Browbeat Documentation

Release

OpenStack Foundation

May 25, 2017

Contents

Contents:

CHAPTER 1

Browbeat 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

Installing Browbeat and running the Overcloud checks can be performed either from your local machine or from the undercloud. The local machine install/check assumes you have ansible installed already.

Before running Browbeat

- Execute the ansible/generate_tripleo_hostfile.sh script (builds ssh-config file)
- Configure browbeat-config.yaml to match your tests
- (Optional) Set your Openstack version metadata in metadata/version.json

Currently Keystone Dashboards only depend on osp_series but may be extended to show build date in the future, thus build is also provided but not required. You can add whatever other version related metadata you would like to metadata/version.json. Typically, whatever automation you have to produce builds should provide this file.

What is necessary

- Ansible

  Why? We started with using bash to make changes to the Overcloud, creating complex sed/awks that we get for free with Ansible (for the most part). Other monitoring and stress test tools are installed by the respective playbooks when run.

Install Browbeat from your local machine

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 <undercloud-ip>
$ vi install/group_vars/all.yml # Make sure to edit the dns_server to the correct ip␣
↪address
$ ansible-playbook -i hosts install/browbeat.yml
$ vi install/group_vars/all.yml # Edit Browbeat network settings
$ ansible-playbook -i hosts install/browbeat_network.yml # For external␣
↪access(required to build Shaker image)
$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: `browbeat-network.yml` 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/10/single/networking-guide/

(Optional) Install collectd

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

(Optional) Install collectd->graphite dashboards

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

(Optional) Install connmon

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

Install Browbeat directly on undercloud

From your undercloud

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

```
[stack@ospd ansible]$ ansible-playbook -i hosts install/browbeat.yml
[stack@ospd ansible]$ vi install/group_vars/all.yml # Edit Browbeat network settings
[stack@ospd ansible]$ ansible-playbook -i hosts install/browbeat_network.yml # For␣
↪external access(required to build Shakerimage)
[stack@ospd ansible]$ ansible-playbook -i hosts install/shaker_build.yml
```

Note: `browbeat-network.yml` 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/10/single/networking-guide/

(Optional) Install collectd

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

(Optional) Install collectd->graphite dashboards

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

(Optional) Install connmon

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

Run Overcloud checks

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

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

CHAPTER 3

Usage

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 venv.

Run performance stress tests through Browbeat on the undercloud:

```
$ ssh undercloud-root
[root@ospd ~]# su - stack
[stack@ospd ~]$ screen -S browbeat
[stack@ospd ~]$ . browbeat-venv/bin/activate
(browbeat-venv)[stack@ospd ~]$ cd browbeat/
(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"
```

Run performance stress tests through Browbeat

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

Running PerfKitBenchmarker

Work is on-going to utilize PerfKitBenchmarker as a workload provider to Browbeat. Many benchmarks work out of the box with Browbeat. You must ensure that your network is setup correctly to run those benchmarks and you will need to configure the settings in ansible/install/group_vars/all.yml for Browbeat public/private networks. 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
```

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/browbeat_network.yml
$ 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

venv venv to execute shaker commands in, `/home/stack/shaker-venv` by default

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
```

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.

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 <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.

CHAPTER 4

Plugins

Rally

Context - browbeat_persist_network

This context creates network resources that persist upon completation of a rally run. It is used in conjunction with the nova_boot_persist_with_network plugin scenario. Beware that removal of the network resources created by this context plugin can be a lengthy process so this is best used on "throw-away-test" clouds.

Scenario - nova_boot_persist

This scenario creates instances without a network that persist upon completation 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 compuete 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.

Scenario - nova_boot_persist_with_network

This scenario creates instances that are attached to a network and persist upon completation 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 compuete 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.

CHAPTER 5

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
```

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 6

Indices and tables

- genindex
- modindex
- search