ESCALATOR USER REQUIREMENTS

Release brahmaputra.1.0 (657dcc4)

OPNFV

May 19, 2016

Contents:

CHAPTER ONE

CONTRIBUTORS

Jie Hu (ZTE, hu.jie@zte.com.cn)
Qiao Fu (China Mobile, fuqiao@chinamobile.com)
Ulrich Kleber (Huawei, Ulrich.Kleber@huawei.com)
Maria Toeroe (Ericsson, maria.toeroe@ericsson.com)
Sama, Malla Reddy (DOCOMO, sama@docomolab-euro.com)
Zhong Chao (ZTE, chao.zhong@zte.com.cn)
Julien Zhang (ZTE, zhang.jun3g@zte.com.cn)
Yuri Yuan (ZTE, yuan.yue@zte.com.cn)
Zhipeng Huang (Huawei, huangzhipeng@huawei.com)
Jia Meng (ZTE, meng.jia@zte.com.cn)
Liyi Meng (Ericsson, liyi.meng@ericsson.com)
Pasi Vaananen (Stratus, pasi.vaananen@stratus.com)

CHAPTER TWO

SCOPE

This document describes the user requirements on the smooth upgrade function of the NFVI and VIM with respect to the upgrades of the OPNFV platform from one version to another. Smooth upgrade means that the upgrade results in no service outage for the end-users. This requires that the process of the upgrade is automatically carried out by a tool (code name: Escalator) with pre-configured data. The upgrade process includes preparation, validation, execution, monitoring and conclusion.

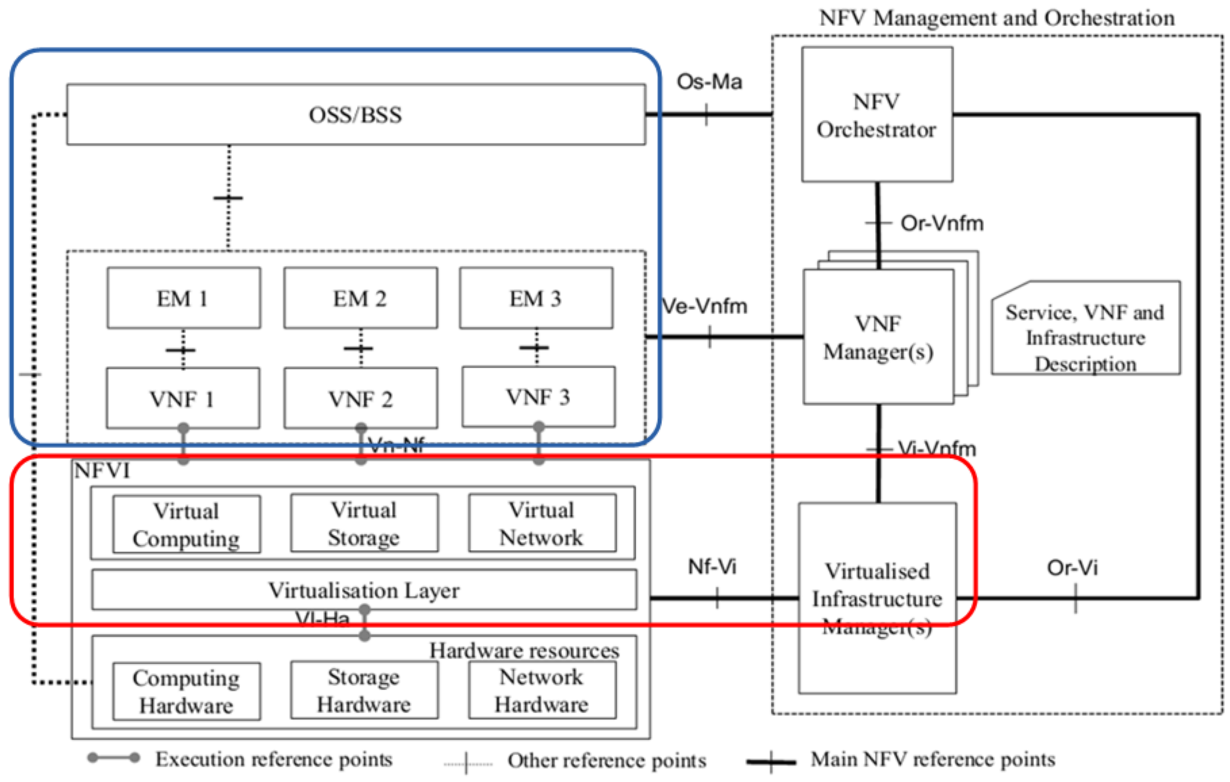
The requirements are defined in a stepwise approach, i.e. in the first phase focusing on the upgrade of the VIM then widening the scope to the NFVI.

