

Better Testing Through Statistics

Matthew Treinish

mtreinish@kortar.org

mtreinish on Freenode

Masayuki Igawa

masayuki.igawa@gmail.com

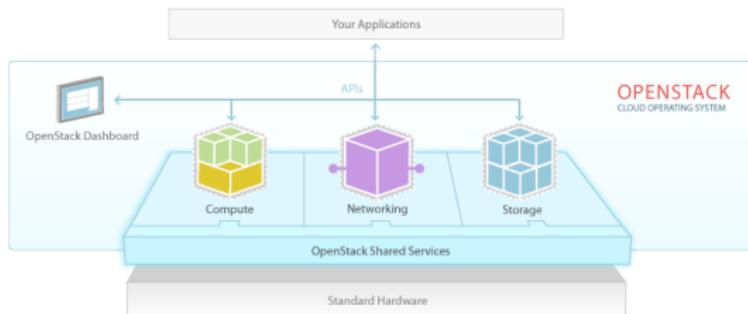
masayukig on Freenode

July 13, 2016

<https://github.com/masayukig/better-testing-through-statistics>

What is “the OpenStack”?

- ▶ Open Source Cloud Software: Apache License Version 2.0
- ▶ consists of a lot of projects: **57 projects**
- ▶ released every 6 month: Latest version is called ‘Mitaka’



What is “the OpenStack QA”?

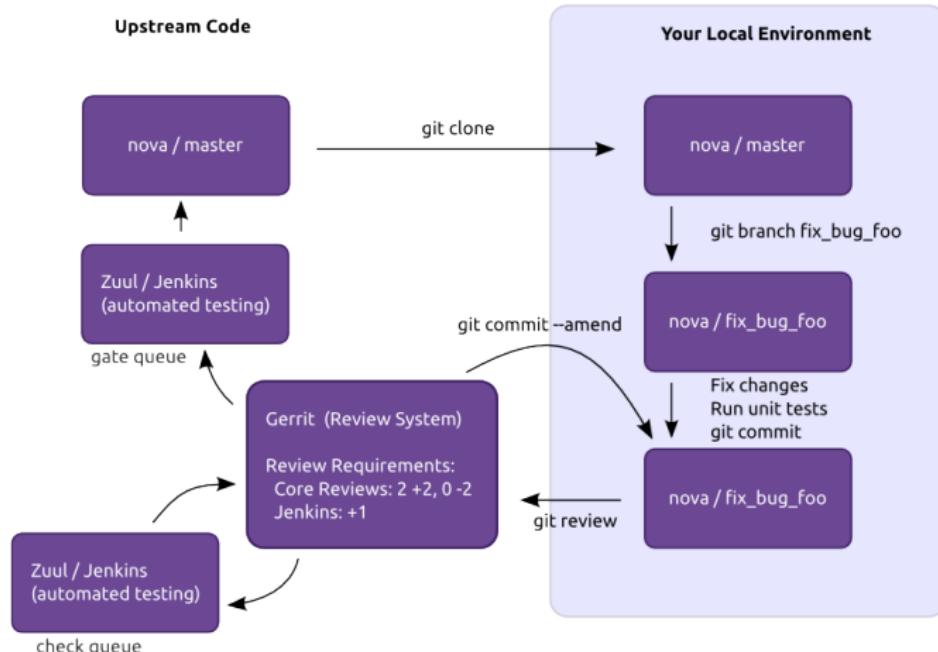
- ▶ An official OpenStack project team
- ▶ Develop, maintain, and initiate tools and plans to ensure the upstream stability and quality of OpenStack, and its release readiness at any point during the release cycle.

