Browbeat Documentation

Release

OpenStack Foundation

Mar 23, 2018

Contents

1 Introduction **3**

2 Installation **5**
- 2.1 Install Browbeat on Undercloud . . . 5
- 2.2 Install Browbeat from your local machine . . . 7
- 2.3 Install/Setup Browbeat Machine . . . 8
- 2.4 Using Keystone Public Endpoint . . . 12
- 2.5 Uploading Images to the overcloud . . . 12

3 Additional Components Installation **13**
- 3.1 Install Monitoring Host (Carbon/Graphite/Grafana) . . . 13
- 3.2 Install ELK Host (ElasticSearch/LogStash/Kibana) . . . 15

4 Usage **19**
- 4.1 Run Overcloud checks . . . 19
- 4.2 Run Browbeat performance tests from Undercloud . . . 19
- 4.3 Running PerfKitBenchmarker . . . 19
- 4.4 Running Shaker . . . 20
- 4.5 Running YODA . . . 21
- 4.6 Interpreting Browbeat Results . . . 22
- 4.7 Working with Multiple Clouds . . . 22
- 4.8 Compare software-metadata from two different runs . . . 23

5 Plugins **25**
- 5.1 Rally . . . 25

6 Browbeat as a CI tool **27**
- 6.1 Browbeat as a Quickstart Extra . . . 27

7 Developing against Quickstart **29**
- 7.1 Why use Quickstart? . . . 29
- 7.2 Limitations . . . 29
- 7.3 Hardware Requirements . . . 29
- 7.4 Localhost Preparation . . . 30
- 7.5 Create a Quickstart cloud . . . 30
- 7.6 Connecting to your Undercloud/Overcloud from your local machine . . . 34
- 7.7 Setup Browbeat against your Quickstart Cloud . . . 34

7.8 Troubleshooting 36

8 Contributing **39**
8.1 Adding functionality 40

9 Indices and tables **43**

Contents:

CHAPTER 1

Introduction

This started as a project to help determine the number of database connections a given OpenStack deployment uses via stress tests. It has since grown into a set of Ansible playbooks to help check deployments for known issues, install tools, run performance stress workloads and change parameters of the overcloud.

CHAPTER 2

Installation

Browbeat is currently installed via an ansible playbook. In a Tripleo environment it can be installed directly on the Undercloud or a separate machine. The installation can be run from either your local machine or directly on the machine you want Browbeat installed on.

2.1 Install Browbeat on Undercloud

This is usually the easiest installation due to many requirements are satisfied on the Undercloud. In some cases it may not be desired to install Browbeat on the Undercloud (Ex. Limited Resource requirements or Non-Tripleo installed cloud)

2.1.1 Requirements

Hardware

- Undercloud Machine (Baremetal or Virtual Machine)

Networking

- Access to Public API endpoints
- Access to Keystone Admin Endpoint

Note: For tripleo, public API endpoints are located on the External Network by default. The Keystone Admin Endpoint is deployed on the ctlplane network by default. These networking requirements should be validated before attempting an installation.

2.1.2 On the Undercloud

```
$ ssh undercloud-root
[root@undercloud ~]# su - stack
[stack@undercloud ~]$ git clone https://github.com/openstack/browbeat.git
[stack@undercloud ~]$ cd browbeat/ansible
[stack@undercloud ansible]$ ./generate_tripleo_hostfile.sh -t localhost
[stack@undercloud ansible]$ sudo easy_install pip
[stack@undercloud ansible]$ sudo pip install ansible
[stack@undercloud ansible]$ vi install/group_vars/all.yml # Make sure to edit the dns_
↪server to the correct ip address
[stack@undercloud ansible]$ ansible-playbook -i hosts install/browbeat.yml
[stack@undercloud ansible]$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: Your default network might not work for you depending on your underlay/overlay network setup. In such cases, user needs to create appropriate networks for instances to allow them to reach the internet. Some useful documentation can be found at: https://access.redhat.com/documentation/en/red-hat-openstack-platform/11/single/networking-guide/

2.1.3 (Optional) Install Browbeat instance workloads

Browbeat instance workloads are orchestrated Rally plugins that ship with Browbeat. We currently support a handful of workloads

- Pbench-Uperf - Networking throughput / RR test
- Linpack - Microbenchmark for CPU load

To enable installation of the Browbeat workloads set install_browbeat_workloads: true in ansible/install/group_vars/all.yml.

It is also required to provide the neutron network id of a private network which has external access. To set this, edit ansible/install/group_vars/all.yml and provide the network id for the browbeat_network:

This work can either be done prior to installation of Browbeat, or after Browbeat has been installed. To skip directly to this task execute:

```
$ ansible-playbook -i hosts install/browbeat.yml --start-at-task "Check browbeat_
↪network"
...
```

2.1.4 (Optional) Install Collectd

```
[stack@ospd ansible]$ ansible-playbook -i hosts install/collectd-openstack.yml
```

2.1.5 (Optional) Install Rsyslogd logging with aggregation

First configure the values rsyslog values and elasticsearch parameters in *ansible/install/group_vars/all.yml*. If you have a large number of hosts deploying an aggregator using *ansible/install/rsyslog-aggregator.yml* is strongly suggested. If you have a small scale, change the value rsyslog_forwarding in *all.yml* to *false*. Once things are configured to your liking deploy logging on the cloud using the *rsyslog-logging.yml* playbook.

Firewall configuration for the aggregator is left up to the user. The logging install playbook will check that the aggregator is up and the port is open if you deploy with aggregation.

```
[stack@ospd ansible]$ vim install/group_vars/all.yml
[stack@ospd ansible]$ ansible-playbook -i hosts install/rsyslog-aggregator.yml
[stack@ospd ansible]$ ansible-playbook -i hosts install/rsyslog-logging.yml
```

2.1.6 (Optional) Install Browbeat Grafana dashboards

```
[stack@ospd ansible]$ ansible-playbook -i hosts install/dashboards-openstack.yml
```

2.1.7 Run Overcloud checks

```
[stack@ospd ansible]$ ansible-playbook -i hosts check/site.yml
```

Your Overcloud check output is located in results/bug_report.log

2.2 Install Browbeat from your local machine

This installs Browbeat onto your Undercloud but the playbook is run from your local machine rather than directly on the Undercloud machine.

2.2.1 From your local machine

```
$ ssh-copy-id stack@<undercloud-ip>
$ git clone https://github.com/openstack/browbeat.git
$ cd browbeat/ansible
$ ./generate_tripleo_hostfile.sh -t <undercloud-ip>
$ vi install/group_vars/all.yml # Review and edit configuration items
$ ansible-playbook -i hosts install/browbeat.yml
$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: Your default network might not work for you depending on your underlay/overlay network setup. In such cases, user needs to create appropriate networks for instances to allow them to reach the internet. Some useful documentation can be found at: https://access.redhat.com/documentation/en/red-hat-openstack-platform/11/single/networking-guide/

2.2.2 (Optional) Install collectd

```
$ ansible-playbook -i hosts install/collectd-openstack.yml
```

2.2.3 (Optional) Install Browbeat Grafana dashboards

```
$ ansible-playbook -i hosts install/grafana-dashboards.yml
```

2.3 Install/Setup Browbeat Machine

This setup is used when running Browbeat on a separate machine than the Undercloud. Using this method, you can create multiple users on the machine and each user can be pointed at a different cloud or the same cloud.

