OpenStack Magnum *Pike* and the CERN cloud

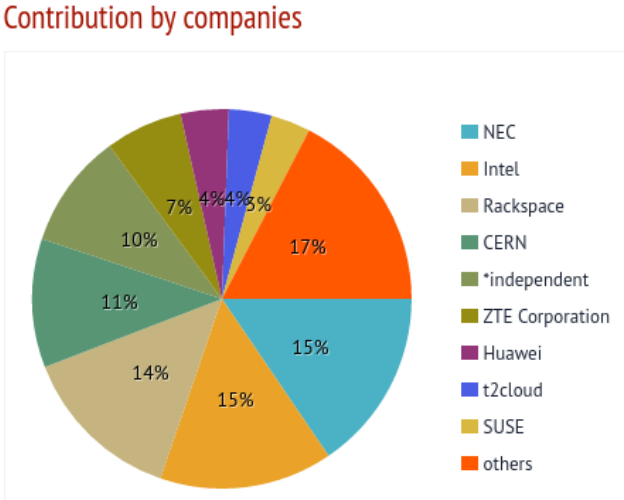
Spyros Trigazis @strigazi

OpenStack Magnum

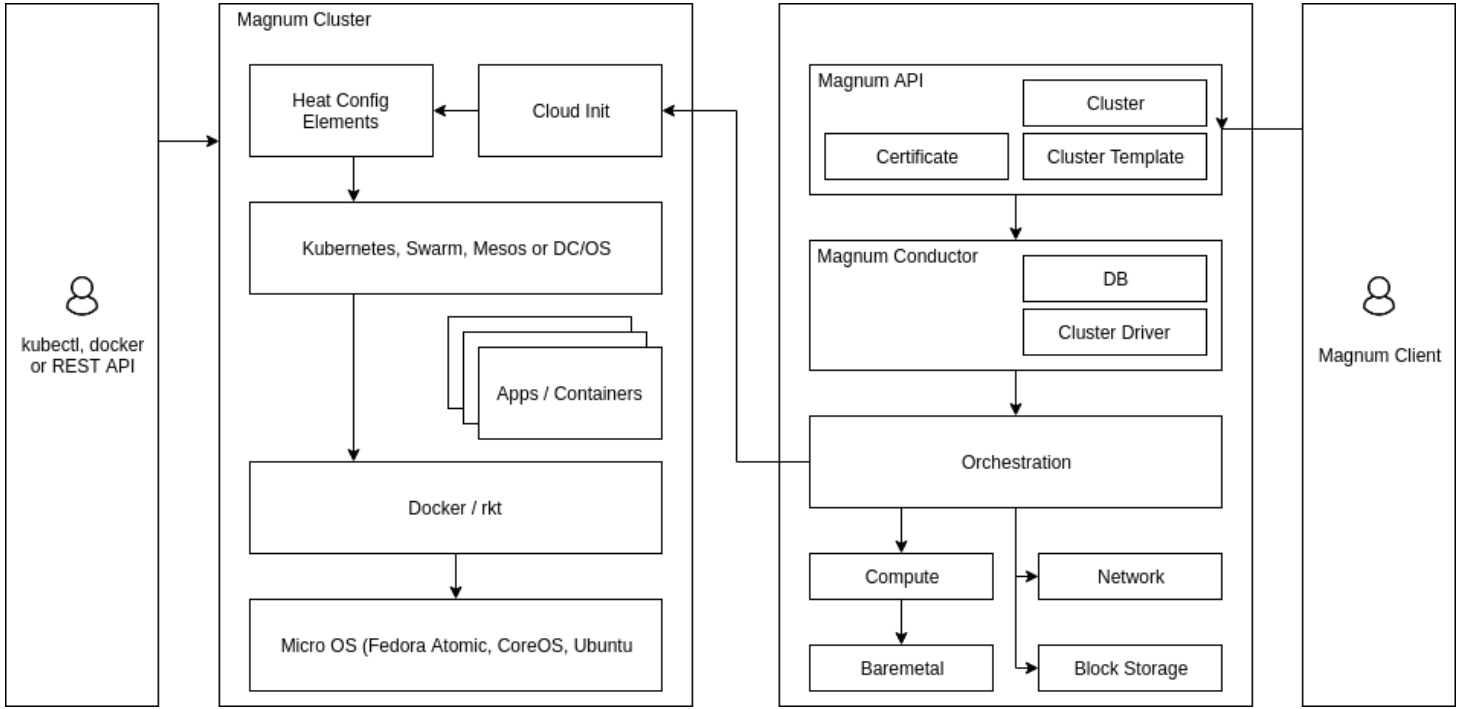
OpenStack Magnum #openstack-containers

Kubernetes, Docker Swarm, Apache Mesos, DC/OS (experimental) aaS
Deep integration of OpenStack with Container technologies:

- Compute Instances
- Networks, Load Balancers
- Storage
- Security
- Native Container API
- Lifecycle cluster operations
  - Scale cluster up and down
  - More WIP



OpenStack Magnum Architecture

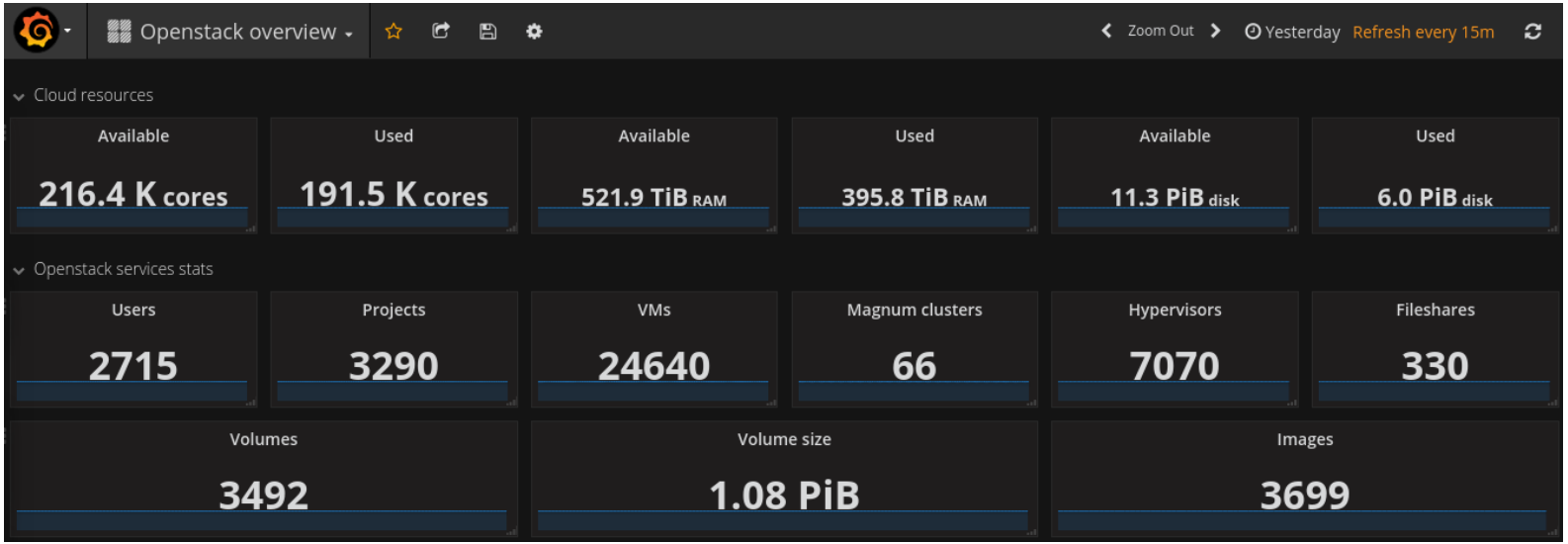


Containers and the CERN Cloud

CERN OpenStack Infrastructure

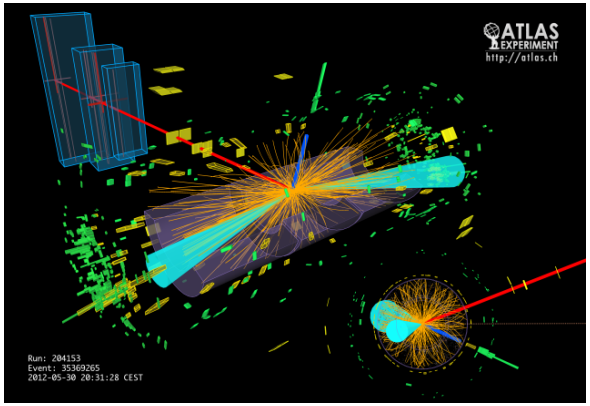
Production since 2013

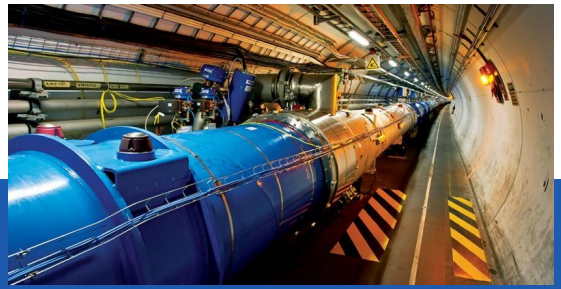
~ 216.000 cores ~4 million vms created ~200 vms per hour

