Advanced Openstack Administration

Activity guide

Release L rev39

Component Soft Ltd.

November 20, 2016

Contents

Preface

About command line outputs

Note:
Here we describe the formatting rules of this book. Take the following commands as examples only. These commands not necessary execute correctly or produce the same output for your actual setup.

Example 1

```
root@controller1 (admin) $> nova list --minimal
+--------------------------------------+--------+
| ID                                   | Name   |
+--------------------------------------+--------+
| 58012abf-1b73-40ec-989c-96f4592cd277 | test_1 |
+--------------------------------------+--------+
```

In this case

- The command runs on the `controller` node, as user `root`.
- The keystone credentials for tenant `admin` are loaded into the shell environment.

In the above case it is required to load the admin credentials in order to make the certain command to work properly. In case of the `admin` user (and tenant), it can be done by sourcing the file `/root/keystonerc_admin`

```
root@controller1 $> source /root/keystonerc_admin
root@controller1 (admin) $> env |grep OS_
OS_REGION_NAME=RegionOne
OS_PASSWORD=makeitso
OS_AUTH_URL=http://10.10.10.51:5000/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

Commands not related to OpenStack (like `ls`) does not take care of the OpenStack credentials at all, so for those, it is irrelevant whether you can `OS_` environment variables or not.

Example 2

```
root@controller1 (admin) $> nova interface-list vm1

// Port ID                              | Net ID                                >>
//-------------------------------------+---------------------------------------->>
// 7912e922-e871-4e7a-a943-34017ee29160 | 41f61be7-40b9-4a67-aeca-feae3a5986ac>>
// 93925c7f-0112-493c-8a39-261449128f5f | e3ee2d62-e216-4e76-b438-733e706f1500>>

 IP addresses //
--------------//
 10.40.40.100 //
 10.30.30.102 //
```

In this one, the output is too long, so the first and last columns are cut off (// symbols) and the third column is presented separately (>> symbol)

Example 3

```
root@controller1 (admin)$> nova host-describe compute2.openstack.local
+--------------------------+------------------+-----+-----------+------+
| HOST                     | PROJECT          | cpu | memory_mb | disk |
+--------------------------+------------------+-----+-----------+------+
| compute2.openstack.local | (total)          | 8   | 15948     | 4    |
| compute2.openstack.local | (used_now)       | 4   | 2560      | 4    |
| compute2.openstack.local | (used_max)       | 4   | 2048      | 4    |
| compute2.openstack.local | 0cb8d/--/c274e67 | 4   | 2048      | 4    |
+--------------------------+------------------+-----+-----------+------+
```

In this case the second column PROJECT is truncated to 20 characters, so UUID 0cb8d6ab778546bbadc69488dc274e67 is shortened to 0cb8d/--/c274e67.

Lab 2: Setup centralized logging

Task 1: Reset your installation

The rest of the lab requires you to have your final setup with two controller, network, compute and proxy nodes. Run script `/labfiles/os_nodes rebuild_nodes` to reset and start nodes accordingly.

```
root@lab_machine $> /labfiles/os_nodes rebuild_nodes

Domain compute1 destroyed
Domain controller1 destroyed
Domain network1 destroyed
Domain controller1 started
Domain controller2 started
Domain compute1 started
Domain compute2 started
Domain network1 started
Domain network2 started
Domain proxy1 started
Domain proxy2 started
```

Note:
If you feel confident, that you **have completed** the manual installation in the previous lab, you can also use the installed `controller1` , `network1` and `compute1` nodes for the rest of the exercises. However, you have to modify the configuration on these nodes, since we make all REST API endpoints as well as the proxy to be accessed through the proxies. Use the Ansible playbook `use_proxy.yml` to

- Update the Keystone endpoint URL for each service
- Update the MySQL connection string for services
- Update the Keystone catalog

```
root@lab_machine $> cd /labfiles/ansible
root@lab_machine $>  ansible-playbook -i os_hosts use_proxy.yml  -t openrc \
> -l '10.10.10.51:10.10.10.52:10.10.10.53'
```

You can start the remaining set of nodes after that.

