Cloud Infrastructure Reference Model
Version 1.0
11 November 2020

This is a Non-binding Permanent Reference Document of the GSMA

Security Classification: Non-confidential

Copyright Notice

Disclaimer

Antitrust Notice

Table of Contents

1 Introduction 5
1.1 Overview 5
1.2 Scope 6
1.3 Principles 7
1.4 Definitions/Terminology 7
1.5 Abbreviations 7
1.6 References 10
1.7 Conventions 12
2 Workload Requirements & Analysis 12
2.1 Workloads Collateral 13
2.2 Use cases 15
2.3 Analysis 19
2.4 Cloud Infrastructure Profiles 20
3 Modelling 21
3.1 Model 22
3.2 Virtual Resources 23
3.2.1 Tenant 24
3.2.2 Compute 24
3.2.3 Storage 24
3.2.4 Availability Zone 25
3.3 Cloud Infrastructure Management 25
3.3.1 Virtual Infrastructure Manager 25
3.3.2 Hardware Infrastructure Manager 26
3.4 Hardware Infrastructure Resources 27
3.5 Network 28
3.6 Storage 29
3.7 Sample reference model realization 30
4 Infrastructure Capabilities, Measurements and Catalogue 30
4.1 Capabilities and Performance Measurements 30
4.1.1 Exposed vs Internal 31
4.1.2 Exposed Infrastructure Capabilities 32
4.1.3 Exposed Infrastructure Performance Measurements 34
4.1.4 Internal Infrastructure Capabilities 34
4.1.5 Cloud Infrastructure Management Capabilities 36
4.1.6 Cloud Infrastructure Management Performance Measurements 37
4.2 Infrastructure Profiles Catalogue 38
4.2.1 Compute Flavours 38
4.2.2 Virtual Network Interface Specifications 39
4.2.3 Storage Extensions 40
4.2.4 Cloud Infrastructure Profiles 40
4.2.5 Cloud Infrastructure Profile Capabilities Mapping 41
4.2.6 Void 43
4.2.7 One stop shop 43

4.3 Networking 46
4.3.1 Network Layering and Concepts 46
4.3.2 Networking Reference Model 50
4.3.3 Deployment examples based on the Networking Reference Model 51
5 Feature set and Requirements from Infrastructure 53
5.1 Cloud Infrastructure Software profile description 53
5.1.1 Virtual Compute 55
5.1.2 Virtual Storage 55
5.1.3 Virtual Networking 56
5.1.4 Security 56
5.1.5 Platform Services 57
5.2 Cloud Infrastructure Software Profiles features and requirements 57
5.2.1 Virtual Compute 57
5.2.2 Virtual Storage 58
5.2.3 Virtual Networking 58
5.3 Cloud Infrastructure Hardware Profile description 59
5.4 Cloud Infrastructure Hardware Profiles features and requirements. 62
5.4.1 Compute Resources 62
5.4.2 Storage Configurations 63
5.4.3 Network Resources 63
6 External Interfaces 64
6.1 Introduction 64
6.2 Cloud Infrastructure APIs 65
6.2.1 Tenant Level APIs 66
6.2.2 Hardware Acceleration Interfaces 68
6.3 Intra-Cloud Infrastructure Interfaces 73
6.3.1 Hypervisor Hardware Interface 73
6.4 Enabler Services Interfaces 73
7 Security 73
7.1 Introduction 73
7.2 Potential attack vectors 74
7.3 Security Scope 74
7.3.1 In-scope and Out-of-Scope definition 74
7.3.2 High level security Requirements 74
7.4 Cloud Infrastructure Security 75
7.4.1 General Platform Security 75
7.4.2 Platform ‘back-end’ access security 78
7.4.3 Platform ‘front-end’ access security 78
7.5 Workload Security - Vendor Responsibility 78
7.5.1 Software Hardening 78
7.5.2 Port Protection 78
7.5.3 Software Code Quality and Security 79
7.5.4 Alerting and monitoring 79
7.5.5 Logging 79

7.5.6 VNF images 79
7.5.7 Vulnerability Management 79
7.6 Workload Security - Cloud Infrastructure Operator Responsibility 79
7.6.1 Remote Attestation/openCIT 80
7.6.2 Workload Image Scanning / Signing 80
7.6.3 Networking Security Zoning 80
7.6.4 Volume Encryption 81
7.6.5 Root of Trust for Measurements (RTM) 81
7.7 Common security standards 82
7.8 Testing & certification 83
7.8.1 Testing demarcation points 83
7.8.2 Certification requirements 84
7.9 Consolidated Security Requirements 84
7.9.1 System Hardening 84
7.9.2 Platform and Access 86
7.9.3 Confidentiality and Integrity 87
7.9.4 Workload Security 88
7.9.5 Image Security 88
7.9.6 Security LCM 88
7.9.7 Monitoring and Security Audit 89
7.9.8 Compliance with Standards 91
7.10 Security References 92
8 Chapter reserved for future use 93
9 Infrastructure Operations and Lifecycle Management 93
9.1 Introduction 93
9.2 Configuration and Lifecycle Management 94
9.3 Assurance 96
9.4 Capacity Management 96
10 Challenges and Gaps 97
10.1 Introduction 97
10.2 Challenges 97
10.3 Gaps 97
10.3.1 Discovery 97
10.3.2 Support Load Balance of VNF/CNFs 98
10.3.3 Service Function Chain 98
10.3.4 Packet Acceleration Request (e.g. Hardware Acceleration) 98
10.3.5 Multi-cloud architecture directions for network workloads 99
Annex A Cloud iNfrastructure Telco Taskforce 100
A.1 Overview 100
A.1.1 Terminology and Glossary 100
A.1.2 Problem Statement 100
A.1.3 Project Goals and Purpose 101
A.1.4 Common Cloud Infrastructure Benefits 102
A.2 Principles 103

A.2.1 Overall Principles 103
A.2.2 Requirements Principles 104
A.2.3 Architectural Principles 105
A.3 Scope 105
A.3.1 Functional Scope 106
A.3.2 Out of Scope Components 106
A.3.3 Specification Types 107
A.3.4 Relationship to other industry projects 108
Annex B Reference Model Glossary 111
B.1 Terminology 111
B.2 Software Layer Terminology 111
B.3 Hardware Layer Terminology 113
B.4 Operational and Administrative Terminology 114
B.5 Container Related Terminology 115
B.6 OpenStack Related Terminology 116
B.7 Cloud Platform Abstraction Related Terminology: 117
B.8 Other Referenced Terminology 117
Annex C Document Management 119
C.1 Document History 119

1 Introduction

1.1 Overview

The Reference Model (RM) described in this document specifies a virtualisation technology agnostic (VM-based and container-based) cloud infrastructure abstraction and acts as a "catalogue" of the exposed infrastructure capabilities, resources, and interfaces required by the workloads. This document has been developed by Cloud iNfrastructure Telco Taskforce (CNTT). For more information about CNTT, its activities and deliverables, see Annex A.

The document starts from the abstract and as it progresses it increasingly gets into more details. It follows the traditional design process where you start from core principles, progress to abstract concepts and models, then finish with operational considerations, such as security and lifecycle management.

- **Chapter 01 - Introduction**: Overall scope of the Reference Model document including the goals and objectives of the project.

  Audience: This chapter is written for a general technical audience with interest in this topic.

- **Chapter 02 - Workload requirements & Analysis**: High level requirements and core principles needed to understand how the model was developed. Addresses the thinking behind the decisions that were made.

  Audience: This chapter is written for architects and others with an interest in how the decisions were made.

- **Chapter 03 - Modelling**: The high-level cloud infrastructure model itself.

  Audience: This chapter is written for architects and others who wants to gain a quick high-level understanding of the model.

