Project Proposal: “DOCTOR” (NFVI Fault Management/Maintenance)

Contact: Ashiq Khan
Nov. 25, 2014
Problem description

- Virtualised Infrastructure Manager (VIM) e.g. OpenStack cannot detect certain Network Functions Virtualisation Infrastructure (NFVI) i.e. Resource Pool faults, which is necessary to detect and notify in order to ensure the proper functioning of EPC VNFs e.g. MME, S/P–GW.
  - EPC VNFs are often in ACT–SBY configuration and need to switch to SBY as soon as relevant faults are detected
  - NFVI encompasses Physical Machines, Hypervisors, Storage and Network elements

- In addition, VIM e.g. OpenStack needs to receive maintenance instruction from the operator/administrator
  - Empty certain Physical Machines (PMs) so that maintenance works could be performed
Features required

- OpenStack shall be able to collect certain fault information about the elements in its resource pool

- OpenStack shall be able to inform the users/client whose VMs are affected by the resource pool faults

- OpenStack shall be able to receive maintenance instruction (p8) for the elements in its resource pool
Use-Case 1: Fault Management

![Fault Management Diagram](https://wiki.opnfv.org/requirements_projects/doctor_fault_management_and_maintenance)

Fig. 1: Steps in Fault Management

[https://wiki.opnfv.org/requirements_projects/doctor_fault_management_and_maintenance](https://wiki.opnfv.org/requirements_projects/doctor_fault_management_and_maintenance)
Scope is OpenStack, internal APIs to resource pool and the northbound I/F to be exposed from OpenStack.

1. Fault notification (PM ID)
2. Fault/event correlation
3. Fault notification (VM ID, Fault ID)
4. Instruction on VM (VM ID, Operation ID)
5. Execute Instruction e.g. migrate/kill VM

Internal feature: Fault detection (1,2) ; northbound I/F: two messaging (3,4)
Use-Case 2: Maintenance

https://wiki.opnfv.org/requirements_projects/doctor_fault_management_and_maintenance
High-level message flow

- Only add #1 on the Fault Management (P4, 5)

**Scope**

- Resource Pool
- OpenStack
- User/Client
- Admin/Physical Resource manager

1. Resource Pool maintenance (Maintenance ID, Server ID)
2. Fault/event correlation
3. Notification (VM ID, Event ID)
4. Instruction on VM (VM ID, Operation ID)
5. Execute Instruction e.g. migrate VM

**The same as in Fault Management**

**Internal feature:** event correlation (2); northbound I/F: three messaging (1, 3, 4)

The objective is to free up the physical server prior to maintenance
Focus Area in Initial Development at VIM

- Receive fault or maintenance message regarding physical resource
- Identify affected logical resources
- Find user’s notification policy and receiver
- Send out fault notification
- Receive instruction/ACK on fault recovery
Scope

- OpenStack as VIM
- Fault items to be detected
  - Hardware faults to start with
- Fault detection mechanism in between OpenStack and the resource pool
- Northbound I/Fs of VIM

- Extensibility
  - Other NFVI elements
    - KVM, ODL
Deliverable (by Mar 2015)

• Documents
  – VIM northbound I/F specifications for fault management and NFVI maintenance
  – Implementation architecture and plan in OpenStack

• For implementation work (collaborative development) in OpenStack
  – Blueprint would be submitted for L-release
  – Blueprint could already be submitted for Kilo based on progress
# Timeline of Doctor Project

<table>
<thead>
<tr>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OPNFV</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Simultaneous release open</td>
<td>Declare participation in SI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Declare final release plan</td>
</tr>
<tr>
<td><strong>Doctor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Declare init</td>
<td>Declare final release plan</td>
</tr>
<tr>
<td></td>
<td>Release plan</td>
<td>Release plan</td>
</tr>
</tbody>
</table>

**Jan**

- CI Freeze
- Documentation starts
- 2nd Spec ready - refinement of the 1st
- Submit blueprint to Kilo

**Feb**

- CI Code Freeze
- API Freeze
- System test starts
- 1st release candidate

**Mar**

- 2nd Spec ready
- Participating PJs hold release review
- 2nd release: Preparation to join OpenStack
- 2nd Blueprint for L-release ready
- 1st Blueprint for L-release ready

- Gap Analysis (OS) Determine implementation scenario
- First Spec. ready
  - NB I/Fs
  - Messaging
  - Information elements
  - Implementation plan

Project DOCTOR
Resources

• Committers:
  – Technical: Ryota, Carlos, Tomi, Zhangyu, Palani and others
    • Responsible for the items shown in ▲
  – Documentation: Ashiq, Uli, Serge, Dirk

• Integration testing: would be determined later (after April 2015)
Appendix:
Infrastructure resources

- Mailing list
- Etherpad
- IRC, MeetingBot
- Telco (GoToMeeting)
- Wiki:
  - Code review: Gerrit (not necessary now)
  - Code repository: Git repo (not necessary now)
Initial Architecture: tentative

1. Set Alarm
2. Detect Failure
3. Find Affected
4. Update State
5. Notify

User-side Manager

Doctor

Nova

(High Health Monitor)

(Ironic)

VM

HV, etc.

H/W

Project DOCTOR
Initial Architecture: Alternative

1. Set Alarm
2. Fault Injection
3. Find Affected
4. Update State
5. Notify

User-side Manager
Nova
Alarm conf
(Health Monitor)
Ironic
VM
HV, etc.
H/W