The requirements may apply to different NFV functions (NFVI, or VIM, or both of them). They will be classified in the Appendix of this document.

The objects being upgraded described in this document are software modules covered by red box in the picture below which includes: VIM and NFVI.

The target of the upgrade is to reduce the impact on the applications in the blue box below as much as possible.

Please keep in mind that the upgrade tool does not take Vi-Vnfm and Or-Vi into consideration. In other words, these two interfaces may not provided service normally during upgrade procedure.


NFV Management and Orchestration
OSS/BSS
Os-Ma
NFV Orchestrator
Or-Vnfm
EM 1
EM 2
EM 3
Ve-Vnfm
VNF Manager(s)
Service, VNF and Infrastructure Description
VNF 1
VNF 2
VNF 3
Vn-Nf
Vi-Vnfm
NFVI
Virtual Computing
Virtual Storage
Virtual Network
Nf-Vi
Virtualised Infrastructure Manager(s)
Or-Vi
Virtualisation Layer
Vl-Ha
Hardware resources
Computing Hardware
Storage Hardware
Network Hardware
Execution reference points
Other reference points
Main NFV reference points

CHAPTER

THREE

TERMINOLOGY

3.1 Terminologies

Operator The term refers to network service providers and Virtual Network Function (VNF) providers.

End-User The term refers to a subscriber of the Operator's services.

Network Service The term refers to a service provided by an Operator to its end-users using a set of (virtualized) Network Functions

Infrastructure Services The term refers to services provided by the NFV Infrastructure to the VNFs as required by the Management & Orchestration functions and especially the VIM. I.e. these are the virtual resources as perceived by the VNFs.

Smooth Upgrade The term refers to an upgrade that results in no service outage for the end-users.

Rolling Upgrade The term refers to an upgrade strategy, which upgrades a node or a subset of nodes at a time in a wave style rolling through the data centre. It is a popular upgrade strategy to maintain service availability.

Parallel Universe Upgrade The term refers to an upgrade strategy, which creates and deploys a new universe - a system with the new configuration - while the old system continues running. The state of the old system is transferred to the new system after sufficient testing of the new system.

Infrastructure Resource Model The term refers to the representation of infrastructure resources, namely: the physical resources, the virtualization facility resources and the virtual resources.

Physical Resource The term refers to a piece of hardware in the NFV infrastructure that may also include firmware enabling this piece of hardware.

Virtual Resource The term refers to a resource, which is provided as services built on top of the physical resources via the virtualization facilities; in particular, virtual resources are the resources on which VNFs are deployed. Examples of virtual resources are: VMs, virtual switches, virtual routers, virtual disks.

Visualization Facility The term refers to a resource that enables the creation of virtual environments on top of the physical resources, e.g. hypervisor, OpenStack, etc.

Upgrade Campaign The term refers to a choreography that describes how the upgrade should be performed in terms of its targets (i.e. upgrade objects), the steps/actions required of upgrading each, and the coordination of these steps so that service availability can be maintained. It is an input to an upgrade tool (Escalator) to carry out the upgrade.

Upgrade Duration The duration of an upgrade characterized by the time elapsed between its initiation and its completion. E.g. from the moment the execution of an upgrade campaign has started until it has been committed. Depending on the upgrade strategy, the state of the configuration and the upgrade target some parts of the system may be in a more vulnerable state with respect to service availbility.

Outage The period of time during which a given service is not provided is referred as the outage of that given service. If a subsystem or the entire system does not provide any service, it is the outage of the given subsystem or the system. Smooth upgrade means upgrade with no outage for the user plane, i.e. no VNF should experience service outage.

Rollback The term refers to a failure handling strategy that reverts the changes done by a potentially failed upgrade execution one by one in a reverse order. I.e. it is like undoing the changes done by the upgrade.

Backup The term refers to data persisted to a storage, so that it can be used to restore the system or a given part of it in the same state as it was when the backup was created assuming a cold restart. Changes made to the system from the moment the backup was created till the moment it is used to restore the (sub)system are lost in the restoration process.