```
root@lab_machine $> for i in {controller,network,compute}2 proxy{1,2}; do
> /labfiles/os_nodes start $i; done
```

Task 2: Pre-flight check

- Login to the **controller1**, and check the health of your OpenStack installation.
  - Check Nova services (`nova service-list`)

```
root@controller1 $> . /root/keystonerc_admin
root@controller1 (admin) $> nova service-list
//------------------+-----------------------------+----------+---------+-------//
// Binary           | Host                        | Zone     | Status  | State //
//------------------+-----------------------------+----------+---------+-------//
// nova-consoleauth | controller1.openstack.local | internal | enabled | up    //
// nova-scheduler   | controller1.openstack.local | internal | enabled | up    //
// nova-conductor   | controller1.openstack.local | internal | enabled | up    //
// nova-cert        | controller1.openstack.local | internal | enabled | up    //
// nova-compute     | compute1.openstack.local    | nova     | enabled | up    //
// nova-compute     | compute2.openstack.local    | nova     | enabled | up    //
// nova-conductor   | controller2.openstack.local | internal | enabled | down  //
// nova-consoleauth | controller2.openstack.local | internal | enabled | down  //
// nova-scheduler   | controller2.openstack.local | internal | enabled | down  //
// nova-cert        | controller2.openstack.local | internal | enabled | down  //
//------------------+-----------------------------+----------+---------+-------//
```

Note:
Node `controller2` will be started in the next module of this training.

- Check Cinder services with `cinder service-list`.

```
root@controller1 (admin) $> cinder service-list
+------------------+-------------------------------+------+---------+-------//
|      Binary      |              Host             | Zone |  Status | State //
+------------------+-------------------------------+------+---------+-------//
| cinder-scheduler |   controller1.openstack.local | nova | enabled |   up  //
| cinder-scheduler |   controller2.openstack.local | nova | enabled |  down //
|  cinder-volume   | controller1.openstack.local@lvm | nova | enabled |   up  //
|  cinder-volume   | controller2.openstack.local@lvm | nova | enabled |  down //
+------------------+-------------------------------+------+---------+-------//
```

- Check Neutron services (`neutron agent-list`)

```
root@controller1 (admin) $> neutron agent-list
//--------------------+-----------------------------+-------//
// agent_type         | host                        | alive //
//--------------------+-----------------------------+-------//
// Open vSwitch agent | compute2.openstack.local    | :-)   //
// Metering agent     | network1.openstack.local    | :-)   //
// Open vSwitch agent | network1.openstack.local    | :-)   //
// Metadata agent     | network1.openstack.local    | :-)   //
// Open vSwitch agent | network2.openstack.local    | :-)   //
// Open vSwitch agent | compute1.openstack.local    | :-)   //
// Metering agent     | network2.openstack.local    | :-)   //
// Loadbalancer agent | network2.openstack.local    | :-)   //
// DHCP agent         | network1.openstack.local    | :-)   //
```



```
// Loadbalancer agent | network1.openstack.local | :-)   //
// Metadata agent     | network2.openstack.local | :-)   //
// L3 agent           | network2.openstack.local | :-)   //
// L3 agent           | network1.openstack.local | :-)   //
// DHCP agent         | network2.openstack.local | :-)   //
//--------------------+--------------------------+-------//
```

- Create Heat stack named `base` from template `/labfiles/lab2/base_infrastructure.yml`.

```
root@controller1 (admin) $> heat stack-create \
> -f /labfiles/lab2/base_infrastructure.yml base
+--------------------------------------+------------+--------------------//
| id                                   | stack_name | stack_status       //
+--------------------------------------+------------+--------------------//
| 01546ac0-08ca-4b61-908c-3dbe3e62fb88 | base       | CREATE_IN_PROGRESS //
+--------------------------------------+------------+--------------------//
```

```
root@controller1 (admin) $> heat resource-list base
+---------------//--------------------+-----------------//
| resource_name // resource_type       | resource_status //
+---------------//--------------------+-----------------//
| CirrOS        // OS::Glance::Image   | CREATE_COMPLETE //
| guest-key     // OS::Nova::KeyPair   | CREATE_COMPLETE //
| net_ext       // OS::Neutron::Net    | CREATE_COMPLETE //
| subnet_ext    // OS::Neutron::Subnet | CREATE_COMPLETE //
+---------------//--------------------+-----------------//
```

- Create a test VM instance using the Heat template `/labfiles/lab2/instance.yml`.