Current QA Projects

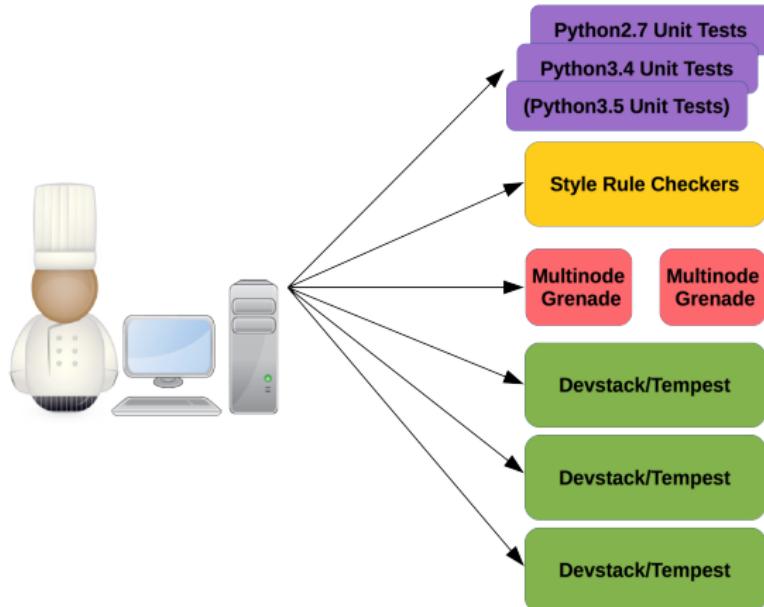
17 repositories (2016/7/8)

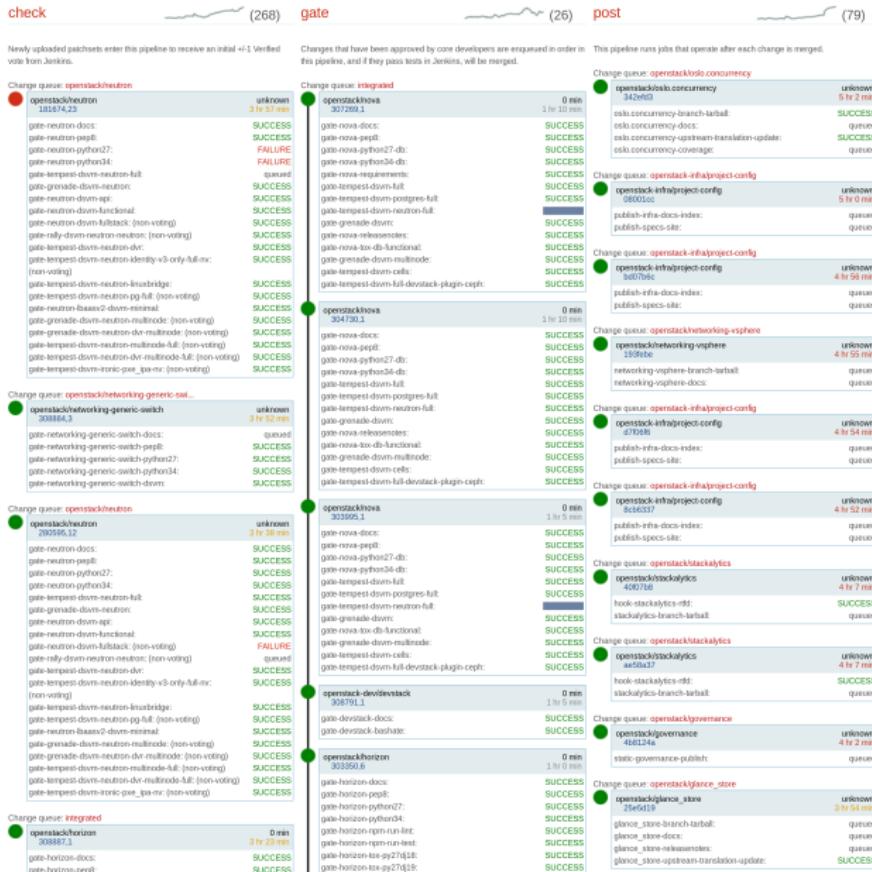
- ▶ devstack
- ▶ devstack-plugin-cookiecutter
- ▶ devstack-plugin-ceph
- ▶ devstack-vagrant
- ▶ grenade
- ▶ tempest
- ▶ tempest-lib
- ▶ tempest-plugin-cookiecutter
- ▶ bashate
- ▶ stackviz
- ▶ hacking
- ▶ eslint-config-openstack
- ▶ os-testr
- ▶ os-performance-tools
- ▶ openstack-health-dashboard
- ▶ karma-subunit-reporter

What is “the OpenStack Gate”?



What Happens when you push a change?