- **Chapter 04 - Infrastructure Capabilities, Metrics, and Catalogue**: Details about the capabilities needed to support the various types of workloads and how the capabilities are applied to the model. The details regarding T-shirt sizes and other considerations are found in this section.

  Audience: This chapter is written for architects, developers and others who need to deploy infrastructure or develop applications.

- **Chapter 05 - Feature set and Requirements from Infrastructure**: This chapter goes into more details on what needs to be part of the cloud infrastructure. It describes the software and hardware capabilities and configurations recommended for the different types of cloud infrastructure profiles.

  Audience: This chapter is written for architects, developers and others who need to deploy infrastructure or develop applications.

- **Chapter 06 - External Interfaces**: This chapter covers APIs and any actual interfaces needed to communicate with the workloads and any other external components.

Audience: This chapter is written for architects, developers and others who need to develop APIs or develop applications that use the APIs.

- **Chapter 07 - Security**: This chapter identifies the security requirements that need to be taken into consideration when designing and implementing a cloud infrastructure environment. It does not cover details related to company specific requirements to meet regulatory requirements.

  Audience: This chapter is written for security professional, architects, developers and others who need to understand the role of security in the cloud infrastructure environment.

- **Chapter 08**: This chapter is reserved for future use
- **Chapter 09 - Life Cycle Management**: This chapter focuses on the operational aspects of the cloud infrastructure. Discussions include deployment considerations, on-going management, upgrades and other lifecycle concerns and requirements. It does not cover details related to company specific operational requirements, nor does it go into how the cloud infrastructure will interface with existing BSS/OSS systems.

  Audience: This chapter is written for lifecycle managers, operational support teams and others who need to support the infrastructure or the applications.

- **Chapter 10 - Challenges and Gaps**: Opportunities for future developments as technology changes over time.

  Audience: This chapter is written for a general technical audience with interest in this topic.

1.2 Scope

This **Reference Model** document focuses on identifying the abstractions, and associated concepts, that are needed to represent the cloud infrastructure. Figure 1 below highlights its scope in more details.

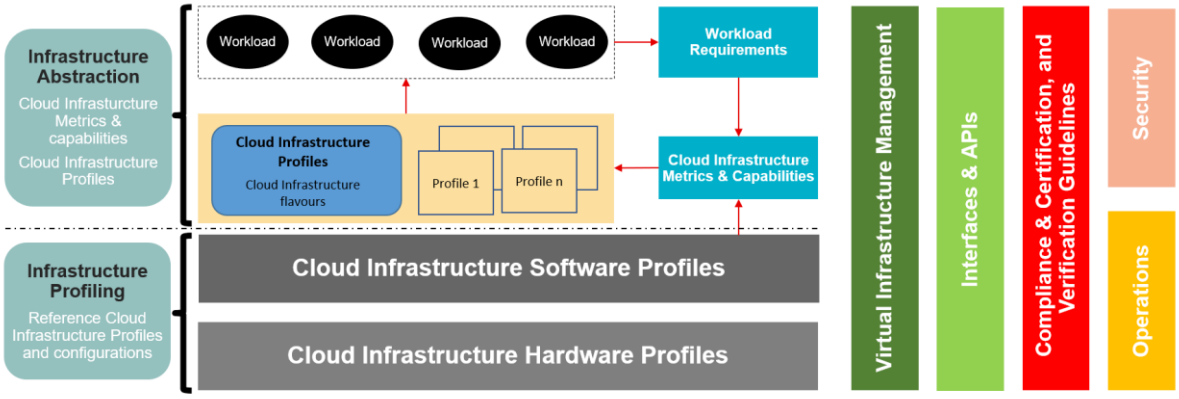

Figure 1: Scope of Reference Model

This document specifies:

- **Cloud Infrastructure abstraction**: in context with how it interacts with the other components required to build a complete cloud system that supports workloads deployed in Virtual Machines (VM) or containers. Network function workloads that are deployed on virtual machines and containers are referred to as virtual network functions (VNF) and containerised network functions (CNF), respectively; please note that it is now more common to refer CNFs as cloud native network functions.

    - **Cloud Infrastructure capabilities & metrics**: A set of cloud infrastructure capabilities and metrics required to perform telco scale network functions and satisfy their performance criterion.
    - **Infrastructure profiles catalogue**: A catalogue of standard infrastructure software and hardware configurations, referred to as profiles; these profiles abstract the infrastructure for the workloads. Only a few profiles, with well-defined characteristics, can meet the operational and performance requirements of all workloads.

- Cloud Infrastructure Software and Hardware profiles:

    - **Cloud Infrastructure software profiles**: These software profiles are components of the corresponding infrastructure profiles within the infrastructure profiles catalogue, and specify the host infrastructure software configurations.
    - **Cloud Infrastructure hardware profiles**: These hardware profiles are components of the corresponding infrastructure profiles within the infrastructure profiles catalogue, and specify the host infrastructure hardware configurations.

- Conformance and verification

    - **Conformance programs**: These define the requirements for verification and validation programs for both the cloud infrastructure and workloads.
    - **Test framework**: Provides test suites to allow conformance of cloud infrastructure and workloads.

1.3 Principles

The Reference Model specifications conform to the overall principles defined in Annex A, section A.2.

1.4 Definitions/Terminology

To help guide the reader, the Reference Model Glossary (see Annex B) provides an introduction to the main terms used within this document and throughout the project in general. These definitions are, with a few exceptions, based on the ETSI GR NFV 003 [1] definitions. In a few cases, they have been modified to avoid deployment technology dependencies only when it seems necessary to avoid confusion.

1.5 Abbreviations

Term	Description
AI	Artificial Intelligence
API	Application Programming Interface

Term	Description
B2B	Business to Business
B2C	Business to Consumer
BIOS	Basic Input Output System
CaaS	Container as a Service
CAPEX	Capital Expenditure
CI/CD	Continuous Integration / Continuous Deployment
CNF	Cloud Native Network Function
CNI	Container Network Interface
CNTT	Cloud iNfrastructure Telco Taskforce
CPU	Central Processing Unit
CRTM	Core Root of Trust for Measurements
CVE	Common Vulnerabilities and Exposures
DC	Data Center
DMA	Direct Memory Access
DNS	Domain Name System
DPDK	Data Plane Development Kit
DRAM	Dynamic Random Access Memory
DRTM	Dynamic Root of Trust for Measurements
E2E	End to End
EPA	Enhanced Platform Awareness
eTOM	enhanced Telecom Operations Map
ETSI	European Telecommunications Standards Institute
EULA	End-User License Agreement
EVPN	Ethernet Virtual Private Network
FCAPS	fault, configuration, accounting, performance, security
FPGA	Field Programmable Gate Array
FTTx	Fiber to the x
GB	Giga Byte
GW	Gateway
GPU	Graphics Processing Unit
GSMA	Groupe Speciale Mobile Association
HDD	Hard Disk Drive
HW	Hardware
IaaS	Infrastructure as a Service
IMS	IP Multimedia Subsystem
IO	Input/Output
IOPS	Input/Output per Second
IOMMU	Input/Output Memory Management Unit

Term	Description
IoT	Internet of Things
IP	Internet Protocol
IPMI	Intelligent Platform Management Interface
IPSec	Internet Protocol Security
IT	Information Technology
K8s	Kubernetes
LAN	Local Area Network
LCM	Lifecycle Management
MANO	Management and Orchestration
ML	Machine Learning
MPLS	Multi-Protocol Label Switching
MVNO	Mobile Virtual Network Operator
NAT	Network Address Translation
NBI	North Bound Interface
NFV	Network Function Virtualisation
NFVI	Network Function Virtualisation Infrastructure
NFVO	Network Function Virtualisation Orchestrator
NIC	Network Interface Card
NPU	Numeric Processing Unit
NUMA	Non-Uniform Memory Access
NVMe	Non-Volatile Memory Express
ONAP	Open Network Automation Platform
OPEX	Operational Expenditure
OPNFV	Open Platform for NFV
ORAN	Open Radio Access Network
OS	Operating System
OVP	OPNFV Verified Program
PCIe	Peripheral Component Interconnect Express
PCI-PT	PCIe PassThrough
PCR	Platform Configuration Register
PF	Physical Function
POD	Point of Delivery
PRD	Permanent Reference Document
QoS	Quality of Service
RA	Reference Architecture
RAM	Random Access Memory
RAN	Radio Access Network
RBAC	Role-bases Access Control

