
{
  "Authenticate.validate_nova": [
    {
      "args": {
        "repetitions": 2
      },
      "context": {
        "users": {
          "project_domain": "default",
          "resource_management_workers": 30,
          "tenants": 1,
          "user_domain": "default",
          "users_per_tenant": 8
        }
      },
      "runner": {
        "concurrency": 1,
        "times": 1,
        "type": "constant"
      }
    }
  ]
}
```

Appendix E: Rally Boot and List Server JSON File

```
{% set flavor_name = flavor_name or "m1.small" %}
{
    "NovaServers.boot_and_list_server": [
        {
            "args": {
                "flavor": {
                    "name": "{{flavor_name}}"
                },
                "nics": [{
                    "net-id": "0fd1b597-7ed0-45cf-b9e2-a5dfbee80377"
                }],
                "image": {
                    "name": "rhel-server7_raw"
                },
                "detailed": true
            },
            "runner": {
                "concurrency": 1,
                "times": 1,
                "type": "constant"
            },
            "context": {
                "users": {
                    "tenants": 1,
                    "users_per_tenant": 1
                },
                "quotas": {
                    "neutron": {
                        "network": -1,
                        "port": -1
                    },
                    "nova": {
                        "instances": -1,
                        "cores": -1,
                        "ram": -1
                    }
                }
            }
        }
    ]
}
```


redhat.