The Size of the Gate

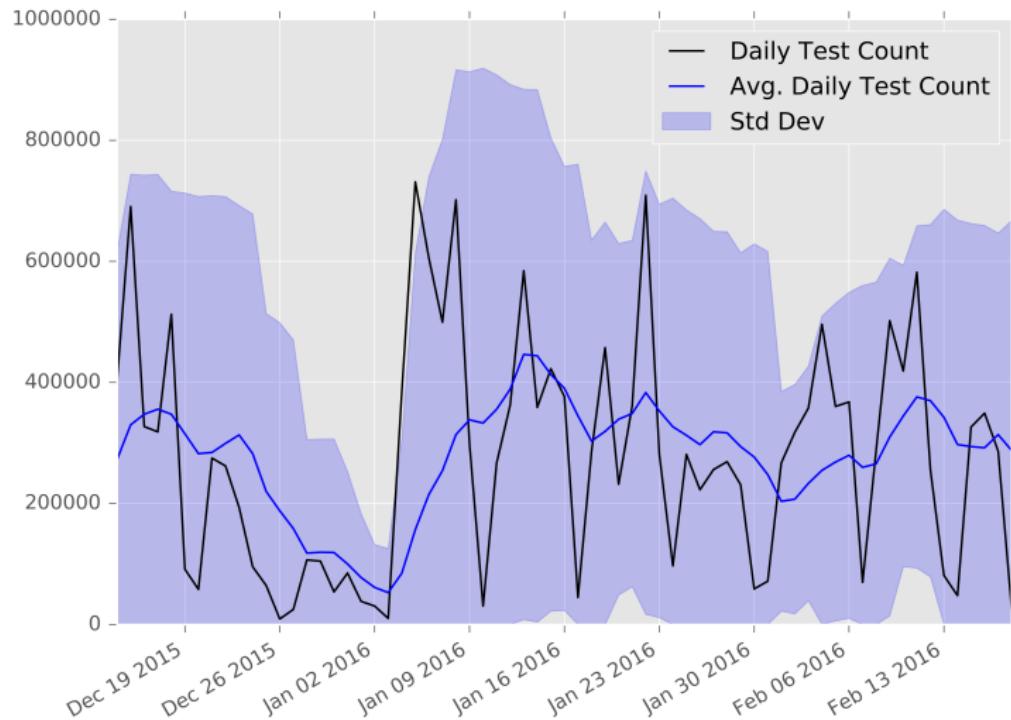
One Proposed Change Generates:

- ▶ 5–25 Devstacks
- ▶ ~10,000 integration tests (roughly 1.5k per devstack)
- ▶ ~151 2nd level guests created in each devstack cloud
- ▶ ~1 GB of logs uncompressed for each run

In aggregate:

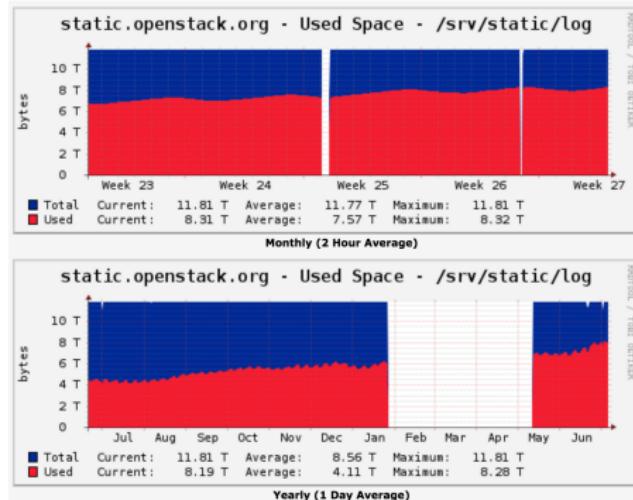
- ▶ ~12,500 jobs run in check and gate daily
- ▶ ~0.01% individual tempest test failure rate
- ▶ ~.77% tempest run failure rate

Number of Tempest Tests per Day in the Gate Queue:



Log Server

- ▶ Log Server: <http://logs.openstack.org/>
- ▶ Archive of all artifacts from all jobs for ~4 months
- ▶ ~8 TB of data compressed