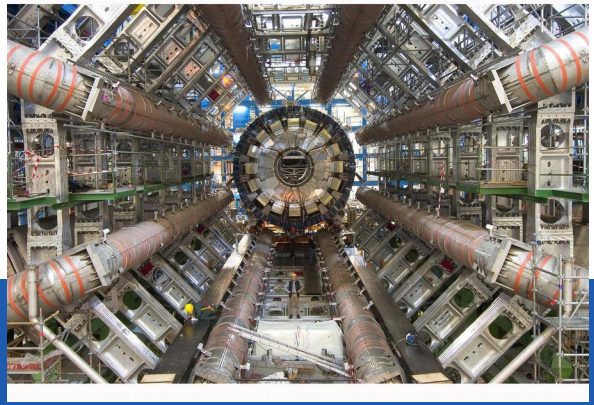


CERN Container Use Cases

- Batch Processing
- End user analysis / Jupyter Notebooks
- Machine Learning / TensorFlow / Keras
- Infrastructure Management
  - Data Movement, Web servers, PaaS …
- Continuous Integration / Deployment
- And many others

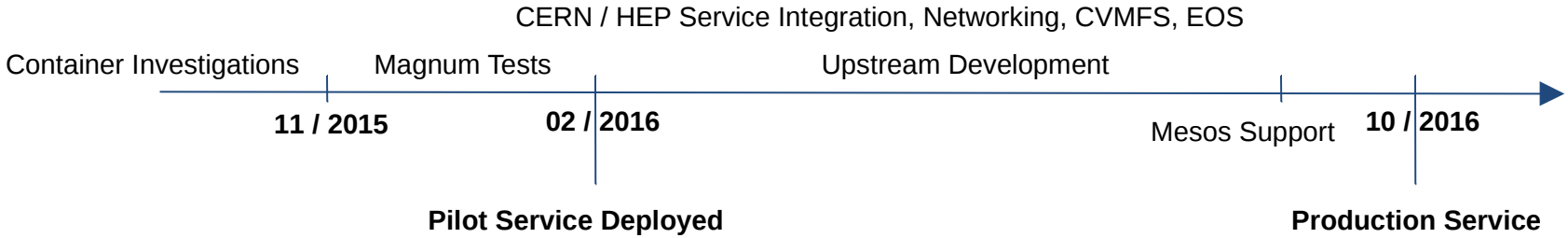






CERN Magnum Deployment

- Integrate containers in the CERN cloud
  - Shared identity, networking integration, storage access, …
- Add CERN services in *system* containers with atomic
- **Fast, Easy to use**



CERN Magnum Deployment

- Clusters are described by *cluster templates*
- Shared/public templates for most common setups, customizable by users

```
$ magnum cluster-template-list
+------+--------------------------+
| uuid | name                     |
+------+--------------------------+
| .... | swarm                    |
| .... | swarm-ha                 |
| .... | kubernetes               |
| .... | kubernetes-ha            |
| .... | mesos                    |
| .... | mesos-ha                 |
+------+--------------------------+
```

CERN Magnum Deployment

- Clusters are described by *cluster templates*
- Shared/public templates for most common setups, customizable by users

```
$ magnum cluster-create --name myswarmcluster --cluster-template swarm --node-count 100
     ~ 5 mins later
$ magnum cluster-list
+------+----------------+------------+--------------+----------------------------+
| uuid | name           | node_count | master_count | keypair  | status          |
+------+----------------+------------+--------------+----------------------------+
| .... | myswarmcluster | 100        | 1            | mysshkey | CREATE_COMPLETE |
+------+----------------+------------+--------------+----------------------------+

$ $(magnum cluster-config myswarmcluster --dir magnum/myswarmcluster)

$ docker info / ps / ...
$ docker run --volume-driver cvmfs -v atlas.cern.ch:/cvmfs/atlas -it centos /bin/bash
[root@32f4cf39128d /]#
```

Magnum Benchmarks

Rally Benchmarks and Kubernetes scalability

- Benchmark the Magnum service
  - How fast can I get my container cluster?
  - Use Rally to measure to performance like any other OpenStack service
- Benchmark the resources
  - Ok, it was reasonably fast, what can I do with it?
  - Use a demo provided by Google to measure the performance of the cluster
    - Rally tests for container are under development and near completion

Deployment Setup at CERN and CNCF

CERN

- 240 hypervisors
  - 32 cores, 64 GB RAM, 10Gb inks
- Container storage in our CEPH cluster
- Magnum / Heat setup
  - Dedicated 3 node controllers, dedicated 3 node RabbitMQ cluster
- Flat Network for vms

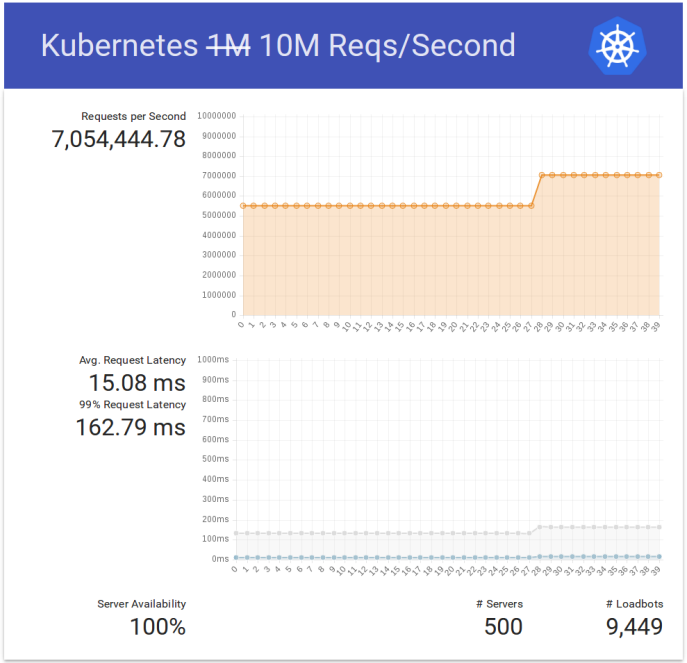
CNCF

- 100 hypervisors
  - 24 cores, 128 GB RAM
- Container storage in local disk
- Magnum / Heat setup
  - Shared 3 node controllers, shared 5 node RabbitMQ cluster
- Private networks with linux bridge

CERN Results

- Second go: rally and 7 million requests / sec
  - Kubernetes **7 million requests / sec**

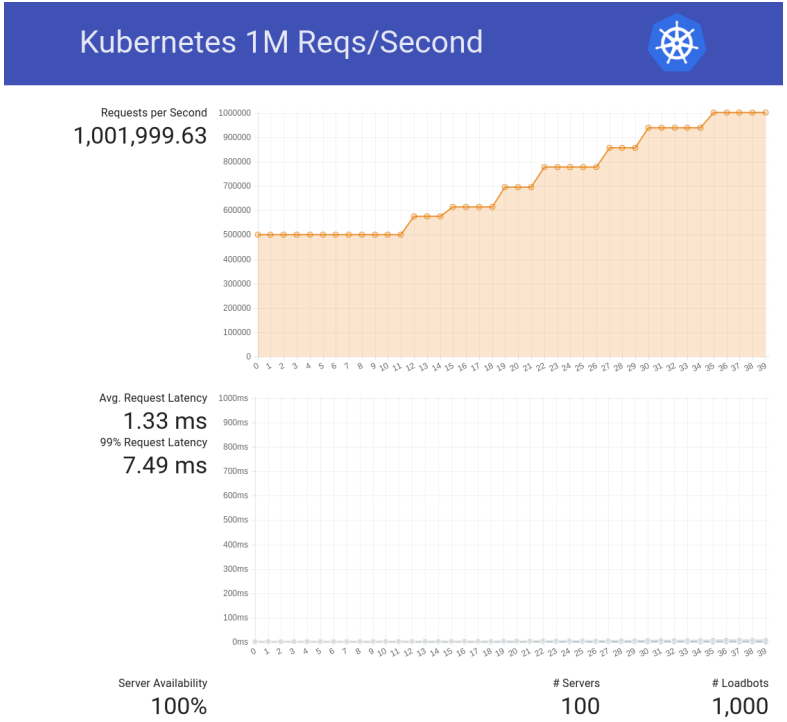
Cluster Size (Nodes)	Concurrency	Deployment Time (min)
2	50	2.5
16	10	4
32	10	4
128	5	5.5
512	1	14
1000	1	23



