EVALUATING TWO YEARS IN PRODUCTION WITH JUJU AND CHARMED OPENSTACK

Ubuntu Summit 2022 | Prague, Czech Republic | 2022-11-08

Johan Hallbäck

Document class: Public

AGENDA

- INTRODUCTION
  - `juju whoami`
  - OpenStack @ Ibeo
- GOOD PRACTICES AND THINGS TO KNOW BEFORE DEPLOYING
  - Networking (Neutron)
  - GPUs in Nova
  - Ceph sizing and tuning
- JUJU SPECIFIC TOPICS
  - Understanding charms
  - Demystifying the Juju controller
- EXPERIENCES FROM UPGRADING USING JUJU
- BRING YOUR OWN CHARMS
  - Making network switches charming
- CURRENT ISSUES AND FUTURE WORK
  - Better NAS services
  - Improving backups

JUJU WHOAMI – ABOUT THE AUTHOR

Johan Hallbäck:

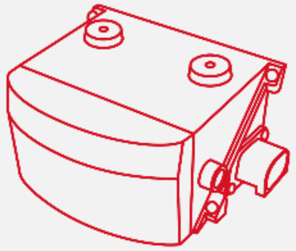
- Linux user since 1998
- Fulltime Linux/Unix/Storage sysadm since 2003
- B.Sc. in Computer Engineering 2004
- Juju user since 2018
- Charmed OpenStack & Ceph @ Ibeo, 2019-

- Charmhub Discourse: hallback
- Charmhub Chat: johan-hallback

ABOUT IBEO

Worldwide technology leader in the field of lidar sensors, validation solutions, and software tools.

- Hamburg, Germany
- Eindhoven, Netherlands
- Detroit, USA
- Coming soon: China



Ibeo technology in serial production since 2017 with Audi, Mercedes

430 + competent and enthusiastic colleagues create innovative solutions in an agile and self-managed work environment.

OPENSTACK @ IBEO (2019-)

Scaling challenges with traditional infrastructure components

- Employee number doubling every 12-18 months
- IT acting resource gatekeeper
- Highly skilled users demanding control
- Initial serices needed: Virtualization, file and block storage
- Workload: Heavy CI/CD, simulation, sensor data processing etc
- Started to build a complete new environent, including network infra

Top-Innovator 2022

PUBLIC

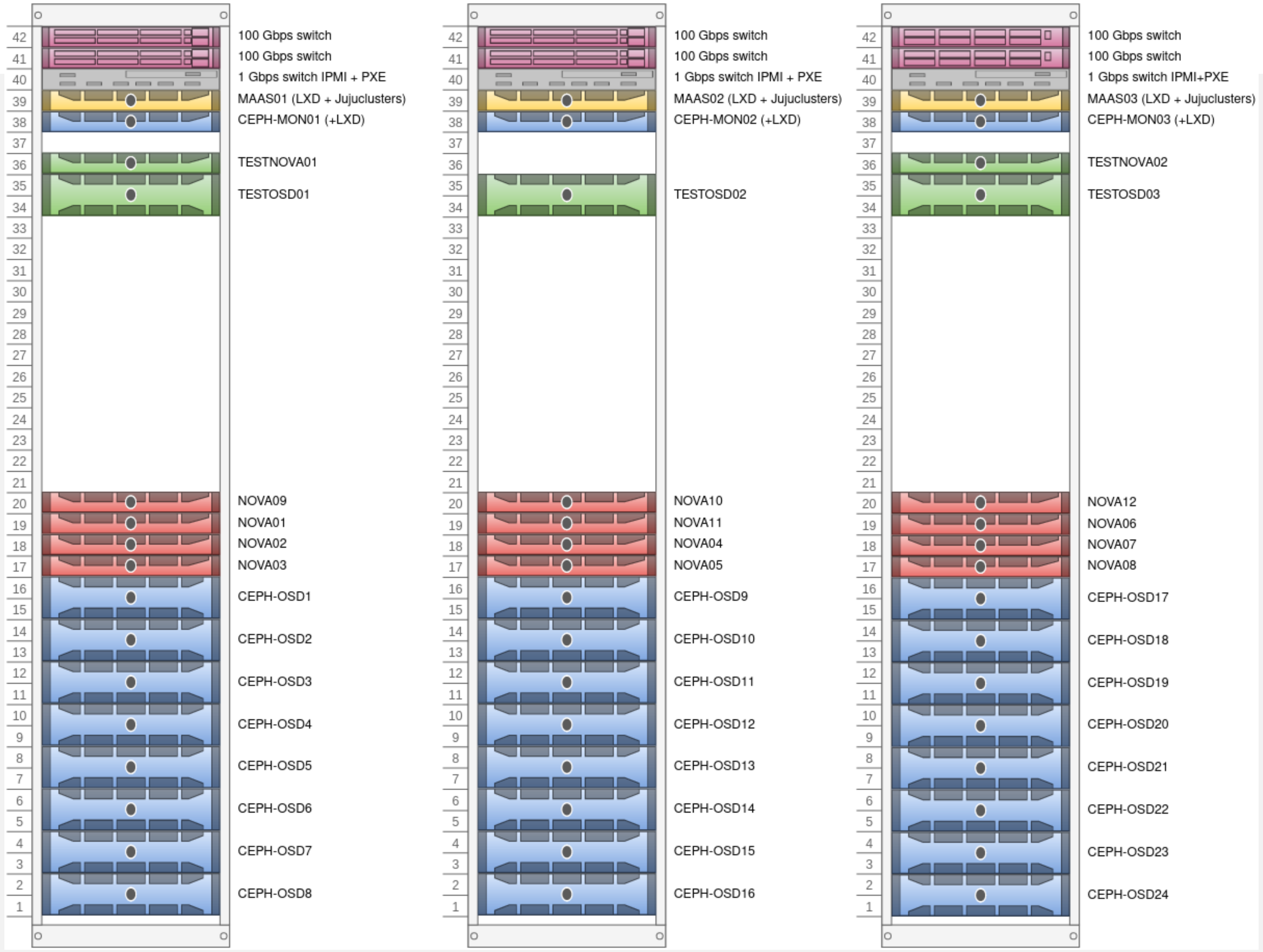
GOOD PRACTICES AND THINGS TO KNOW BEFORE DEPLOYING

Where to start?

- OpenStack Charms Deployment Guide: https://docs.openstack.org/project-deploy-guide/charm-deployment-guide
- OpenStack Charm guide: https://docs.openstack.org/charm-guide
- Charmed Ceph Documentation: https://ubuntu.com/ceph/docs

Our approach: Strip down the OpenStack Telemetry Bundle anno 2019, customize and deploy `cloud:bionic-train`.

