ETSI GR NFV-REL 014 V5.1.1 (2023-10)

Network Functions Virtualisation (NFV) Release 5; Reliability; Report on evaluating reliability for cloud-native VNFs

Disclaimer

The present document has been produced and approved by the Network Functions Virtualisation (NFV) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGR/NFV-REL014

Keywords
cloud-native, reliability

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

The present document can be downloaded from:
https://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

If you find a security vulnerability in the present document, please report it through our
Coordinated Vulnerability Disclosure Program:
https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document studies the reliability evaluation for cloud-native VNFs (as defined in ETSI GS NFV-EVE 011 [i.2]). It identifies key use cases (containerized VNF software modification, containerized VNF scaling out/in) of cloud-native VNF lifecycle management which may impact reliability. In these use cases, possible new functionalities to support the resiliency assurance of these VNF operations are discussed, using Kubernetes® as an example containerization technology. Then possible solutions are presented related to the new functionalities, together with the potential architectural options.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GR NFV 003: "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV".

[i.2] ETSI GS NFV-EVE 011 "Network Functions Virtualisation (NFV) Release 3; Virtualised Network Function; Specification of the Classification of Cloud Native VNF implementations".

[i.3] ETSI GS NFV-SEC 023: "Network Functions Virtualisation (NFV) Release 4; Security; Container Security Specification".

[i.4] ETSI GR NFV-IFA 029: "Network Functions Virtualisation (NFV) Release 3; Architecture; Report on the Enhancements of the NFV architecture towards "Cloud-native" and "PaaS"".

[i.5] ETSI GS NFV-IFA 010: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Functional requirements specification".

[i.6] ETSI GS NFV-IFA 040: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Requirements for service interfaces and object model for OS container management and orchestration specification".

[i.7] ETSI GS NFV-IFA 036: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Requirements for service interfaces and object model for container cluster management and orchestration specification".

[i.8] Kubernetes® document.

[i.9] Kubernetes® API conventions document.

[i.10] ETSI GS NFV 006: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Architectural Framework Specification".

[i.11] ETSI GS NFV-REL 003: " Network Functions Virtualisation (NFV); Reliability; Report on Models and Features for End-to-End Reliability".

[i.12] ETSI GS NFV-SOL 001: "Network Functions Virtualisation (NFV) Release 4; Protocols and Data Models; NFV descriptors based on TOSCA specification".

[i.13] ETSI GR NFV-IFA 041: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Report on enabling autonomous management in NFV-MANO".

[i.14] ETSI GS NFV-IFA 027: "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; Performance Measurements Specification".

[i.15] ETSI GS NFV-SOL 018: "Network Functions Virtualisation (NFV) Release 4; Protocols and Data Models; Profiling specification of protocol and data model solutions for OS Container management and orchestration".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GR NFV 003 [i.1] and the following apply:

NOTE: A term defined in the present document takes precedence over the definition of the same term, if any, in ETSI GR NFV 003 [i.1].

cloud-native VNF: VNF designed to be deployed and managed in a cloud computing environment for efficient operation

NOTE: The present document assumes that cloud-native VNFs have (but are not limited to) the following characteristics:

- dynamic (e.g. with frequent changes);
- scalable;
- fine-granular (e.g. composed of microservices, containerized).

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI GR NFV 003 [i.1] apply.

4 Overview and background

4.1 General

The telecom industry is experiencing a transformation towards cloud-native. Cloud-native VNFs may use technologies such as containerized functions, micro-service based architecture, self-management, scalability, etc. The management of VNFs following cloud-native principles is bringing profound changes to the operations and maintenance of telecom cloud-based networks, e.g. the combination of DevOps and Cloud increases the software delivery and efficiency. These new changes introduce new challenges on managing cloud-native VNFs, especially for the non-functional aspects like performance, reliability and security.