Restore The term refers to a failure handling strategy that reverts the changes done, for example, by an upgrade by restoring the system from some backup data. This results in the loss of any change and data persisted after the backup was been taken. To recover those additional measures need to be taken if necessary (e.g. rollforward).

Rollforward The term refers to a failure handling strategy applied after a restore (from a backup) opertaion to recover any loss of data persisted between the time the backup has been taken and the moment it is restored. Rollforward requires that data that needs to survive the restore operation is logged at a location not impacted by the restore so that it can be re-applied to the system after its restoration from the backup.

Downgrade The term refers to an upgrade in which an earlier version of the software is restored through the upgrade procedure. A system can be downgraded to any earlier version and the compatibility of the versions will determine the applicable upgrade strategies and whether service outage can be avoided. In particular any data conversion needs special attention.

3.2 Abbreviations

NFVI The term is an abbreviation for Network Function Virtualization Infrastructure; sometimes it is also referred as data plane in this document. The NFVI provides the virtual resources to the virtual network functions under the control of the VIM.

VIM The term is an abbreviation for Virtual Infrastructure Manager; sometimes it is also referred as control plane in this document. The VIM controls and manages the NFVI compute, network and storage resources to provide the required virtual resources to the VNFs.

CHAPTER

FOUR

BACKGROUND

4.1 Upgrade Objects

4.1.1 Physical Resource

Most cloud infrastructures support the dynamic addition and removal of hardware. Accordingly a hardware upgrade could be done by adding the new piece of hardware and removing the old one. From the persepctive of smooth upgrade the orchestration/scheduling of these actions is the primary concern.

Upgrading a physical resource may involve as well the upgrade of its firmware and/or modifying its configuration data. This may require the restart of the hardware.

4.1.2 Virtual Resources

Addition and removal of virtual resources may be initiated by the users or be a result of an elasticity action. Users may also request the upgrade of their virtual resources using a new VM image.

On the other hand changes in the infrastructure, namely, in the hardware and/or the virtualization facility resources may result in the upgrade of the virtual resources. For example if by some reason the hypervisor is changed and the current VMs cannot be migrated to the new hypervisor - they are incompatible - then the VMs need to be upgraded too. This is not something the NFVI user (i.e. VNFs) would know about.

4.1.3 Virtualization Facility Resources

Based on the functionality they provide, virtualization facility resources could be divided into computing node, networking node, storage node and management node.

The possible upgrade objects in these nodes are considered below: (Note: hardware based virtualization may be considered as virtualization facility resource, but from escalator perspective, it is better to consider it as part of the hardware upgrade.)

Computing node

1. OS Kernel
2. Hypvervisor and virtual switch
3. Other kernel modules, like drivers
4. User space software packages, like nova-compute agents and other control plane programs.

Updating 1 and 2 will cause the loss of virtualzation functionality of the compute node, which may lead to the interruption of data plane services if the virtual resource is not redudant.

Updating 3 might have the same result.

Updating 4 might lead to control plane services interruption if not an HA deployment.

Networking node

1. OS kernel, optional, not all switches/routers allow the upgrade their OS since it is more like a firmware than a generic OS.
2. User space software package, like neutron agents and other control plane programs

Updating 1 if allowed will cause a node reboot and therefore leads to data plane service interruption if the virtual resource is not redundant.

Updating 2 might lead to control plane services interruption if not an HA deployment.

Storage node

1. OS kernel, optional, not all storage nodes allow the upgrade their OS since it is more like a firmware than a generic OS.
2. Kernel modules
3. User space software packages, control plane programs

Updating 1 if allowed will cause a node reboot and therefore leads to data plane services interruption if the virtual resource is not redundant.

Update 2 might result in the same.

Updating 3 might lead to control plane services interruption if not an HA deployment.

Management node

1. OS Kernel
2. Kernel modules, like driver
3. User space software packages, like database, message queue and control plane programs.

Updating 1 will cause a node reboot and therefore leads to control plane services interruption if not an HA deployment. Updating 2 might result in the same.

Updating 3 might lead to control plane services interruption if not an HA deployment.