Problems/Issues when Running in the Gate

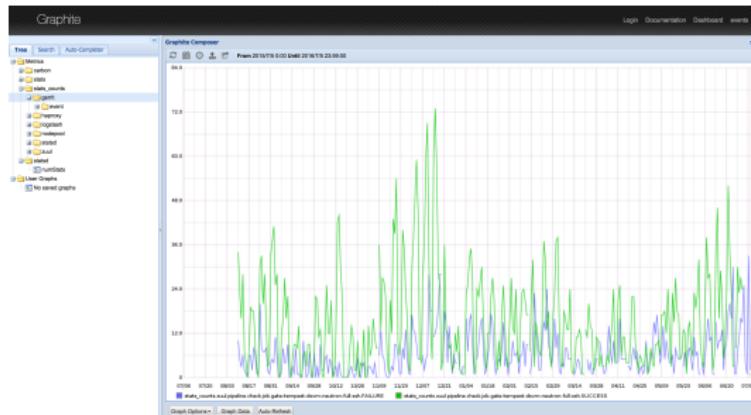
- ▶ Works fine for looking at results 1 at a time
- ▶ Difficult to find a non-deterministic failures
- ▶ Difficult to find performance regressions
- ▶ Finding out how often something passes or fails is impossible

General Approach

- ▶ Look at things on the larger scale
- ▶ Use statistics and data mining to find previously unknown trends in OpenStack
- ▶ Make the data from test runs open and accessible to everyone
- ▶ Ensure there are APIs available for accessing everything

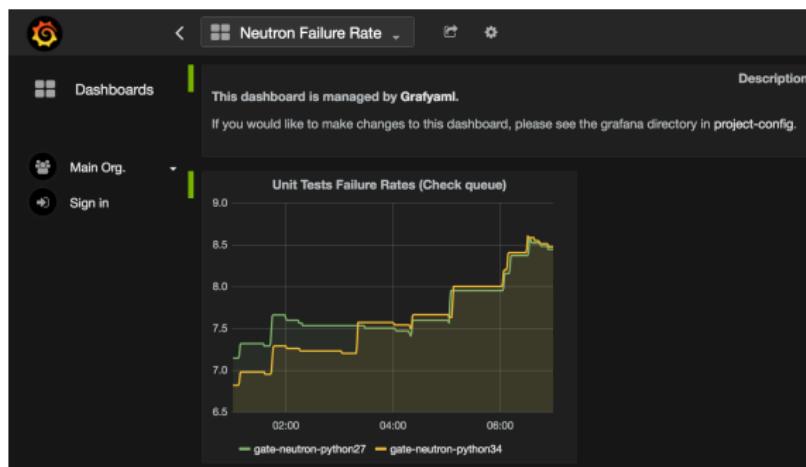
Graphite

- ▶ <http://graphite.openstack.org/>
- ▶ Infra services report to graphite
- ▶ Include job results
- ▶ Limited to job level data
- ▶ Time based, can't be linked to an individual job



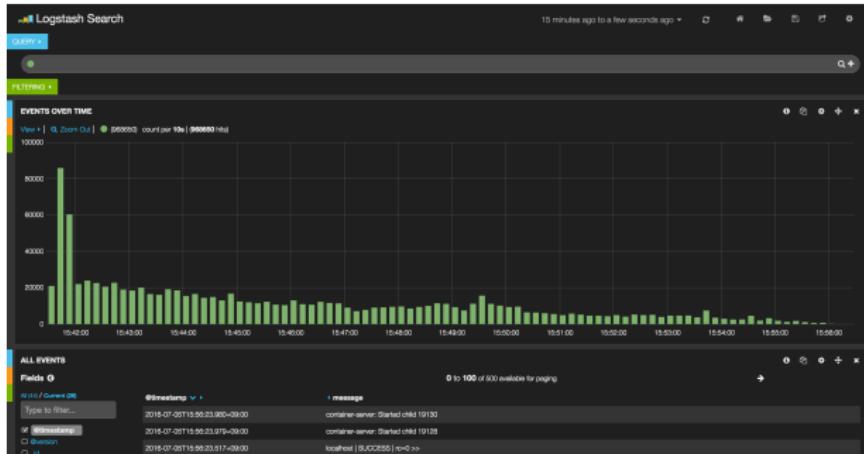
Grafana

- ▶ <http://grafana.openstack.org/>
- ▶ Provides a layer on top of graphite to easily make useful visualizations
- ▶ Adds a number of dashboards
- ▶ Some projects using this to track job failure rates