```
root@controller1 (admin) $> heat stack-create \
> -f /labfiles/lab2/instance.yml lab1
+--------------------------------------+------------+--------------------//
| id                                   | stack_name | stack_status       //
+--------------------------------------+------------+--------------------//
| 01546ac0-08ca-4b61-908c-3dbe3e62fb88 | base       | CREATE_COMPLETE    //
| c817b254-c85d-44e8-97dd-edac8679889f | lab1       | CREATE_IN_PROGRESS //
+--------------------------------------+------------+--------------------//
```

```
root@controller1 (admin) $> heat resource-list lab1
+---------------------//-------------------------------+-----------------//
| resource_name       // resource_type                  | resource_status //
+---------------------//-------------------------------+-----------------//
| fl_ip1              // OS::Neutron::FloatingIP        | CREATE_COMPLETE //
| int_gre1_router     // OS::Neutron::Router            | CREATE_COMPLETE //
| int_router_interface // OS::Neutron::RouterInterface | CREATE_COMPLETE //
| net_int             // OS::Neutron::Net               | CREATE_COMPLETE //
| port1               // OS::Neutron::Port              | CREATE_COMPLETE //
| subnet_int          // OS::Neutron::Subnet            | CREATE_COMPLETE //
| test_1              // OS::Nova::Server               | CREATE_COMPLETE //
+---------------------//-------------------------------+-----------------//
```

```
root@controller1 (admin) $> nova list
+--------------------------------------+--------+--------+------------>>
| ID                                   | Name   | Status | Task State >>
+--------------------------------------+--------+--------+------------>>
| d9fd57b5-90f3-4102-87a9-88ab3e8debd8 | lab1   | ACTIVE | -          >>
+--------------------------------------+--------+--------+------------>>
```



```
+-------------+----------------------------------------+
| Power State | Networks                               |
+-------------+----------------------------------------+
| Running     | int_gre1=10.30.30.101, 192.168.100.101 |
+-------------+----------------------------------------+
```

```
root@controller1 (admin) $> nova console-log --length 10 lab1
=== cirros: current=0.3.2 uptime=2.63 ===
  ____               ____  ____
 / __/ __ ____ ____ / __ \/ __/
/ /__ / // __// __// /_/ /\ \
\___//_//_/  /_/   \____/___/
   http://cirros-cloud.net


login as 'cirros' user. default password: 'cubswin:)' or root/makeitso.
lab1 login:
```

- Delete stack `lab1` but keep the stack `base` running.

```
root@controller1 (admin) $> heat stack-delete lab1
+--------------------------------------+------------+--------------------//
| id                                   | stack_name | stack_status       //
+--------------------------------------+------------+--------------------//
| 3ba91888-2b87-45db-be22-4b6a33ce14bb | lab1       | DELETE_IN_PROGRESS //
+--------------------------------------+------------+--------------------//
```

```
root@controller1 (admin) $> heat stack-list
+--------------------------------------+------------+--------------------//
| id                                   | stack_name | stack_status       //
+--------------------------------------+------------+--------------------//
| 01546ac0-08ca-4b61-908c-3dbe3e62fb88 | base       | CREATE_COMPLETE    //
+--------------------------------------+------------+--------------------//
```

Task 3: Configure Openstack nodes with FileBeat

- Start node `monitor` node.

```
root@lab_machine $> /labfiles/os_nodes start monitor
Domain monitor started
```

Note:
In your setup, the monitor node is preconfigured with Elasticsearch/Kibana. Check Appendix 1 for details.

The following steps sets up FileBeat for `controller1`.

- Setup a yum repo `/etc/yum.repos.d/beats.repo`

```
[beats]
name=Elastic Beats Repository
baseurl=https://packages.elastic.co/beats/yum/el/$basearch
```



```
enabled=1
gpgkey=https://packages.elastic.co/GPG-KEY-elasticsearch
gpgcheck=1
```

- Install `filebeat`