4.2 Upgrade Granularity

The granularity of an upgrade can be characterized from two perspective: - the physical dimension and - the software dimension

4.2.1 Physical Dimension

The physical dimension characterizes the number of similar upgrade objects targeted by the upgrade, i.e. whether it is full / partial upgrade of a data centre, cluster, zone. Because of the upgrade of a data centre or a zone, it may be divided into several batches. Thus there is a need for efficiency in the execution of upgrades of potentially huge number of upgrade objects while still maintain availability to fulfill the requirement of smooth upgrade.

The upgrade of a cloud environment (cluster) may also be partial. For example, in one cloud environment running a number of VNFs, we may just try to upgrade one of them to check the stability and performance, before we upgrade all of them. Thus there is a need for proper organization of the artifacts associated with the different upgrade objects. Also the different versions should be able to coextist beyond the upgrade period.

From this perspective special attention may be needed when upgrading objects that are collaborating in a redundancy schema as in this case different versions not only need to coexist but also collaborate. This puts requirement on the upgrade objects primarily. If this is not possible the upgrade campaign should be designed in such a way that the proper isolation is ensured.

4.2.2 Software Dimension

The software dimension of the upgrade characterizes the upgrade object type targeted and the combination in which they are upgraded together.

Even though the upgrade may initially target only one type of upgrade object, e.g. the hypervisor the dependency of other upgrade objects on this initial target object may require their upgrade as well. I.e. the upgrades need to be combined. From this perspective the main concern is compatibility of the dependent and sponsor objects. To take into consideration of these dependencies they need to be described together with the version compatility information. Breaking dependencies is the major cause of outages during upgrades.

In other cases it is more efficient to upgrade a combination of upgrade objects than to do it one by one. One aspect of the combination is how the upgrade packages can be combined, whether a new image can be created for them before hand or the different packages can be installed during the upgrade independently, but activated together.

The combination of upgrade objects may span across layers (e.g. software stack in the host and the VM of the VNF). Thus, it may require additional coordination between the management layers.

With respect to each upgrade object type and even stacks we can distingush major and minor upgrades:

Major Upgrade

Upgrades between major releases may introducing significant changes in function, configuration and data, such as the upgrade of OPNFV from Arno to Brahmaputra.

Minor Upgrade

Upgrades inside one major releases which would not leads to changing the structure of the platform and may not infect the schema of the system data.

4.3 Scope of Impact

Considering availability and therefore smooth upgrade, one of the major concerns is the predictability and control of the outcome of the different upgrade operations. Ideally an upgrade can be performed without impacting any entity in the system, which means none of the operations change or potentially change the behaviour of any entity in the system in an uncotrolled manner. Accordingly the operations of such an upgrade can be performed any time while the system is running, while all the entities are online. No entity needs to be taken offline to avoid such adverse effects. Hence such upgrade operations are referred as online operations. The effects of the upgrade might be activated next time it is used, or may require a special activation action such as a restart. Note that the activation action provides more control and predictability.

If an entity's behavior in the system may change due to the upgrade it may be better to take it offline for the time of the relevant upgrade operations. The main question is however considering the hosting relation of an upgrade object what hosted entities are impacted. Accordingly we can identify a scope which is impacted by taking the given upgrade object offline. The entities that are in the scope of impact may need to be taken offline or moved out of this scope i.e. migrated.

If the impacted entity is in a different layer managed by another manager this may require coordination because taking out of service some infrastructure resources for the time of their upgrade which support virtual resources used by VNFs that should not experience outages. The hosted VNFs may or may not allow for the hot migration of their VMs. In case of migration the VMs placement policy should be considered.

CHAPTER

FIVE

REQUIREMENTS

5.1 Upgrade duration

As the OPNFV end-users are primarily Telecom operators, the network services provided by the VNFs deployed on the NFVI should meet the requirement of 'Carrier Grade'.:

```
In telecommunication, a "carrier grade" or"carrier class" refers to a
system, or a hardware or software component that is extremely reliable,
well tested and proven in its capabilities. Carrier grade systems are
tested and engineered to meet or exceed "five nines" high availability
standards, and provide very fast fault recovery through redundancy
(normally less than 50 milliseconds). [from wikipedia.org]
```

"five nines" means working all the time in ONE YEAR except 5'15".

```
We have learnt that a well prepared upgrade of OpenStack needs 10
minutes. The major time slot in the outage time is used spent on
synchronizing the database. [from ' Ten minutes OpenStack Upgrade? Done!
' by Symantec]
```

This 10 minutes of downtime of the OpenStack services however did not impact the users, i.e. the VMs running on the compute nodes. This was the outage of the control plane only. On the other hand with respect to the preparations this was a manually tailored upgrade specific to the particular deployment and the versions of each OpenStack service.

