Doctor project - Graduation proposal

The OPNFV Doctor team

August 25, 2017

LINUX FOUNDATION
COLLABORATIVE PROJECTS

AGENDA

- Introduction to OPNFV Doctor project
- Achievements
- Metrics

OPNFV

OPNFV Doctor project - Introduction

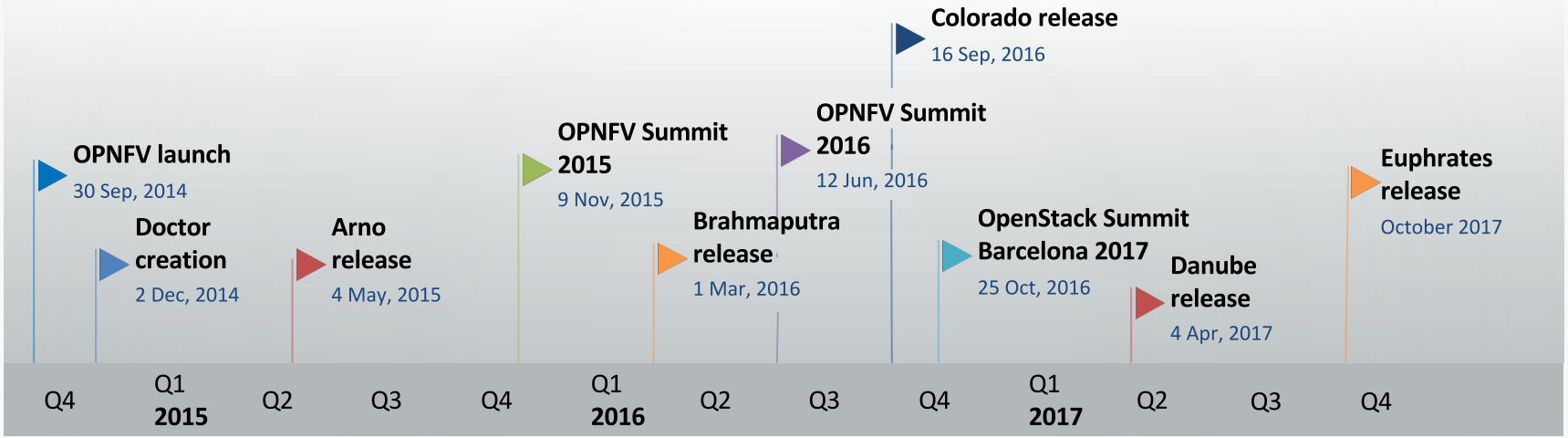
- Goal:
  - Develop and build fault management and maintenance framework for high availability of Network Services running on top of virtualized infrastructure.

  ➔ Proposed with a very clear target / key feature:

  - Immediate notification of unavailability of virtualized resources from VIM to Consumer
- Contributing organizations:
  - NEC (PTL: Ryota Mibu), AT&T, Cisco, Cloudbase Solutions, Corenova, Ericsson, Huawei, Intel, KDDI, KT, Nokia, NTT DOCOMO, Spirent, Sprint, Telecom Italia, ZTE
- https://wiki.opnfv.org/display/doctor/

OPNFV

OPNFV Doctor project – Timeline



ARNO
- Requirement document

BRAHMAPUTRA
- Ceilometer "Immediate Notification"
- Nova "Mark Host Down"
- Functional test cases
- PoC demo at OPNFV Summit
- Documentation updates

COLORADO
- Nova: "Get valid server state" and "Add notification for service status change"
- Integration of Congress as Doctor Inspector
- Extended functional tests
- PoC demo at OPNFV Summit and OpenStack Summit
- Documentation updates
- OPNFV Plugfest

DANUBE
- Neutron "Port Status update"
- Inspector design guidelines
- Performance profiler
- Documentation updates
- OPNFV Plugfest

EUPHRATES
- Congress: parallel policy action execution for faster fencing, notification and recovery
- Maintenance specs
- Code refactoring to Python
- Collectd as Doctor Monitor