Reliability for cloud-native VNFs is really challenging because of their highly dynamic nature. Thus, highly dynamic management is needed due to the nature of cloud-native VNFs. In the scope of ISG NFV, ETSI GS NFV-EVE 011 [i.2] specifies non-functional aspects for cloud-native VNFs, e.g. resiliency, scaling and composition. ETSI GS NFV-SEC 023 [i.3] specifies the security and hardening requirements for VNFs running in a containerized environment. ETSI GR NFV-IFA 029 [i.4] investigates the use cases for application of cloud-native design principles, but it lacks the use cases analysis and has no recommendations on NFV-MANO functional enhancement derived from the use cases.

The present document focuses on the study of reliability aspects for supporting the management of cloud-native VNFs. Clause 5 introduces some cloud-native configuration capabilities and metrics related to reliability. Clause 6 studies a number of use cases for the purpose of deriving corresponding criteria and their associated configuration capabilities and metrics to evaluate the reliability for cloud-native VNFs. Clause 7 further elaborates on the identified criteria and their associated metrics of reliability evaluation. Finally, Clause 8 summarizes the recommendations for normative work in the future.

4.2 Containerized VNFs

In a cloud-native VNF environment, OS-container becomes the recommended technology for the infrastructure services in support of the VNFs, even though VM-based virtualisation is still an option for fulfilling cloud-native objectives. The introduction of OS-containers has an impact on the NFV-MANO architecture, as specified in ETSI GS NFV-IFA 010 [i.5] and ETSI GS NFV-IFA 040 [i.6], namely the introduction of new managed objects and a management function related to OS-container management and orchestration, i.e. MCIO and CISM. A Managed Container Infrastructure Object (MCIO) is a managed object representing the desired and actual state of a containerized workload for the OS-container management and orchestration. As specified in ETSI GS NFV-IFA 040 [i.6], an MCIO requesting compute/storage resources can be mapped to a VNFC.

Kubernetes®, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications. ETSI GS NFV-SOL 018 [i.15] profiles the Kubernetes® API as NFV protocol and data model solution for OS container management and orchestration. As defined in ETSI GR NFV-IFA 029 [i.4] and ETSI GS NFV-IFA 040 [i.6], MCIO could be realized as Pod, Deployment, StatefulSet, etc. in Kubernetes®.

Figure 4.2-1 shows the OS-container infrastructure service management architecture, as described in ETSI GS NFV 006 [i.10]. The Container Infrastructure Service (CIS) is responsible for providing the virtualised infrastructure as OS-containers. The Container Infrastructure Service Management (CISM) is responsible for the management of containerized workloads as MCIOs running in the CIS. The Container Cluster Management (CCM) is responsible for the management of CIS clusters.