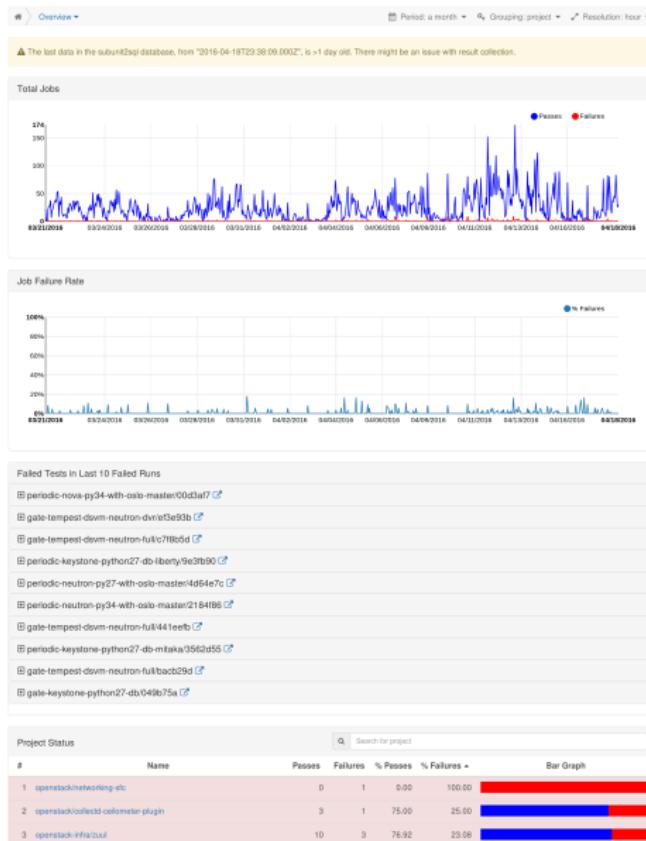
ELK

- ▶ Elasticsearch, Logstash, Kibana
 - ▶ <http://logstash.openstack.org>
 - ▶ Provides a search engine on top of job artifacts
 - ▶ Limited to 10 days of results



Using OpenStack Health

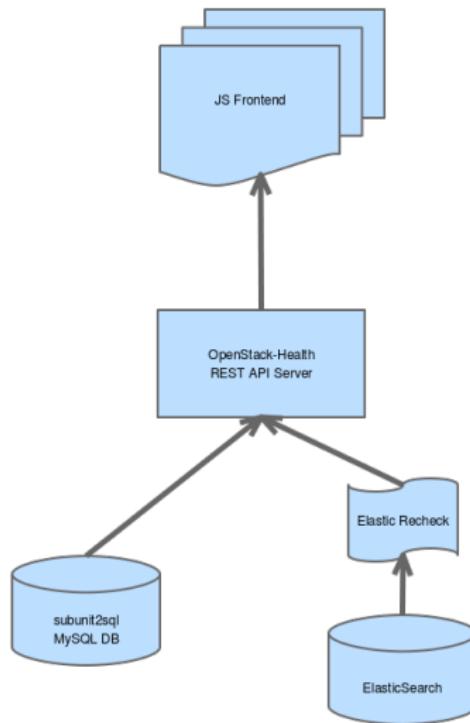
OpenStack Health is a dashboard for visualizing test results of OpenStack CI jobs.



openstack-health

- ▶ <http://status.openstack.org/openstack-health/#/>
- ▶ Designed to be a single point of access for all the data about the gate
- ▶ Currently can leverage subunit2sql and elastic-recheck