OPNFV

Workflow

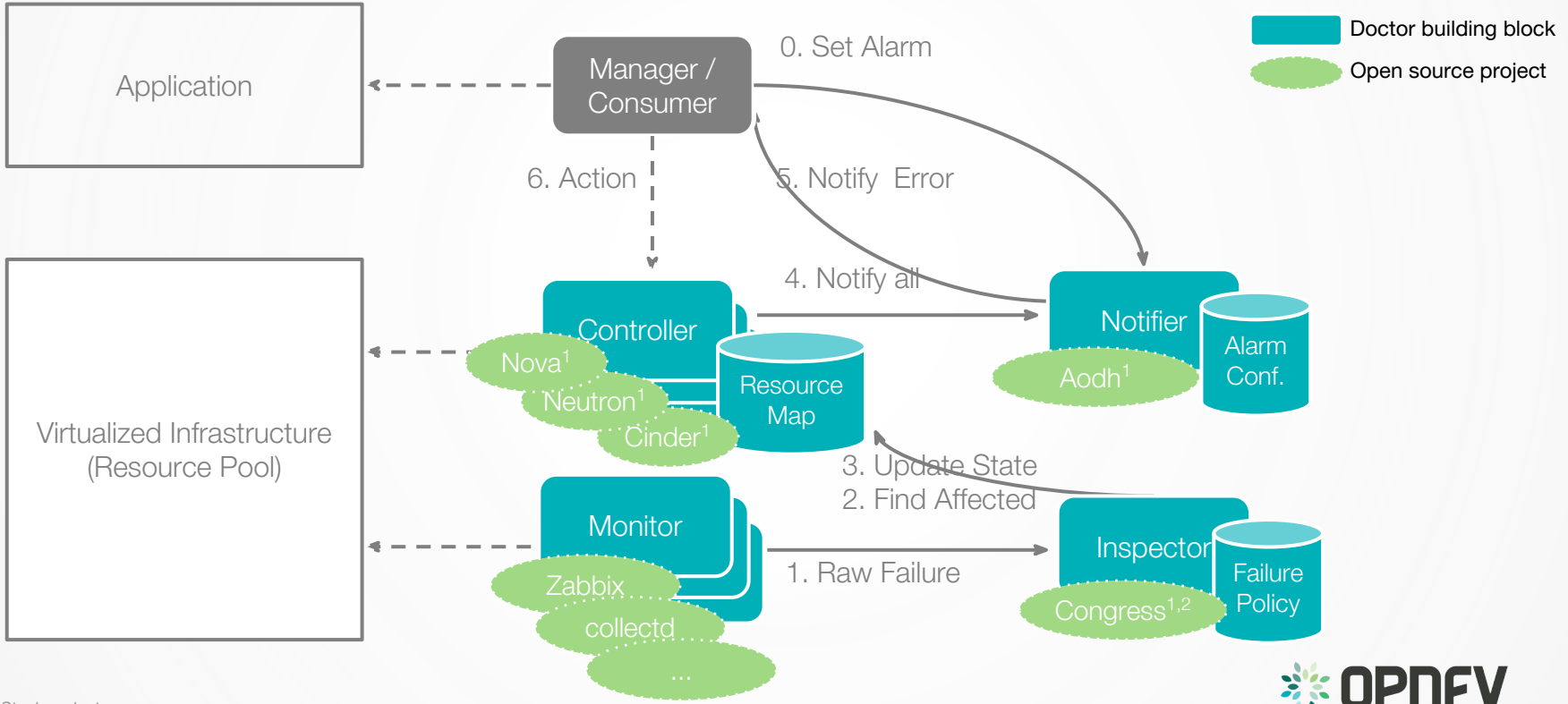
OPNFV

1. Doctor requirements document

http://artifacts.opnfv.org/doctor/docs/development_requirements/index.html

- Use cases and scenarios
  - Active-Standby configuration (1+1 redundancy):
    - Consumer of infrastructure has configured ACT-STBY
    - Fault in virtualized infrastructure (NFVI) → inform the Consumer to switch to STBY instance
  - Prevention actions based on fault prediction: Switch to STBY in case of a predicted fault
  - NFVI maintenance: inform Consumer(s) of affected hardware about planned maintenance
- Requirements
  1. **Monitor** physical and virtual resources and **detect** problems
  2. **Correlate** faults and **identify** affected virtual resources
  3. **Notification** of Consumer(s) of affected virtual resources
  4. Execute steps 1-3 in less than e.g. 1 second to avoid service disruption

2. Doctor architecture and integrated (OpenStack) projects



[1] OpenStack project
[2] Vitrage could be an alternative to Congress

3. Gap analysis and solution brainstorming (examples 1)

Resource state – *missing feature*

To be:
- Nova API shall support to change nova-compute state
- User shall be able to read OpenStack states and trust they are correct

As is (Kilo release):
- When a VM goes down due to a host HW, host OS or hypervisor failure, nothing happens in OpenStack. The VMs of a crashed host/hypervisor are reported to be live and OK through the OpenStack API.
- nova-compute state might change too slowly or the state is not reliable if expecting also VMs to be down.