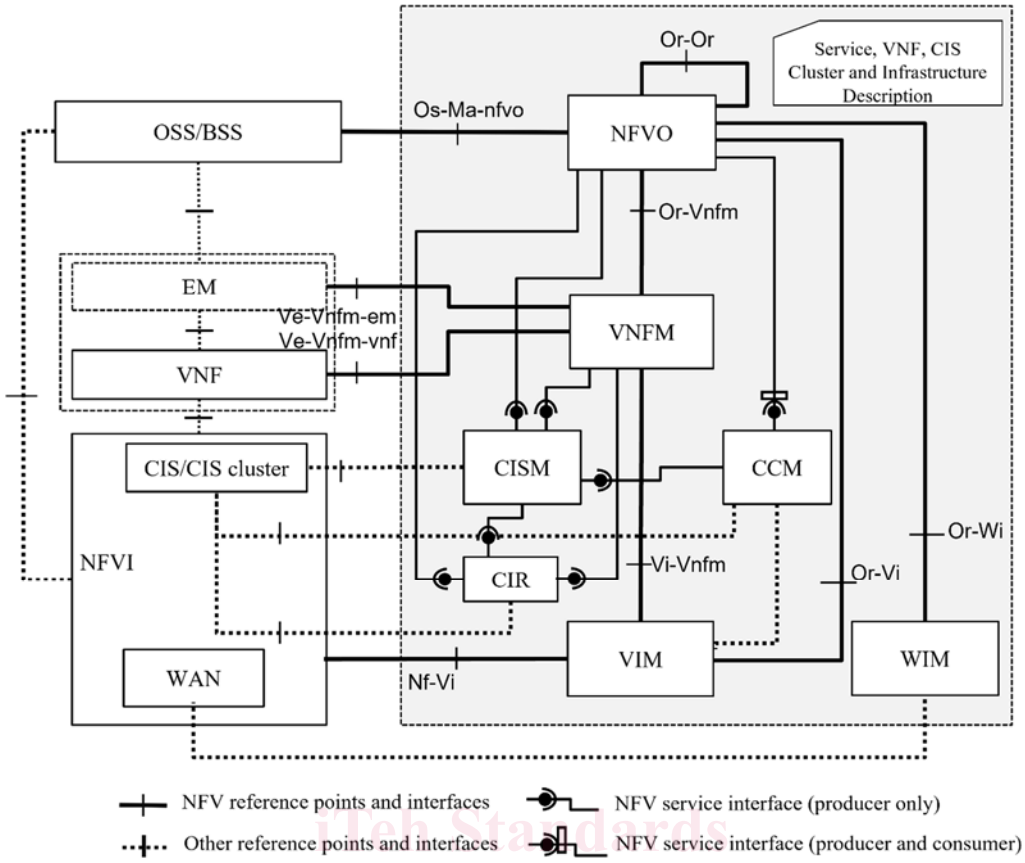

Figure 4.2-1: Container-based NFV architectural framework

5 Cloud-native configuration capabilities and metrics related to reliability

5.1 Kubernetes® objects

Kubernetes® objects are persistent managed objects representing the managed container cluster and its different resources such as pods. Kubernetes® objects can describe among others which containerized applications are running (and on which nodes), the resources available to those applications, and the policies applicable to those applications [i.8]. In the context of NFV, Kubernetes objects are MCIOs.

Most importantly, a Kubernetes® object includes two fields: the spec field and the status field [i.9]. The spec field is set by the user and characterizes the desired status of the entity represented by the Kubernetes® object. The status field shows the current status of the entity represented by the Kubernetes® object as supplied and updated by Kubernetes®. Kubernetes® continually and actively manages the actual status of each entity to match its desired status.

5.2 Management of clusters

5.2.1 Introduction

Current industry solutions for OS container management expect that a cluster of machines (running either in virtual machines or on bare-metal servers) is provided for their use. In the context of NFV, a cluster is called a Container Infrastructure Service (CIS) cluster and is composed of one or multiple CIS cluster nodes. A CIS cluster node, which can be realized as a VM or a bare-metal server, is a compute resource that runs a CIS instance or a CISM instance, or both.

The CCM (CIS Cluster Management) is responsible for the lifecycle management and FCAPS management of the CIS cluster. The CCM consumer can define the essential cluster information, including the description of the CIS cluster nodes, the placement constraints and the affinity or anti-affinity rules, etc., in the CIS Cluster Descriptor (CCD) that is interpreted by the CCM.

The concept of CIS cluster and the functionalities of CCM are detailed in ETSI GS NFV-IFA 036 [i.7].

5.2.2 Relevant configuration capabilities

ETSI GS NFV-IFA 036 [i.7] defines the requirements for CIS cluster management. CCM is responsible for lifecycle management of the CIS cluster, including applying changes to the CIS cluster configuration, scaling the CIS cluster, and modification of CIS cluster software.

CIS cluster configuration attributes include CIS cluster nodes to be used in the CIS cluster, number of CIS cluster nodes, scaling characteristics, placement constraints, cluster networking, and cluster storage. For more details, refer to clause 4.2.4 of ETSI GS NFV-IFA 036 [i.7].

CCM can construct CISM with high availability. For details of how CISM high availability can be achieved, refer to clause 4.2.12 of ETSI GS NFV-IFA 036 [i.7].

5.2.3 Relevant metrics

CCM reports information related to the CIS cluster configuration and CIS cluster status as defined in clause 4.2.4 of ETSI GS NFV-IFA 036 [i.7].

CCM provides performance measurements for CIS cluster nodes, CIS cluster storage, and CIS cluster nodes network, but not at the overall CIS cluster level.

5.3 Management of pods

5.3.1 Overview

Pods are the most basic deployable resources in Kubernetes® which are represented by Kubernetes® objects. Pods contain one or more containers, such as Docker™ containers. When a pod runs multiple containers, the containers share the pod's resources. Pods run on nodes organized in a certain container cluster.

