Tales From The Gate
How Debugging The Gate Helps Your Enterprise

Matthew Treinish (irc: mtreinish)
Matt Riedemann (irc: mriedem)
Sean Dague (irc: sdague)

August 18, 2015
What is “The Gate”? 

- Colloquialism for OpenStack’s pre-merge continuous integration (CI) system.
- The jobs run can be different between projects.
- Can be thought of as a reference configuration.
- Hosted on community infrastructure.
- We gate on unit test jobs but the majority of testing happens with integrated testing using devstack + Tempest.
- There are multiple queues (check, gate, experimental, periodic).
What happens when you submit code?
CI Workflow

Upstream Code

nova / master

Zuul / Jenkins (automated testing)

gate queue

Gerrit (Review System)

Review Requirements:
Core Reviews: 2 +2, 0 -2
Jenkins: +1

Your Local Environment

nova / master

git clone

git branch fix_bug_foo

Fix changes
Run unit tests

git commit

nova / fix_bug_foo

git commit --amend

nova / fix_bug_foo

git review

check queue
Gate Scale

- >80M tempest tests run in gate queue during kilo
- Each proposed patch spins up between 4 and 20 devstack environments for running tests
- Each tempest run starts ~130 guests in the devstack environment
- ~1.73% run failure rate
- ~.019% individual test failure rate
What could possibly go wrong...

- Dozens of jobs with different configurations and multiple services (and multiple API versions) running together.
- Often race failures occur at a small frequency so they sometimes are not caught on gating jobs for the change which introduced them.
- Don’t forget that dependent libraries have race bugs also, e.g. libvirt/qemu.
Types of failures

**Upstream Service Breaks**
- Examples:
  - pypi bad cert
  - github outages
  - iaas dns blacklisting
  - iaas provider network
  - upstream apt mirrors
- Assume touching network is poison, cache or bring resources local

**Infra Breaks**
- Examples:
  - bad nodepool images
  - service outages
  - mirrors broken
- Fixes:
  - Make infra more resilient and self healing

**Bugs in OpenStack**
- Examples:
  - state corruption
  - races w/ async mess aging
  - races w/ multiple workers
  - db deadlocks
- Fixes:
  - Ferret out races in the code

**Bugs in Tests**
- Examples:
  - Unsafe global state expectations
  - Comparing timestamps
- Fixes:
  - Fix the tests

**Bugs in Dependencies**
- Examples:
  - kernel nbd vs. ovswitch
  - libvirt wedging
  - httplib2
- Fixes:
  - Get bug reported upstream, try to provide work around for buggy versions in OpenStack
Configuration Differences

- Database
- Storage
- Networking
- Miscellaneous
  - Upgrade
  - Large Ops
  - Multi-node
Devstack + Tempest

MySQL
  Also includes:
  - Force config drive
  - Keystone in Apache

PostgreSQL
  Also includes:
  - Metadata service
  - Keystone w/ eventlet

Nova Network

Neutron

Ceph

LVM

Grenade
  - Full
  - Partial-ncpu

nova network

neutron

Large Ops

Multi-node
What could possibly go wrong...

- Running $ncpu workers on multiple projects at once in a single-node devstack causing out-of-memory errors. We found out that is not a sane default. (Bug: 1366931)

- LVM operations locking up for over 60 seconds within a synchronized call causing RPC timeouts. (Bug: 1373513)

- nbd kernel panic with network namespaces (Bug: 1273386)

- Resize/restart with neutron breaks connectivity (Bug: 1323658 current gate failure with real world examples)
Debugging

- So Jenkins is unhappy, let’s check the gate-tempest-dsvm-full job.

<table>
<thead>
<tr>
<th>Jenkins gate</th>
<th>May 15, 2015 10:27 AM</th>
</tr>
</thead>
<tbody>
<tr>
<td>gate-horizon-docs</td>
<td>SUCCESS in 4m 38s</td>
</tr>
<tr>
<td>gate-horizon-pep8</td>
<td>SUCCESS in 3m 03s</td>
</tr>
<tr>
<td>gate-horizon-python27</td>
<td>SUCCESS in 5m 24s</td>
</tr>
<tr>
<td>gate-horizon-python27-django14</td>
<td>SUCCESS in 6m 52s</td>
</tr>
<tr>
<td>gate-horizon-selenium</td>
<td>SUCCESS in 3m 38s</td>
</tr>
<tr>
<td>gate-horizon-dsvm-integration</td>
<td>FAILURE in 42m 36s (non-voting)</td>
</tr>
<tr>
<td>gate-tempest-dsvm-full</td>
<td>FAILURE in 47m 07s</td>
</tr>
<tr>
<td>gate-grenade-dsvm</td>
<td>SUCCESS in 37m 33s</td>
</tr>
</tbody>
</table>
Debugging