Term	Description
RFC	Request for Change
RC	Reference Conformance
RI	Reference Implementation
RM	Reference Model
RTM	Root of Trust for Measurements
SATA	Serial AT Attachment
SBA	Service Based Architecture
SBI	South Bound Interface
SDN	Software-Defined Networking
SDS	Software-Defined Storage
SFC	Service Function Chaining
SLA	Service Level Agreement
SMT	Simultaneous multithreading
SR-IOV	Single Root Input Output Virtualisation
SRTM	Static Root of Trust for Measurements
SSD	Solid State Drive
SW	Software
TLS	Transport Layer Security
TOSCA	Topology and Orchestration Specification for Cloud Applications
TPM	Trusted Platform Module
UEFI	Unified Extensible Firmware Interface
UI	User Interface
vCPU	Virtual CPU
VF	Virtual Function
VIM	Virtualised Infrastructure Manager
VLAN	Virtual LAN
VM	Virtual Machine
VNF	Virtual Network Function
VNFC	Virtual Network Function Component
vNIC	Virtual Network Interface Card
vRAN	Virtual Radio Access Network
VxLAN	Virtual Extensible LAN
vXYZ	virtual XYZ, e.g., as in vNIC
Wi-Fi	Wireless Fidelity

1.6 References

Ref	Doc Number	Title
[1]	ETSI GR NFV	"Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV", January 2020. Available at

Ref	Doc Number	Title
	003 V1.5.1	https://www.etsi.org/deliver/etsi_gr/NFV/001_099/003/01.05.01_60/gr_NFV003v010501p.pdf
[2]	RFC 2119	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt
[3]	ETSI GS NFV 002 V1.2.1	"Network Functions Virtualisation (NFV); Architectural Framework". Available at https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf
[4]	ETSI GR NFV-IFA 029 V3.3.1	"Network Functions Virtualisation (NFV) Release 3; Architecture; Report on the Enhancements of the NFV architecture towards "Cloud-native" and "PaaS" ". Available at https://www.etsi.org/deliver/etsi_gr/NFV-IFA/001_099/029/03.03.01_60/gr_NFV-IFA029v030301p.pdf
[5]	ETSI GS NFV-TST 008 V3.2.1	"Network Functions Virtualisation (NFV) Release 3; Testing; NFVI Compute and Network Metrics Specification". Available at https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/008/03.02.01_60/gs_NFV-TST008v030201p.pdf
[6]	ETSI GS NFV-IFA 027 V2.4.1	"Network Functions Virtualisation (NFV) Release 2; Management and Orchestration; Performance Measurements Specification". Available at https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/027/02.04.01_60/gs_nfv-ifa027v020401p.pdf
[7]	ETSI GS NFV-IFA 002 V2.1.1	"Network Functions Virtualisation (NFV);Acceleration Technologies; VNF Interfaces Specification". Available at https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/002/02.01.01_60/gs_NFV-IFA002v020101p.pdf
[8]	ETSI NFV-IFA 019 V3.1.1	"Network Functions Virtualisation (NFV); Acceleration Technologies; Acceleration Resource Management Interface Specification; Release 3". Available at https://www.etsi.org/deliver/etsi_gs/nfv-ifa/001_099/019/03.01.01_60/gs_nfv-ifa019v030101p.pdf
[9]	ETSI GS NFV-INF 004 V1.1.1	"Network Functions Virtualisation (NFV); Infrastructure; Hypervisor Domain". Available at https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/004/01.01.01_60/gs_NFV-INF004v010101p.pdf
[10]	ETSI GS NFV-IFA 005 V3.1.1	"Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Or-Vi reference point - Interface and Information Model Specification". Available at https://www.etsi.org/deliver/etsi_gs/nfv-ifa/001_099/005/03.01.01_60/gs_nfv-ifa005v030101p.pdf
[11]	DMTF RedFish	"DMTF RedFish Specification". Available at https://www.dmtf.org/standards/redfish
[12]	NGMN Overview on 5GRAN Functional Decomposition ver 1.0	"NGMN Overview on 5GRAN Functional Decomposition". Available at https://www.ngmn.org/wp-content/uploads/Publications/2018/180226_NGMN_RANFSX_D1_V20_Final.pdf
[13]	ORAN-WG4.IOT.0-v01.00	"Front haul Interoperability Test Specification(IOT)". Available at https://static1.squarespace.com/static/5ad774cce74940d7115044b0/t/5db36ffa820b8d29022b6d08/1572040705841/ORAN-WG4.IOT.0-v01.00.pdf/2018/180226_NGMN_RANFSX_D1_V20_Final.pdf

Ref	Doc Number	Title
[14]	ETSI GS NFV-TST 009 V3.1.1	"Network Functions Virtualisation (NFV) Release 3; Testing; Specification of Networking Benchmarks and Measurement Methods for NFVI". Available at https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/009/03.01.01_60/gs_NFV-TST009v030101p.pdf
[15]	ETSI GR NFV IFA-012	"Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Report on Os-Ma-Nfvo reference point - application and service management use cases and recommendations". Available at https://www.etsi.org/deliver/etsi_gr/NFV-IFA/001_099/012/03.01.01_60/gr_NFV-IFA012v030101p.pdf
[16]	ETSI GS NFV-SEC 001 V1.1.1	"Network Functions Virtualisation (NFV); NFV Security; Problem Statement". Available at https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/001/01.01.01_60/gs_nfv-sec001v010101p.pdf
[17]	ETSI GS NFV-SEC 003 V1.1.1	"Network Functions Virtualisation (NFV); NFV Security; Security and Trust Guidance". Available at https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/003/01.01.01_60/gs_NFV-SEC003v010101p.pdf
[18]	ETSI GS NFV-SEC 013 V3.1.1	"Network Functions Virtualisation (NFV) Release 3; NFV Security; Security Specification for MANO Components and Reference points". Available at https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/014/03.01.01_60/gs_NFV-SEC014v030101p.pdf
[19]	ETSI GS NFV-SEC 013 V2.6.1	"Network Functions Virtualisation (NFV) Release 2; Security; VNF Package Security Specification ". Available at https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/021/02.06.01_60/gs_nfv-sec021v020601p.pdf

1.7 Conventions

"The key words "must", "must not", "required", "shall", "shall not", "should", "should not", "recommended", "may", and "optional" in this document are to be interpreted as described in RFC2119 [2]."

2 Workload Requirements & Analysis

The Cloud Infrastructure is the totality of all hardware and software components which build up the environment in which VNFs/CNFs (workloads) are deployed, managed and executed. It is, therefore, inevitable that different workloads would require different capabilities and have different expectations from it.

One of the main targets of the CNTT is to define an agnostic cloud infrastructure, to remove any dependencies between workloads and the deployed cloud infrastructure, and offer infrastructure resources to workloads in an abstracted way with defined capabilities and metrics.

This means, operators will be able to host their Telco workloads (VNFs/CNFs) with different traffic types, behaviour and from any vendor on a unified consistent cloud infrastructure.

Additionally, a well-defined cloud infrastructure is also needed for other type of workloads such as IT, Machine Learning, and Artificial Intelligence.

This chapter analyses various telco workloads and their requirements, and recommends certain cloud infrastructure parameters needed to specify the desired performance expected by these workloads.