The project targets to achieve a more generic methodology, which however requires that the upgrade objects fulfil certain requirements. Since this is only possible on the long run we target first the upgrade of the different VIM services from version to version.

Questions:

1. Can we manage to upgrade OPNFV in only 5 minutes?
2. Is it acceptable for end users ? Such as a planed service interruption will lasting more than ten minutes for software upgrade.
3. Will any VNFs still working well when VIM is down?

5.1.1 The maximum duration of an upgrade

The duration of an upgrade is related to and proportional with the scale and the complexity of the OPNFV platform as well as the granularity (in function and in space) of the upgrade.

5.1.2 The maximum duration of a roll back when an upgrade is failed

The duration of a roll back is short than the corresponding upgrade. It depends on the duration of restore the software and configure data from pre-upgrade backup / snapshot.

5.1.3 The maximum duration of a VNF interruption (Service outage)

Since not the entire process of a smooth upgrade will affect the VNFs, the duration of the VNF interruption may be shorter than the duration of the upgrade. In some cases, the VNF running without the control from of the VIM is acceptable.

5.2 Pre-upgrading Environment

System is running normally. If there are any faults before the upgrade, it is difficult to distinguish between upgrade introduced and the environment itself.

The environment should have the redundant resources. Because the upgrade process is based on the business migration, in the absence of resource redundancy,it is impossible to realize the business migration, as well as to achieve a smooth upgrade.

Resource redundancy in two levels:

NFVI level: This level is mainly the compute nodes resource redundancy. During the upgrade, the virtual machine on business can be migrated to another free compute node.

VNF level: This level depends on HA mechanism in VNF, such as: active-standby, load balance. In this case, as long as business of the target node on VMs is migrated to other free nodes, the migration of VM might not be necessary.

The way of redundancy to be used is subject to the specific environment. Generally speaking, During the upgrade, the VNF's service level availability mechanism should be used in higher priority than the NFVI's. This will help us to reduce the service outage.

5.3 Release version of software components

This is primarily a compatibility requirement. You can refer to Linux/Python Compatible Semantic Versioning 3.0.0:

Given a version number MAJOR.MINOR.PATCH, increment the:

MAJOR version when you make incompatible API changes,

MINOR version when you add functionality in a backwards-compatible manner,

PATCH version when you make backwards-compatible bug fixes.

Some internal interfaces of OpenStack will be used by Escalator indirectly, such as VM migration related interface between VIM and NFVI. So it is required to be backward compatible on these interfaces. Refer to "Interface" chapter for details.

5.4 Work Flows

Describes the different types of requirements. To have a table to label the source of the requirements, e.g. Doctor, Multi-site, etc.

5.5 Basic Actions

This section describes the basic functions may required by Escalator.

5.5.1 Preparation (offline)

This is the design phase when the upgrade plan (or upgrade campaign) is being designed so that it can be executed automatically with minimal service outage. It may include the following work:

1. Check the dependencies of the software modules and their impact, backward compatibilities to figure out the appropriate upgrade method and ordering.
2. Find out if a rolling upgrade could be planned with several rolling steps to avoid any service outage due to the upgrade some parts/services at the same time.
3. Collect the proper version files and check the integration for upgrading.
4. The preparation step should produce an output (i.e. upgrade campaign/plan), which is executable automatically in an NFV Framework and which can be validated before execution.
    - The upgrade campaign should not be referring to scalable entities directly, but allow for adaptation to the system configuration and state at any given moment.
    - The upgrade campaign should describe the ordering of the upgrade of different entities so that dependencies, redundancies can be maintained during the upgrade execution
    - The upgrade campaign should provide information about the applicable recovery procedures and their ordering.
    - The upgrade campaign should consider information about the verification/testing procedures to be performed during the upgrade so that upgrade failures can be detected as soon as possible and the appropriate recovery procedure can be identified and applied.
    - The upgrade campaign should provide information on the expected execution time so that hanging execution can be identified
    - The upgrade campaign should indicate any point in the upgrade when coordination with the users (VNFs) is required.

5.5.2 Validation the upgrade plan / Checking the pre-requisites of System(offline / online)

The upgrade plan should be validated before the execution by testing it in a test environment which is similar to the product environment.