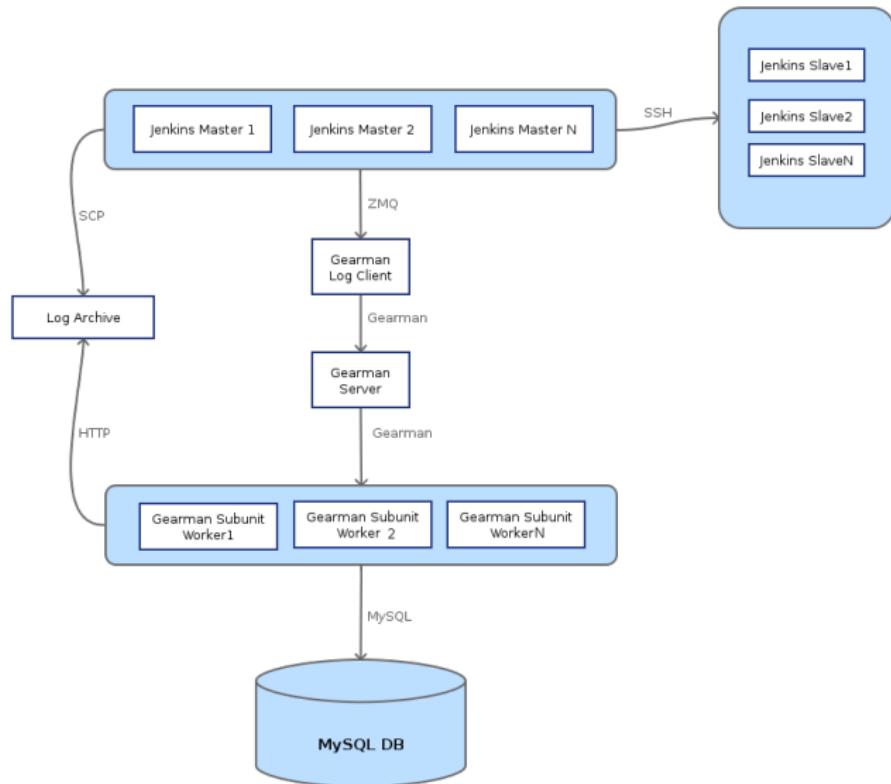
OpenStack-Health Architecture



subunit2sql

- ▶ Designed to store test results data in a sql database
- ▶ Provides a DB schema and a python API for interacting with the database
- ▶ CLI utilities for storing and retrieving results in the DB as subunit v2
- ▶ A public database of everything with subunit output run in OpenStack-Infra
- ▶ Used to store the results from test runs for 6 months

subunit2sql in OpenStack Infrastructure



Elastic Recheck

- ▶ Designed to answer the question “Have you seen this recently?”
- ▶ Leverages elastic search to identify failures with known fingerprints
- ▶ Contains a repository of elastic-search queries with known failures
- ▶ Has 2 parts:
 - ▶ A bot which watches changes and reports identified failures to gerrit and IRC
 - ▶ A dashboard which shows failure categorization

Elastic Recheck

<http://status.openstack.org/elastic-recheck/>

All Pipelines

Gate Pipeline

Uncategorized

The elastic-recheck project uses Elasticsearch to classify and track OpenStack gate failures. Documentation can be found [here](#). You can also learn more by reading this post on the Elasticsearch blog: [OpenStack elastic-recheck: powered by the elk stack](#)

Data Last Updated: Tue Jul 12 2016 13:00:01 GMT+0900 (JST)

Last Elastic Search Index Update: Tue Jul 12 2016 12:59:20 GMT+0900 (JST)

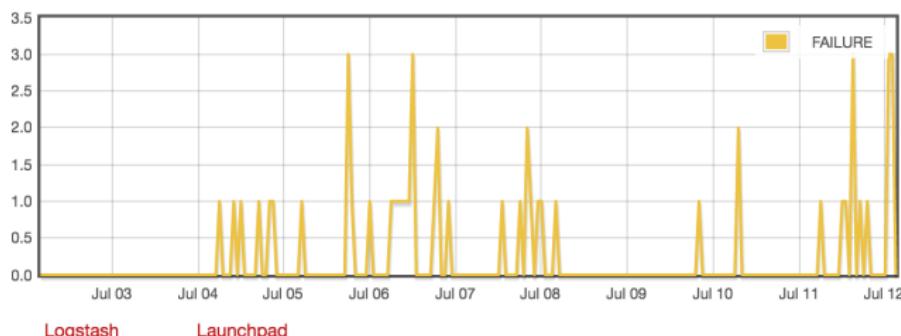
Delay in Elastic Search: Up to date

Cluster Health: green

Bug 1539271 - Libvirt live block migration migration stalls

14 fails in 24 hrs / 50 fails in 10 days

Projects: (nova - Confirmed)