CNCF Results

Cluster Size (Nodes)	Concurrency	Number of Clusters	Deployment Time (min)
2	10	100	3.02
2	10	1000	Able to create 219 clusters
32	5	100	Able to create 28 clusters

nodes	containers	reqs/sec	latency	flannel
35	1100	1M	83.2 ms	udp
80	1100	1M	1.33 ms	host-gw
80	3100	3M	26.1 ms	host-gw



Plans for Magnum Pike

- Rolling upgrades of clusters
  - Upgrade to new versions of Kubernetes, Docker Swarm, DC/OS etc
- Heterogeneous Clusters
  - Mix of VMs and Baremetal, spread across AZs
- Docker Swarm Mode
- Container Monitoring
  - Work in Progress for a cadvisor, prometheus and grafana stack
- Container engine logging
- Full support for custom cluster drivers
  - Allow ops deploy easier driver with independent packages
- Baremetal support for all drivers

Cluster Upgrades

Cluster Upgrades

Number One priority for Pike!

1. Populate the cluster resource with all cluster_template attributes (started in Ocata)
2. Extend the driver_plugin interface
3. Create a new *versioned* Driver resource, improve the interaction of magnum with the driver_plugins
4. Bump cluster version after each operation
5. Handle upgrades with the new Driver resource

Driver Plugin Interface

1. validate_config
2. get_default_config NEW
3. create
4. create_dry_run NEW
5. update_dry_run NEW
6. update
7. upgrade NEW
8. delete
9. get_status NEW
10. get_scale_manager
11. get_monitor
12. rotate_ca_certificate existing interface, implementation WIP

Driver Resource

- uuid (immutable)
- name (string)
- public (boolean, default=false)
- version (immutable auto-increment integer)
- plugin_name (immutable string, supplied at create time, required)
- cluster_count (integer, derived by database query that counts clusters)
- config (blob of JSON text defaults to output of get_default_config method of the related driver plugin)
- enabled (boolean, default=true)
- latest_version (string uuid of the latest driver version, not visible in a list)

Driver Resource and Cluster versions

- Allow soft Driver resource delete
- On resource update, create a new one and bump the version

- Add auto-increment version field in Clusters
- Add an extra descriptive status/reason field Clusters
- After each cluster operation, bump the version and update the reason accordingly

Updating a Driver

- If no driver_plugin update is needed, update the Driver resource to change the config blob to change an attribute such as an image or COE version
- If the driver_plugin does need to be changed, update the driver package (which should have a new driver plugin version), update the Driver resource

Upgrading a Cluster (end user perspective)

- magnum cluster-upgrade <cluster name or id> --version x

Sounds complicated for Devs and Ops?

It is a little more that before... for a few reasons:

- It must be simple for end users
- Managing versioned objects is not trivial for a developers perspective
- We need to allow ops to do proper accounting
  - Without proper accounting and versioning, managing and supporting a lot of clusters, becomes an operational nightmare

Sounds complicated for Devs and Ops?

It is, for a few reasons:

- It must be simple for end users
- Managing versioned objects is not trivial for a developers perspective
- We need to allow ops to do proper accounting
  - Without proper accounting and versioning, managing and supporting a lot of clusters, becomes an operational nightmare

Heterogeneous Clusters phase 1

- Generate existing resource groups dynamically in Magnum
  - Currently, we are locked down to two resource groups per cluster (could be worse, only one :))
- No API changes, leverage existing labels and the new Driver resource to define nodegroups

COE status monitoring and Cluster healing

- Introduce a configurable periodic task to check the status of the COE
  - Are all expected nodes available?
- Add user triggered operation to heal the cluster
  - Driver specific, node replacement, services restart
    - PoC node replacement:
      http://clouddocs.web.cern.ch/clouddocs/containers/maintenance.html

Built-in monitoring and logging

- Add a Prometeus, grafana, cadvisor, node-exporter stack
  - Hosted on the COE
  - K8s and Docker swarm implementations are under review
- Leverage the existing docker logging mechanism to advertise its logs
  - Ops will be able to collect them in ElasticSearch, influxDB etc

Other features/optimizations

- Self-Hosted Kubernetes
  - Requires fully containerized kubernetes
- TLS credential caching
- Add more options to tune magnum’s periodic tasks
- OpenStack CI improvement, support openSUSE and CoreOS CI

Timeline?

- First working prototype April ‘17, before the OpenStack Summit in Boston

CERN