2.3.1 Requirements

Hardware

- Baremetal or Virtual Machine

Networking

- Access to Public API endpoints
- Access to Keystone Admin Endpoint

RPM

- epel-release
- ansible
- git

OpenStack

- overcloudrc file placed in browbeat user home directory

Note: For tripleo, public API endpoints are located on the External Network by default. The Keystone Admin Endpoint is deployed on the ctlplane network by default. These networking requirements should be validated before attempting an installation.

2.3.2 Preparing the Machine (CentOS 7)

1. Install Machine either from Image, ISO, or PXE
2. Check for Required Network Connectivity

Determine Overcloud Keystone endpoints

```
[stack@undercloud-1 ~]$ . overcloudrc
[stack@undercloud-1 ~]$ openstack catalog show identity
+-----------+-----------------------------------------+
| Field     | Value                                   |
+-----------+-----------------------------------------+
| endpoints | regionOne                               |
|           |   publicURL: http://172.21.0.10:5000    |
|           |   internalURL: http://172.16.0.16:5000  |
|           |   adminURL: http://192.168.24.61:35357  |
|           |                                         |
| name      | keystone                                |
```

```
| type      | identity                                |
+-----------+-----------------------------------------+
```

Check network connectivity

```
$ ssh root@browbeatvm
[root@browbeatvm ~]$ # Ping Keystone Admin API IP Address
[root@browbeatvm ~]# ping -c 2 192.168.24.61
PING 192.168.24.61 (192.168.24.61) 56(84) bytes of data.
64 bytes from 192.168.24.61: icmp_seq=1 ttl=64 time=1.60 ms
64 bytes from 192.168.24.61: icmp_seq=2 ttl=64 time=0.312 ms

--- 192.168.24.61 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.312/0.957/1.603/0.646 ms
[root@browbeatvm ~]$ # Ping Keystone Public API IP Address
[root@browbeatvm ~]# ping -c 2 172.21.0.10
PING 172.21.0.10 (172.21.0.10) 56(84) bytes of data.
64 bytes from 172.21.0.10: icmp_seq=1 ttl=64 time=0.947 ms
64 bytes from 172.21.0.10: icmp_seq=2 ttl=64 time=0.304 ms

--- 172.21.0.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.304/0.625/0.947/0.322 ms
```

3. Create user for Browbeat and generate SSH key

```
[root@browbeatvm ~]# useradd browbeat1
[root@browbeatvm ~]# passwd browbeat1
Changing password for user browbeat1.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@browbeatvm ~]# echo "browbeat1 ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/
↪browbeat1; chmod 0440 /etc/sudoers.d/browbeat1
browbeat1 ALL=(root) NOPASSWD:ALL
[root@browbeatvm ~]# su - browbeat1
[browbeat1@browbeatvm ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/browbeat1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/browbeat1/.ssh/id_rsa.
Your public key has been saved in /home/browbeat1/.ssh/id_rsa.pub.
The key fingerprint is:
c2:b2:f0:cd:ef:d2:2b:a8:9a:5a:bb:ca:ce:c1:8c:3b browbeat1@browbeatvm
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|                 |
|     .           |
|  . . o S        |
|+  o = .         |
|.+. o.o.         |
|E+... o..        |
|OB+o   ++.       |
+-----------------+
```

4. Enable passwordless SSH into localhost and Undercloud then copy overcloudrc over to Browbeat VM

```
[browbeat1@browbeatvm ansible]$ ssh-copy-id browbeat1@localhost
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any␣
↪that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it␣
↪is to install the new keys
browbeat1@localhost's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'browbeat1@localhost'"
and check to make sure that only the key(s) you wanted were added.

[browbeat1@browbeatvm ~]$ ssh-copy-id stack@undercloud-1
The authenticity of host 'undercloud-1 (undercloud-1)' can't be established.
ECDSA key fingerprint is fa:3a:02:e8:8e:92:4d:a7:9c:90:68:6a:c2:eb:fe:e1.
Are you sure you want to continue connecting (yes/no)? yes
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any␣
↪that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it␣
↪is to install the new keys
stack@undercloud-1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'stack@undercloud-1'"
and check to make sure that only the key(s) you wanted were added.

[browbeat1@browbeatvm ~]$ scp stack@undercloud-1:/home/stack/overcloudrc .
overcloudrc                                   100%  553     0.5KB/s   00:00
```

Note: In SSL environments, you must copy the certificate over and check that the "OS_CA_CERT" variable is set correctly to the copied certificate location

5. Install RPM requirements

```
[browbeat1@browbeatvm ~]$ sudo yum install -y epel-release
[browbeat1@browbeatvm ~]$ sudo yum install -y ansible git
```

6. Clone Browbeat

```
[browbeatuser1@browbeat-vm ~]$ git clone https://github.com/openstack/browbeat.git
Cloning into 'browbeat'...
remote: Counting objects: 7425, done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 7425 (delta 14), reused 12 (delta 12), pack-reused 7398
Receiving objects: 100% (7425/7425), 5.23 MiB | 0 bytes/s, done.
Resolving deltas: 100% (4280/4280), done.
```

7. Generate hosts, ssh-config, and retrieve heat-admin-id_rsa.

```
[browbeat1@browbeatvm ~]$ cd browbeat/ansible/
[browbeat1@browbeatvm ansible]$ ./generate_tripleo_hostfile.sh -t undercloud-1 --
↪localhost
...
```

```
[browbeat1@browbeatvm ansible]$ ls ssh-config hosts heat-admin-id_rsa
heat-admin-id_rsa  hosts  ssh-config
```

Note use of "–localhost" to indicate the desire to install browbeat on the localhost rather than the undercloud.

8. Edit installation variables

```
[browbeat1@browbeatvm ansible]$ vi install/group_vars/all.yml
```

In this case, adjust browbeat_user, iptables_file and dns_server. Each environment is different and thus your configuration options will vary.

Note: If you require a proxy to get outside your network, you must configure http_proxy, https_proxy, no_proxy variables in the proxy_env dictionary in install/group_vars/all.yml

---

9. Run Browbeat install playbook

```
[browbeat1@browbeatvm ansible]$ ansible-playbook -i hosts install/browbeat.yml
```

10. Setup browbeat-config.yaml and test run Rally against cloud

```
[browbeat1@browbeatvm ansible]$ cd ..
[browbeat1@browbeatvm browbeat]$ vi browbeat-config.yaml
[browbeat1@browbeatvm browbeat]$ . .browbeat-venv/bin/activate
(browbeat-venv) [browbeat1@browbeatvm browbeat]$ python browbeat.py rally
```

11. Build Shaker image

```
[browbeatuser1@browbeat-vm ~]$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: Your default network might not work for you depending on your underlay/overlay network setup. In such cases, user needs to create appropriate networks for instances to allow them to reach the internet. Some useful documentation can be found at: https://access.redhat.com/documentation/en/red-hat-openstack-platform/11/single/networking-guide/

2.3.3 (Optional) Install collectd

```
[browbeatuser1@browbeat-vm ~]$ ansible-playbook -i hosts install/collectd-openstack.
↪yml
```

2.3.4 (Optional) Install Browbeat Grafana dashboards

```
[browbeatuser1@browbeat-vm ~]$ ansible-playbook -i hosts install/grafana-dashboards.
↪yml
```

2.3.5 Considerations for additional Browbeat Installs

If it is desired to run Browbeat against multiple clouds from the same machine. It is recommended to create a second user (Ex. browbeat2) and repeat above instructions. In order to expose the second user's Browbeat results via httpd,

change the port (Variable browbeat_results_port) and thus each user's results will be available via http on different ports.

Note: Keep in mind that running multiple sets of control plane workloads from multiple Browbeat users at the same time will introduce variation into resulting performance data if the machine on which Browbeat is installed is resource constrained.

2.4 Using Keystone Public Endpoint

If your Browbeat installation can not reach the Keystone Admin API endpoint due to the networking, you can use Keystone V3 options. In your overcloudrc or rc file you can add the following environment variables.

