From How-To to POC to Production: Learning by Building

Presented By: Grant Kirkwood, CTO – Unitas Global

March 4, 2017 – SCALE 15x – Pasadena, Calif
WHAT IS OPENSTACK?

- “Science experiment”
- “A mess of competing projects and priorities”
- “De facto choice for new private clouds”
ONE DOES NOT SIMPLY
LEARN OPENSTACK
FROM HOW-TO: TO POC: TO PRODUCTION!

LEARNING BY BUILDING...
STEP 1:
WHAT IS IT?

- Scripted install
- All-in-one
- Up and running in minutes
DEVSTACK

WHAT YOU NEED

• Laptop!
• VirtualBox
• Ubuntu
• DevStack
DOWNLOAD

• VirtualBox:
  https://www.virtualbox.org/wiki/Downloads

• Ubuntu:
  http://releases.ubuntu.com/16.04/ubuntu-16.04.10-server-amd64.iso
INSTRUCTIONS

• Install VirtualBox
• Create an Ubuntu VM
• Install DevStack

• Help found here:
  http://ronaldbradford.com/blog/downloading-and-installing-devstack-2016-04-02/
Downloading and installing devstack

April 2, 2016 by ronald

The following instructions assume you have a running Linux virtual machine that can support the installation of devstack to demonstrate a simple working OpenStack cloud.

For more information about the preparation needed for this step, see these pre-requisite instructions:

- installing VirtualBox
- Setting up an Ubuntu virtual machine using VirtualBox or Setting up Ubuntu using vagrant

Pre-requisites

You will need to login to your Linux virtual machine as a normal user (e.g. stack if you followed these instructions).

To verify the IP address of your machine you can run:

```bash
$ ifconfig eth1
```

NOTE: This assumes you configured a second network adapter as detailed.

You need to determine the IP address assigned. If this is your first-time using VirtualBox and this was configured with default settings, the value will be 192.168.56.101

```
eth1 Link encap:Ethernet  HWaddr 00:00:00:00:00:00
    inet addr:192.168.56.101  Bcast:192.168.36.255  Mask:255.255.255.0
    inet6 addr: fe00::0/64 Scope:Link
     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:396500 errors:0 dropped:0 overruns:0 frame:0
     TX packets:282820 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0  txqueuelen:0
```

http://ronaldbradford.com/blog/downloading-and-installing-devstack-2016-04-02/

- Step by step instructions here
- Tested…. it works!
- Thanks Ronald Bradford
WHAT WE LEARNED

• CLI basics
• How Horizon (etc) *should* work (and look)
STEP 2:

POÄNG

1. [Diagram of a person thinking]
2. [Diagram of a person building]
3. [Diagram of a person assembling]
4. [Diagram of a person tightening]
5. [Diagram of a person carrying]
6. [Diagram of a completed chair]
WHAT IS IT?

• Step-by-step walkthrough
• “Vanilla” / trunk-based
• Not scripted – each component built by hand
WHAT YOU NEED

- Some hardware ideal
- Intel NUCs are good
- VLAN-capable switch
- Patience!
DOWNLOAD

- Ubuntu:
  http://releases.ubuntu.com/16.04/ubuntu-16.04.10-server-amd64.iso

- Documentation:
  http://docs.openstack.org/mitaka/install-guide-ubuntu/

- A good text editor:
  https://atom.io

- A note on versions…
DOWNLOAD

• Documentation:

1. Disable the keystone service from starting automatically after installation:

   ```
   # echo "manual" > /etc/init/keystone.override
   ```

2. Run the following command to install the packages:

   ```
   # apt-get install keystone apache2 libapache2-mod-wsgi
   ```

3. Edit the `/etc/keystone/keystone.conf` file and complete the following actions:

   - In the `[DEFAULT]` section, define the value of the initial administration token:
     ```
     [DEFAULT]
     ...
     admin_token = ADMIN_TOKEN
     ```

     Replace `ADMIN_TOKEN` with the random value that you generated in a previous step.