```
root@controller1 $> yum install -y filebeat
```

- Configure `/etc/filebeat/filebeat.yml` to send log entries to LogStash

```
filebeat:
 prospectors:
    - paths:
        - "/var/log/neutron/*.log"
      fields:
         service: neutron
    - paths:
        - "/var/log/glance/*.log"
      fields:
         service: glance
    - paths:
        - "/var/log/cinder/*.log"
      fields:
         service: cinder
    - paths:
        - "/var/log/nova/*.log"
      fields:
         service: nova
    - paths:
        - "/var/log/heat/*.log"
      fields:
         service: heat
    - paths:
        - "/var/log/horizon/*.log"
      fields:
         service: horizon
    - paths:
        - "/var/log/keystone/*.log"
      fields:
         service: keystone
    - paths:
        - "/var/log/swift/*.log"
      fields:
         service: swift
    - paths:
        - "/var/log/mariadb/*.log"
      fields:
         service: mysql
    - paths:
        - "/var/log/ceilometer/*.log"
      fields:
         service: ceilometer
shipper:
  tags: ["controller"]

output:
  logstash:
```



```
    hosts: ["10.10.10.110:5044"]
```

Note:
You can find the above config file at `/labfiles/lab2/filebeat.yml`.

Note:
In production, both FileBeat and LogStash has to use SSL. Check the example config and the configuration reference for the complete set of options.

- Validate the FileBeat configuration

```
root@controller1 $> filebeat -c /etc/filebeat/filebeat.conf --configtest
```

- Rotate logs the avoid transferring all previous log entries into LogStash

```
$> logrotate  -f  /etc/logrotate.conf
```

- Start service `filebeat`

```
root@controller1 $> systemctl enable filebeat
root@controller1 $> systemctl start filebeat
root@controller1 $> systemctl status filebeat
filebeat.service - LSB: Sends log files to Logstash or directly to Elasticsearch.
   Loaded: loaded (/etc/rc.d/init.d/filebeat)
   Active: active (running) since Tue 2015-11-24 09:56:50 UTC; 6s ago
```

- On the `monitor` node, verify that new data has arrived into ElasticSearch.

```
root@monitor $> curl 10.10.10.110:9200/_cat/indices
yellow open .kibana              1 1    93 3  76.6kb  76.6kb
yellow open filebeat-2015.11.24 5 1  2337 0 822.8kb 822.8kb
```

- Use the `/labfiles/ansible/filebeat.yml` Ansible playbook on `lab_machine` to deploy `filebeat` to the rest of the nodes

```
root@lab_machine $> ansible-playbook -i /labfiles/ansible/os_hosts \
>/labfiles/ansible/filebeat.yml
```

Task 4: Using Kibana

- Load sample dashboards


```
root@monitor $> cd
root@monitor $> curl -L -O \
> http://download.elastic.co/beats/dashboards/beats-dashboards-1.0.0.tar.gz
root@monitor $> tar -zxvf beats-dashboards-1.0.0.tar.gz
root@monitor $> cd ./beats-dashboards-1.0.0/
root@monitor $> ./load.sh 10.10.10.110:9200
```

- Visit http://10.10.10.110:5601 to check logs and metrics in Kibana

Note:
For remote lab use SSH redirection at http://localhost:35601, and ensure that you have at least one active connection to the `lab_machine`.

- Start a new VM instance using Heat template `/labfiles/lab2/instance.yml`, and search for its UUID in Kibana.

```
root@controller1 (admin) $> heat stack-create -f /labfiles/lab2/instance.yml lab1
+--------------------------------------+------------+--------------------//
| id                                   | stack_name | stack_status       //
+--------------------------------------+------------+--------------------//
...
| 50792edd-281f-49e3-b339-28feeaeaf6ef | lab1       | CREATE_IN_PROGRESS //
+--------------------------------------+------------+--------------------//
```



```
root@controller1 (admin) $> heat resource-list lab1|grep -i server
| test_1   | 0c1c5b3b-979b-4d9a-ac16-0f8a26d31be0   | OS::Nova::Server                 //
```

- Check logs for the last 15 minutes
- Add fields `loglevel` and `message` to the listing.
- Search for the UUID and filter out event from `ceilometer`.

The output should look like as follows: