OpenStack HA - reliability and scalability

Michał Dulko

Intel Technology Poland

September 26th, 2016

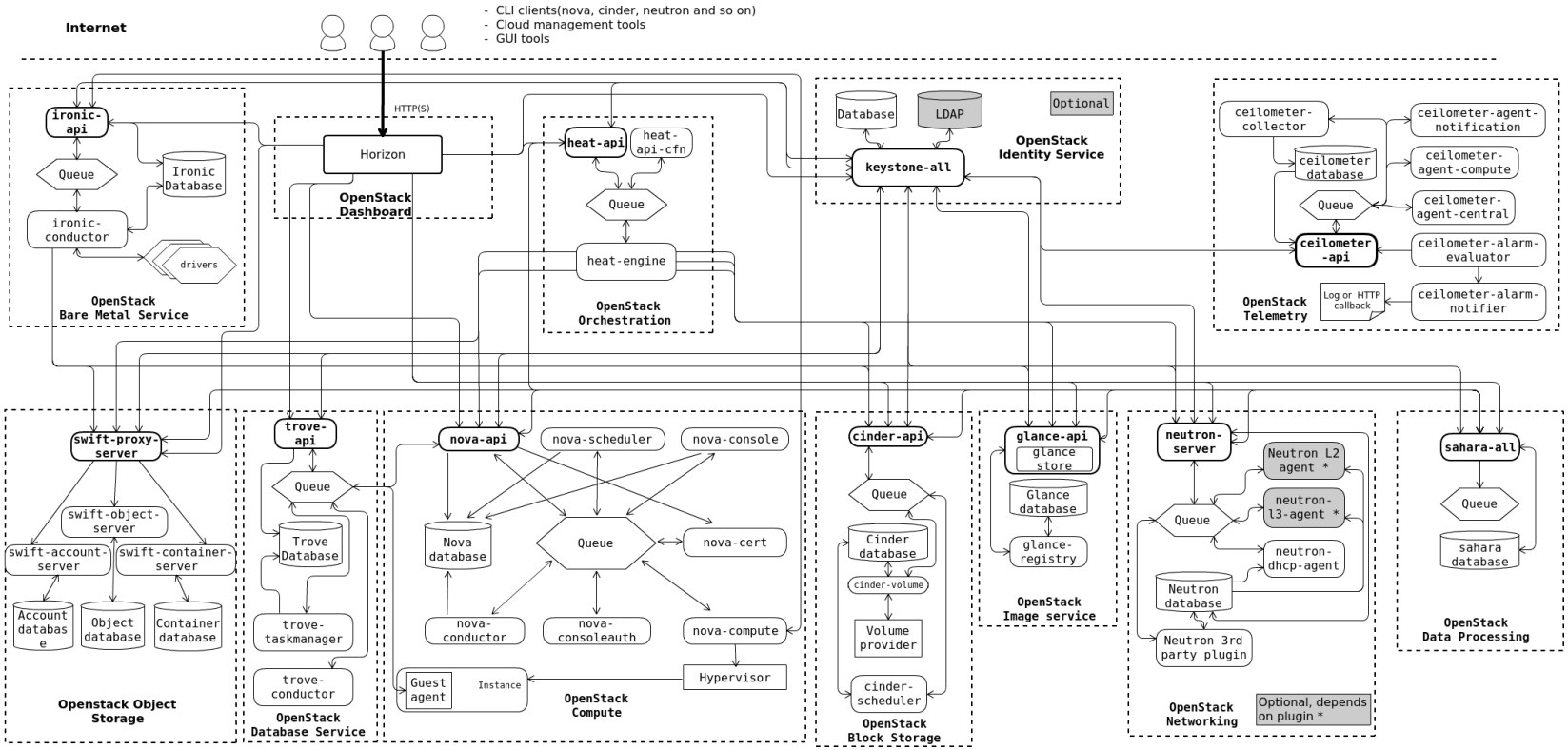
Introduction

High availability

High availability is a characteristic of a system, which aims to ensure an agreed level of operational performance for a higher than normal period.
There are three principles of system design in high availability engineering:

1. Elimination of single points of failure.
2. Reliable crossover.
3. Detection of failures as they occur.

OpenStack Architecture



What's actually in there?

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)
- Database (MySQL, PostgreSQL)

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)
- Database (MySQL, PostgreSQL)
- Message Queue (RabbitMQ, zmq, Apache Kafka))

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)
- Database (MySQL, PostgreSQL)
- Message Queue (RabbitMQ, zmq, Apache Kafka))
- Object storage (Swift, Ceph)

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)
- Database (MySQL, PostgreSQL)
- Message Queue (RabbitMQ, zmq, Apache Kafka))
- Object storage (Swift, Ceph)
- Block storage (Ceph, . . .)

What's actually in there?

- OpenStack services (Keystone, Nova, Neutron, Glance, *Cinder*, *Swift*, . . .)
- Database (MySQL, PostgreSQL)
- Message Queue (RabbitMQ, zmq, Apache Kafka))
- Object storage (Swift, Ceph)
- Block storage (Ceph, . . .)
- *Virtualized networking layer*

Shared services

Database

- Typically - Galera cluster (it's magic!).
- Running on 3, 5, 7, . . . nodes for quorum.
- OpenStack is battle-tested on Galera.

Message Queue

- Clustered RabbitMQ.
- Again running on 3, 5, 7, ... nodes for quorum.
- Erlang's internal database (Mnesia) is responsible for keeping state consistent.
- Running RabbitMQ, especially in HA, is considered non-trivial.

Object store

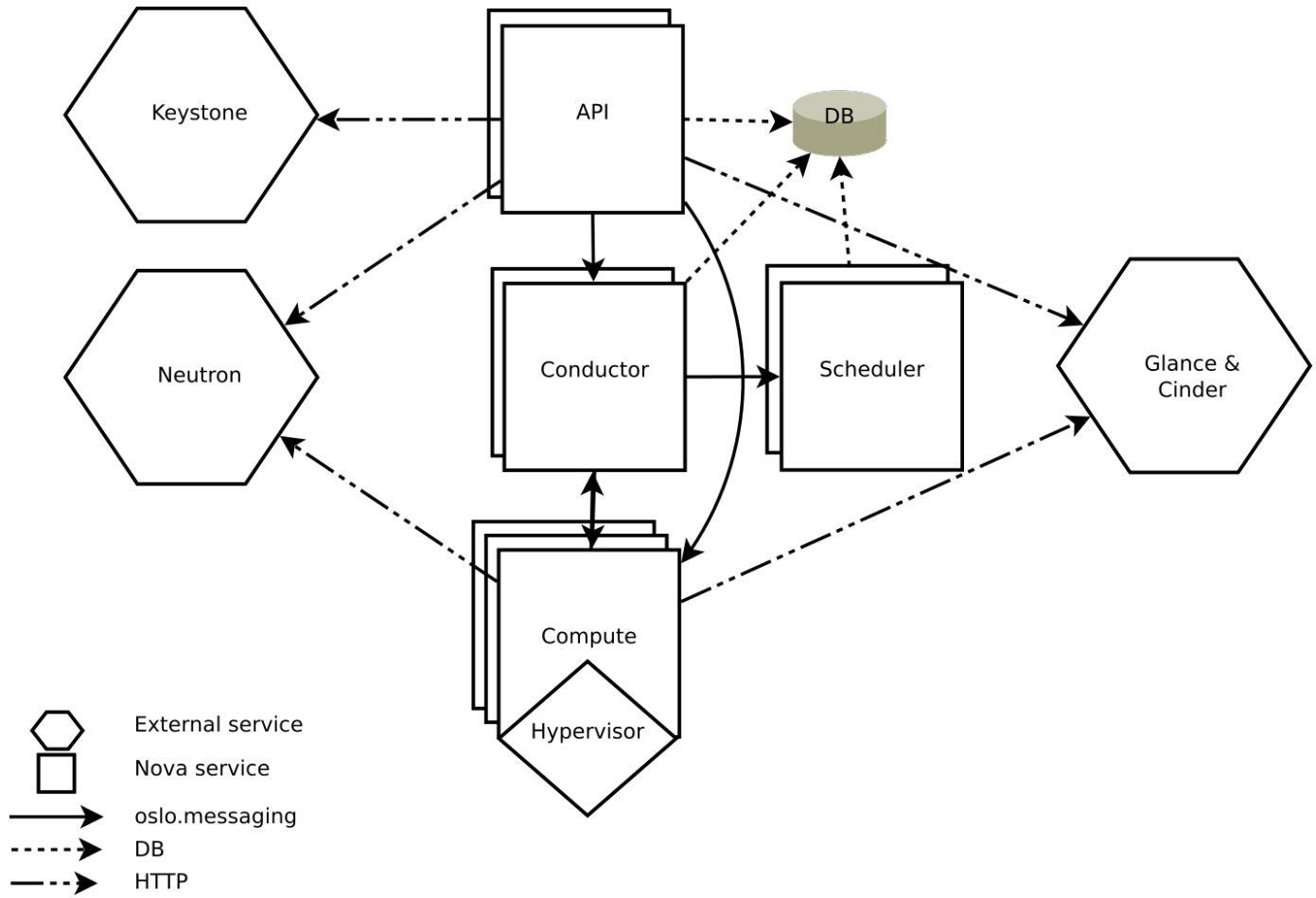
- Ceph
  - Has it's own ways of being reliable.

Object store

- Ceph
  - Has it's own ways of being reliable.
- Swift
  - Runs a "ring", which is basically a consistent hash ring.
  - You need to make sure to configure Swift to replicate objects.

OpenStack services

Nova architecture



OpenStack services types

- Communication
  - REST API services (nova-api, cinder-api, glance-api, Keystone)
  - Message-queue bound services (**nova-conductor**, nova-compute, **cinder-volume**)

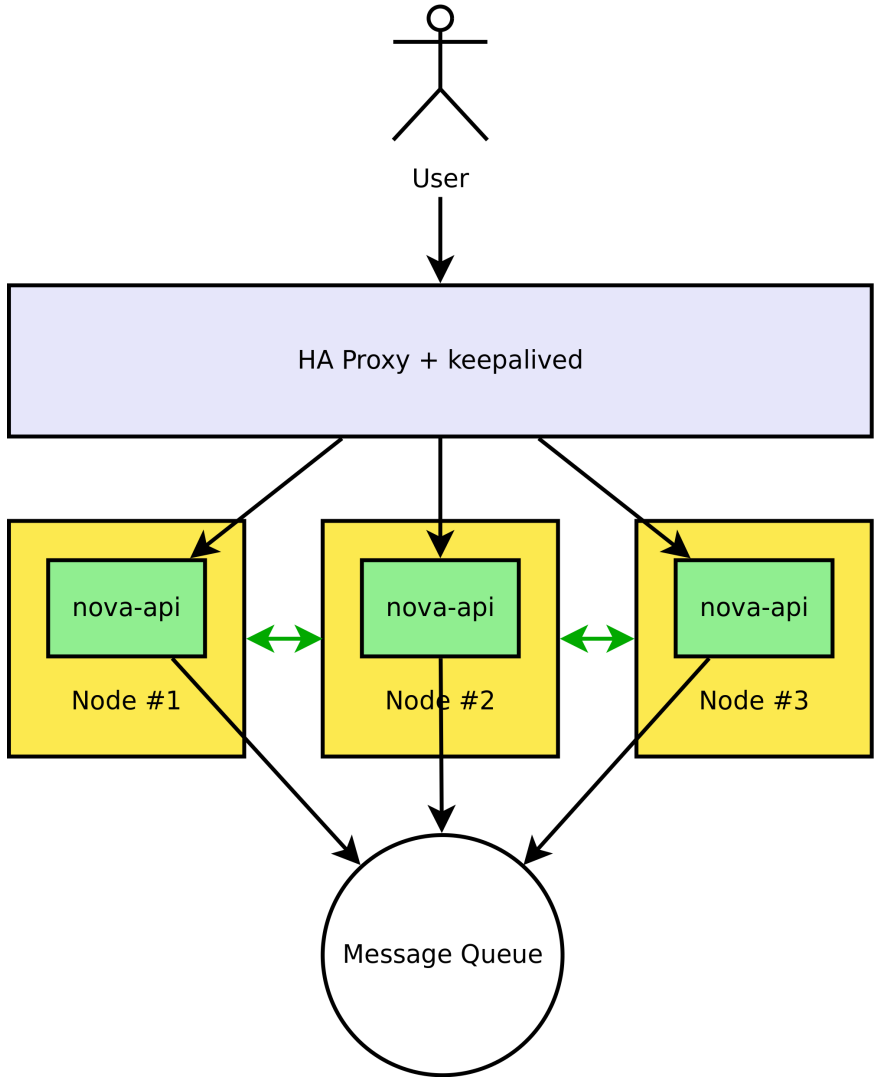
OpenStack services types

- Communication
  - REST API services (nova-api, cinder-api, glance-api, Keystone)
  - Message-queue bound services (**nova-conductor**, nova-compute, **cinder-volume**)
- Statefullness
  - Stateless, *shared state* (nova-api, **nova-conductor**)
  - Stateful (**cinder-volume**, nova-compute)

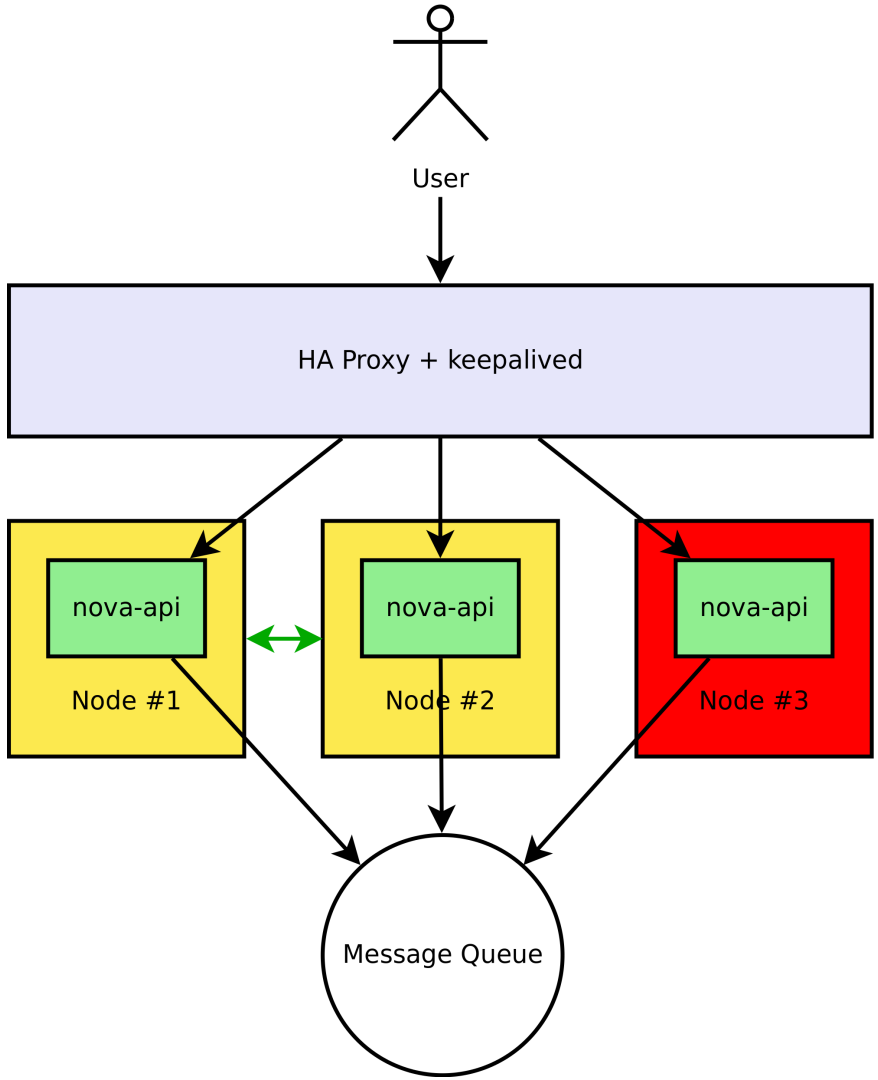
REST API services

- Keystone is run on Apache, rest are either standalone Python services or both.
- You're supposed to run them behind HAProxy.

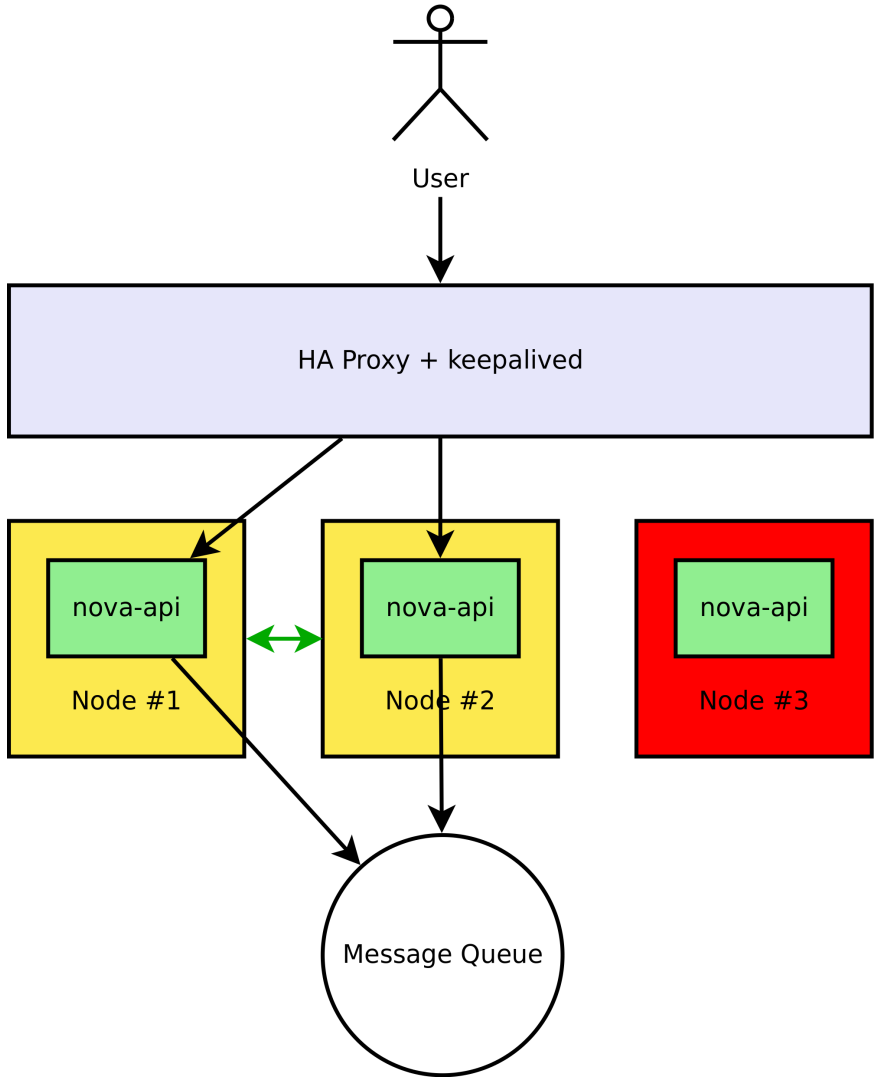
HAProxy + REST API



HAProxy + REST API



HAProxy + REST API



Message queue-bound services

- If stateless - just run multiple of them on controller nodes in A/A mode.

Message queue-bound services

- If stateless - just run multiple of them on controller nodes in A/A mode.
- If stateful - uh, oh...

Message queue-bound services

- If stateless - just run multiple of them on controller nodes in A/A mode.
- If stateful - uh, oh...
  - You need to run it as A/P service...

Message queue-bound services

- If stateless - just run multiple of them on controller nodes in A/A mode.
- If stateful - uh, oh...
  - You need to run it as A/P service...
  - ...so you'll need some cluster management software like Pacemaker to monitor and keep it running...

Message queue-bound services

- If stateless - just run multiple of them on controller nodes in A/A mode.
- If stateful - uh, oh...
  - You need to run it as A/P service...
  - ...so you'll need some cluster management software like Pacemaker to monitor and keep it running...
  - ...and some fencing software to protect from split-brains and zombie services.

Pacemaker 101

- Distributed cluster management software
- Features include:
  - awareness of other applications in the stack
  - a shared quorum implementation and calculation
  - data integrity through fencing
  - automated recovery of instances to ensure capacity
- Configurable and extendable through OCF (*Open Cluster Framework*) agents/scripts.
- Pacemaker is rather heavy, so OpenStack projects are aiming to get as many services A/A capable.

Fencing

- In case of non-responding application we don't know if it's dead or a network partition occurred.
- To make sure that we won't have two A/P service instances running, we need to fence the node where dead service instance resides on.

Fencing

- In case of non-responding application we don't know if it's dead or a network partition occurred.
- To make sure that we won't have two A/P service instances running, we need to fence the node where dead service instance resides on.
- Software solution: STONITH - *Shoot The Other Node In The Head*.
  - UPS (Uninterruptible Power Supply)
  - PDU (Power Distribution Unit)
  - Blade power control devices
  - Lights-out devices
- Be aware - this complicates the system even more!

Neutron HA

- Liberty
  - DVR - Distributed Virtual Router - east/west routing on compute node, floating IP resolved on compute node also. The SNAT (for fixed IP) is on centralized network node, only one network node can be available per installation
  - L3 HA - multiple network node in active/passive configuration (VRRP, keepalived) - this is single network node resolving east/west routing, floating and and SNAT on single node.
  - DHCP, metadata agents - A/A, multiple agents are working on each network node.
- Mitaka
  - DVR-HA - Use DVR and multiple active/passive network nodes for SNAT.

Resources and further help

- OpenStack High Availability Guide
- Mirantis OpenStack 7.0 Reference Architecture (might be a little outdated)
- Pacemaker documentation
- Neutron DVR Documentation
- Neutron L3 VRRP Documentation
- #openstack-ha IRC channel (freenode)

Thank you!

https://github.com/dulek/openstack-meetup-wroclaw-ha

remind me to switch to next slide for Q&A

Legal Notices and Disclaimers

- Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.
- Intel, the Intel logo and others are trademarks of Intel Corporation in the U.S. and/or other countries.
- *Other names and brands may be claimed as the property of others.