   - In the `[database]` section, configure database access:
     ```
     [database]
     ...
     connection = mysql+pymysql://keystone:KEystone_DBPASS@controller/keystone
     ```
HOW-TO

DOCUMENTING

• Why the good text editor?
• Documenting your work critical
• You WILL do things wrong
• You WILL want to do things differently next time around… yes next time
DOCUMENTING

• Leave yourself some notes!
• Will save you grief later…
• And helps with learning
• …AND will need for HA
INSTRUCTIONS

• Internet (2 networks)
• Switch
• 3+ Servers
• Architecture decisions in official how-to doc
• Use provider networks with self-service option
INSTRUCTIONS

• Initial network setup
• Translate to home network…

http://docs.openstack.org/mitaka/install-guide-ubuntu/environment-networking.html
HOW-TO

TOPOLOGY

- 192.168.10.0/24
  - Cut it in half…
- 10.0.10.0/24
Core Components

- Contents from How-To Tutorial:
- Essential: Environment, Identity, Image, Compute, Networking, Dashboard, Block Storage
OPENSTACK TIME MACHINE
LIFTOFF

• Time to launch your first VM...
• Good chance it won’t work
• Where to look for clues
• Remember Devstack…

Contents

• Conventions
• Overview
  ◦ Example architecture
  ◦ Networking
• Environment
• Identity service
• Image service
• Compute service
• Networking service
• Dashboard
• Block Storage service
• Shared File Systems service
• Object Storage service
• Orchestration service
• Telemetry service
• Database service
• Launch an instance
• Community support
• Glossary
WHEN YOU (SUCCESSFULLY) LAUNCH YOUR FIRST VM...

LOOKS LIKE THIS:

FEELS LIKE THIS:

It's alive! It's alive!
WHAT WE LEARNED

• OpenStack under the hood
• How all the components work together
• Where to look for clues when something goes wrong
• CLI in-depth
SHORTCOMINGSS

- Not designed to scale
- Not optimized for performance
- Not built highly available
- Wouldn’t use this in production
NEXT STEPS

• Tear it down and start over! 😊
• Do it from your notes this time (how-to phone-a-friend)
• Fix the things you want to improve
• Make sure documentation complete – will need for POC
NOT QUITE READY FOR POC...

• A proof of concept has to be built with business use in mind
• That means we have a bunch of new stuff to learn
SCALABILITY & RELIABILITY

• Now the fun begins…
• First, time to upgrade storage: Ceph!
• Start by changing Glance to a Ceph backend
• Keep it simple – only change one thing at a time!
• Then change Cinder to use Ceph for volumes
SCALABILITY & RELIABILITY

- Building for high availability: wrapping services in more services… service burritos?
- What components do we use? Where do they fit?
- Can’t really build on top of the existing config
- This is just one way to do it…
HIGH AVAILABILITY
HIGH AVAILABILITY

MariaDB

RabbitMQ
HIGH AVAILABILITY

MARIA DB

RABBIT MQ

PRIVATE VLAN FOR
MARIA DB

PRIVATE VLAN FOR
RABBIT MQ

GALERA

MARIA DB

GALERA

RABBIT MQ
HIGH AVAILABILITY

Private VLAN for MariaDB

MariaDB

RabbitMQ

Private VLAN for RabbitMQ

Galera

MariaDB

RabbitMQ

Linux Bridge network

VIP

HAPROXY

Keepalived

Linux Bridge network

VIP

Private VLAN for OpenStack Management
SCALABILITY & RELIABILITY

- Ok, so that’s the controllers!
- Now that we know how it fits together…
- Back to your build notes!
SCALABILITY & RELIABILITY

• Notable changes – service endpoints
• Update hosts file with pointer to HA endpoint
• …which means build the service endpoints to IPs that don’t initially exist!
• Update haproxy as you go along
SCALABILITY & RELIABILITY

- Example:

```bash
# NOTE: need to update things below with -ha version
openstack service create --name keystore --description "OpenStack Identity" id
openstack endpoint create --publicurl http://keystone-ha:5000/v2.0 --internalurl...
openstack project create --description "Admin Project" admin
openstack user create --password-prompt admin
openstack role create admin
openstack role add --project admin --user admin admin
openstack project create --description "Service Project" service
#openstack project create --description "Demo Project" demo
```
TESTING

• Break it!
• Looping VM launching
• Power cycle controllers mid-cycle
• Haproxy stats helpful
• Tempest, Rally, etc.
# How-to

## Statistics Report for pid 3740 on node1

### General process information

- **pid** = 3740 (process #4, nproc = 4)
- **uptime** = 1844 23h01m26s
- **system limits:** memmax = unlimited; ulimit-n = 8075
- **maxsock = 8075, maxconn = 4000, maxpipes = 0**
- Current conns = 295; current pipes = 0; conn rate = 0/sec
- Running tasks: 1/334; idle = 100%