Immediate notification – *deficiency in operation*

To be:
- VIM to immediately notify unavailability of virtual resource to VIM user.
- User shall only receive fault notifications related to owned resource(s).

As is (Kilo release):
- OpenStack Metering 'Ceilometer' can notify unavailability of resource.
- Due to innerworking of Ceilometer (polling of events), notification of faults takes seconds to few minutes.
- Performance issue for Ceilometer in medium to large scale deployments.

Solution brainstorming:
- Discussion with experts on best way to address the gap
- Outcome: e.g. BP spec for „mark nova-compute down"

OPNFV

3. Gap analysis and solution brainstorming (examples 2)

Maintenance discussions:

There has been discussions about planned maintenance together with OpenStack operators and with Nova and Craton project. There is yet no complete implementation plan as Nova will not accept the feature inside and operator tool project Craton is lacking contributors.

- OPS session in Austin summit:
  https://etherpad.openstack.org/p/AUS-ops-Nova-maint
- OPS session in Barcelona summit:
  https://etherpad.openstack.org/p/BCN-ops-informal-meetup
- Ops sessions Milan mid-cycle summit:
  https://etherpad.openstack.org/p/MIL-ops-telco-nfv
  https://etherpad.openstack.org/p/MIL-ops-inventory-and-fleet-management
- OpenStack Nova Blueprint:
  https://blueprints.launchpad.net/nova/+spec/maintenance-reason-to-server

4. Test cases and user manual

Run Test Script

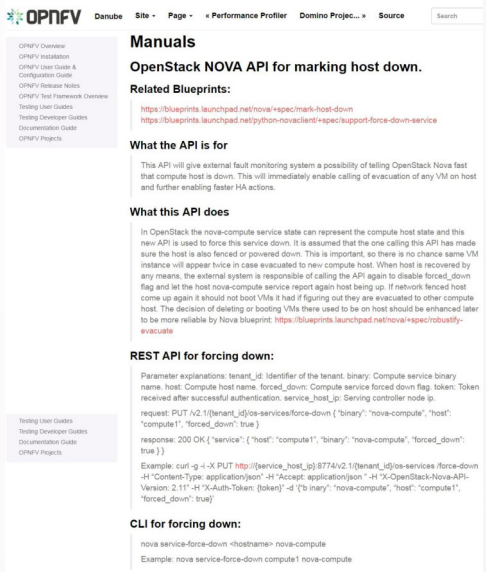
Doctor project has own testing script under doctor/tests. This test script can be used for functional testing against an OPNFV deployment.

Before running this script, make sure OpenStack env parameters are set properly following OpenStack CLI manual, so that Doctor Inspector can operate OpenStack services.

Then, you can run the script as follows:

```
git clone https://gerrit.opnfv.org/gerrit/doctor
cd doctor/tests
export INSTALLER_TYPE=local
export INSPECTOR_TYPE=sample
./run.sh
```

- End to end test cases
  - Upstream: unit tests and scope-restricted functional tests upstream
  - Downstream: E2E functional tests will validate full systems integration
- Manuals
  - Requirement and design documents
  - User guide, config guide, API guide
  - How to use implemented blueprints
  - How to run the tests and interpret results
  - Doctor project “Solution brief”



OPNFV Danube Site Page « Performance Profiler Domino Projec... » Source

Manuals

OpenStack NOVA API for marking host down.

Related Blueprints:

https://blueprints.launchpad.net/nova/+spec/mark-host-down
https://blueprints.launchpad.net/python-novaclient/+spec/support-force-down-service

What the API is for

This API will give external fault monitoring system a possibility of telling OpenStack Nova fast that compute host is down. This will immediately enable calling of evacuation of any VM on host and further enabling faster HA actions.