```
export OS_IDENTITY_API_VERSION=3
export OS_INTERFACE=public
```

2.5 Uploading Images to the overcloud

Browbeat by default uploads CentOS and CirrOS images to the cloud for use in Rally and other workloads. It is recommended to upload RAW images if using ceph and hence the convert_to_raw variable must be set to true as shown below in ansible/install/group_vars/all.yml. The default is false.

```
images:
  centos7:
    name: centos7
    url: http://cloud.centos.org/centos/7/images/CentOS-7-x86_64-GenericCloud.qcow2
    type: qcow2
    convert_to_raw: true
```

CHAPTER 3

Additional Components Installation

3.1 Install Monitoring Host (Carbon/Graphite/Grafana)

A monitoring host exposes System and Application performance metrics to the Browbeat user via Grafana. It helps expose what may be causing your bottleneck when you encounter a performance issue.

3.1.1 Prerequisites

Hardware

- Baremetal or Virtual Machine
- SSD storage

Operating System

- RHEL 7
- CentOS 7

Repos

- Red Hat Enterprise Linux 7Server - x86_64 - Server
- Red Hat Enterprise Linux 7Server - x86_64 - Server Optional

RPM

- epel-release
- ansible
- git

3.1.2 Installation

1. Deploy machine (RHEL7 is used in this example)
2. Install RPMS

```
[root@dhcp23-93 ~]# yum install -y https://download.fedoraproject.org/pub/epel/epel-
↪release-latest-7.noarch.rpm
...
[root@dhcp23-93 ~]# yum install -y ansible git
```

3. Clone Browbeat

```
[root@dhcp23-93 ~]# git clone https://github.com/openstack/browbeat.git
Cloning into 'browbeat'...
remote: Counting objects: 7533, done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 7533 (delta 30), reused 36 (delta 23), pack-reused 7469
Receiving objects: 100% (7533/7533), 5.26 MiB | 5.79 MiB/s, done.
Resolving deltas: 100% (4330/4330), done.
```

4. Add a hosts file into ansible directory

```
[root@dhcp23-93 ~]# cd browbeat/ansible/
[root@dhcp23-93 ansible]# vi hosts
```

Content of hosts file should be following

```
[graphite]
localhost

[grafana]
localhost
```

5. Setup SSH config, SSH key and exchange for Ansible

```
[root@dhcp23-93 ansible]# touch ssh-config
[root@dhcp23-93 ansible]# ssh-keygen
Generating public/private rsa key pair.
...
[root@dhcp23-93 ansible]# ssh-copy-id root@localhost
...
```

6. Edit install variables

```
[root@dhcp23-93 ansible]# vi install/group_vars/all.yml
```

Depending on the environment you may need to edit more than just the following variables - graphite_host and grafana_host

---

Note: If you require a proxy to get outside your network, you must configure http_proxy, https_proxy, no_proxy variables in the proxy_env dictionary in install/group_vars/all.yml

---

7. Install Carbon and Graphite via Ansible playbook

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/graphite.yml
...
```

8. Install Grafana via Ansible playbook

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/grafana.yml
...
```

9. Install Grafana dashboards via Ansible playbook

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/grafana-dashboards.yml -e
↪'cloud_dashboards=false'
...
```

10. (Optional) Monitor the Monitor Host

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/collectd-generic.yml --
↪tags graphite
...
```

Now navigate to http://monitoring-host-address:3000 to verify Grafana is installed, the Graphite data source exists and custom dashboards are uploaded.

You can now point other clouds at this host in order to view System and Application performance metrics. Depending on the number of clouds and machines pointed at your monitoring server, you may need to add more disk IO capacity, disk storage or carbon-cache+carbon-relay processes depending entirely on the number of metrics and your environments capacity. There is a Graphite dashboard included and it is recommended to install collectd on your monitoring host such that you can see if you hit resource issues with your monitoring host.

3.2 Install ELK Host (ElasticSearch/LogStash/Kibana)

An ELK server allows you to publish resulting benchmark data into ElasticSearch which allows you to build querys and dashboards to examine your benchmarking result data over various metadata points.

3.2.1 Prerequisites

Hardware

- Baremetal or Virtual Machine

Operating System

- RHEL 7
- CentOS 7

Repos

- Red Hat Enterprise Linux 7Server - x86_64 - Server
- Red Hat Enterprise Linux 7Server - x86_64 - Server Optional

RPM

- epel-release
- ansible
- git

3.2.2 Installation

1. Deploy machine (RHEL7 is used in this example)
2. Install RPMS

```
[root@dhcp23-93 ~]# yum install -y https://download.fedoraproject.org/pub/epel/epel-
↪release-latest-7.noarch.rpm
...
[root@dhcp23-93 ~]# yum install -y ansible git
```

3. Clone Browbeat

```
[root@dhcp23-93 ~]# git clone https://github.com/openstack/browbeat.git
Cloning into 'browbeat'...
remote: Counting objects: 7533, done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 7533 (delta 30), reused 36 (delta 23), pack-reused 7469
Receiving objects: 100% (7533/7533), 5.26 MiB | 5.79 MiB/s, done.
Resolving deltas: 100% (4330/4330), done.
```

4. Add a hosts file into ansible directory

```
[root@dhcp23-93 ~]# cd browbeat/ansible/
[root@dhcp23-93 ansible]# vi hosts
```

Content of hosts file should be following

```
[elk]
localhost
```

5. Setup SSH config, SSH key and exchange for Ansible

```
[root@dhcp23-93 ansible]# touch ssh-config
[root@dhcp23-93 ansible]# ssh-keygen
Generating public/private rsa key pair.
...
[root@dhcp23-93 ansible]# ssh-copy-id root@localhost
...
```

6. Edit install variables

```
[root@dhcp23-93 ansible]# vi install/group_vars/all.yml
```

Depending on the environment you may need to edit more than just the following variables - es_ip

If you are deploying using a machine that is not an OSP undercloud, be sure to edit the home_dir/browbeat_path to match its actual path.

Note: If you require a proxy to get outside your network, you must configure http_proxy, https_proxy, no_proxy variables in the proxy_env dictionary in install/group_vars/all.yml

---

7. Install ELK via Ansible playbook

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/elk.yml
...
```

8. Install Kibana Visualizations via Ansible playbook

```
[root@dhcp23-93 ansible]# ansible-playbook -i hosts install/kibana-visuals.yml
...
```

Now navigate to http://elk-host-address to verify Kibana is installed and custom visualizations are uploaded.

CHAPTER 4

Usage

4.1 Run Overcloud checks

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

Your Overcloud check output is located in results/bug_report.log

NOTE: It is strongly advised to not run the ansible playbooks in a virtual environment.