RACK LAYOUT

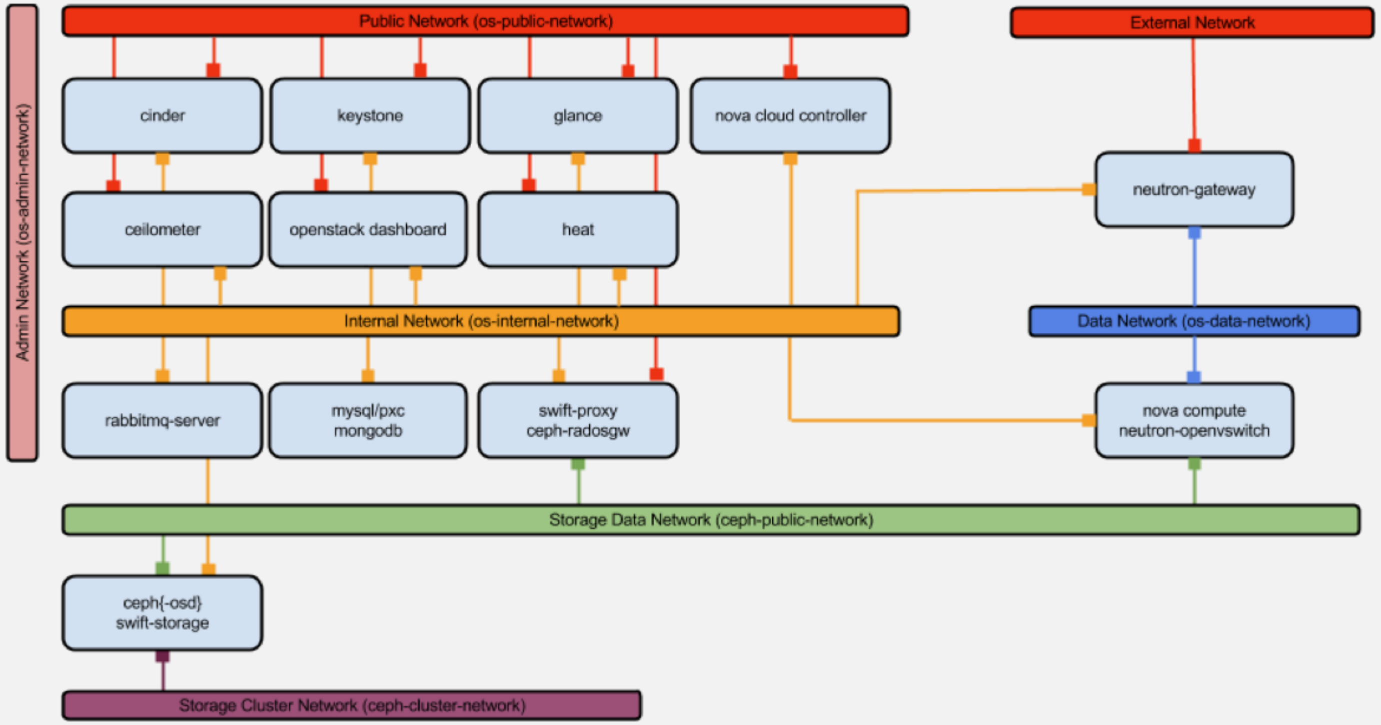

42
41
40
39
38
37
36
35
34
33
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
100 Gbps switch
100 Gbps switch
1 Gbps switch IPMI + PXE
MAAS01 (LXD + Jujuclusters)
CEPH-MON01 (+LXD)
TESTNOVA01
TESTOSD01
NOVA09
NOVA01
NOVA02
NOVA03
CEPH-OSD1
CEPH-OSD2
CEPH-OSD3
CEPH-OSD4
CEPH-OSD5
CEPH-OSD6
CEPH-OSD7
CEPH-OSD8
100 Gbps switch
100 Gbps switch
1 Gbps switch IPMI + PXE
MAAS02 (LXD + Jujuclusters)
CEPH-MON02 (+LXD)
TESTOSD02
NOVA10
NOVA11
NOVA04
NOVA05
CEPH-OSD9
CEPH-OSD10
CEPH-OSD11
CEPH-OSD12
CEPH-OSD13
CEPH-OSD14
CEPH-OSD15
CEPH-OSD16
100 Gbps switch
100 Gbps switch
1 Gbps switch IPMI+PXE
MAAS03 (LXD + Jujuclusters)
CEPH-MON03 (+LXD)
TESTNOVA02
TESTOSD03
NOVA12
NOVA06
NOVA07
NOVA08
CEPH-OSD17
CEPH-OSD18
CEPH-OSD19
CEPH-OSD20
CEPH-OSD21
CEPH-OSD22
CEPH-OSD23
CEPH-OSD24

OPENSTACK NETWORKING USING MAAS+JUJU

As dictated by the tools:

- MAAS: Define L2 constructs (bonds, VLANs), subnets and spaces
- Juju: Bind endpoints to spaces or assign CIDRs to `os-public-network` etc
- Neutron OVS: map Charm options like `data-ports`/`bridge-mappings` (or OVN similar) to bonds from MAAS

(Image by James Page: source: https://maas.io/blog/deploying-openstack-on-maas-1-9-with-juju)


Admin Network (os-admin-network)
Public Network (os-public-network)
External Network
cinder
keystone
glance
nova cloud controller
neutron-gateway
ceilometer
openstack dashboard
heat
Internal Network (os-internal-network)
Data Network (os-data-network)
rabbitmq-server
mysql/pxc mongodb
swift-proxy ceph-radosgw
nova compute neutron-openvswitch
Storage Data Network (ceph-public-network)
ceph{-osd} swift-storage
Storage Cluster Network (ceph-cluster-network)

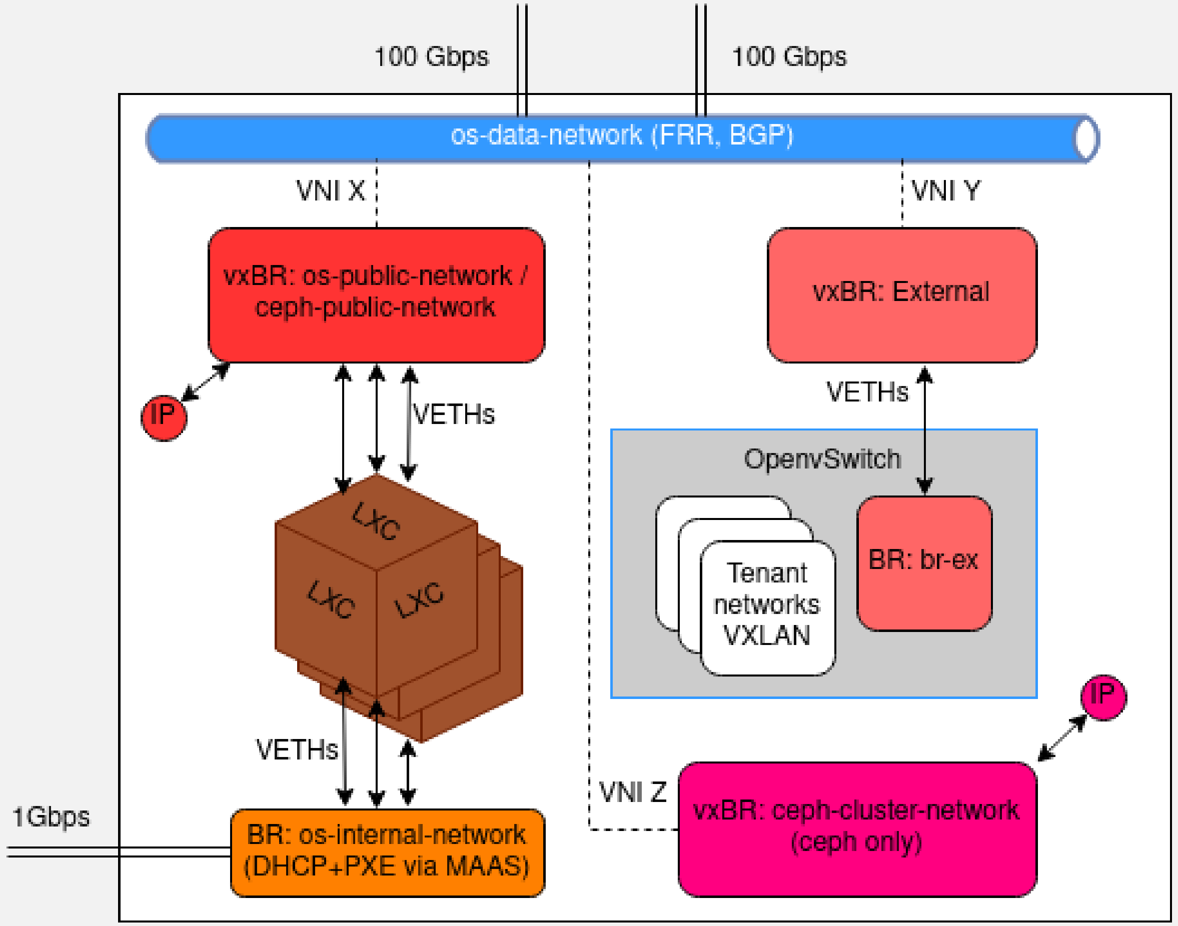
L3 APPROACH AUTOMATED OUTSIDE JUJU

Features:

- MAAS managed internal (deploy) network
- Routing-on-the-Host using FRR (BGP+EVPN)
- VXLAN tenant networks in OVS (DVR)
- 2x100Gbps and MTU=9000 everywhere

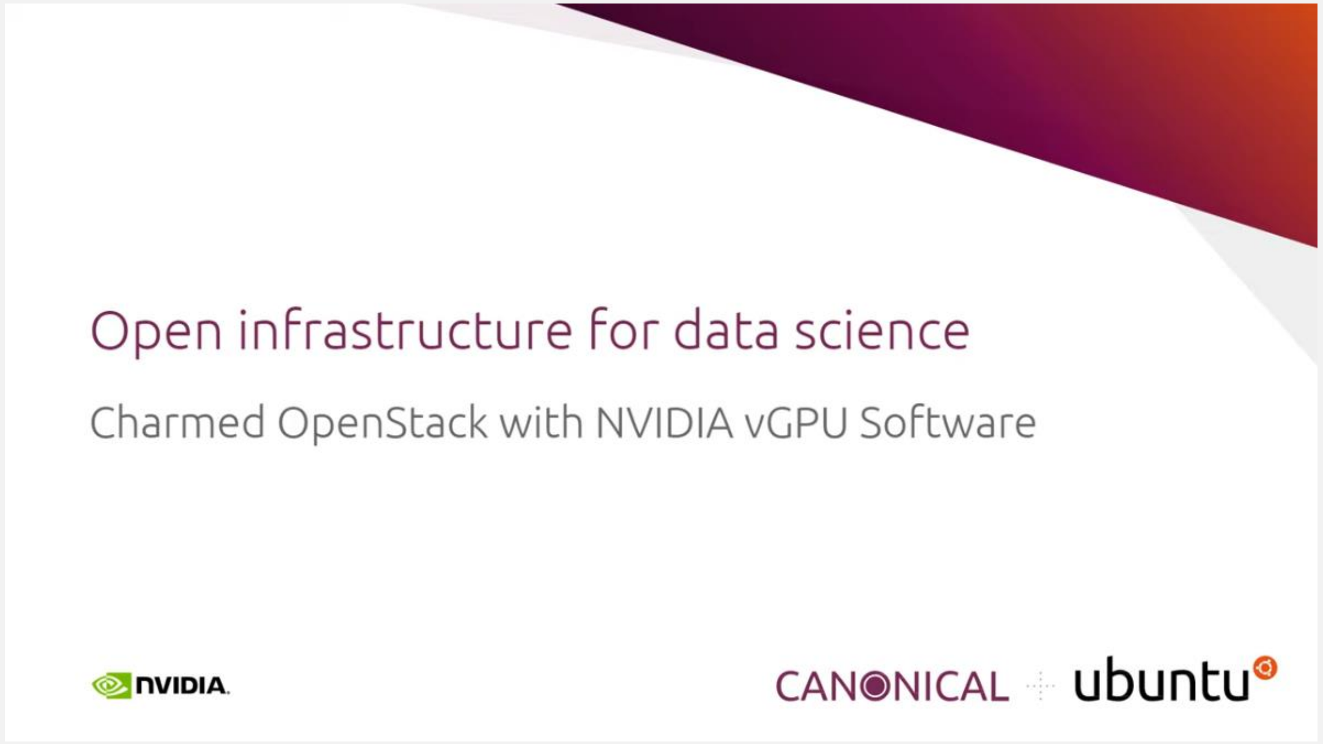
Breaks the deploy process in three:

1. Deploy OpenStack bundle without relations
2. Run script for FRR setup, host bridges, IP allocation incl LXD veths
3. Add bundle overlay with relations & spaces


100 Gbps
100 Gbps
os-data-network (FRR, BGP)
VNI X
VNI Y
vxBR: os-public-network / ceph-public-network
vxBR: External
IP
VETHs
VETHs
OpenvSwitch
LXC
LXC
LXC
Tenant networks VXLAN
BR: br-ex
IP
VETHs
VNI Z
1Gbps
BR: os-internal-network (DHCP+PXE via MAAS)
vxBR: ceph-cluster-network (ceph only)

(V)GPUS IN NOVA THE JUJU WAY

- Choose the right cards with vGPU support: https://docs.nvidia.com/grid
- Add the nova-compute-nvidia-vgpu subordinate to nova-compute
- Divide your GPU to x vGPUs using Juju
- Create a trait, add to a flavor and boot!
- Demo by James page at https://youtu.be/TiAe3L_H4W8 ➔


Open infrastructure for data science

Charmed OpenStack with NVIDIA vGPU Software

GPU PCI PASSTHROUGH USING JUJU (1/2)

Still doable with a few commands!

1. Create tag in MAAS with kernel options for PCI passthrough and tag machine:

```
$ maas kraken tags create name='Nvidia-RTX3090' comment='Nvidia RTX3090 PCI Passthrough' \
    kernel_opts='amd_iommu=on iommu=pt kvm.ignore_msrs=1 vfio-pci.ids=10de:2204,10de:1aef modprobe.blacklist=nouveau'
```

2. Configure PCI aliases and whitelist in the Nova charms:

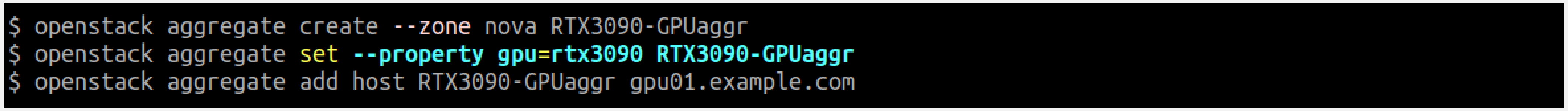
```
$ juju config nova-compute pci-alias='{"vendor_id":"10de","product_id":"2204","name":"Nvidia-RTX3090"}'
$ juju config nova-cloud-controller pci-alias='{"vendor_id":"10de","product_id":"2204","name":"Nvidia-RTX3090"}'
$ juju config nova-compute pci-passthrough-whitelist='{"vendor_id": "10de", "product_id": "2204"}'
```

3. Adjust filters in the nova-cloud-controller charm:

```
$ FILTERS=$(juju ssh nova-cloud-controller/leader 'sudo grep enabled_filters /etc/nova/nova.conf | cut -d" " -f3')
$ juju config nova-cloud-controller \
    scheduler-default-filters="$FILTERS,AggregateInstanceExtraSpecsFilter,PciPassthroughFilter"
```

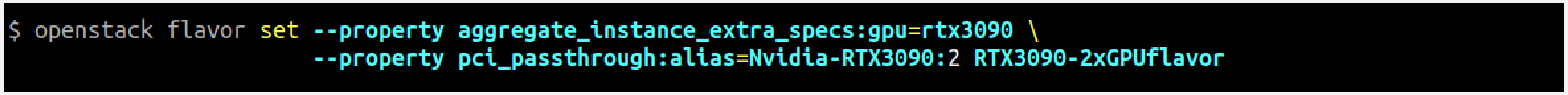
GPU PCI PASSTHROUGH USING JUJU (2/2)

4. Create a host aggregate, set a GPU property and add your GPU host to it:



```
$ openstack aggregate create --zone nova RTX3090-GPUaggr
$ openstack aggregate set --property gpu=rtx3090 RTX3090-GPUaggr
$ openstack aggregate add host RTX3090-GPUaggr gpu01.example.com
```

5. Set PCI passthrough and host aggregate specs on a flavor:



```
$ openstack flavor set --property aggregate_instance_extra_specs:gpu=rtx3090 \
                       --property pci_passthrough:alias=Nvidia-RTX3090:2 RTX3090-2xGPUflavor
```

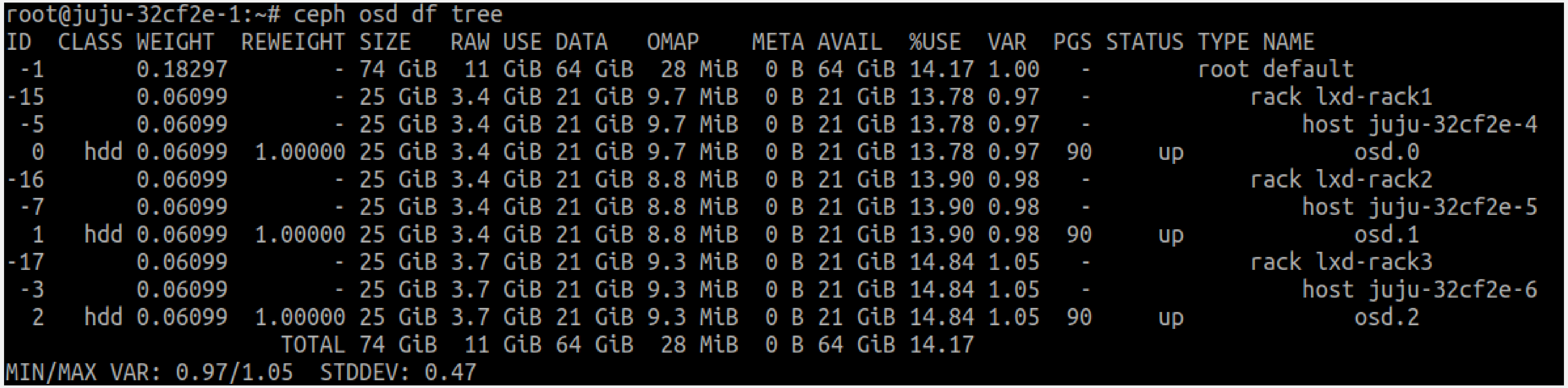
6. Boot with new flavor!

Results:

- No way to split (share) GPUs without shelving
- Still, ”hardware” and VMs manageable in the same cloud substrate
- Groundwork for introducing Omnivector’s SLURM charms using our Candid enabled Juju controller cluster in OpenStack

CEPH TRAP #1: WRONG FAILURE DOMAIN & REPLICATION RULE

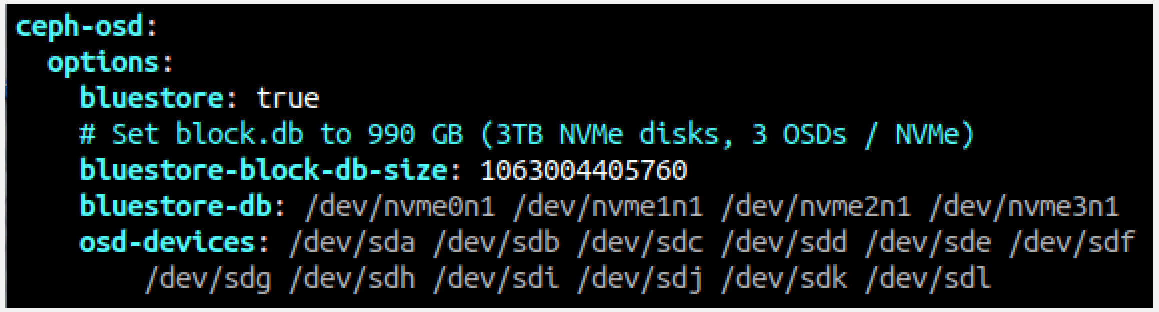
- With replication and multiple racks, data should be replicated between racks, not hosts (which is default)
- Define availability zones in MAAS, add hosts to them sorted by rack
- `juju config ceph-mon customize-failure-domain=true` (same for ceph-osd)
- Possible to modify CRUSH map and replication rule afterwards, but takes a LONG time
- Verify:



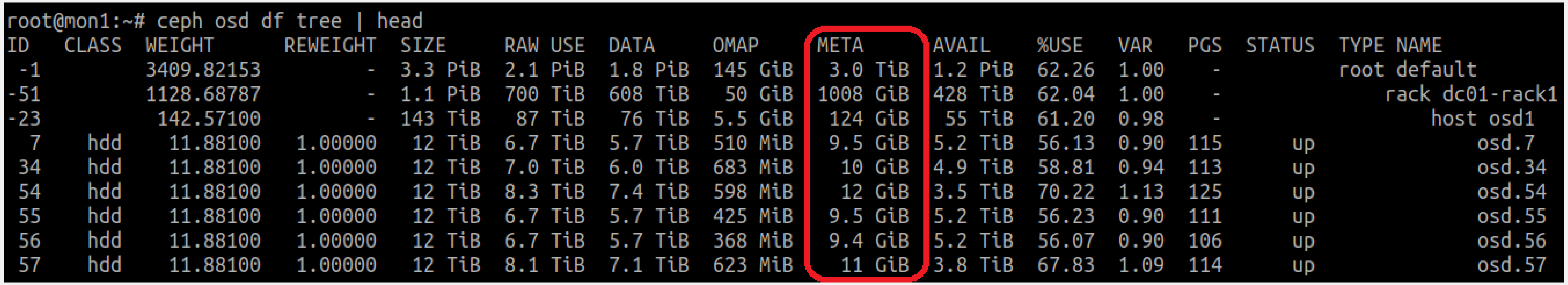
```
root@juju-32cf2e-1:~# ceph osd df tree
ID  CLASS WEIGHT   REWEIGHT SIZE    RAW USE DATA    OMAP     META AVAIL  %USE  VAR  PGS STATUS TYPE NAME
 -1       0.18297         - 74 GiB  11 GiB 64 GiB  28 MiB  0 B 64 GiB 14.17 1.00   -        root default
-15       0.06099         - 25 GiB 3.4 GiB 21 GiB 9.7 MiB  0 B 21 GiB 13.78 0.97   -            rack lxd-rack1
 -5       0.06099         - 25 GiB 3.4 GiB 21 GiB 9.7 MiB  0 B 21 GiB 13.78 0.97   -                host juju-32cf2e-4
  0   hdd 0.06099   1.00000 25 GiB 3.4 GiB 21 GiB 9.7 MiB  0 B 21 GiB 13.78 0.97  90     up              osd.0
-16       0.06099         - 25 GiB 3.4 GiB 21 GiB 8.8 MiB  0 B 21 GiB 13.90 0.98   -            rack lxd-rack2
 -7       0.06099         - 25 GiB 3.4 GiB 21 GiB 8.8 MiB  0 B 21 GiB 13.90 0.98   -                host juju-32cf2e-5
  1   hdd 0.06099   1.00000 25 GiB 3.4 GiB 21 GiB 8.8 MiB  0 B 21 GiB 13.90 0.98  90     up              osd.1
-17       0.06099         - 25 GiB 3.7 GiB 21 GiB 9.3 MiB  0 B 21 GiB 14.84 1.05   -            rack lxd-rack3
 -3       0.06099         - 25 GiB 3.7 GiB 21 GiB 9.3 MiB  0 B 21 GiB 14.84 1.05   -                host juju-32cf2e-6
  2   hdd 0.06099   1.00000 25 GiB 3.7 GiB 21 GiB 9.3 MiB  0 B 21 GiB 14.84 1.05  90     up              osd.2
                      TOTAL 74 GiB  11 GiB 64 GiB  28 MiB  0 B 64 GiB 14.17
MIN/MAX VAR: 0.97/1.05  STDDEV: 0.47
```

CEPH TRAP #2: WRONG BLUESTORE BLOCK.DB SIZING

- Plan about 4% of SATA capacity on NVMe/SSD for metadata
- Charm default `bluestore-block-db-size` is 1GB (!)
- Unwanted result: BlueFS metadata spillover on **slow disks**
- Easiest fix using Juju actions, for each ceph-osd unit:
  - out, down, purge, zap all OSDs
  - remove-unit and add back



```
ceph-osd:
  options:
    bluestore: true
    # Set block.db to 990 GB (3TB NVMe disks, 3 OSDs / NVMe)
    bluestore-block-db-size: 1063004405760
    bluestore-db: /dev/nvme0n1 /dev/nvme1n1 /dev/nvme2n1 /dev/nvme3n1
    osd-devices: /dev/sda /dev/sdb /dev/sdc /dev/sdd /dev/sde /dev/sdf
        /dev/sdg /dev/sdh /dev/sdi /dev/sdj /dev/sdk /dev/sdl
```



```
root@mon1:~# ceph osd df tree | head
ID   CLASS  WEIGHT      REWEIGHT  SIZE     RAW USE  DATA     OMAP     META      AVAIL    %USE   VAR   PGS  STATUS  TYPE NAME
 -1         3409.82153         -  3.3 PiB  2.1 PiB  1.8 PiB  145 GiB   3.0 TiB  1.2 PiB  62.26  1.00    -          root default
-51         1128.68787         -  1.1 PiB  700 TiB  608 TiB   50 GiB  1008 GiB  428 TiB  62.04  1.00    -              rack dc01-rack1
-23          142.57100         -  143 TiB   87 TiB   76 TiB  5.5 GiB   124 GiB   55 TiB  61.20  0.98    -                  host osd1
  7    hdd    11.88100   1.00000   12 TiB  6.7 TiB  5.7 TiB  510 MiB   9.5 GiB  5.2 TiB  56.13  0.90  115      up              osd.7
 34    hdd    11.88100   1.00000   12 TiB  7.0 TiB  6.0 TiB  683 MiB    10 GiB  4.9 TiB  58.81  0.94  113      up              osd.34
 54    hdd    11.88100   1.00000   12 TiB  8.3 TiB  7.4 TiB  598 MiB    12 GiB  3.5 TiB  70.22  1.13  125      up              osd.54
 55    hdd    11.88100   1.00000   12 TiB  6.7 TiB  5.7 TiB  425 MiB   9.5 GiB  5.2 TiB  56.23  0.90  111      up              osd.55
 56    hdd    11.88100   1.00000   12 TiB  6.7 TiB  5.7 TiB  368 MiB   9.4 GiB  5.2 TiB  56.07  0.90  106      up              osd.56
 57    hdd    11.88100   1.00000   12 TiB  8.1 TiB  7.1 TiB  623 MiB    11 GiB  3.8 TiB  67.83  1.09  114      up              osd.57
```

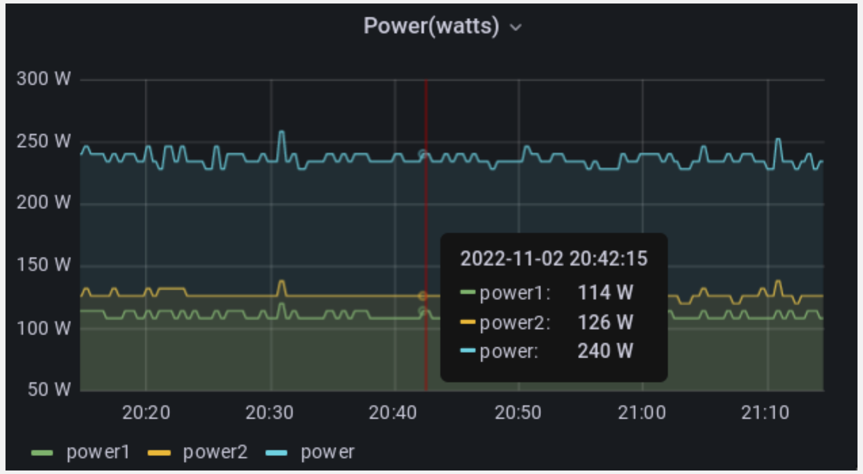
- Docs: https://docs.ceph.com/en/quincy/rados/configuration/bluestore-config-ref/#sizing

BASIC TUNING: CPU GOVERNOR

Make sure your hardware performance mode matches your workload

- In BIOS, ensure EIST (P-state) is enabled ➔ OS control
- Let Nagios monitor all hosts: `juju config nrpe-host cpu-governor=performance`
- Observe throttling issues and fan speed settings
- TIP: `juju config telegraf collect_ipmi_sensor_metrics=true`
- Perfect task for `juju model-config cloud-init-userdata`:

```
if [ ! -f /etc/default/cpufrequtils ] ; then
    sudo systemctl stop ondemand
    sudo systemctl disable ondemand
    sudo apt install -y cpufrequtils
    echo "GOVERNOR=performance"| sudo tee /etc/default/cpufrequtils
    sudo systemctl restart cpufrequtils
fi
```



2022-11-02 20:42:15
power1: 114 W
power2: 126 W
power: 240 W

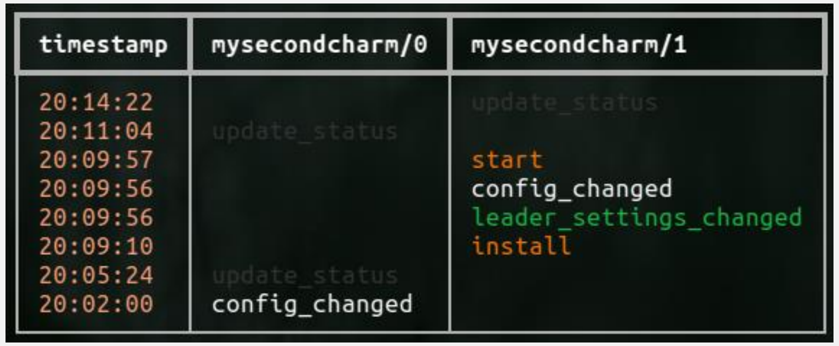
JUJU SPECIFIC TOPICS

- Understanding charms – everything is easy until your first hook failure
- Operating a rock solid black box – The Juju controller

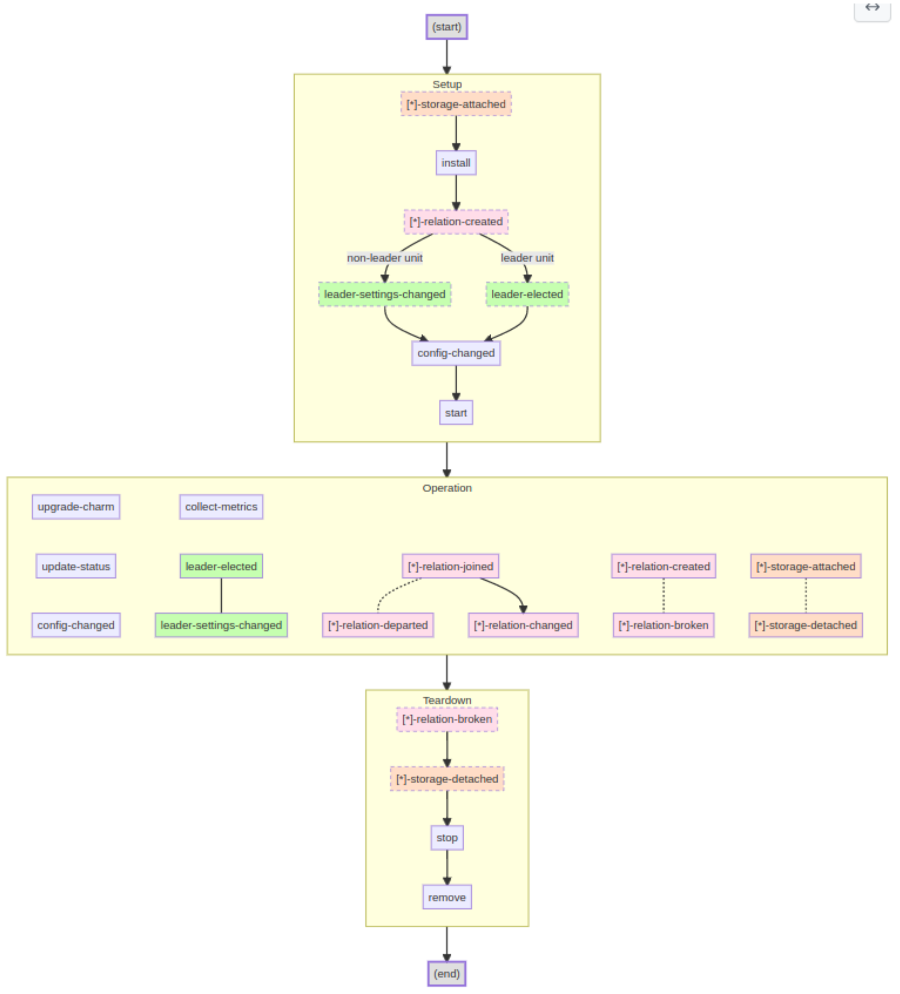
UNDERSTANDING CHARMS (1/2): WHAT AWAITS THE OPERATOR?

A charm’s life – events and hooks

- The charm lifecycle finally official: https://juju.is/docs/sdk/a-charms-life
- **Code reading skills are required**
- Manually: `juju run --unit mycharm/0 hooks/config-changed`
- Use `jhack tail` to visualize events in model
- Emergency workaround: `sys.exit(0)`



```
timestamp   mysecondcharm/0    mysecondcharm/1
20:14:22                       update_status
20:11:04    update_status
20:09:57                       start
20:09:56                       config_changed
20:09:56                       leader_settings_changed
20:09:10                       install
20:05:24    update_status
20:02:00    config_changed
```



PUBLIC

UNDERSTANDING CHARMS (2/2): WHAT AWAITS THE OPERATOR?

Template rendering debugging

- Config not as expected? Check templates in `/var/lib/juju/agents/unit-<charmname>-<unitnumber>/charm/templates`
- Trace config variables back to the charm source code
- **Code reading skills are required**
- Need to hack here? Probably on the wrong path...

THE JUJU CONTROLLER (1/5): GENERAL REMARKS

A rock solid black box on which your infrastructure is dependent

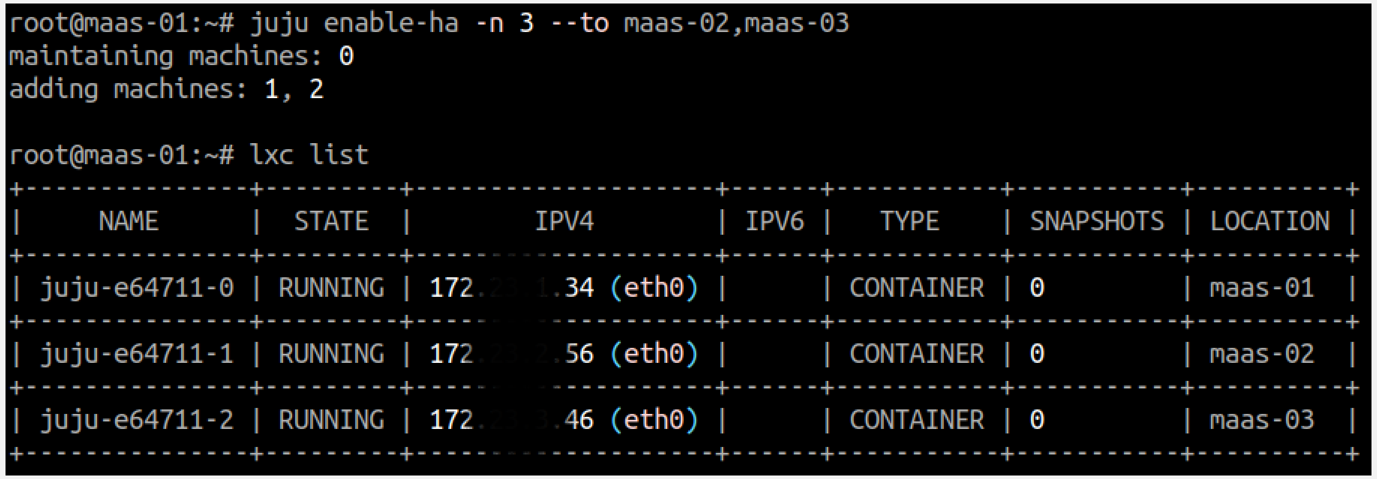
- What can you do without it?
- Can you wait for community help?
- Consider support

THE JUJU CONTROLLER (2/5): JUJU HA

ibeo automotive

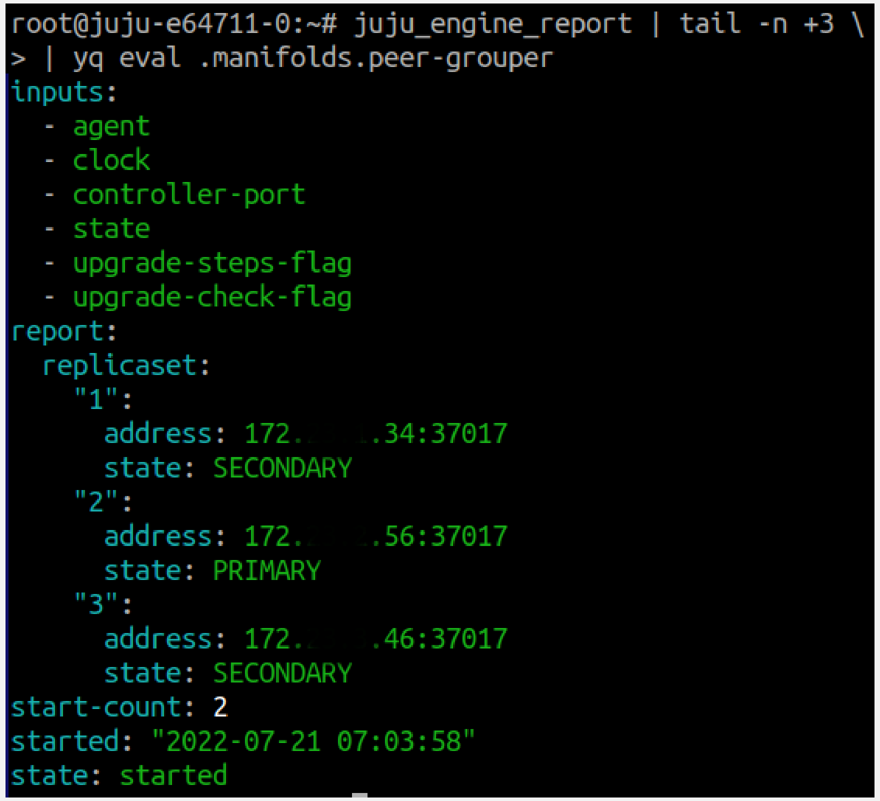
Enable HA!

- Best feature: ”instant restore” of MongoDB trough re-replication:
- Stop `mongod` and `jujud`, clear `/var/lib/juju/db` and restart
- Observe the MongoDB replicaset states



```
root@maas-01:~# juju enable-ha -n 3 --to maas-02,maas-03
maintaining machines: 0
adding machines: 1, 2

root@maas-01:~# lxc list
+---------------+---------+--------------------+------+-----------+-----------+----------+
|     NAME      |  STATE  |        IPV4        | IPV6 |   TYPE    | SNAPSHOTS | LOCATION |
+---------------+---------+--------------------+------+-----------+-----------+----------+
| juju-e64711-0 | RUNNING | 172.[redacted].34 (eth0) |      | CONTAINER | 0         | maas-01  |
+---------------+---------+--------------------+------+-----------+-----------+----------+
| juju-e64711-1 | RUNNING | 172.[redacted].56 (eth0) |      | CONTAINER | 0         | maas-02  |
+---------------+---------+--------------------+------+-----------+-----------+----------+
| juju-e64711-2 | RUNNING | 172.[redacted].46 (eth0) |      | CONTAINER | 0         | maas-03  |
+---------------+---------+--------------------+------+-----------+-----------+----------+
```



```
root@juju-e64711-0:~# juju_engine_report | tail -n +3 \
> | yq eval .manifolds.peer-grouper
inputs:
  - agent
  - clock
  - controller-port
  - state
  - upgrade-steps-flag
  - upgrade-check-flag
report:
  replicaset:
    "1":
      address: 172.[redacted].34:37017
      state: SECONDARY
    "2":
      address: 172.[redacted].56:37017
      state: PRIMARY
    "3":
      address: 172.[redacted].46:37017
      state: SECONDARY
start-count: 2
started: "2022-07-21 07:03:58"
state: started
```

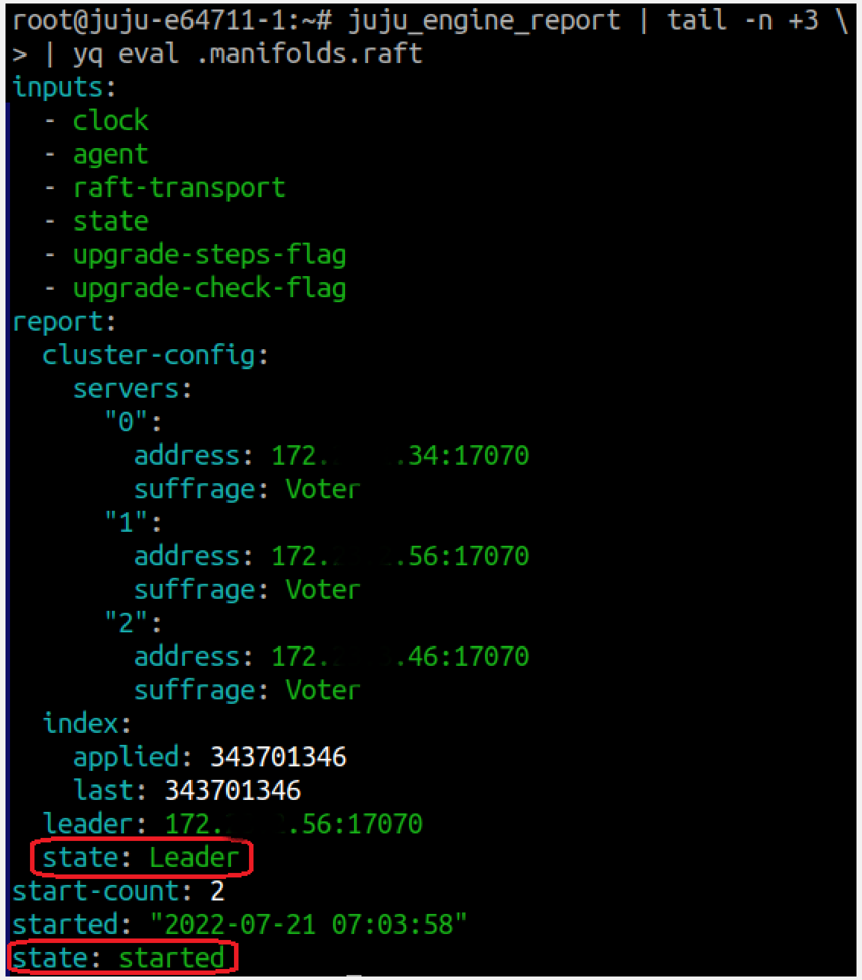
TOP 100 top100.de Top-Innovator 2022

PUBLIC

THE JUJU CONTROLLER (3/5): LEADERSHIP AND RAFT

Everything depends on leadership

- Leadership leases through consensus algorithm called raft
- State is stored in `/var/lib/juju/raft/logs` on the controller
- State replicated from leader, the rest are followers
- On all controllers: `state: started`

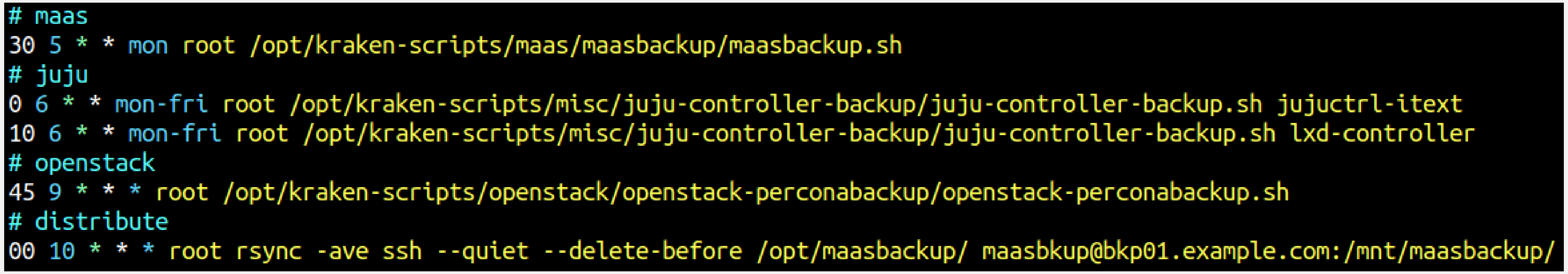


```
root@juju-e64711-1:~# juju_engine_report | tail -n +3 \
> | yq eval .manifolds.raft
inputs:
  - clock
  - agent
  - raft-transport
  - state
  - upgrade-steps-flag
  - upgrade-check-flag
report:
  cluster-config:
    servers:
      "0":
        address: 172.[redacted].34:17070
        suffrage: Voter
      "1":
        address: 172.[redacted].56:17070
        suffrage: Voter
      "2":
        address: 172.[redacted].46:17070
        suffrage: Voter
  index:
    applied: 343701346
    last: 343701346
  leader: 172.[redacted].56:17070
  state: Leader
start-count: 2
started: "2022-07-21 07:03:58"
state: started
```

THE JUJU CONTROLLER (4/5): BACKUP

There is no reason not to backup

- Be sure to run scheduled backups using `juju create-backup`
- Always before you `juju upgrade-controller` (and models)
- Send the backups elsewhere
- Tip: juju users can have a line `password: xxxxyyyy` in `~/.local/share/juju/accounts.yaml`

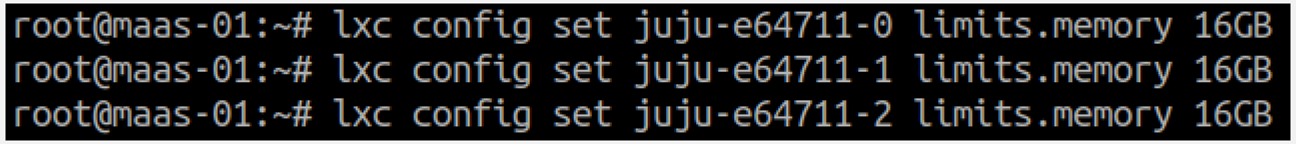


```
# maas
30 5 * * mon root /opt/kraken-scripts/maas/maasbackup/maasbackup.sh
# juju
0 6 * * mon-fri root /opt/kraken-scripts/misc/juju-controller-backup/juju-controller-backup.sh jujuctrl-itext
10 6 * * mon-fri root /opt/kraken-scripts/misc/juju-controller-backup/juju-controller-backup.sh lxd-controller
# openstack
45 9 * * * root /opt/kraken-scripts/openstack/openstack-perconabackup/openstack-perconabackup.sh
# distribute
00 10 * * * root rsync -ave ssh --quiet --delete-before /opt/maasbackup/ maasbkup@bkp01.example.com:/mnt/maasbackup/
```

THE JUJU CONTROLLER (5/5): GENERAL ADVICE

Some personal best practices:

- When upgrading, upgrade all models directly after the controller
- Always before you `juju upgrade-controller` (and models)
- Normally, `juju debug-log -m controller` should not be full of errors
- Display current controller in shell prompt to not mess up wrong model
- Controller in LXD on host with much RAM? Set container limits:



```
root@maas-01:~# lxc config set juju-e64711-0 limits.memory 16GB
root@maas-01:~# lxc config set juju-e64711-1 limits.memory 16GB
root@maas-01:~# lxc config set juju-e64711-2 limits.memory 16GB
```

- Using ZFS? Monitor your pools! https://charmhub.io/prometheus-zfs-exporter
- With Juju 3.0 controller will be a charm, https://charmhub.io/juju-controller. Cross controller relationship with Nagios?

LIFECYCLE MANAGEMENT FOR REAL: UPGRADING WITH JUJU

- OpenStack
- Ceph
- Ubuntu (series)
- Verdict: Epic success

OUR PROGRESS (2019-)

Rock solid, no reinstallations so far:

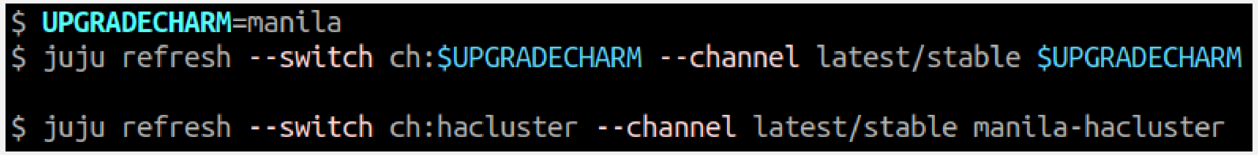
- MAAS: 2.6 (Bionic) ➔ 3.1 (Focal)
- Juju: 2.6.9 ➔ 2.9.32 (pending dist upgrade)
- LXD: 3.0.3 (deb) ➔ 5.6 (snap)

Dist	TEST	PROD	Ceph	Src
18.04	bionic-train	bionic-train	Nautilus	cs:
	bionic-ussuri	bionic:ussuri	Octopus	cs:
20.04	distro (ussuri)	distro (ussuri)	Octopus	cs:
	focal-victoria		Pacific	ch:
	focal-wallaby		Pacific	ch:
	focal-xena		Quincy	**ch:**
	focal-yoga		Quincy	**ch:**

UPGRADING CHARMS

Verdict: Can be done without risk

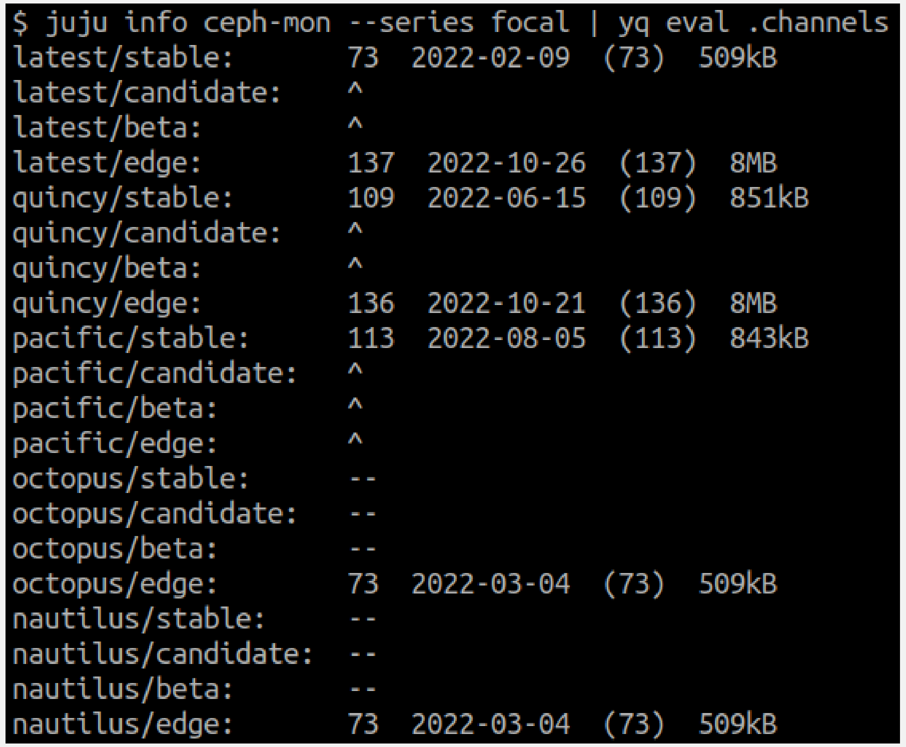
- OpenStack charms seem stale on Charmstore (cs:) since Feb 2022
- OpenStack deployments < Xena could easily switch to Charmhub:



```
$ UPGRADECHARM=manila
$ juju refresh --switch ch:$UPGRADECHARM --channel latest/stable $UPGRADECHARM

$ juju refresh --switch ch:hacluster --channel latest/stable manila-hacluster
```

- Migrating to channels (e.g. xena/stable) for Xena and Ceph Pacific and above seems to not have settled yet (September 2022)

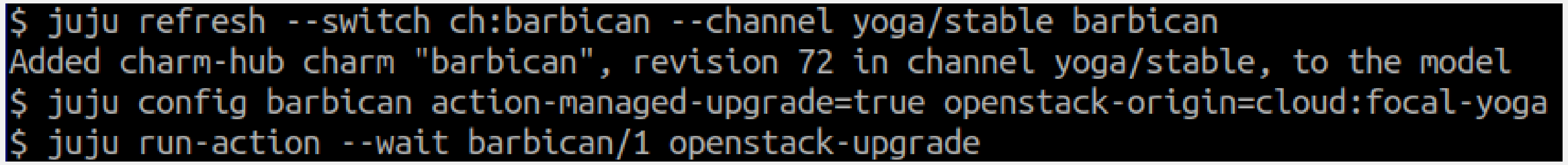


```
$ juju info ceph-mon --series focal | yq eval .channels
latest/stable:       73   2022-02-09  (73)   509kB
latest/candidate:    ^
latest/beta:         ^
latest/edge:         137  2022-10-26  (137)  8MB
quincy/stable:       109  2022-06-15  (109)  851kB
quincy/candidate:    ^
quincy/beta:         ^
quincy/edge:         136  2022-10-21  (136)  8MB
pacific/stable:      113  2022-08-05  (113)  843kB
pacific/candidate:   ^
pacific/beta:        ^
pacific/edge:        ^
octopus/stable:      --
octopus/candidate:   --
octopus/beta:        --
octopus/edge:        73   2022-03-04  (73)   509kB
nautilus/stable:     --
nautilus/candidate:  --
nautilus/beta:       --
nautilus/edge:       73   2022-03-04  (73)   509kB
```

OPENSTACK & CEPH UPGRADES

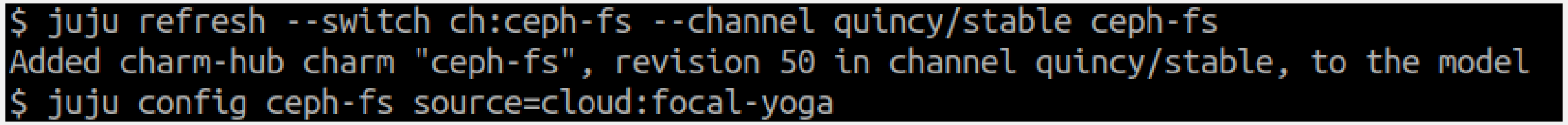
Verdict: This is where Juju excel. Daytime activity!

- Standardized: Same process for most charms
- OpenStack charms (leader first when HA):



```
$ juju refresh --switch ch:barbican --channel yoga/stable barbican
Added charm-hub charm "barbican", revision 72 in channel yoga/stable, to the model
$ juju config barbican action-managed-upgrade=true openstack-origin=cloud:focal-yoga
$ juju run-action --wait barbican/1 openstack-upgrade
```

- Ceph charms:



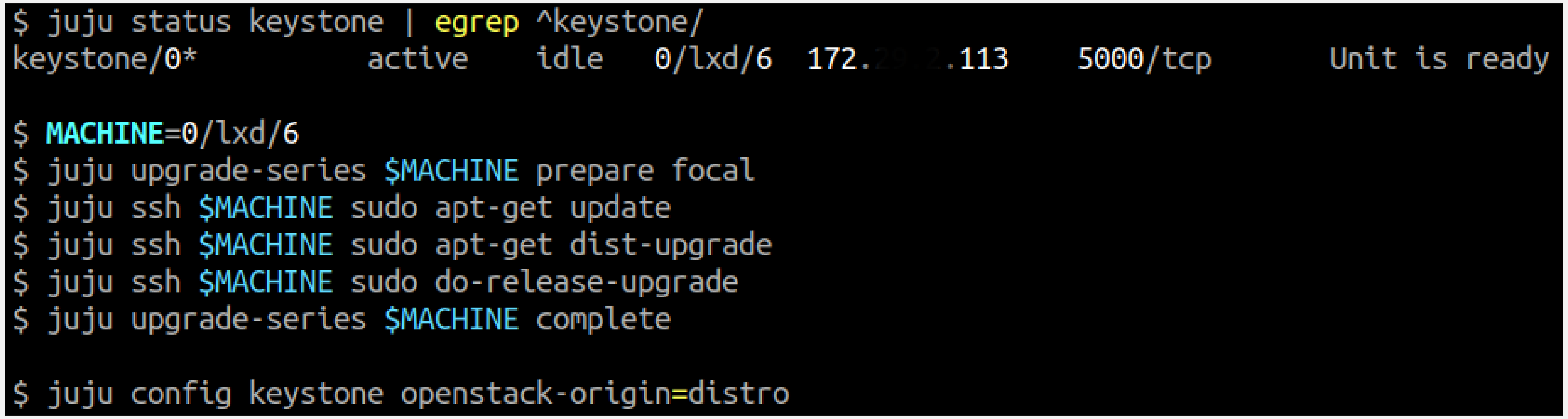
```
$ juju refresh --switch ch:ceph-fs --channel quincy/stable ceph-fs
Added charm-hub charm "ceph-fs", revision 50 in channel quincy/stable, to the model
$ juju config ceph-fs source=cloud:focal-yoga
```

- Verify package versions and services
- Minor corner case issues

SERIES UPGRADES

Verdict: This is where Juju excel. Low risk activity.

- Standardized: Same process for **ALL** charms
- No software version change (e.g. versions in UCA `cloud:bionic-ussuri` matches Focal `distro`)



```
$ juju status keystone | egrep ^keystone/
keystone/0*          active    idle   0/lxd/6  172.[redacted].113    5000/tcp        Unit is ready

$ MACHINE=0/lxd/6
$ juju upgrade-series $MACHINE prepare focal
$ juju ssh $MACHINE sudo apt-get update
$ juju ssh $MACHINE sudo apt-get dist-upgrade
$ juju ssh $MACHINE sudo do-release-upgrade
$ juju upgrade-series $MACHINE complete

$ juju config keystone openstack-origin=distro
```

- Tip: use `screen` for `do-release-upgrade`
- Tip: MAAS APT proxy

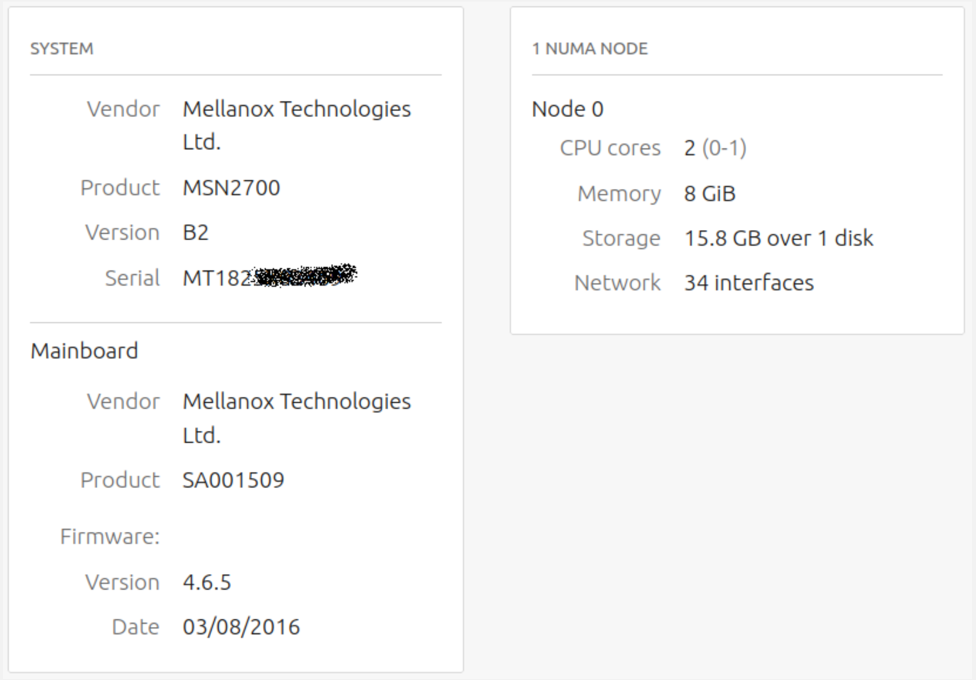
BRING YOUR OWN CHARMS

- “Juju has such a steep learning curve” – challenge accepted
- If you can script it, you can charm it!
- You get basic monitoring for free!

DEPLOYING MELLANOX SWITCHES USING JUJU

Two days later:

- Kernel parameters through MAAS tags
- Set switches to PXE, deploy Ubuntu on them
- Hook based charm (240 lines) installs FRR, config in Git
- MTU management as `juju config` using `netplan`



SYSTEM

Vendor	Mellanox Technologies Ltd.
Product	MSN2700
Version	B2
Serial	MT182[redacted]

Mainboard

Vendor	Mellanox Technologies Ltd.
Product	SA001509

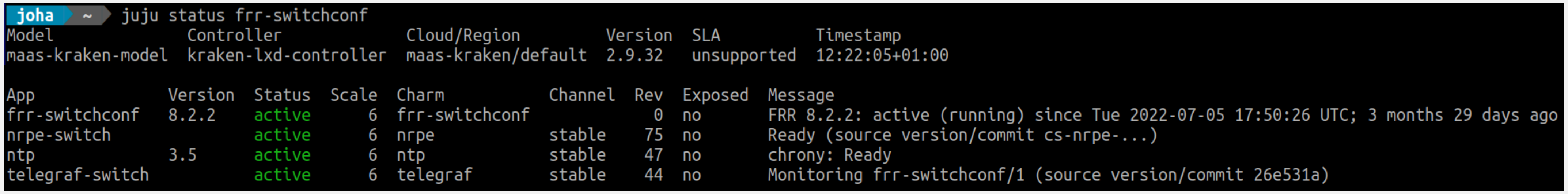
Firmware:

Version	4.6.5
Date	03/08/2016

1 NUMA NODE

Node 0

CPU cores	2 (0-1)
Memory	8 GiB
Storage	15.8 GB over 1 disk
Network	34 interfaces



```
 joha  ~  juju status frr-switchconf
Model             Controller            Cloud/Region          Version  SLA          Timestamp
maas-kraken-model  kraken-lxd-controller  maas-kraken/default  2.9.32   unsupported  12:22:05+01:00

App             Version  Status  Scale  Charm           Channel  Rev  Exposed  Message
frr-switchconf  8.2.2    active      6  frr-switchconf             0  no       FRR 8.2.2: active (running) since Tue 2022-07-05 17:50:26 UTC; 3 months 29 days ago
nrpe-switch              active      6  nrpe            stable    75  no       Ready (source version/commit cs-nrpe-...)
ntp             3.5      active      6  ntp             stable    47  no       chrony: Ready
telegraf-switch          active      6  telegraf        stable    44  no       Monitoring frr-switchconf/1 (source version/commit 26e531a)
```

PUBLIC

CURRENT ISSUES AND FUTURE WORK

Deploying Charmed OpenStack does not necessarily give you all services you need

OPENSTACK NAS SHORTCOMINGS

- NFS Ganesha using the manila-ganesha charm integrated well but very unstable
- No (?) known charms providing SMB/CIFS services. Our fix:
  - Installed Samba CTDB cluster in empty Juju units on LXD, with underlying CephFS
  - SSSD (Active Directory) integration
  - Should definitely be charmed
  - Manila/Dashboard integration would be great
- https://charmhub.io/ceph-nfs looks promising (Pacific and above) but does not seem to integrate with Manila

BACKUP IS YOUR PROBLEM

- CephFS backups (over Samba) solved traditionally with backup appliance (Rubrik). Super performance!
- Instances will be backed by TrilioVault
  - Is charmed! Deploy via Juju to LXDs
  - Has dashboard plugin
  - Backup and restore by the user
  - Software only solution
- Buckets in ceph-radosgw is also hard:
  - Versioning+replication ➔ complex, requires two Ceph clusters
  - Something create with `rclone`?

THANK YOU FOR YOUR ATTENTION!

Johan Hallbäck | Senior Cloud Architect
johan.hallbaeck@ibeo-as.com | Charmhub Discourse: hallback | Charmhub Chat: johan-hallback