Logstash

Launchpad

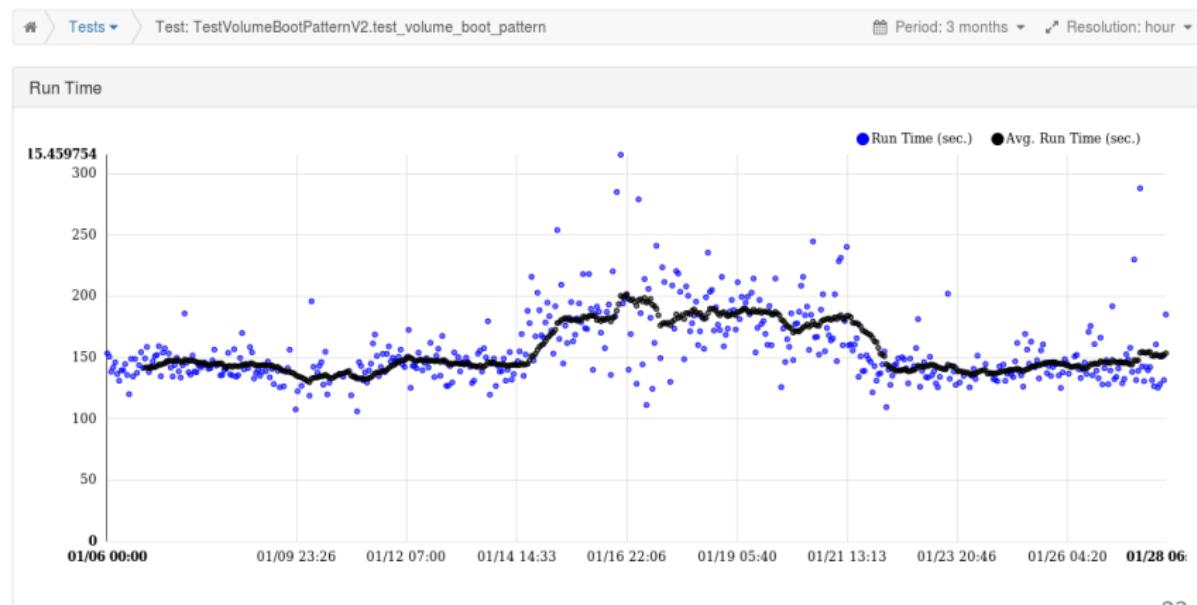
Data Driven Decision Making

- ▶ Determine when it's time to skip a test
- ▶ Identify tests that are actually catching bugs
- ▶ Determine if failures are isolated to region, config, etc.

Finding trends amongst the noise

- ▶ Catch performance regressions
- ▶ Identify relational trends in the data
- ▶ Find and identify non-deterministic bug/race conditions

TestVolumeBootPatternV2.test_volume_boot_pattern



Issues

- ▶ Too many varied data sources each with unique limitations:
 - ▶ Only Gate and Periodic Job data (subunit2sql)
 - ▶ No views for infra failure (subunit2sql)
 - ▶ Limited to 1 line for searching (elastic search)
- ▶ Limited contribution in this space

Future work

- ▶ Integrate all the things in openstack-health
- ▶ Use the data to optimize our test runner scheduler
- ▶ Enable automation around failure detection

Where to get more information

- ▶ openstack-dev ML openstack-dev@lists.openstack.org
- ▶ #openstack-qa on Freenode
- ▶ <http://git.openstack.org/cgit/openstack/openstack-health/>
- ▶ <http://git.openstack.org/cgit/openstack-infra/subunit2sql>
- ▶ <http://git.openstack.org/cgit/openstack-infra/elastic-recheck/>

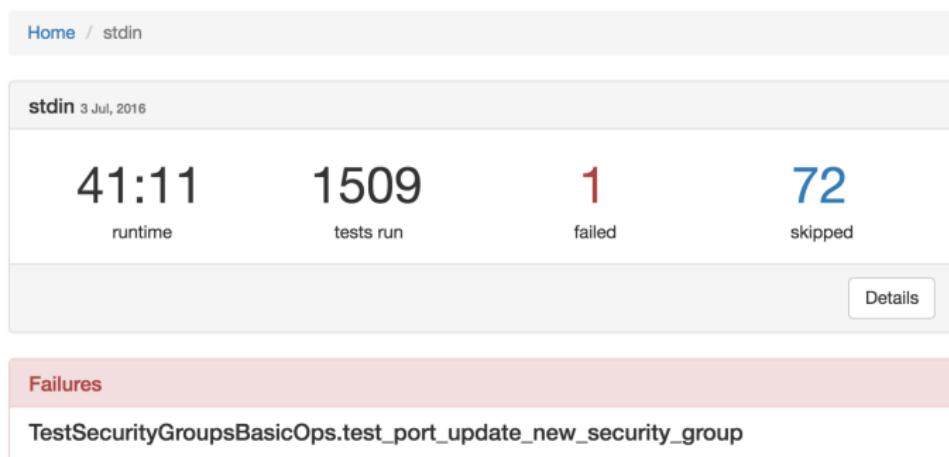
Questions?