4.2 Run Browbeat performance tests from Undercloud

```
$ ssh undercloud-root
[root@ospd ~]# su - stack
[stack@ospd ~]$ cd browbeat/
[stack@ospd browbeat]$ . .browbeat-venv/bin/activate
(browbeat-venv)[stack@ospd browbeat]$ vi browbeat-config.yaml # Edit browbeat-config.
↪yaml to control how many stress tests are run.
(browbeat-venv)[stack@ospd browbeat]$ ./browbeat.py <workload> #perfkit, rally,␣
↪shaker or "all"
```

4.3 Running PerfKitBenchmarker

Many benchmarks work out of the box with Browbeat. You must ensure that your network is setup correctly to run those benchmarks. Currently tested benchmarks include: aerospike, bonnie++, cluster_boot, copy_throughput(cp,dd,scp), fio, iperf, mesh_network, mongodb_ycsb, netperf, object_storage_service, ping, scimark2, and sysbench_oltp.

To run Browbeat's PerfKit Benchmarks, you can start by viewing the tested benchmark's configuration in conf/browbeat-perfkit-complete.yaml. You must add them to your specific Browbeat config yaml file or enable/disable the benchmarks you wish to run in the default config file (browbeat-config.yaml). There are many flags exposed in

the configuration files to tune how those benchmarks run. Additional flags are exposed in the source code of PerfKitBenchmarker available on the Google Cloud Github.

Example running only PerfKitBenchmarker benchmarks with Browbeat from browbeat-config.yaml:

```
(browbeat-venv)[stack@ospd browbeat]$ ./browbeat.py perfkit -s browbeat-config.yaml
```

4.4 Running Shaker

Running Shaker requires the shaker image to be built, which in turn requires instances to be able to access the internet. The playbooks for this installation have been described in the installation documentation but for the sake of convenience they are being mentioned here as well.

```
$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: The playbook to setup networking is provided as an example only and might not work for you based on your underlay/overlay network setup. In such cases, the exercise of setting up networking for instances to be able to access the internet is left to the user.

Once the shaker image is built, you can run Shaker via Browbeat by filling in a few configuration options in the configuration file. The meaning of each option is summarized below:

shaker:

enabled Boolean `true` or `false`, enable shaker or not

server IP address of the shaker-server for agent to talk to (undercloud IP by default)

port Port to connect to the shaker-server (undercloud port 5555 by default)

flavor OpenStack instance flavor you want to use

join_timeout Timeout in seconds for agents to join

sleep_before Time in seconds to sleep before executing a scenario

sleep_after Time in seconds to sleep after executing a scenario

shaker_region OpenStack region you want to use

external_host IP of a server for external tests (should have `browbeat/util/shaker-external.sh` executed on it previously and have iptables/firewalld/selinux allowing connections on the ports used by network testing tools netperf and iperf)

scenarios: List of scenarios you want to run

- name Name for the scenario. It is used to create directories/files accordingly

enabled Boolean `true` or `false` depending on whether or not you want to execute the scenario

density Number of instances

compute Number of compute nodes across which to spawn instances

placement `single_room` would mean one instance per compute node and `double_room` would give you two instances per compute node

progression `null` means all agents are involved, `linear` means execution starts with one agent and increases linearly, `quadratic` would result in quadratic growth in number of agents participating in the test concurrently

time Time in seconds you want each test in the scenario file to run

file The base shaker scenario file to use to override options (this would depend on whether you want to run L2, L3 E-W or L3 N-S tests and also on the class of tool you want to use such as flent or iperf3)

To analyze results sent to Elasticsearch (you must have Elasticsearch enabled and the IP of the Elasticsearch host provided in the browbeat configuration file), you can use the following playbook to setup some prebuilt dashboards for you:

```
$ ansible-playbook -i hosts install/kibana-visuals.yml
```

Alternatively you can create your own visualizations of specific shaker runs using some simple searches such as:

```
shaker_uuid: 97092334-34e8-446c-87d6-6a0f361b9aa8 AND record.concurrency: 1 AND␣
↪result.result_type: bandwidth
shaker_uuid: c918a263-3b0b-409b-8cf8-22dfaeeaf33e AND record.concurrency:1 AND record.
↪test:Bi-Directional
```

4.5 Running YODA

YODA (Yet Openstack Deployment tool, Another) is a workload integrated into Browbeat for benchmarking TripleO deployment. This includes importing baremetal nodes, running introspections and overcloud deployements of various kinds. Note that YODA assumes it is on the undercloud of a TripleO instance post undercloud installation and introspection.

4.5.1 Configuration

For examples of the configuration see *browbeat-complete.yaml* in the repo root directory. Additional configuration documentation can be found below for each subworkload of YODA.

4.5.2 Overcloud

For overcloud workloads, note that the nodes dictionary is dynamic, so you don't have to define types you aren't using, this is done in the demonstration configurations for the sake of completeness. Furthermore the node name is taken from the name of the field, meaning custom role names should work fine there.

The step parameter decides how many nodes can be distributed between the various types to get from start scale to end scale, if these are the same it won't matter. But if they are different up to that many nodes will be distributed to the different node types (in no particular order) before the next deploy is performed. The step rule is violated if and only if it is required to keep the deployment viable, for example if the step dictates that 2 control nodes be deployed it will skip to 3 even if it violates step.

YODA has basic support for custom templates and more advanced roles, configure the *templates:* paramater in the overcloud benchmark section with a string for template paths.

templates: "-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml"

Note that *–templates* is passed to the *overcloud deploy* command before this, then nodes sizes, ntp server and timeout are passed after, so your templates will override the defaults, but not scale, timeout, or ntp settings from the YODA config. If you want to use scheduling hints for your overcloud deploy you will need to pip install [ostag](https://github.com/jkilpatr/ostag) and set *node_pinning: True* in your config file. Ostag will be used before every deploy to clean all tags and tag the appropriate nodes. If you set *node_pinning: False* tags will be cleaned before the deploy. If

you need more advanced features view the ostag readme for how to tag based on node properties. If you don't want YODA to edit your node properties, don't define *node_pinning* in your configuration.

4.5.3 Introspection

Introspection workloads have two modes, batch and individual, the batch workload follows the documentation exactly, nodes are imported, then bulk introspection is run. Individual introspection has it's own custom batch size and handles failures more gracefully (individual instead of group retries). Both have a timeout configured in seconds and record the amount of time required for each node to pxe and the number of failures.

timeout is how long we wait for the node to come back from introspection this is hardware variable. Although the default 900 seconds has been shown to be the 99th percentile for success across at least two stes of hardware. Adjust as required.

Note that *batch_size* can not produce a batch of unintrospected ndoes if none exist so the last batch may be below the maximum size. When nodes in a batch fail the *failure_count* is incremented and the nodes are returned to the pool. So it's possible that same node will fail again in another batch. There is a saftey mechanism that will kill Yoda if a node exceeds 10 retries as that's pretty much garunteed to be misconfigured. For bulk introspection all nodes are tried once and what you get is what you get.

If you wish to change the introspection workload failure threshold of 10% you can set *max_fail_amnt* to any floating point value you desire.

I would suggest bulk introspection for testing documented TripleO workflows and individual introspection to test the performance of introspection itself.

4.6 Interpreting Browbeat Results

By default results for each test will be placed in a timestamped folder *results/* inside your Browbeat folder. Each run folder will contain output files from the various workloads and benchmarks that ran during that Browbeat run, as well as a report card that summarizes the results of the tests.

Browbeat for the most part tries to restrict itself to running tests, it will only exit with a nonzero return code if a workload failed to run. If, for example, Rally where to run but not be able to boot any instances on your cloud Browbeat would return with RC 0 without any complaints, only by looking into the Rally results for that Browbeat run would you determine that your cloud had a problem that made benchmarking it impossible.

Likewise if Rally manages to run at a snails pace, Browbeat will still exit without complaint. Be aware of this when running Browbeat and take the time to either view the contents of the results folder after a run. Or setup Elasticsearch and Kibana to view them more easily.

4.7 Working with Multiple Clouds

If you are running playbooks from your local machine you can run against more than one cloud at the same time. To do this, you should create a directory per-cloud and clone Browbeat into that specific directory:

```
[browbeat@laptop ~]$ mkdir cloud01; cd cloud01
[browbeat@laptop cloud01]$ git clone git@github.com:openstack/browbeat.git
...
[browbeat@laptop cloud01]$ cd browbeat/ansible
[browbeat@laptop ansible]$ ./generate_tripleo_hostfile.sh -t <cloud01-ip-address>
```

```
[browbeat@laptop ansible]$ ansible-playbook -i hosts (Your playbook you wish to run...
↪)
[browbeat@laptop ansible]$ ssh -F ssh-config overcloud-controller-0  # Takes you to␣
↪first controller
```

Repeat the above steps for as many clouds as you have to run playbooks against your clouds.

4.8 Compare software-metadata from two different runs

Browbeat's metadata is great to help build visuals in Kibana by querying on specific metadata fields, but sometimes we need to see what the difference between two builds might be. Kibana doesn't have a good way to show this, so we added an option to Browbeat CLI to query ElasticSearch.

To use :

```
$ python browbeat.py --compare software-metadata --uuid "browbeat-uuid-1" "browbeat-
↪uuid-2"
```

Real world use-case, we had two builds in our CI that used the exact same DLRN hash, however the later build had a 10x performance hit for two Neutron operations, router-create and add-interface-to-router. Given we had exactly the same DLRN hash, the only difference could be how things were configured. Using this new code, we could quickly identify the difference – TripleO enabled l3_ha.

```
[rocketship:browbeat] jtaleric:browbeat$ python browbeat.py --compare software-
↪metadata --uuid "3fc2f149-7091-4e16-855a-60738849af17" "6738eed7-c8dd-4747-abde-
↪47c996975a57"
2017-05-25 02:34:47,230 - browbeat.Tools -    INFO - Validating the configuration␣
↪file passed by the user
2017-05-25 02:34:47,311 - browbeat.Tools -    INFO - Validation successful
2017-05-25 02:34:47,311 - browbeat.Elastic -    INFO - Querying Elastic : index [_
↪all] : role [controller] : uuid [3fc2f149-7091-4e16-855a-60738849af17]
2017-05-25 02:34:55,684 - browbeat.Elastic -    INFO - Querying Elastic : index [_
↪all] : role [controller] : uuid [6738eed7-c8dd-4747-abde-47c996975a57]
2017-05-25 02:35:01,165 - browbeat.Elastic -    INFO - Difference found : Host␣
↪[overcloud-controller-2] Service [neutron] l3_ha [False]
2017-05-25 02:35:01,168 - browbeat.Elastic -    INFO - Difference found : Host␣
↪[overcloud-controller-1] Service [neutron] l3_ha [False]
2017-05-25 02:35:01,172 - browbeat.Elastic -    INFO - Difference found : Host␣
↪[overcloud-controller-0] Service [neutron] l3_ha [False]
```

CHAPTER 5

Plugins

5.1 Rally

5.1.1 Context - browbeat_delay

This context allows a setup and cleanup delay to be introduced into a scenario.

5.1.2 Context - browbeat_persist_network

This context creates network resources that persist upon completion of a rally run. It is used in conjunction with the nova_boot_persist_with_network and nova_boot_persist_with_network_volume plugin scenarios. You can also use *neutron purge* command to purge a project/tenant of neutron network resources.

5.1.3 Scenario - nova_boot_persist

This scenario creates instances without a network that persist upon completion of a rally run. This scenario is best used for excerising the Telemetry systems within an OpenStack Cloud. Alternatively, it can be used to put idle instances on a cloud for other workloads to compete for resources. The scenario is referenced in the Telemetry Browbeat configurations in order to build a "stepped" workload that can be used to analyze Telemetry performance and scalability.

5.1.4 Scenario - nova_boot_persist_with_volume

This scenario creates instances that have an attached volume and persist upon completion of a rally run. This scenario is best used for excerising the Telemetry systems within an OpenStack Cloud. It increases the Telemetry workload by creating more resources that the Telemetry services must collect and process metrics over. Alternatively, it can be used to put idle instances on a cloud for other workloads to compete for resources. The scenario is referenced in the Telemetry Browbeat configurations in order to build a "stepped" workload that can be used to analyze Telemetry scalability.

5.1.5 Scenario - nova_boot_persist_with_network

This scenario creates instances that are attached to a network and persist upon completion of a rally run. This scenario is best used for excerising the Telemetry systems within an OpenStack Cloud. It increases the Telemetry workload by creating more resources that the Telemetry services must collect and process metrics over. Alternatively, it can be used to put idle instances on a cloud for other workloads to compete for resources. The scenario is referenced in the Telemetry Browbeat configurations in order to build a "stepped" workload that can be used to analyze Telemetry scalability.

5.1.6 Scenario - nova_boot_persist_with_network_fip

This scenario creates instances with a nic and associates a floating ip that persist upon completion of a rally run. It is used as a workload with Telemetry by spawning many instances that have many metrics for the Telemetry subsystem to collect upon.

5.1.7 Scenario - nova_boot_persist_with_network_volume

This scenario create instances with a nic and a volume that persist upon completion of a rally run. It is used as a workload with Telemetry by spawning many instances that have many metrics for the Telemetry subsystem to collect upon.

5.1.8 Scenario - nova_boot_persist_with_network_volume_fip

This scenario creates instances with a nic, a volume and associates a floating ip that persist upon completion of a rally run. It is used as a workload with Telemetry by spawning many instances that have many metrics for the Telemetry subsystem to collect upon.

CHAPTER 6

Browbeat as a CI tool

If you would like to make your own CI job add your bootstrapping script to *ci-scripts/<openstack installer>* and Ansible/Python components into normal locations in the repository. Try and provide as many defaults as possible so that job definitions on the Jenkins end can remain small and easily defined. this will help us keep script changes in the repository and better test them before merging.

6.1 Browbeat as a Quickstart Extra

TripleO Quickstart provides an extensible interface to allow "Extras" to add to to its core functionality of generating an entirely virtual Openstack Deployment using TripleO. The focus of this script is to deploy baremetal clouds in continuous integration (CI) for effective and extensible automated benchmarking.

6.1.1 Invoking Locally

Please read The Extras Documentation for a general background on how TripleO Quickstart Extras operate. Please also reference The Baremetal Environments Documentation if you need to configure your job to run on baremetal.

Browbeat provides two playbooks for use with Quickstart `quickstart-browbeat.yml` and `baremetal-virt-undercloud-tripleo-browbeat.yml` the first playbook is for deploying an entierly virtual setup on a single baremetal machine. The second playbook creates a virtual undercloud on a undercloud host machine and deploys a baremetal overcloud as configured by the users hardware environment.

Dependencies for this script (at least for Fedora 25) are

```
$ sudo dnf install ansible git python-virtualenv gcc redhat-rpm-config openssl-devel
```

To run virtual TripleO Quickstart CI set the following environmental vars and run *quickstart-virt.sh* this will create a TripleO environment and run a short Browbeat test. Since this is a all virtual setup it is not suggested for serious benchmarking.

```
export WORKSPACE={TripleO Quickstart Workspace}
export RELEASE={release}
```

```
export VIRTHOST={undercloud-fqdn}

pushd $WORKSPACE/browbeat/ci-scripts/tripleo

bash quickstart-virt.sh
```

To run the baremetal CI follow the requisite steps to setup a hardware environment (this is nontrival) then create a workspace folder and clone TripleO Quickstart and Browbeat into that workspace. Set the variables below and then run `microbrow.sh`. There must be an all.yml file in the HW_ENV directory for overriding some browbeat variables with ones specific to the CI environment.

```
export WORKSPACE={TripleO Quickstart Workspace}
export HW_ENV={hw-env}
export RELEASE={release}
export GRAPH_HOST={Graphite + grafana host}
export GRAFANA_USER={username}
export GRAFANA_PASS={password}
export CLOUD_NAME={cloud-name}
export BENCHMARK={benchmark config file ex browbeat-basic.yaml.j2}
export ELASTIC_HOST={elastic host}
export VIRTHOST={undercloud-fqdn}

pushd $WORKSPACE/browbeat/ci-scripts/tripleo

bash microbrow.sh
```

6.1.2 Configurable Options

By default a cloud will be setup and a very basic benchmark will be run and all results will be placed only in the `browbeat/results` folder on the virtual undercloud.

If configured to use Elasticsearch metadata and benchmarks results will be inserted into Elasticsearch for easier visualization and storage. If Graphana is enabled performance metrics will be gathered from all cloud nodes and stored into the configured graphite instance to be processed by the Grafana dashboards created using the given username and password.

If enabled these dashboards will be automatically overwritten each run to reflect the number of nodes in your cloud and other changes that may occur between runs.

CHAPTER 7

Developing against Quickstart

This document helps you with creating a Tripleo Virtual Cloud on your local machine to assist with developing/testing Browbeat.

7.1 Why use Quickstart?

Tripleo-Quickstart enables us to have an entire tiny cloud to run Browbeat against. It gives you a virtual Undercloud, virtual Overcloud Controller and Computes and other virtual nodes as well. This allows you (with understood limitations) to run Browbeat, test commits, or develop actively with new code without requring a full set of hardware or to run code through CI.

7.2 Limitations

Since everything is virtualized on your local hardware, any performance results are subject to the limitations of your hardware as well as performance behaving with "noisy neighbors". This is only recommended for testing Browbeat and/or gaining familiarity with OpenStack Tripleo Clouds.

7.3 Hardware Requirements

Memory will most likely be your limitation:

- 16GiB Memory+Swap
  - Undercloud, 1 Controller
- 32GiB Memory is recommended
  - Undercloud, 1 Controller
  - Undercloud, 1 Controller, 1 Compute

- Undercloud, 3 Controllers

4 physical cpu cores is recommended with at least 50GB of free disk space ideally on an SSD.

7.4 Localhost Preparation

Ensure that sshd is running on your localhost

```
[akrzos@bithead ~]$ sudo systemctl enable sshd
[akrzos@bithead ~]$ sudo systemctl start sshd
```

Map 127.0.0.2 to your local host

```
[akrzos@bithead ~]$ sudo cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4 127.0.
↪0.2
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```

7.5 Create a Quickstart cloud

7.5.1 Download quickstart.sh

```
[akrzos@bithead ~]$ curl -O https://raw.githubusercontent.com/openstack/tripleo-
↪quickstart/master/quickstart.sh
```

7.5.2 Install dependencies

```
[akrzos@bithead ~]$ bash quickstart.sh --install-deps
```

7.5.3 Create Configuration and Nodes YAML Files

For this usage of Tripleo-quickstart, there are two configuration files to build a cloud, quickstart_config.yml and quickstart_nodes.yml configuration file. Quickstart_config.yml contains some basic options you may configure for your under/over clouds including ssl, cached image urls, enabling telemetry, and the networking setup. The nodes configuration file defines the amount of resources for your virtual overcloud including node count, Three examples are included here.

quickstart_config.yml

```
# Allow unsupported distros to deploy QuickStart (Ex. Fedora 24)
supported_distro_check: false

# Turn off Undercloud SSL
undercloud_generate_service_certificate: false

# Turn off Overcloud SSL
ssl_overcloud: false

# Turn off introspection
```

```
step_introspect: false

# Version of OpenStack (Ex: newton, ocata, pike)
release: ocata

#overcloud_as_undercloud: false
#force_cached_images: true
#dlrn_hash: current-passed-ci

# Use cached images when possible
#undercloud_image_url: http://walkabout.foobar.com/ci-images/ocata/current-passed-ci/
↪undercloud.qcow2
#ipa_image_url: http://walkabout.foobar.com/ci-images/ocata/current-passed-ci/ironic-
↪python-agent.tar
#overcloud_image_url: http://walkabout.foobar.com/ci-images/ocata/current-passed-ci/
↪overcloud-full.tar

# Tell tripleo how we want things done.
extra_args: >-
  --ntp-server pool.ntp.org

# This config is extremely resource intensive, so we disable telemetry
# in order to reduce the overall memory footprint
# This is not required in newton
telemetry_args: >-
   {% if release != 'newton' %}
   -e {{ overcloud_templates_path }}/environments/disable-telemetry.yaml
   {% endif %}

network_isolation: true
network_isolation_type: 'single-nic-vlans'

# Network setting on the virthost
external_network_cidr: 192.168.23.0/24
networks:
  - name: overcloud
    bridge: brovc
    address: "{{ undercloud_network_cidr|nthhost(2) }}"
    netmask: "{{ undercloud_network_cidr|ipaddr('netmask') }}"

  - name: external
    bridge: brext
    forward_mode: nat
    address: "{{ external_network_cidr|nthhost(1) }}"
    netmask: "{{ external_network_cidr|ipaddr('netmask') }}"
    dhcp_range:
      - "{{ external_network_cidr|nthhost(10) }}"
      - "{{ external_network_cidr|nthhost(50) }}"
    nat_port_range:
      - 1024
      - 65535

# Below are the networking options you will most likely need to adjust for your local␣
↪environment
# some are dervived from other vars and do not need to be adjusted.
undercloud_external_network_cidr: 172.21.0.0/24
undercloud_networks:
  external:
```

```
    address: "{{ undercloud_external_network_cidr|nthhost(1) }}"
    netmask: "{{ undercloud_external_network_cidr|ipaddr('netmask') }}"
    address6: "{{ undercloud_external_network_cidr6|nthhost(1) }}"
    device_type: ovs
    type: OVSIntPort
    ovs_bridge: br-ctlplane
    ovs_options: '"tag=10"'
    tag: 10

network_environment_args:
  ControlPlaneSubnetCidr: "{{ undercloud_network_cidr|ipaddr('prefix') }}"
  ControlPlaneDefaultRoute: "{{ undercloud_network_cidr|nthhost(1) }}"
  EC2MetadataIp: "{{ undercloud_network_cidr|nthhost(1) }}"

  ExternalNetCidr: 172.21.0.0/24
  ExternalAllocationPools: [{"start": "172.21.0.10", "end": "172.21.0.100"}]
  ExternalInterfaceDefaultRoute: 172.21.0.1
  NeutronExternalNetworkBridge: "''"

  InternalApiNetCidr: 172.16.0.0/24
  InternalApiAllocationPools: [{"start": "172.16.0.10", "end": "172.16.0.200"}]

  StorageNetCidr: 172.18.0.0/24
  StorageAllocationPools: [{"start": "172.18.0.10", "end": "172.18.0.200"}]

  StorageMgmtNetCidr: 172.19.0.0/24
  StorageMgmtAllocationPools: [{"start": "172.19.0.10", "end": "172.19.0.200"}]

  TenantNetCidr: 172.17.0.0/24
  TenantAllocationPools: [{"start": "172.17.0.10", "end": "172.17.0.250"}]
  DnsServers: [ '{{ external_network_cidr6|nthhost(1) }}' ]
```