Pods follow a defined lifecycle, starting in the **Pending** phase, moving through the **Running** phase if at least one of its containers is running, or in the process of starting or restarting. Pods complete their lifecycle through either the **Succeeded** or **Failed** phases depending on whether any container in the pod has terminated in a failure. A pod will spend most of its operational life in the Running phase.

In a pod, Kubernetes® tracks the container(s) status and determines what action(s) to take to keep the pod's actual status as desired. Kubernetes® is able to restart containers to handle some failures, e.g. Out-Of-Memory (OOM) failure when a container exceeds its resource limit. Container failures, which cannot be resolved by restart, are escalated to the pod level and cause the pod to be terminated in the Failed phase.

For stateless containerized applications, Kubernetes® provides the **Deployment** object to realize declarative configuration for a set of pods. Users can create a Deployment to roll out a group of identical pods, scale the Deployment to increase or decrease the number of its pods, and update the Deployment which means changing this Deployment's pod template by updating the pods' metadata and spec configurations (see clause 5.3.2), e.g. pods' labels or container images.

For stateful containerized applications, the Kubernetes® object **StatefulSet** is used to manage the roll-out and scaling of a set of pods. A StatefulSet provides guarantees about the ordering and uniqueness of these pods. Like a Deployment, a StatefulSet contains pods that are based on the same container spec. Unlike a Deployment, a StatefulSet maintains a sticky identity for each of its pods. Even though these pods are created from the same spec, they are not interchangeable: each has a persistent identifier that it maintains even across rescheduling [i.8].

Pods in a StatefulSet have a unique identity that is comprised of an ordinal and a stable network identity, and a stable storage. When pods are being deployed, they are created sequentially, which defines a succession. When pods are deleted, they are terminated in reverse order. Pods of a StatefulSet can be updated in a rolling fashion, which means an automated rolling update of their containers, labels, resource requests/limits, and annotations.

The Kubernetes® objects expose some configurable attributes to request resources for running pods and their desired limits. These attributes allow capacity planning, assessment of current or historical scheduling limits, quick identification of workloads that cannot be scheduled due to a lack of resources, and comparison of actual usage to the pod's request.

For more details on pods management, refer to [i.8] and [i.9].

5.3.2 Existing configuration capabilities

The Kubernetes® object representing a pod includes the following configuration capabilities:

- **Metadata:** which contains the identification and description of the pod. Commonly used metadata attributes for a pod are:
  - **Name**.
  - **Namespace:** which configures the namespace to which the pod belongs.
  - **Labels:** which lists the pod's custom labels.
- **Spec:** which is a detailed configuration of the pod's various resources and handling options. The key attributes for the pod's resource configuration are the following:
  - **Containers:** which is used to configure the containers of the pod.
  - **NodeName:** name of the node on which Kubernetes® can schedule this pod.
  - **NodeSelector:** which defines the node labels. Kubernetes® has to schedule this pod to a node with these labels.
  - **Volumes:** which provides the storage volume information of the pod.
  - **RestartPolicy:** which configures the strategy for this pod in case of pod failure.

Kubernetes® provides the capability to configure a single pod; however, it is more common to deploy a set of identical pods as a Deployment or a StatefulSet using pod templates.

For managing a set of stateless pods, the Kubernetes® object Deployment includes the following configuration capabilities:

- **Metadata:** which contains the identification and description of the Deployment. Commonly used metadata attributes for a Deployment are its name and labels.
- **Spec:** which is a detailed configuration of the Deployment's various resources. The key attributes for the Deployment's resource configuration are the following:
  - **Replicas:** which is used to configure how many pod replicas are needed.
  - **Selector:** which defines how the Deployment controller finds which pods it manages. For example, the user can select a label that is defined in the pod template, then the Deployment controller will manage pods with this label.
  - **Template:** which provides template for pods in the Deployment.
  - **Strategy:** which specifies the strategy used to replace pods of the old template by new ones when the Deployment is upgraded. It has two options "**Recreate**" or "**RollingUpdate**".