2.1 Workloads Collateral

There are different ways that workloads can be classified, for example:

- **By function type:**
    - Data Plane (a.k.a., User Plane, Media Plane, Forwarding Plane)
    - Control Plane (a.k.a, Signalling Plane)
    - Management Plane

Note: Data plane workloads also include control and management plane functions; control plane workloads also include management plane functions.

- **By service offered:**
    - Mobile broadband service
    - Fixed broadband Service
    - Voice Service
    - Value-Added-Services
- **By technology:** 2G, 3G, 4G, 5G, IMS, FTTx, Wi-Fi...

The list of, most likely to be virtualised, Network Functions below, covering almost ***95%*** of the Telco workloads, is organised by network segment and function type.

- **Radio Access Network (RAN)**
    - Data Plane
        - BBU: BaseBand Unit
        - CU: Centralised Unit
        - DU: Distributed Unit
- **2G/3G/4G mobile core network**
    - Control Plane
        - MME: Mobility Management Entity
        - 3GPP AAA: Authentication, Authorisation, and Accounting
        - PCRF: Policy and Charging Rules Function
        - OCS: Online Charging system
        - OFCS: Offline Charging System
        - HSS: Home Subscriber Server
        - DRA: Diameter Routing Agent
        - HLR: Home Location Register
        - SGW-C: Serving GateWay Control plane
        - PGW-C: Packet data network GateWay Control plane

- Data Plane
    - SGW: Serving GateWay
    - SGW-U: Serving GateWay User plane
    - PGW: Packet data network GateWay
    - PGW-U: Packet data network GateWay User plane
    - ePDG: Evolved Packet Data GateWay
    - MSC: Mobile Switching Center
    - SGSN: Serving GPRS Support Node
    - GGSN: Gateway GPRS Support Node
    - SMSC : SMS Center

- **5G core network** 5G core nodes are virtualizable by design and strong candidate to be on boarded onto Telco Cloud as "cloud native application"
    - Data Plane
        - UPF: User Plane Function
    - Control Plane
        - AMF: Access and Mobility management Function
        - SMF: Session Management Function
        - PCF: Policy Control Function
        - AUSF: Authentication Server Function
        - NSSF: Network Slice Selection Function
        - UDM: Unified Data Management
        - UDR: Unified Data Repository
        - NRF: Network Repository Function
        - NEF: Network Exposure Function

  Note: for Service-based Architecture (SBA) all Network Functions are stateless (store all sessions/ state on unified data repository UDR)

- **IP Multimedia Subsystem (IMS)**
    - Data Plane
        - MGW: Media GateWay
        - SBC: Session Border Controller
        - MRF: Media Resource Function
    - Control Plane
        - CSCF: Call Session Control Function
        - MTAS: Mobile Telephony Application Server
        - BGCF: Border Gateway Control Function
        - MGCF: Media Gateway Control Function

- **Fixed network**
    - Data Plane

        - MSAN: MultiService Access Node
        - OLT: Optical Line Termination
        - WLC: WLAN Controller
        - BNG: Border Network Gateway
        - BRAS: Broadband Remote Access Server
        - RGW: Residential GateWay
        - CPE: Customer Premises Equipment

    - Control Plane

        - AAA: Authentication, Authorisation, and Accounting

- **Other network functions**

    - Data Plane

        - LSR: Label Switching Router
        - DPI: Deep Packet Inspection
        - CG-NAT: Carrier-Grade Network Address Translation
        - ADC: Application Delivery Controller
        - FW: FireWall
        - Sec-GW: Security GateWay
        - CDN: Content Delivery Network

    - Control plane

        - RR: Route Reflector
        - DNS: Domain Name System

    - Management Plane

        - NMS: Network Management System

2.2 Use cases

The intend of this section is to describe some important use cases that are pertinent to this Reference Model. We start with some typical Edge related use cases. The list of use cases will be extended in the future releases.

Telco Edge is commonly coupled with 5G use cases, seen as one of the ingredients of the Ultra-Reliable Low-latency Communication (URLLC) and Enhanced Mobile Broadband (eMBB) Network Slicing. The requirements for user plane Local Breakout / Termination are common mandating that Value Added Services (VASs) & Any Gi-LAN applications are locally hosted at the Edge. The Telco Edge is a perfect fit for centralized vRAN deployment and vDU/vCU hosting that satisfy the latency requirements.

- **Use Case #1 - Edge CDN with eMBB Core Network Slicing**

    - **Business Objectives**

      Monetizing 5G by provisioning eMBB network slice with distributed Content Delivery Network (CDN) as a service, that enables Ultra-HD (UHD) streaming,

Video Optimization, caching for large files, and other capabilities that can either bundled by the Network Slice offering or implicitly enabled by the operator.

- **Targeted Segments**
    - B2C (Targeting high Tier Packages & Bundles)
    - Content Owners (Potential revenue sharing model)
    - Mobile Virtual Network Operators (MVNOs - Wholesale)
    - Stadiums and Venues.
- **Architecture**

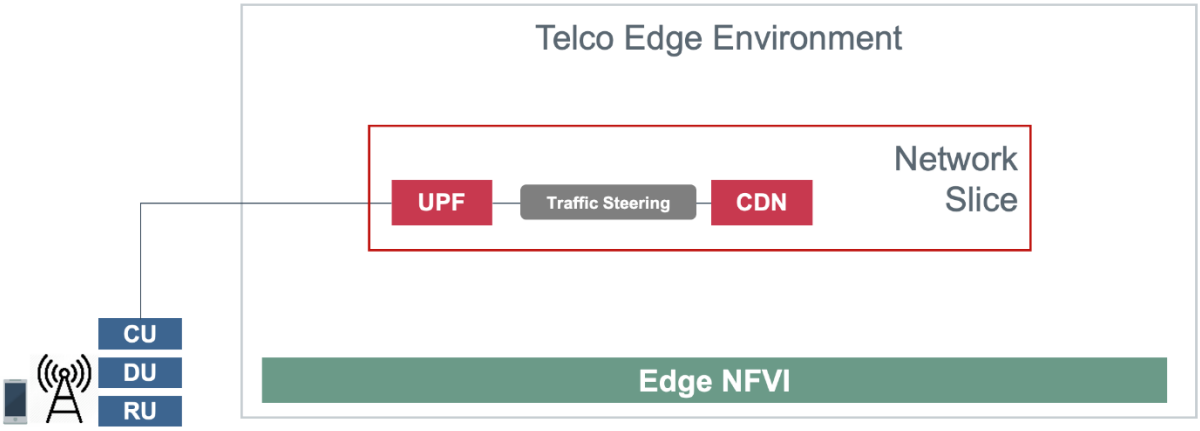

Figure 2: Edge CDN with eMBB Core Network Slicing.

- **Use Case #2 - Edge Private 5G with Core Network Slicing**
    - **Business Objectives**

      Private 5G is considered one of the most anticipated Business use cases in the coming few years enabling Mobile Operators to provide a standalone private Mobile Network to enterprises that may include all the ingredients of PLMN such as Radio, Core, Infrastructure & Services covering the business requirements in terms of security, performance, reliability, & availability.

    - **Targeted Segments**
        - Governmental Sectors & Public Safety (Mission critical applications)
        - Factories and Industry sector.
        - Enterprises with Business-critical applications.
        - Enterprises with strict security requirements with respect to assets reachability.
        - Enterprises with strict KPIs requirements that mandate the on-premise deployment.