quickstart_nodes.yml - 1 Controller

```
# Undercloud Virtual Hardware
undercloud_memory: 8192
undercloud_vcpu: 2

# Controller Virtual Hardware
control_memory: 6144
control_vcpu: 2

# Define a single controller node
overcloud_nodes:
  - name: control_0
    flavor: control
    virtualbmc_port: 6230

node_count: 1

deployed_server_overcloud_roles:
  - name: Controller
    hosts: "$(sed -n 1,1p /etc/nodepool/sub_nodes)"

topology: >-
  --compute-scale 0
```

quickstart_nodes.yml - 1 Controller, 1 Compute

```
# Undercloud Virtual Hardware
undercloud_memory: 8192
undercloud_vcpu: 2

# Controller Virtual Hardware
control_memory: 6144
control_vcpu: 2

# Compute Virtual Hardware
compute_memory: 4096
compute_vcpu: 1

overcloud_nodes:
  - name: control_0
    flavor: control
    virtualbmc_port: 6230
  - name: compute_0
    flavor: compute
    virtualbmc_port: 6231

node_count: 2

deployed_server_overcloud_roles:
  - name: Controller
    hosts: "$(sed -n 1,1p /etc/nodepool/sub_nodes)"

topology: >-
  --compute-scale 1
  --control-scale 1
```

quickstart_nodes.yml - 3 Controllers

```
# Undercloud Virtual Hardware
undercloud_memory: 8192
undercloud_vcpu: 2

# Controller Virtual Hardware
control_memory: 6144
control_vcpu: 1

# Define a single controller node
overcloud_nodes:
  - name: control_0
    flavor: control
    virtualbmc_port: 6230
  - name: control_1
    flavor: control
    virtualbmc_port: 6231
  - name: control_2
    flavor: control
    virtualbmc_port: 6232

node_count: 3

deployed_server_overcloud_roles:
  - name: Controller
    hosts: "$(sed -n 1,1p /etc/nodepool/sub_nodes)"
```

```
topology: >-
  --compute-scale 0
  --control-scale 3
```

Run quickstart.sh playbooks

You can change version of OpenStack (Ex. newton, ocata, master) you need by editing the *release* yaml parameter in quickstart_config.yaml (above).

```
time bash quickstart.sh -v -c quickstart_config.yml -N quickstart_nodes.yml -I -t all␣
↪-p quickstart.yml -T all -X 127.0.0.2
```

```
time bash quickstart.sh -v -c quickstart_config.yml -N quickstart_nodes.yml -I -t all␣
↪-p quickstart-extras-undercloud.yml -T none 127.0.0.2
```

```
time bash quickstart.sh -v -c quickstart_config.yml -N quickstart_nodes.yml -I -t all␣
↪-p quickstart-extras-overcloud-prep.yml -T none 127.0.0.2
```

```
time bash quickstart.sh -v -c quickstart_config.yml -N quickstart_nodes.yml -I -t all␣
↪-p quickstart-extras-overcloud.yml -T none 127.0.0.2
```

If all 4 playbooks completed without errors, you should have a local tripleo quickstart cloud. In order to validate, I would recommend ssh-ing into the Undercloud and issuing various openstack cli commands against the overcloud to verify the health of your quickstart-deployment.

7.6 Connecting to your Undercloud/Overcloud from your local machine

Create a vlan10 for external network access

```
[root@bithead network-scripts]# cat ifcfg-brovc.10
DEVICE=brovc.10
ONBOOT=yes
HOTPLUG=no
NM_CONTROLLED=no
VLAN=yes
IPADDR=172.21.0.2
NETMASK=255.255.255.0
BOOTPROTO=none
MTU=1500
[root@bithead network-scripts]# ifup brovc.10
```

You can now access the overcloud's external/public api endpoints from your local machine and install Browbeat for benchmarking against it.