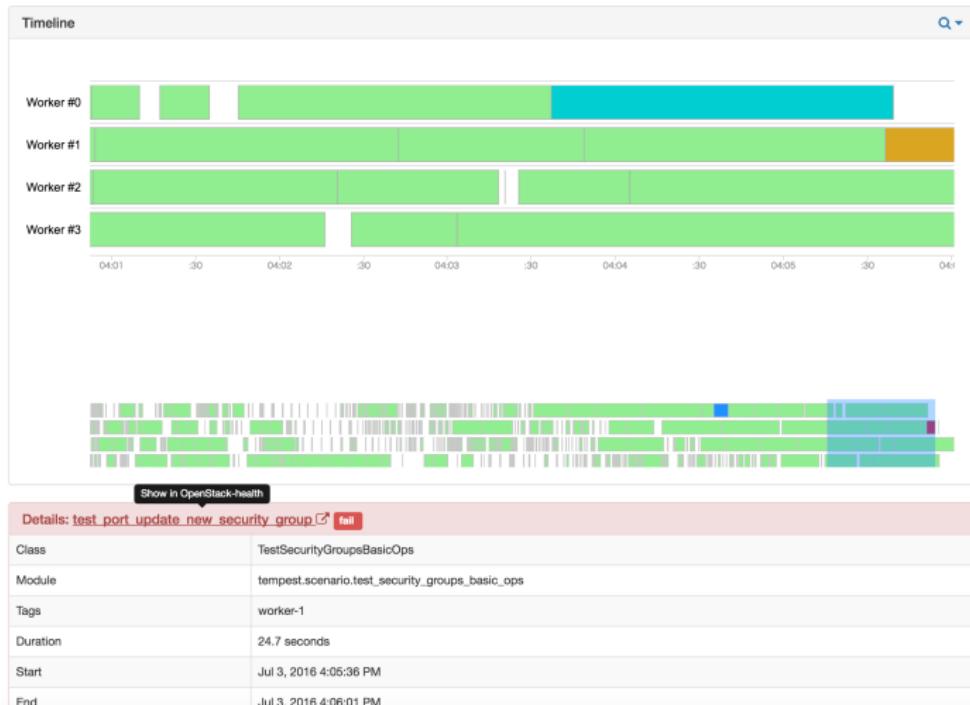
Appendix: StackViz

Visualization tool of individual CI build results

- ▶ git.openstack.org/cgit/openstack/stackviz

Datasets





TestSecurityGroupsBasicOps.test_port_update_new_security_group

Summary traceback pythonlogging



```
Traceback (most recent call last):
  File "tempest/scenario/test_security_groups_basic_ops.py", line 188, in setUp
    self._deploy_tenant(self.primary_tenant)
  File "tempest/scenario/test_security_groups_basic_ops.py", line 352, in _deploy_tenant
    self._set_access_point(tenant)
  File "tempest/scenario/test_security_groups_basic_ops.py", line 320, in _set_access_point
    self._assign_floating_ips(tenant, server)
  File "tempest/scenario/test_security_groups_basic_ops.py", line 326, in _assign_floating_ips
    client=tenant.manager.floating_ips_client)
  File "tempest/scenario/manager.py", line 868, in create_floating_ip
    port_id, ip4 = self._get_server_port_id_and_ip4(thing)
  File "tempest/scenario/manager.py", line 847, in _get_server_port_id_and_ip4
    "No IPv4 addresses found in: %s" % ports)
File "/opt/stack/new/tempest/.tox/tempest/local/lib/python2.7/site-packages/unittest2/case.py", line 845, in assertNotEqual
  raise self.failureException(msg)
AssertionError: 0 == 0 : No IPv4 addresses found in: [{u'extra_dhcp_opts': [], u'admin_state_up': True, u'mac_address': u'fa:16:3e:ee:7f:bc'}
```

Timeline