Before the upgrade plan being executed, the system healthy of the online product environment should be checked and confirmed to satisfy the requirements which were described in the upgrade plan. The sysinfo, e.g. which included system alarms, performance statistics and diagnostic logs, will be collected and analogized. It is required to resolve all of the system faults or exclude the unhealthy part before executing the upgrade plan.

5.5.3 Backup/Snapshot (online)

For avoid loss of data when a unsuccessful upgrade was encountered, the data should be back-upped and the system state snapshot should be taken before the execution of upgrade plan. This would be considered in the upgrade plan.

Several backups/Snapshots may be generated and stored before the single steps of changes. The following data/files are required to be considered:

1. running version files for each node.
2. system components' configuration file and database.
3. image and storage, if it is necessary.

Although the upper layer, which include VNFs and VNFMs, is out of the scope of Escalator, but it is still recommended to let it ready for a smooth system upgrade. The escalator could not guarantee the safe of VNFs. The upper layer should have some safe guard mechanism in design, and ready for avoiding failure in system upgrade.

5.5.4 Execution (online)

The execution of upgrade plan should be a dynamical procedure which is controlled by Escalator.

1. It is required to supporting execution ether in sequence or in parallel.
2. It is required to check the result of the execution and take the action according the situation and the policies in the upgrade plan.
3. It is required to execute properly on various configurations of system object. I.e. stand-alone, HA, etc.
4. It is required to execute on the designated different parts of the system. I.e. physical server, virtualized server, rack, chassis, cluster, even different geographical places.

5.5.5 Testing (online)

The testing after upgrade the whole system or parts of system to make sure the upgraded system(object) is working normally.

1. It is recommended to run the prepared test cases to see if the functionalities are available without any problem.
2. It is recommended to check the sysinfo, e.g. system alarms, performance statistics and diagnostic logs to see if there are any abnormal.

5.5.6 Restore/Roll-back (online)

When upgrade is failure unfortunately, a quick system restore or system roll-back should be taken to recovery the system and the services.

1. It is recommend to support system restore from backup when upgrade was failed.
2. It is recommend to support graceful roll-back with reverse order steps if possible.

5.5.7 Monitoring (online)

Escalator should continually monitor the process of upgrade. It is keeping update status of each module, each node, each cluster into a status table during upgrade.

1. It is required to collect the status of every objects being upgraded and sending abnormal alarms during the upgrade.
2. It is recommend to reuse the existing monitoring system, like alarm.
3. It is recommend to support pro-actively query.
4. It is recommend to support passively wait for notification.

Two possible ways for monitoring:

Pro-Actively Query requires NFVI/VIM provides proper API or CLI interface. If Escalator serves as a service, it should pass on these interfaces.

Passively Wait for Notification requires Escalator provides callback interface, which could be used by NFVI/VIM systems or upgrade agent to send back notification.

5.5.8 Logging (online)

Record the information generated by escalator into log files. The log file is used for manual diagnostic of exceptions.

1. It is required to support logging.
2. It is recommended to include time stamp, object id, action name, error code, etc.

5.5.9 Administrative Control (online)

Administrative Control is used for control the privilege to start any escalator's actions for avoiding unauthorized operations.

1. It is required to support administrative control mechanism
2. It is recommend to reuse the system's own secure system.
3. It is required to avoid conflicts when the system's own secure system being upgraded.

5.6 Requirements on Object being upgraded

Escalator focus on smooth upgrade. In practical implementation, it might be combined with installer/deplorer, or act as an independent tool/service. In either way, it requires targeting systems(NFVI and VIM) are developed/deployed in a way that Escalator could perform upgrade on them.

On NFVI system, live-migration is likely used to maintain availability because OPNFV would like to make HA transparent from end user. This requires VIM system being able to put compute node into maintenance mode and then isolated from normal service. Otherwise, new NFVI instances might risk at being schedule into the upgrading node.

On VIM system, availability is likely achieved by redundancy. This impose less requirements on system/services being upgrade (see PVA comments in early version). However, there should be a way to put the target system into standby mode. Because starting upgrade on the master node in a cluster is likely a bad idea.