- **Architecture**

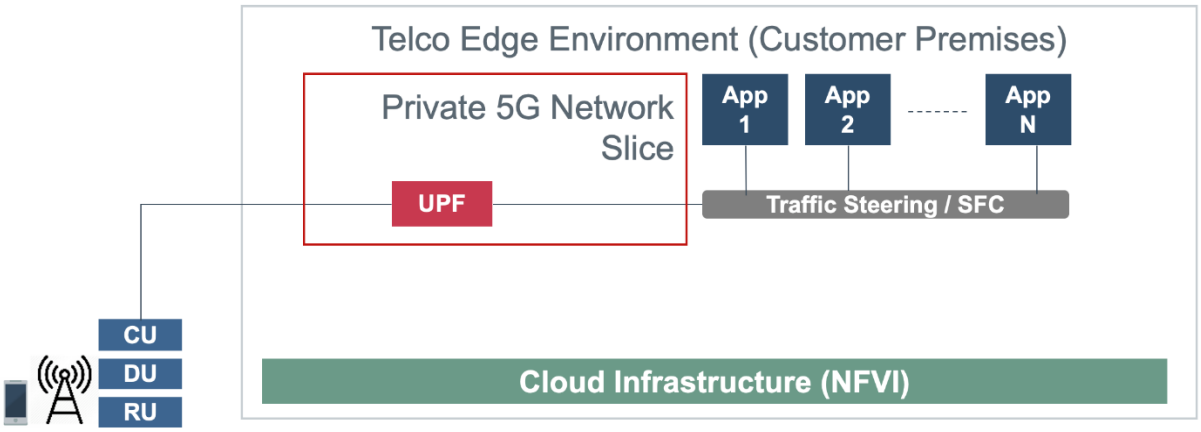

Figure 3: Edge Private 5G with Core Network Slicing.

- There are multiple flavours for Private 5G deployments or NPN, Non-Public Network as defined by 3GPP.
- The use case addresses the technical realization of NPN as a Network Slice of a PLMN as per Annex D – 3GPP TS 23.501 R16 and not covering the other scenarios of deployment.
- The use case assumes a network slice that is constructed from a single UPF deployed on Customer premises while sharing the 5G Control Plane (AMF, SMF, & other CP Network Functions) with the PLMN.
- The use case doesn't cover the requirements of the private Application Servers (ASs) as they may vary with each customer setup.
- Hosting the CU/DU on-Customer Infrastructure depends on the enterprise offering by the Mobile Operator and the selected Private 5G setup.
- The Edge Cloud Infrastructure can be governed by the client or handled by the Service Provider (Mobile Operator) as part of Managed-services model.

- **Use Case #3 - Edge Automotive (V2X) with uRLLC Core Network Slicing**

  - **Business Objectives**

    The V2X (Vehicle-to-everything) set of use cases provides a 5G monetization framework for Mobile Operators developing 5G URLLC business use cases targeting the Automotive Industry, Smart City Regulators, & Public Safety.

  - **Targeted Segments**

    - Automotive Industry.
    - Governmental Departments (Smart Cities, Transport, Police, Emergency Services, etc.).
    - Private residencies (Compounds, Hotels and Resorts).
    - Enterprise and Industrial Campuses.

- **Architecture**

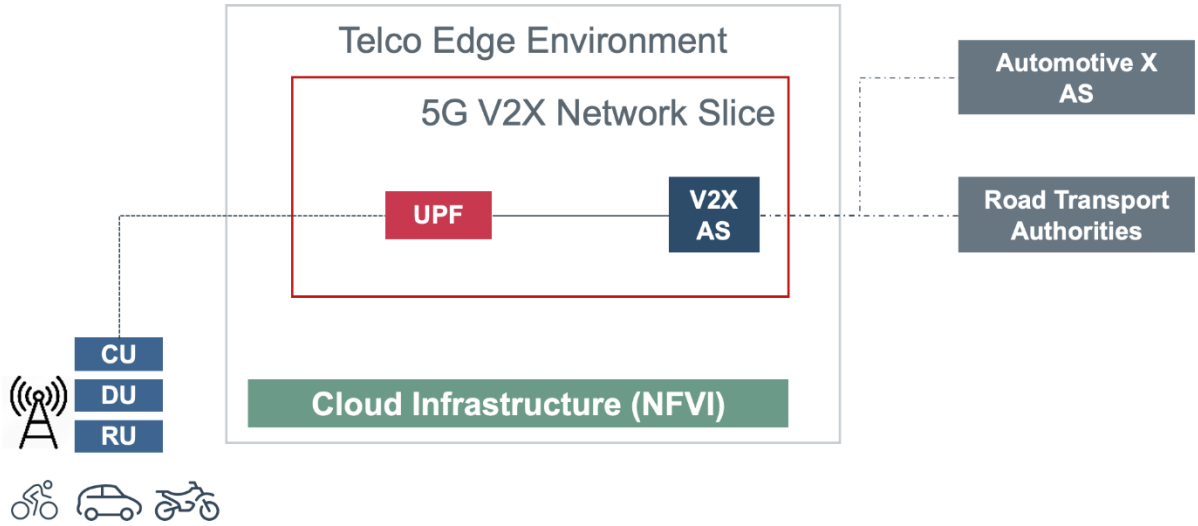

Figure 4: Edge Automotive (V2X) with uRLLC Core Network Slicing.

- 5G NR-V2X is a work item in 3GPP Release 16 that is not completed yet by the time of writing this document.
- C-V2X, Cellular V2X has two modes of communications
    - Direct Mode (Commonly described by SL, Sidelink by 3GPP): This includes the V2V, V2I, & V2P using a direct Interface (PC5) operating in ITS, Intelligent Transport Bands (e.g. 5.9 GHZ).
    - Network Mode (UL/DL): This covers the V2N while operating in the common telecom licensed spectrum. This use case is capitalizing on this mode.
- The potential use cases that may consume services from Edge is the Network Model (V2N) and potentially the V2I (According on how the Infrastructure will be mapped to an Edge level)

- **Use Case #4 – Edge vRAN Deployments**
    - **Business Objectives**

      vRAN is one of the trending technologies of RAN deployment that fits for all Radio Access Technologies. vRAN helps to provide coverage for rural & uncovered areas with a compelling CAPEX reduction compared to Traditional and legacy RAN deployments. This coverage can be extended to all area types with 5G greenfield deployment as a typical example.

    - **Targeted Segments**
        - Private 5G Customers (vRAN Can be part of the Non-Public Network, NPN)
        - B2B Customers & MVNOs (vRAN Can be part of an E2E Network Slicing)
        - B2C (Mobile Consumers Segment).

- **Architecture**

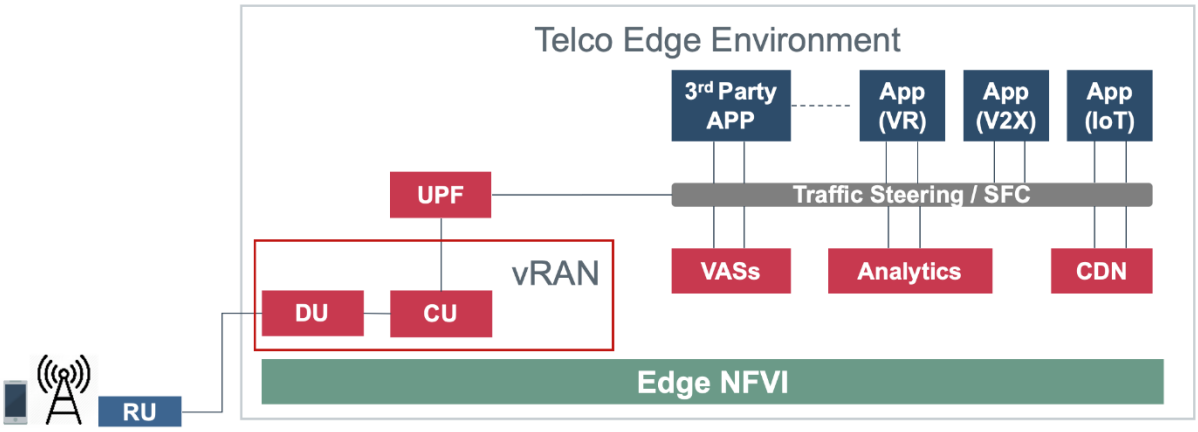

Figure 5: Edge vRAN Deployments.