What this API does

In OpenStack the nova-compute service state can represent the compute host state and this new API is used to force this service down. It is assumed that the one calling this API has made sure the host is also fenced or powered down. This is important, so there is no chance same VM instance will appear twice in case evacuated to new compute host. When host is recovered by any means, the external system is responsible of calling the API again to disable forced_down flag and let the host nova-compute service report again host being up. If network fenced host come up again it should not boot VMs it had if figuring out they are evacuated to other compute host. The decision of deleting or booting VMs there used to be on host should be enhanced later to be more reliable by Nova blueprint. https://blueprints.launchpad.net/nova/+spec/robustify-evacuate

REST API for forcing down:

Parameter explanations: tenant_id: Identifier of the tenant. binary: Compute service binary name. host: Compute host name. forced_down: Compute service forced down flag. token: Token received after successful authentication. service_host_ip: Serving controller node ip.

request: PUT /v2.1/{tenant_id}/os-services/force-down { "binary": "nova-compute", "host": "compute1", "forced_down": true }

response: 200 OK { "service": { "host": "compute1", "binary": "nova-compute", "forced_down": true } }

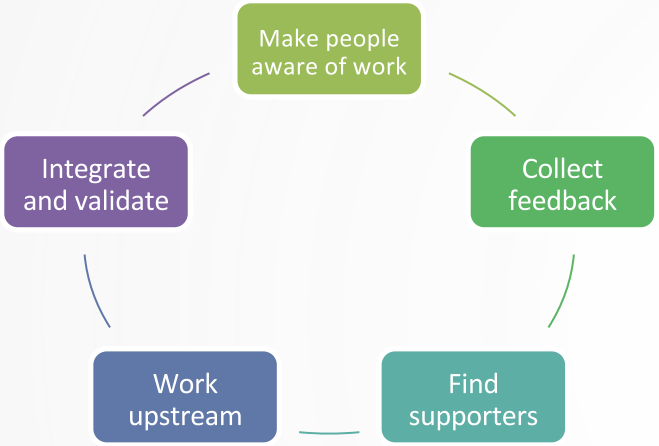
Example: curl -g -i -X PUT http://{service_host_ip}:8774/v2.1/{tenant_id}/os-services /force-down -H "Content-Type: application/json" -H "Accept: application/json " -H "X-OpenStack-Nova-API-Version: 2.11" -H "X-Auth-Token: {token}" -d '{"b inary": "nova-compute", "host": "compute1", "forced_down": true}'

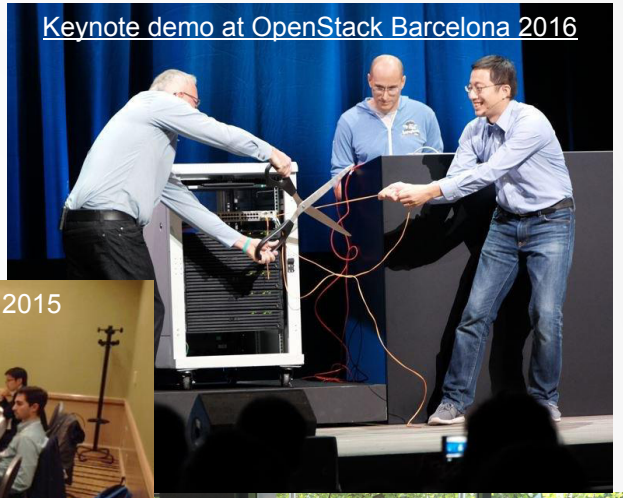
CLI for forcing down:

nova service-force-down <hostname> nova-compute

Example: nova service-force-down compute1 nova-compute

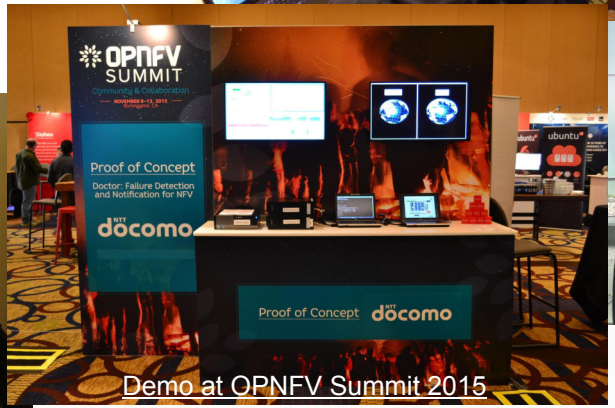
5. PoCs, demos and hackfests




Keynote demo at OpenStack Barcelona 2016


Meetup at OPNFV Summit 2015

PoC at OPNFV Summit 2016



Demo at OPNFV Summit 2015

PoC at OPNFV Summit 2017

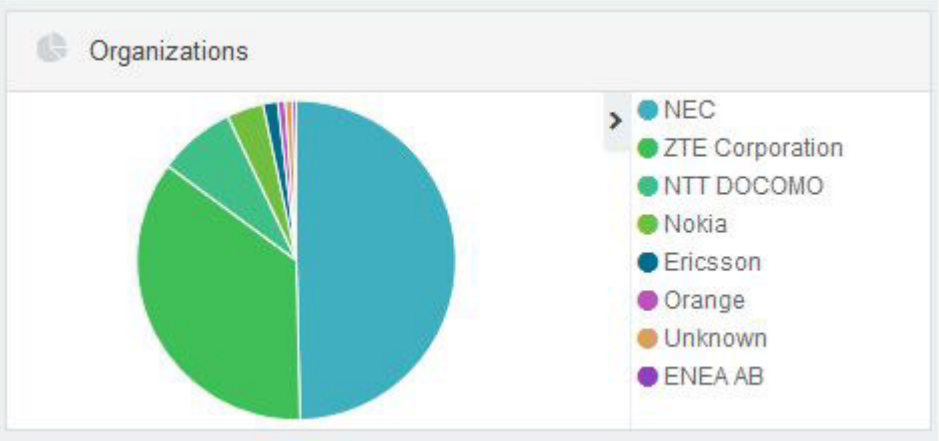
OPNFV

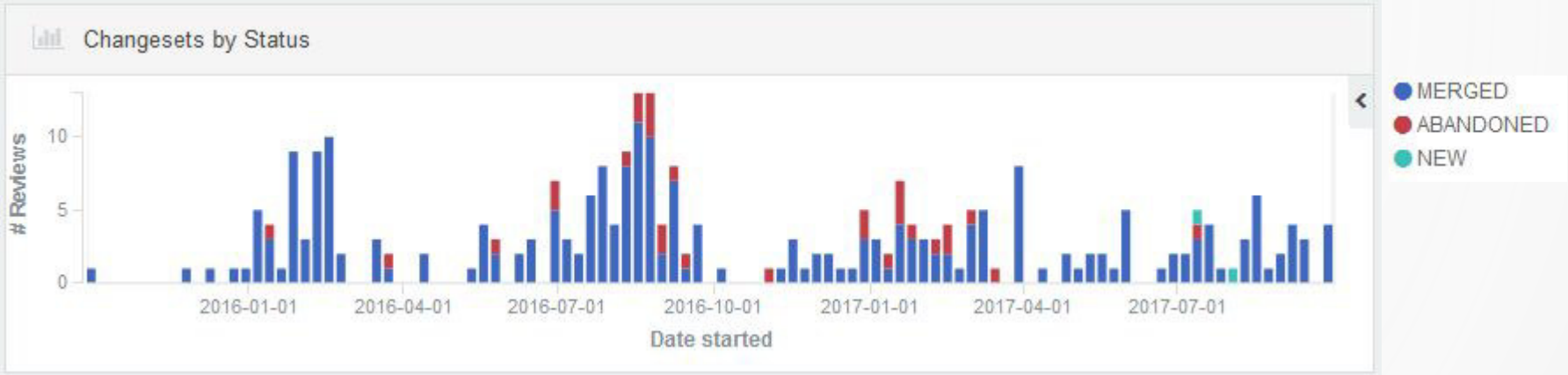
6. Upstream achievements

Project	Blueprint	Spec Drafter	Lead Developer	Status
Aodh	Event Alarm Evaluator	Ryota Mibu (NEC)	Ryota Mibu (NEC)	Completed (Liberty)
Nova	New nova API call to mark nova-compute down	Tomi Juvonen (Nokia)	Roman Dobosz (Intel)	Completed (Liberty)
	Support forcing service down	Tomi Juvonen (Nokia)	Carlos Goncalves (NEC)	Completed (Liberty)
	Get valid server state	Tomi Juvonen (Nokia)	Tomi Juvonen (Nokia)	Completed (Mitaka)
	Add notification for service status change	Balazs Gibizer (Ericsson)	Balazs Gibizer (Ericsson)	Completed (Mitaka)
Congress	Push Type Datasource Driver	Masahito Muroi (NTT)	Masahito Muroi (NTT)	Completed (Mitaka)
	Adds Doctor Driver	Masahito Muroi (NTT)	Masahito Muroi (NTT)	Completed (Mitaka)
Neutron	Port data plane status	Carlos Goncalves (NEC)	Carlos Goncalves (NEC)	Completed (Pike)

Metrics

268 # changesets
17 # changeset submitters





Future plans

- Fault management:
  - Extend framework with automated failure handling / healing
  - Work with proposed OpenStack SIG
    https://etherpad.openstack.org/p/self-healing-rocky-forum

- Maintenance
  - Keep working on maintenance topics
    in cooperation with OpenStack operators, Nova, Craton, ...