1. It is required for NFVI/VIM to support **service handover** mechanism that minimize interruption to 0.001%(i.e. 99.999% service availability). Possible implementations are live-migration, redundant deployment, etc, (Note: for VIM, interruption could be less restrictive)
2. It is required for NFVI/VIM to restore the early version in a efficient way, such as **snapshot**.
3. It is required for NFVI/VIM to **migration data** efficiently between base and upgraded system.
4. It is recommend for NFV/VIM's interface to support upgrade orchestration, e.g. reading/setting system state.

5.7 Functional Requirements

Availability mechanism, etc.

5.8 Non-functional Requirements

CHAPTER SIX

USE CASES

This section describes the use cases in different system configuration to verify the requirements of Escalator.

6.1 Use case #1: Smooth upgrade in a HA configuration

For a system with HA configuration, the operator can use Escalator to smooth-upgrade NFVI/VIM components into a new version without any service outage.

When a compute node being upgraded, the VMs on the node may need to be migrated to other compute nodes to avoid service outage, so it is requred that there are enough redundant resources to migrate VMs on this compute node.

Before upgrade, the operator can use Escalator to check whether smooth upgrade conditions are all satisfied. These conditions include whether there are enough idle resources to migrate VMs during updrading, and whether the new version is compatible with the current one, etc. If there are some conditions not satisfied, Escalator will show them. Escalator can also provide the solutions if there is any, such as the number and configuration of spare compute nodes which are needed.

When upgrade starts, Escalator will also automatically check whether smooth upgrade conditions are all satisfied. If some smooth upgrade conditions are not satisfied, Escalator will show the failure of smooth upgrade.

- Pre-Conditions
  1. The system is running as normal.
  2. The VNFs are providing services as usual.
- Upgrading steps
  1. The VNFs are continually providing services during the upgrade.
  2. The operator successfully logged in the GUI of Escalator to select the software packages including Linux OS, Hypervisor, OpenStack, ODL and other OPNFV components, ect. (All or part of components could be selected.)
  3. Select the nodes to be upgraded. i.e. controller node, network node, storage node and compute node, etc.
  4. Select "Disable Scale-up". It will limit the scale-up operation when upgrade is in progress to prevent failures due to the shortage of resources.
  5. Select "Check Smooth Upgrade Conditions". If Escalator shows that there are some conditions not satisfied, try to resolve them according to the solutions provided.
  6. Select "Smooth Upgrade", then apply the upgrade operation.
  7. Select "Restore Scale-up" after the upgrade. It will restore scale-up to the original enabled/disabled state before upgrade.
- Post-Conditions

1. The system is upgraded successfully.
2. There is no service outage during the upgrade.
3. The VNFs are providing services as usual after the upgrade.

6.2 Use case #2: Roll-back after a failed smooth upgrade in a HA configuration

For a system with HA configuration, if the upgrade fails when the operator is smooth-upgrading NFVI/VIM components into a new version using Escalator, the operator can roll-back the system without any service outage.

- Pre-Conditions
  1. The system is running as normal.
  2. The VNFs are providing services as usual.
  3. Scale-up operation is disabled.
  4. Smooth upgrade failed.
- Roll-back steps
  1. Escalator concludes that the upgrade has failed and provides the operator with the reason.
  2. Select the “Roll-back” operation.
  3. If the roll-back is successful, go to step 4, otherwise the operator can select “Restore Backup” to restore the system from the backup data.
  4. Select “Restore Scale-up” after the roll-back. It will restore scale-up to the original enabled/disabled state before upgrade.
- Post-Conditions
  1. The system is rolled-back successfully when the upgrade failed.
  2. There is no service outage during the roll-back.
  3. The VNFs are providing services as usual after the roll-back.

6.3 Use case #3: Roll-back after a successful smooth upgrade in a HA configuration

When a smooth upgrade in a HA configuration is successful, the operator may want to roll-back for some reasons, such as performance issues. Escalator supports roll-back after a successful smooth upgrade without any service outage.

- Pre-Conditions
  1. The system is running as normal.
  2. The VNFs are providing services as usual.
  3. Smooth upgrade succeeded.
- Roll-back steps
  1. Select “Disable Scale-up”. It will limit the scale-up operation when roll- back is in progress to prevent failures due to the shortage of resources.

2. Select "Check Smooth Roll-back Conditions". If Escalator shows that there are some conditions not satisfied, try to resolve them according to the solutions provided.
3. Select "Roll-back", then apply the roll-back operation.
4. If the roll-back is successful, go to step 5, otherwise the operator can select "Restore Backup" to restore the system from the backup data.
5. Select "Restore Scale-up" after the roll-back. It will restore scale-up to the original enabled/disabled state before roll-back.