- There are multiple deployment models for Centralized Unit (CU) & Distributed Unit (DU). This use case covers the placement case of having the DU & CU collocated & deployed on Telco Edge, see NGMN Overview on 5GRAN Functional Decomposition ver 1.0 [12]
- The use case covers the 5G vRAN deployment. However, this can be extended to cover 4G vRAN as well.
- Following Split Option 7.2, the average market latency for RU-DU (Fronthaul) is 100 microsec – 200 microsec while the latency for DU-CU (Midhaul) is tens of milliseconds, see ORAN-WG4.IOT.0-v01.00 [13].

2.3 Analysis

Studying various requirements of workloads helps understanding what expectation they will have from the underlying cloud infrastructure. Following are *some* of the requirement types on which various workloads might have different expectation levels:

- **Computing**
  - Speed (e.g., CPU clock and physical cores number)
  - Predictability (e.g., CPU and RAM sharing level)
  - Specific processing (e.g., cryptography, transcoding)
- **Networking**
  - Throughput (i.e., bit rate and/or packet rate)
  - Latency
  - Connection points / interfaces number (i.e., vNIC and VLAN)
  - Specific traffic control (e.g., firewalling, NAT, cyphering)
  - Specific external network connectivity (e.g., MPLS, VXLAN)
- **Storage**
  - IOPS (i.e., input/output rate and/or byte rate)
  - Volume
  - Ephemeral or Persistent

    - Specific features (e.g., object storage, local storage)

By trying to sort workloads into different categories based on the requirements observed, below are the different profiles concluded, which are mainly driven by expected performance levels:

- **Profile One**
  - Workload types
    - Control plane functions without specific need, and management plane functions
    - Examples: OFCS, AAA, NMS
  - No specific requirement
- **Profile Two**
  - Workload types
    - Data plane functions (i.e., functions with specific networking and computing needs)
    - Examples: BNG, S/PGW, UPF, Sec-GW, DPI, CDN, SBC, MME, AMF, IMS-CSCF, UDR
  - Requirements
    - Predictable computing
    - High network throughput
    - Low network latency

2.4 Cloud Infrastructure Profiles

Based on the above analysis, following cloud infrastructure profiles are proposed (also shown in Figure 6 below)

- **Basic**: for Workloads that can tolerate resource over-subscription and variable latency.
- **Network Intensive**: for Workloads that require predictable computing performance, high network throughput and low network latency.

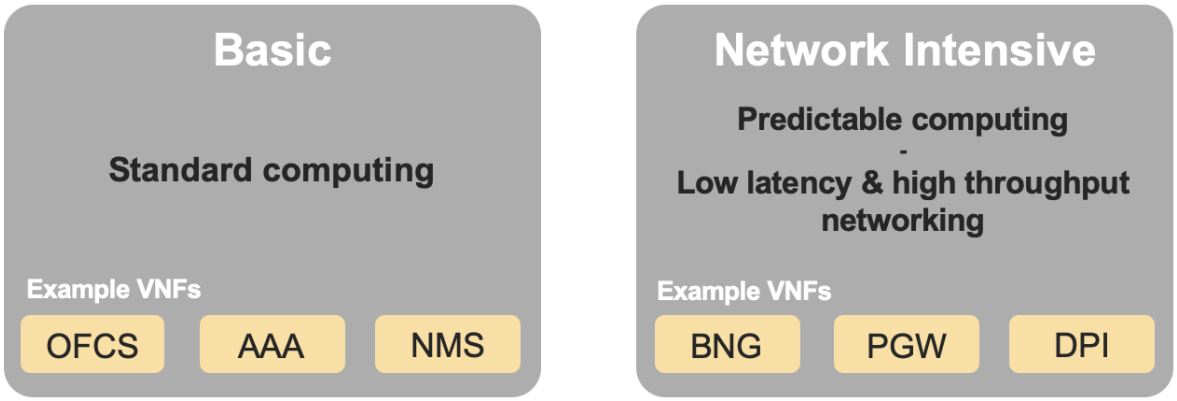

Figure 6: Infrastructure profiles proposed based on VNFs categorisation.

In Chapter 4 these **B (Basic)** and **N (Network intensive)** infrastructure profiles will be defined in greater detail for use by workloads:

Note: This is an initial set of proposed profiles and it is expected that more profiles will be added as more requirements are gathered and as technology enhances and matures. For instance, the following profiles may be added in future releases:

- **Compute Intensive**: for Workloads that require predictable computing performance and low network latency.
- ***Storage Intensive****: for Workloads that require low storage latency and/or high storage IOPS.*
- ***Enhanced Compute Intensive****: for compute intensive Workloads that require higher computing performance and/or specific compute resource (e.g., GPU).*
- ***Enhanced Network Intensive****: for network intensive Workloads that require higher network performance and/or specific network resource (e.g., crypto acceleration).*

3 Modelling

It is necessary to clearly define the infrastructure resources and their capabilities a shared cloud infrastructure (network function virtualisation infrastructure, NFVI) will provide for hosting workloads including virtual network functions (VNFs) and/or cloud-native network functions (CNFs). The lack of a common understanding of which resources and corresponding capabilities a suitable cloud infrastructure should provide may lead to several issues which could negatively impact the time and the cost for on-boarding and maintaining these solutions on top of a virtualised infrastructure.

The abstraction model presented in this Reference Model (RM) specifies a common set of virtual infrastructure resources that a cloud infrastructure will need to provide to be able to host most of the typical VNF/CNF telco workloads. The intention of this Reference Model is to follow the following principles:

- **Scope:** the model should describe the most relevant virtualised infrastructure resources (incl. acceleration technologies) a cloud infrastructure needs to host Telco workloads
- **Separation of Concern:** the model should support a clear distinction between the responsibilities related to maintaining the network function virtualisation infrastructure and the responsibilities related to managing the various VNF workloads
- **Simplicity:** the amount of different types of resources (including their attributes and relationships amongst one another) should be kept to a minimum to reduce the configuration spectrum which needs to be considered
- **Declarative:** the model should allow for the description of the intended state and configuration of the cloud infrastructure resources for automated life cycle management
- **Explicit:** the model needs to be rich enough to allow for the instantiation and the on-going operation of the cloud infrastructure
- **Lifecycle:** the model must distinguish between resources which have independent lifecycles but should group together those resources which share a common lifecycle
- **Aligned:** the model should clearly highlight the dependencies between its components to allow for a well-defined and simplified synchronisation of independent automation tasks.

To summarise: the abstraction model presented in this document will build upon existing modelling concepts and simplify and streamline them to the needs of telco operators who intend to distinguish between infrastructure related and workload related responsibilities.