7.7 Setup Browbeat against your Quickstart Cloud

After you have your Quickstart cloud up and the networking connectivity working, you will want to run Browbeat against it so you can begin contributing. Simply run the script in the utils folder to install Browbeat for usage on the new Tripleo Quickstart cloud.

```
[akrzos@bithead ~]$ git clone git@github.com:openstack/browbeat.git
Cloning into 'browbeat'...
Warning: Permanently added 'github.com,192.30.253.112' (RSA) to the list of known␣
↪hosts.
remote: Counting objects: 8567, done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 8567 (delta 19), reused 18 (delta 15), pack-reused 8523
Receiving objects: 100% (8567/8567), 5.52 MiB | 3.44 MiB/s, done.
Resolving deltas: 100% (4963/4963), done.
Checking connectivity... done.
[akrzos@bithead ~]$ cd browbeat/
[akrzos@bithead browbeat]$ ./utils/oooq-browbeat-install.sh
Installing Browbeat on localhost
...(Truncated)
~/code/browbeat-refactor/browbeat
[akrzos@bithead browbeat]$ . .browbeat-venv/bin/activate
(.browbeat-venv) [akrzos@bithead browbeat]$ ./browbeat.py -s conf/quickstart.yml rally
2017-12-13 15:46:34,648 - browbeat.config -    INFO - Config conf/quickstart.yml␣
↪validated
2017-12-13 15:46:34,655 - browbeat.config -    INFO - Workload ping-m1-tiny-centos␣
↪validated as perfkit
2017-12-13 15:46:34,657 - browbeat.config -    INFO - Workload quickstart-shaker-l2␣
↪validated as shaker
2017-12-13 15:46:34,665 - browbeat.config -    INFO - Workload quickstart-rally␣
↪validated as rally
2017-12-13 15:46:34,665 - browbeat -    INFO - Browbeat test suite kicked off
2017-12-13 15:46:34,665 - browbeat -    INFO - Browbeat UUID: 8e869626-a596-4ec7-b0b1-
↪ac7f2bf915a7
2017-12-13 15:46:34,666 - browbeat -    INFO - Running workload(s): rally
2017-12-13 15:46:34,666 - browbeat -    INFO - perfkit workload ping-m1-tiny-centos␣
↪disabled via cli
2017-12-13 15:46:34,666 - browbeat -    INFO - shaker workload quickstart-shaker-l2␣
↪disabled via cli
2017-12-13 15:46:34,666 - browbeat -    INFO - rally workload quickstart-rally is␣
↪enabled
2017-12-13 15:46:34,666 - browbeat.rally -    INFO - Running Rally workload:␣
↪quickstart-rally
2017-12-13 15:46:34,666 - browbeat.rally -    INFO - Running Scenario: authentic-
↪keystone
2017-12-13 15:46:34,669 - browbeat.rally -    INFO - Running with scenario_args: {
↪'concurrency': 1, 'times': 1}
2017-12-13 15:47:08,665 - browbeat.rally -    INFO - Generating Rally HTML for task_
↪id : 399b90d9-5bc2-431c-b7c9-b7782fef2dde
2017-12-13 15:47:10,224 - browbeat.rally -    INFO - Running Scenario: create-list-
↪network
2017-12-13 15:47:10,226 - browbeat.rally -    INFO - Running with scenario_args: {
↪'concurrency': 1, 'times': 1}
2017-12-13 15:47:45,781 - browbeat.rally -    INFO - Generating Rally HTML for task_
↪id : 544b7cc4-b15c-4308-8f1b-158f06f1b002
2017-12-13 15:47:47,414 - browbeat.rally -    INFO - Running Scenario: boot-list-
↪cirros
2017-12-13 15:47:47,417 - browbeat.rally -    INFO - Running with scenario_args: {
↪'flavor_name': 'm1.xtiny', 'concurrency': 1, 'image_name': 'cirros', 'times': 1}
2017-12-13 15:53:42,181 - browbeat.rally -    INFO - Generating Rally HTML for task_
↪id : 52c348d4-edba-4a3e-bfd9-48ee97cd6613
2017-12-13 15:53:44,566 - browbeat.workloadbase -    INFO - Total scenarios executed:3
2017-12-13 15:53:44,566 - browbeat.workloadbase -    INFO - Total tests executed:3
2017-12-13 15:53:44,566 - browbeat.workloadbase -    INFO - Total tests passed:3
```

```
2017-12-13 15:53:44,566 - browbeat.workloadbase -    INFO - Total tests failed:0
2017-12-13 15:53:44,568 - browbeat -    INFO - Saved browbeat result summary to /home/
↪akrzos/code/browbeat-refactor/browbeat/results/20171213-154634.report
2017-12-13 15:53:44,568 - browbeat -    INFO - Browbeat finished successfully, UUID:␣
↪8e869626-a596-4ec7-b0b1-ac7f2bf915a7
(.browbeat-venv) [akrzos@bithead browbeat]$ ls results/
20171213-154634  20171213-154634.report  browbeat-Rally-run.log
```

7.8 Troubleshooting

7.8.1 View Undercloud and Overcloud Instance

```
[root@bithead ~]# sudo su - stack -c 'virsh list --all'
 Id    Name                           State
----------------------------------------------------
 1     undercloud                     running
 3     compute_0                      running
 4     control_0                      running
```

7.8.2 Accessing Virtual Baremetal Nodes consoles

```
[root@bithead ~]# sudo su - stack -c 'virsh -c qemu:///session console undercloud'
Connected to domain undercloud
Escape character is ^]

Red Hat Enterprise Linux Server 7.3 (Maipo)
Kernel 3.10.0-514.26.2.el7.x86_64 on an x86_64

undercloud login:
```

7.8.3 Get to Undercloud via ssh

```
[akrzos@bithead ~]$ ssh -F ~/.quickstart/ssh.config.ansible undercloud
Warning: Permanently added '127.0.0.2' (ECDSA) to the list of known hosts.
Warning: Permanently added 'undercloud' (ECDSA) to the list of known hosts.
Last login: Tue Sep 19 13:25:33 2017 from gateway
[stack@undercloud ~]$
```

7.8.4 Get to Overcloud nodes via ssh

```
[akrzos@bithead ~]$ ssh -F ~/.quickstart/ssh.config.ansible overcloud-controller-0
Warning: Permanently added '127.0.0.2' (ECDSA) to the list of known hosts.
Warning: Permanently added 'undercloud' (ECDSA) to the list of known hosts.
Last login: Tue Sep 19 13:25:33 2017 from gateway
[heat-admin@overcloud-controller-0 ~]$
```

7.8.5 Other gotchas

Make sure your / partition does not fill up with cached images as they can take a large amount of space

```
[root@bithead ~]# df -h /var/cache/tripleo-quickstart/
Filesystem                          Size  Used Avail Use% Mounted on
/dev/mapper/fedora_dhcp23--196-root   50G   40G  6.9G  86% /
[root@bithead ~]# du -sh /var/cache/tripleo-quickstart/
5.4G  /var/cache/tripleo-quickstart/
```

7.8.6 Further Documentation

Tripleo Quickstart docs

CHAPTER 8

Contributing

Contributions are most welcome! You must first create a Launchpad account and follow the instructions here to get started as a new OpenStack contributor.

Once you've signed the contributor license agreement and read through the above documentation, add your public SSH key under the 'SSH Public Keys' section of review.openstack.org.

You can view your public key using:

```
$ cat ~/.ssh/id_*.pub
```

Set your username and email for review.openstack.org:

```
$ git config --global user.email "example@example.com"
$ git config --global user.name "example"
$ git config --global --add gitreview.username "example"
```

Next, Clone the github repository:

```
$ git clone https://github.com/openstack/browbeat.git
```

You need to have git-review in order to be able to submit patches using the gerrit code review system. You can install it using:

```
$ sudo yum install git-review
```

To set up your cloned repository to work with OpenStack Gerrit

```
$ git review -s
```

It's useful to create a branch to do your work, name it something related to the change you'd like to introduce.

```
$ cd browbeat
$ git branch my_special_enhancement
$ git checkout !$
```

Make your changes and then commit them using the instructions below.

```
$ git add /path/to/files/changed
$ git commit
```

Use a descriptive commit title followed by an empty space. You should type a small justification of what you are changing and why.

Now you're ready to submit your changes for review:

```
$ git review
```

If you want to make another patchset from the same commit you can use the amend feature after further modification and saving.

```
$ git add /path/to/files/changed
$ git commit --amend
$ git review
```

If you want to submit a new patchset from a different location (perhaps on a different machine or computer for example) you can clone the Browbeat repo again (if it doesn't already exist) and then use git review against your unique Change-ID:

```
$ git review -d Change-Id
```

Change-Id is the change id number as seen in Gerrit and will be generated after your first successful submission.

The above command downloads your patch onto a separate branch. You might need to rebase your local branch with remote master before running it to avoid merge conflicts when you resubmit the edited patch. To avoid this go back to a "safe" commit using:

```
$ git reset --hard commit-number
```

Then,

```
$ git fetch origin
```

```
$ git rebase origin/master
```

Make the changes on the branch that was setup by using the git review -d (the name of the branch is along the lines of review/username/branch_name/patchsetnumber).

Add the files to git and commit your changes using,

```
$ git commit --amend
```

You can edit your commit message as well in the prompt shown upon executing above command.

Finally, push the patch for review using,

```
$ git review
```

8.1 Adding functionality

If you are adding new functionality to Browbeat please add testing for that functionality in.

```
$ ci-scripts/install-and-check.sh
```

See the README.rst in the ci-scripts folder for more details on the structure of the script and how to add additional tests.

CHAPTER 9

Indices and tables

- genindex
- modindex
- search