- Post-Conditions
  1. The system is rolled-back successfully.
  2. There is no service outage during the roll-back.
  3. The VNFs are providing services as usual after the roll-back.

6.4 Use case #4: Non-smooth upgrade in a non-HA/HA configuration

For a system with non-HA configuration, the operator can also use Escalator to upgrade NFVI/VIM components into a new version. In this case, the upgrade may result in service outage. In other words, the upgrade is non-smooth. For a system with HA configuration, if the service outage is acceptable or inevitable, the operator can also use Escalator to non-smoothly upgrade the system.

- Pre-Conditions
  1. The system is running as normal.
- Upgrading steps
  1. The operator successfully logged in the GUI of Escalator to select the software packages including Linux OS, Hypervisor, OpenStack, ODL and other OPNFV components, ect. (All or part of components could be selected.)
  2. Select the nodes to be upgraded. i.e. controller node, network node, storage node and compute node, etc.
  3. Select "Non-Smooth Upgrade", then apply the upgrade operation.
- Post-Conditions
  1. The system is upgraded successfully.

6.5 Use case #5: Roll-back after a failed non-smooth upgrade in a non-HA/HA configuration

For a system with non-HA/HA configuration, if the upgrade fails when the operator is non-smoothly upgrading NFVI/VIM components into a new version using Escalator, the operator can roll-back the system. In this case, the roll-back may result in service outage.

- Pre-Conditions
  1. The system is running as normal.
  2. Non-smooth upgrade failed.
- Roll-back steps
  1. Escalator concludes that the upgrade has failed and provides the operator with the reason.

2. Select the "Roll-back" operation.
3. If the roll-back fails, the operator can select "Restore Backup" to restore the system from the backup data.

- Post-Conditions
    1. The system is rolled-back successfully when the upgrade failed.

6.6 Use case #6: Roll-back after a successful non-smooth upgrade in a non-HA/HA configuration

When a non-smooth upgrade in a non-HA/HA configuration is successful, the operator may want to roll-back for some reasons, such as performance issues. Escalator supports roll-back after a successful non-smooth upgrade. In this case,the roll-back may result in service outage.

- Pre-Conditions
    1. The system is running as normal.
    2. Non-smooth upgrade succeeded.
- Roll-back steps
    1. Select the "Roll-back" operation.
    2. If the roll-back fails, the operator can select "Restore Backup" to restore the system from the backup data.
- Post-Conditions
    1. The system is rolled-back successfully when the upgrade failed.

CHAPTER

SEVEN

REFERENCE

[1] ETSI GS NFV 002 (V1.1.1): “Architectural Framework”

[2] ETSI GS NFV 003 (V1.1.1): “Terminology for Main Concepts in NFV”

[3] ETSI GS NFV-SWA001:“Virtual Network Function Architecture”

[4] ETSI GS NFV-MAN001:“Management and Orchestration”

[5] ETSI GS NFV-REL001:”Resiliency Requirements”

[6] QuEST Forum TL-9000:”Quality Management System Requirement Handbook”

[7] Service Availability Forum AIS:”Software Management Framework”

CHAPTER

EIGHT

REQUIREMENTS FROM OTHER PROJECTS

8.1 Doctor Project

The scope of Doctor project also covers maintenance scenario in which

1. The VIM administrator requests host maintenance to VIM.
2. VIM will notify it to consumer such as VNFM to trigger application level migration or switching active-standby nodes.
3. VIM waits response from the consumer for a short while.

- VIM should send out notification of VM migration to consumer (VNFM) as abstracted message like "maintenance".
- VIM could wait VM migration until it receives "VM ready to maintenance" message from the owner (VNFM)

8.2 HA Project

8.3 Multi-site Project

- Escalator upgrade one site should at least not lead to the other site API token validation failed.

CHAPTER

NINE

QUESTIONNAIRE OF ESCALATOR

A Questionnaire was created for collecting requirements from other projects.

Escalator Questionnaire: https://wiki.opnfv.org/_media/wiki/opnfv_escalator_questionnaire_20150723.pptx

Answer the questionnaire: https://docs.google.com/forms/d/11o1mt15zcq0WBtXYK0n6lKF8XuIzQTwvv8ePTjmcoF0/viewform?usp=

- search

Revision:

Build date: May 19, 2016