Summary

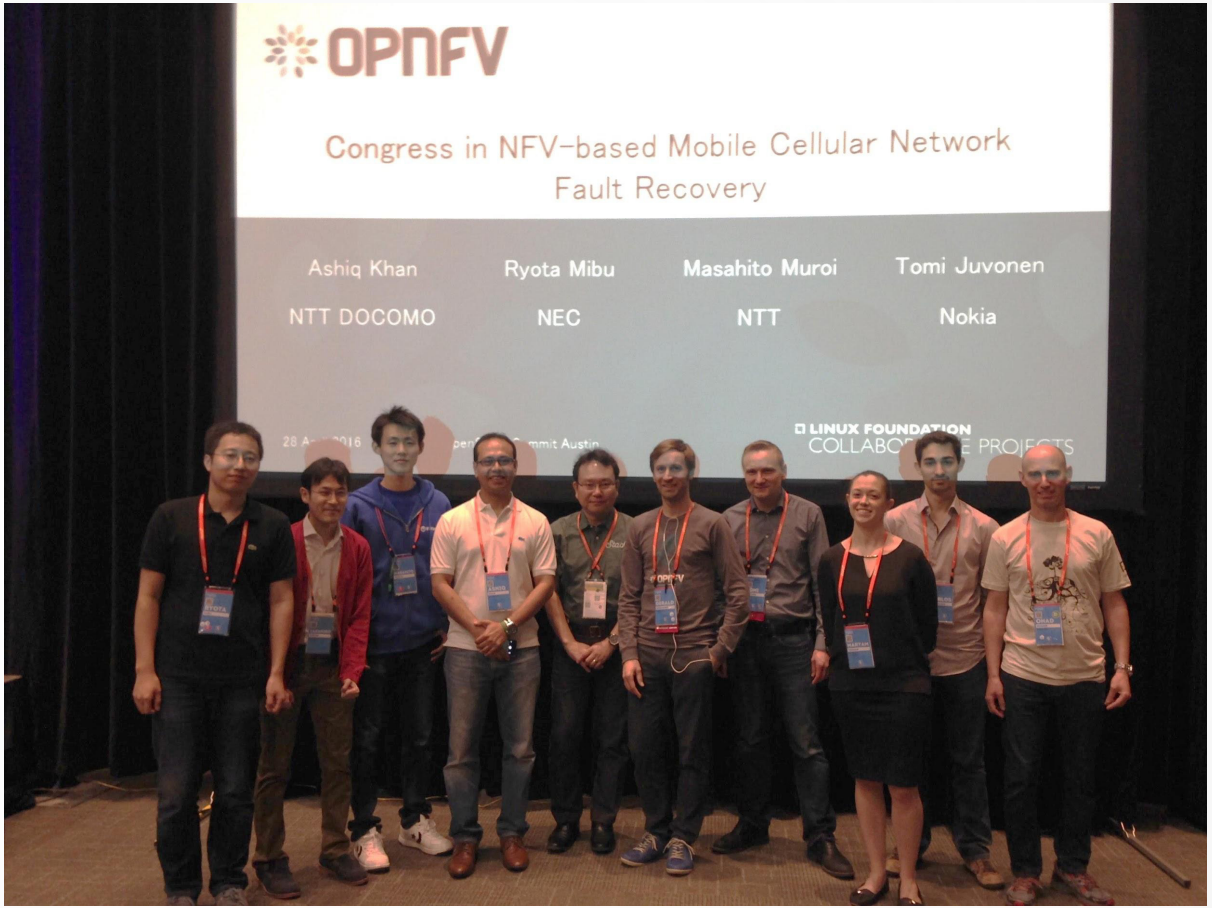
- Describe the problem being solved by project @ Project creation
  - Lack of fault detection, notification and recovery mechanism in OpenStack
  - OpenStacks inability in receiving and executing maintenance instructions
  - Requirements shall be produced to solve the problems above

- Project requirements

1. **Monitor** physical and virtual resources and **detect** problems/planned maintenance
2. **Correlate** faults and **identify** affected virtual resources
3. **Notification** of Consumer(s) of affected virtual resources
4. Execute steps 1-3 in less than e.g. 1 second to avoid service disruption

OPNFV

THANKS



OPNFV