3.1 Model

The abstraction model for the cloud infrastructure is divided into two logical layers: the virtual infrastructure layer and the hardware infrastructure layer, with the intention that only the virtual infrastructure layer will be directly exposed to workloads (VNFs/CNFs):

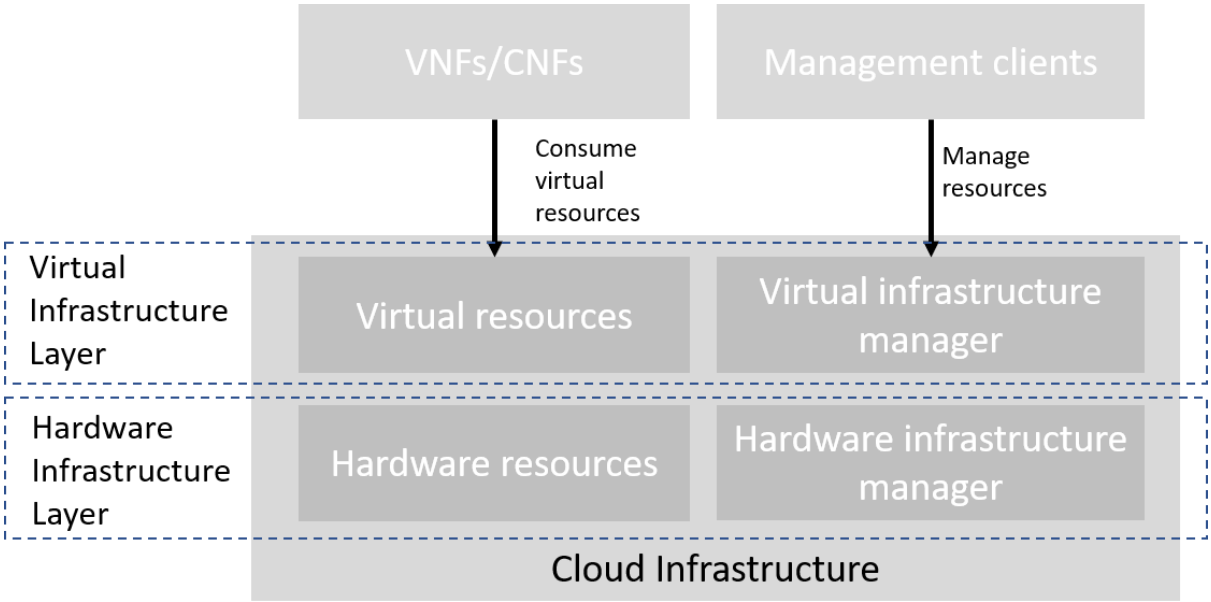

Figure 7: Cloud Infrastructure Model Overview.

The functionalities of each layer are as follows:

- **Virtual infrastructure resources:** These are all the infrastructure resources (compute, storage and networks) which the cloud infrastructure provides to the workloads such as VNFs/CNFs. These virtual resources can be managed by the tenants and tenant workloads directly or indirectly via an application programming interface (API).
- **Virtual infrastructure manager:** This consists of the software components that manage the virtual resources and make those management capabilities accessible via one or more APIs. The responsibilities of this functionality include the management of logical constructs such as tenants, tenant workloads, resource catalogues, identities, access controls, security policies, etc.
- **Hardware infrastructure manager:** This is a logical block of functionality responsible for the management of the abstracted hardware resources (compute, network and storage) and as such it is shielded from the direct involvement with server host software.
- **Physical Infrastructure Resources:** These consist of physical hardware components such as servers, (including random access memory, local storage, network ports, and hardware acceleration devices), storage devices, network devices, and the basic input output system (BIOS).
- **Workloads (VNFs/CNFs):** These consist of workloads such as virtualized and/or containerized network functions that run within a virtual machine (VM) or as a set of containers.

3.2 Virtual Resources

The virtual infrastructure resources provided by the Cloud Infrastructure can be grouped into four categories as shown in the diagram below:

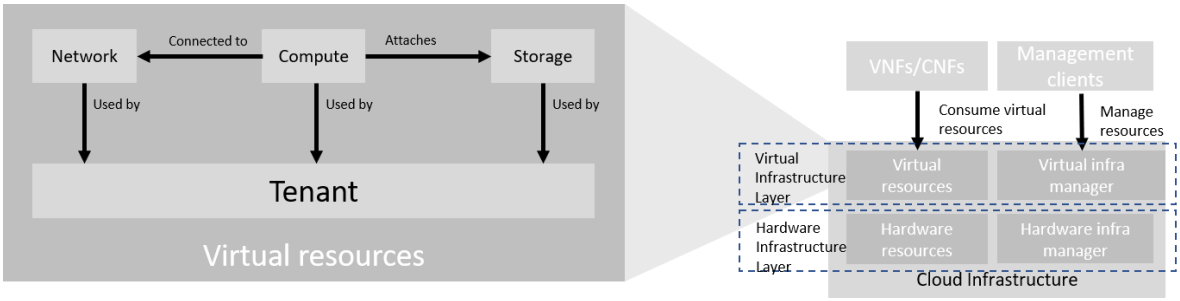

Figure 8: Virtual Infrastructure Resources provide virtual compute, storage and networks in a tenant context.

- **Tenants:** represent an isolated and independently manageable elastic pool of compute, storage and network resources
- **Compute resources:** represent virtualised computes for workloads and other systems as necessary
- **Storage resources:** represent virtualised resources for persisting data
- **Network resources:** represent virtual resources providing layer 2 and layer 3 connectivity

The virtualised infrastructure resources related to these categories are listed below.

3.2.1 Tenant

A cloud infrastructure needs to be capable of supporting multiple tenants and has to isolate sets of infrastructure resources dedicated to specific workloads (VNF/CNF) from one another. Tenants represent an independently manageable logical pool of compute, storage and network resources abstracted from physical hardware.

Example: a tenant within an OpenStack environment or a Kubernetes cluster.

Attribute	Description
name	name of the logical resource pool
type	type of tenant (e.g. OpenStack tenant, Kubernetes cluster, …)
vcpus	max. number of virtual CPUs
ram	max. size of random access memory in GB
disk	max. size of ephemeral disk in GB
networks	description of external networks required for inter-domain connectivity
metadata	key/value pairs for selection of the appropriate physical context (e.g. location, availability zone, …)

Table 1: Attributes of a tenant

3.2.2 Compute

A virtual machine or a container/pod is used by a tenant capable of hosting the application components of workloads (VNFs). A virtual compute therefore requires a tenant context and, since it will need to communicate with other communication partners, it is assumed that the networks have been provisioned in advance.

Example: a virtual compute descriptor as defined in TOSCA Simple Profile for NFV.

Attribute	Description
name	name of the virtual host
vcpus	number of virtual cpus
ram	size of random access memory in GB
disk	size of root disc in GB
nics	sorted list of network interfaces connecting the host to the virtual networks
acceleration	key/value pairs for selection of the appropriate acceleration technology
metadata	key/value pairs for selection of the appropriate redundancy domain

Table 2: Attributes of compute resources

3.2.3 Storage

A workload can request different types of storage based on data longevity: persistent or ephemeral storage. Persistent storage outlives the compute instance whereas ephemeral storage is linked to compute instance lifecycle.

There are multiple storage performance requirements such as latency, IOPS and capacity. For example, a workload may require one of its storage device to provide low latency, high

IOPS and very large/huge storage capacity (terabytes of data). Low Latency storage is for workloads which have strong constraints on the time to access the storage. High IOPS oriented storage is for workloads requiring lots of read/write actions. Capacity oriented storage is for workloads that need lots of volumetry without strong performance constraints.

Storage resources have the following attributes:

Attribute	Description
name	name of storage resources
data availability	persistent or ephemeral
performance	latency, IOPS, capacity
enhanced features	replication, encryption
type	block, object or file
size	size in GB

Table 3: Attributes of storage resources

3.2.4 Availability Zone

An availability zone is a logical pool of physical resources (e.g. compute, block storage, and network). These logical pools segment the physical resources of a cloud based on factors chosen by the cloud operator. The cloud operator may create availability zones based on location (rack, datacentre), or indirect failure domain dependencies like power sources. Workloads can leverage availability zones to utilise multiple locations or avoid sharing failure domains for a workload, and thus increase its fault-tolerance.

As a logical group with operator-specified criteria, the only mandatory attribute for an Availability Zone is the name.

Attribute	Description
name	name of the availability zone

Table 4: Attributes of availability zones

3.3 Cloud Infrastructure Management

Cloud infrastructure provides the capability to manage virtual and hardware resources via Application Programmable Interfaces or graphical user interfaces.

3.3.1 Virtual Infrastructure Manager

The virtual infrastructure manager allows to:

- setup, manage and delete tenants,
- setup, manage and delete user- and service-accounts,
- manage access privileges and
- provision, manage, monitor and delete virtual resources.