- Start with the console log to see which test(s) failed so we know which service logs to check. Note: tempest timeouts are tricky.
  - `tempest.api.compute.servers.test_delete_server.DeleteServersTestJSON.test_delete_server_while_in_verify_resize_state [119.765416s] ... FAILED`
  - `tempest.exceptions.BuildErrorException: Server e79e417a-885b-4468-b3d0-cf52e1a0af90 failed to build and is in ERROR status`
  - Details: `{u'code': 500, u'message': u'No valid host was found. There are not enough hosts available.', u'created': u'2015-05-15T15:05:54Z'}`
Debugging

- Failed to build a server so let’s check the nova compute logs.

```
[instance: e79e417a-885b-4468-b3d0-cf52e1a0af90] Instance failed to spawn
Traceback (most recent call last):
  File "/opt/stack/new/nova/nova/compute/manager.py", line 2442, in _build_resources
    yield resources
  File "/opt/stack/new/nova/nova/compute/manager.py", line 2314, in _build_and_run_instance
    block_device_info, block_device_info
  File "/opt/stack/new/nova/virt/libvirt/driver.py", line 2354, in spawn
    block_device_info, block_device_info
  File "/opt/stack/new/nova/virt/libvirt/driver.py", line 4391, in _create_domain_and_network_power_on_power_on
  File "/opt/stack/new/nova/virt/libvirt/driver.py", line 4322, in _create_domain
LOG.error(err)
File "/usr/local/lib/python2.7/dist-packages/oslo_utils/executils.py", line 85, in __exit__
  six.reraise(type_, value, traceback)
File "/opt/stack/new/nova/virt/libvirt/driver.py", line 4306, in _create_domain
  domain = self._conn.defineXML(xml)
File "/usr/local/lib/python2.7/dist-packages/eventlet/tpool.py", line 183, in do
  result = proxy_call(self._autowrap, f, *args, **kwargs)
File "/usr/local/lib/python2.7/dist-packages/eventlet/tpool.py", line 141, in proxy_call
  rv = execute(f, *args, **kwargs)
File "/usr/local/lib/python2.7/dist-packages/eventlet/tpool.py", line 122, in execute
  six.reraise(c, e, tb)
File "/usr/local/lib/python2.7/dist-packages/libvirt.py", line 80, in tworker
  rv = meth(*args, **kwargs)
File "/usr/local/lib/python2.7/dist-packages/libvirt.py", line 3263, in defineXML
  if ret is None: raise libvirtError('virDomainDefineXML() failed', conn=self)
libvirtError: internal error: received hangup / error event on socket
```
Debugging

- We found an error so run it through logstash to see if it’s hitting on multiple changes, especially in the gate queue. < 10 days is key.

- Check launchpad for a previously reported bug. If not found, create a new one. (Bug: 1353939)
Debugging

- Push a query to elastic-recheck for tracking.

**Bug 1353939 - Rescue fails with 'Failed to terminate process: Device or resource busy' in the n-cpu log**

- 41 fails in 24 hrs / 293 fails in 10 days
- Projects: (nova - In Progress)

**Open Reviews:**

![Graph showing failure and success over time with labels Logstash and Launchpad]
Debugging

- elastic-recheck is a project that uses Elasticsearch to check Jenkins (voting) job failures against indexed job logs in logstash.openstack.org.

- Uses fingerprints for known race bugs to classify the failure.

- Comments on changes in Gerrit when tests fail for known bugs:

  Elastic Recheck
  Patch Set 6:

  I noticed jenkins failed, I think you hit bug(s):

  - check-grenade-dsvm-partial-ncpu: unrecognized error
  - gate-tempest-dsvm-large-ops: https://bugs.launchpad.net/bugs/1311066 https://bugs.launchpad.net/bugs/1288618
  - gate-tempest-dsvm-neutron-large-ops: https://bugs.launchpad.net/bugs/1311066

  Some of the tests failed in a way that we did not understand. Please help us classify these issues so that they can be part of Elastic Recheck http://status.openstack.org/elastic-recheck/ For more details on this and other bugs, please see http://status.openstack.org/elastic-recheck/
Debugging

- [http://status.openstack.org/elastic-recheck/data/uncategorized.html](http://status.openstack.org/elastic-recheck/data/uncategorized.html)
Lessons Learned

● We need sane defaults given the configuration nightmare.

● Just rechecking without looking at failures causes more issues long term.

● Keeping stable branches stable is hard but is important for end consumers/deployers/operators that are not doing continuous deployment from trunk.

● Adequate logging is critical for post-mortem analysis. Projects should be following the logging guidelines.

● We should fix code rather than devstack and at least document warnings/workarounds in release notes for config/deploy.
Where to get more information

● #openstack-qa channel on Freenode IRC

● openstack-dev mailing list: http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack-dev

● http://status.openstack.org/elastic-recheck/

● OpenStack Bootstrapping Hour session on debugging the gate: https://www.youtube.com/watch?v=fowBDdLGBIU

● Infra presentations: http://docs.openstack.org/infra/publications/
Questions?