### Stats

<table>
<thead>
<tr>
<th>Queue</th>
<th>Session rate</th>
<th>Sessions</th>
<th>Bytes</th>
<th>Denied</th>
<th>Errors</th>
<th>WARNs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
</tr>
<tr>
<td>Frontend</td>
<td>0</td>
<td>4</td>
<td>-</td>
<td>1</td>
<td>3</td>
<td>4000</td>
</tr>
<tr>
<td>Backend</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>400</td>
</tr>
</tbody>
</table>

### db

<table>
<thead>
<tr>
<th>Queue</th>
<th>Session rate</th>
<th>Sessions</th>
<th>Bytes</th>
<th>Denied</th>
<th>Errors</th>
<th>WARNs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
</tr>
<tr>
<td>Frontend</td>
<td>0</td>
<td>33</td>
<td>-</td>
<td>0</td>
<td>291</td>
<td>400</td>
</tr>
<tr>
<td>node1</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>33</td>
<td>291</td>
</tr>
<tr>
<td>node2</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>node3</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Backend</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>291</td>
<td>400</td>
</tr>
</tbody>
</table>

### keystone_admin

<table>
<thead>
<tr>
<th>Queue</th>
<th>Session rate</th>
<th>Sessions</th>
<th>Bytes</th>
<th>Denied</th>
<th>Errors</th>
<th>WARNs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
</tr>
<tr>
<td>Frontend</td>
<td>0</td>
<td>11</td>
<td>-</td>
<td>1</td>
<td>51</td>
<td>10000</td>
</tr>
<tr>
<td>node1</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>node2</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>node3</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Backend</td>
<td>0</td>
<td>0</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

### keystone_api

<table>
<thead>
<tr>
<th>Queue</th>
<th>Session rate</th>
<th>Sessions</th>
<th>Bytes</th>
<th>Denied</th>
<th>Errors</th>
<th>WARNs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
<td>Cur</td>
<td>Max</td>
<td>Limit</td>
</tr>
<tr>
<td>Frontend</td>
<td>0</td>
<td>11</td>
<td>-</td>
<td>0</td>
<td>8</td>
<td>10000</td>
</tr>
<tr>
<td>node1</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>node2</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>node3</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Backend</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>7</td>
<td>10000</td>
</tr>
</tbody>
</table>
STEP 3:

YOU WANT PROOF?
I'LL GIVE YOU PROOF!
NOW THE FUN BEGINS

• You know how it all works now… but…
• Throw everything you just learned out the window!
QUESTION:

• Are you in the business of building and operating technology infrastructure?
PROOF OF CONCEPT

DECISIONS

• Are we building for a business requirement?
• Must assume POC will be used
  (after all, if not, it wasn’t a successful POC!)
• Therefore, build as if for production
• Don’t want to be doing this by hand…
PROOF OF CONCEPT

DECISIONS

• Deployment method
• Distribution?
  (refer to question 1)
• Automation
  ie; OpenStack-Ansible
• Hardware
BUSINESS REQUIREMENTS

- Define business requirements & success criteria
- Keep it simple to start!
- Each service adds complexity, whether or not used
- Conduct surveys: regimented/consistent process for adding new services
PROOF OF CONCEPT

GETTING STARTED

• Build small environment using distribution of choice now that you understand how it works inside (or non-distribution with OpenStack-Ansible)
• Test business applications in environment
PROOF OF CONCEPT

UPDATES & UPGRADES

• Brain surgery…
STEP 4:
DEFINITION

• Supporting business-critical workload
• Downtime not an option
• Tied to revenue (and thus our jobs!)
OPERATIONAL REQUIREMENTS

- Monitoring
- Tracking
- Measuring performance
- Updates & upgrades
AUTOMATION

• Much of what you did is automated with major distributions
• Still critical to understand how it all works
• Serves as foundation for production grade operational excellence
LEARNING BY BUILDING
ROADMAP / SUMMARY

Testing the waters with DevStack

• VirtualBox and a laptop
• Up and running in minutes
• Get comfortable with how it should work

Build with How-To

• Build it by hand with openstack.org tutorial
• Learn how the pieces work together
• Document and take notes like crazy!

High availability and scalability

• Upgrade storage to Ceph
• Make all services HA
• Try to break it

Proof of Concept

• Define business objectives
• Desire to build/operate or consume?
• Switch to distribution
• Keep it simple, add services slowly/thoughtfully
• Test, break, break again

Production

• Monitoring
• Tracking
• Updates & upgrades
Contact Us

Presenter
Grant Kirkwood - CTO, Unitas Global
grant.kirkwood@unitasglobal.com
IRC: tnarg
Twitter: @prolixity

Visit Us!
Booth #610 (on the left when you walk in – next to Disney)