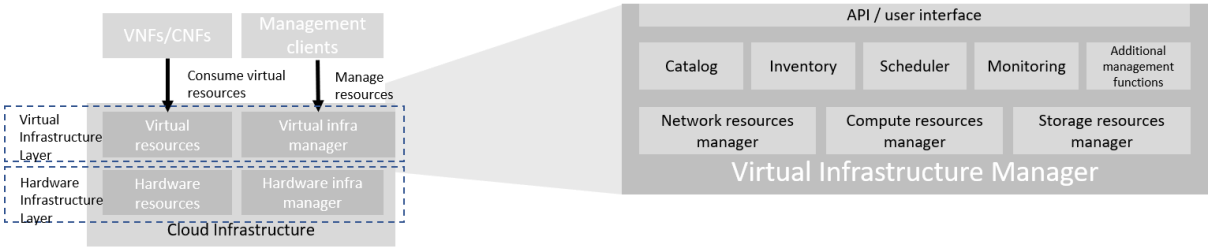

Figure 9: Virtual Infrastructure Manager.

The virtual infrastructure manager needs to support the following functional aspects:

- **API/UI**: an application programming interface / user interface providing access to the virtual resource management function
- **Catalogue**: manages the collection of available templates for virtual resource the cloud infrastructure can provide
- **Inventory**: manages the information related to virtual resources of a cloud infrastructure
- **Scheduler**: receives requests via API/UI, provisions and manages virtual resources by coordinating the activities of the compute-, storage- and network resources managers
- **Monitoring**: monitors and collects information on all events and the current state of all virtual resources
- **Additional Management Functions**: include identity management, access management, policy management (e.g. to enforce security policies), etc.
- **Compute Resources Manager**: provides a mechanism to provision virtual resources with the help of hardware compute resources
- **Storage Resources Manager**: provides a mechanism to provision virtual resources with the help of hardware storage resources
- **Network Resources Manager**: provides a mechanism to provision virtual resources with the help of hardware network resources

3.3.2 Hardware Infrastructure Manager

The hardware infrastructure manager allows to:

- provision, manage, monitor and delete hardware resources (underlay network, physical compute, physical storage, accelerators)
- manage hardware resource discovery and topology
- manage equipment
- manage hardware infrastructure telemetry and log collection services

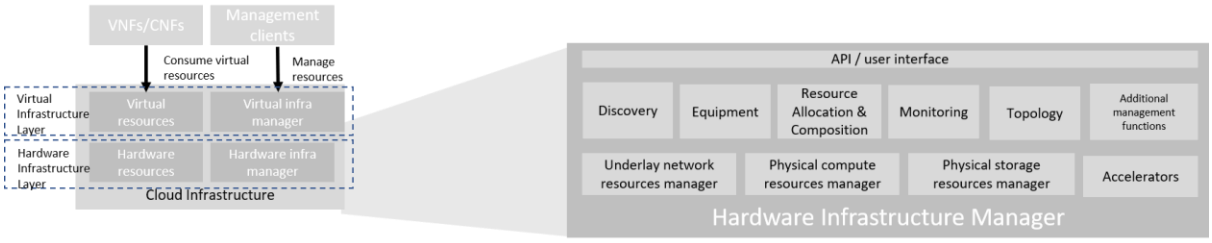

Figure 10: Hardware Infrastructure Manager.

The hardware infrastructure manager needs to support the following functional aspects:

- **API/UI**: an application programming interface / user interface providing access to the hardware resource management functions
- **Discovery**: discover and manages the information related to hardware resources of a cloud infrastructure
- **Equipment**: discover and manages the information related to hardware resources of a cloud infrastructure
- **Resource Allocation and Composition**: creates and allocates abstracted hardware resources
- **Monitoring**: monitors and collects information on all events and the current state of all hardware resources
- **Topology**: manages topological view of hardware resources
- **Additional Management Functions**: include identity management, access management, policy management (e.g. to enforce security policies), etc.
- **Underlay Network Resources Manager**: provides a mechanism to provision hardware resources for the use by the underlay network (e.g. switch fabric, smartNICs)
- **Physical Compute Resources Manager**: provides a mechanism to provision hardware compute resources
- **Physical Storage Resources Manager**: provides a mechanism to provision hardware storage resources
- **Accelerators**: provide a mechanism to provision hardware accelerator services

3.4 Hardware Infrastructure Resources

The physical compute, storage and network resources serve as the foundation of the cloud infrastructure. They are as such not directly exposed to the workloads (VNFs/CNFs).

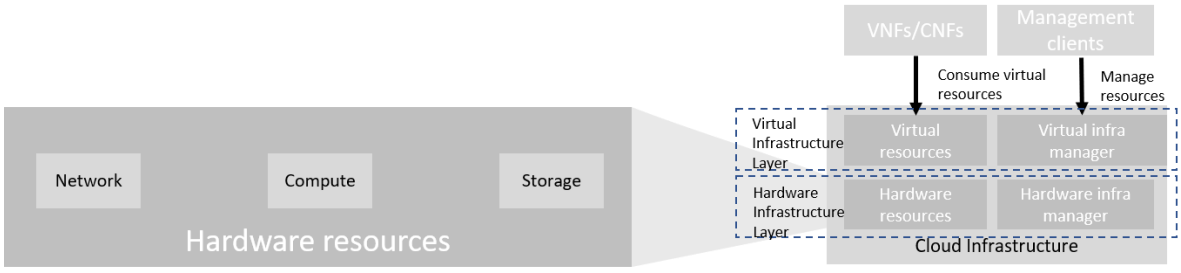

Figure 11: Hardware Infrastructure Resources

3.5 Network

Networking, alongside Compute and Storage, is an integral part of the Cloud Infrastructure (Network Function Virtualisation Infrastructure). The general function of networking in this context is to provide the connectivity between various virtual and physical resources required for the delivery of a network service. Such connectivity may manifest itself as a virtualised network between VMs and/or containers (e.g. overlay networks managed by SDN controllers, and/or programmable network fabrics) or as an integration into the infrastructure hardware level for offloading some of the network service functionality.

Normalization of the integration reference points between different layers of the Cloud Infrastructure architecture is one of the main concerns. In the networking context the primary focus is directed on the packet flow and control flow interfaces between the virtual resources (referred to as Software (SW) Virtualisation Layer) and physical resources (referred to as Hardware (HW) Infrastructure Layer), as well as on related integration into the various MANO reference points (hardware/network infrastructure management, orchestration). The identification of these two different layers (SW Virtualisation Layer and HW Infrastructure Layer) remains in alignment with the separation of resources into virtual and physical resources, generally used in this document, see e.g. Figure 7. The importance of understanding the separation of concerns between SW Virtualisation Layer and HW Infrastructure Layer is important because without it, the cardinality of having multiple CaaS and IaaS instances executing on their own private virtual resources from the single shared HW Infrastructure Layer cannot be expressed into separate administrative domains.

Principles that should be followed during the development and definition of the networking scope for the Reference Model, Reference Architectures, Reference Implementations and Reference Conformance test suites:

- Abstraction: A standardized network abstraction layer between the Virtualisation Layers and the Network Physical Resources Layer that hides (or abstracts) the details of the Network Physical resources from the Virtualisation Layers.

Note: In deployment phases this principle may be applied in many different ways e.g. depending on target use case requirements, workload characteristics, different algorithm implementations of pipeline stages and available platforms. The network abstraction layer supports, for example, physical resources with or without programmable hardware acceleration, or programmable network switches

- Agnosticism: Define Network Fabric concepts and models that can carry any type of traffic in terms of:
    - Control, User and Management traffic types
    - Acceleration technologies that can support multiple types of infrastructure deployments and network function workloads
- Automation: Enable end-to-end automation, from Physical Fabric installation and provisioning to automation of workloads (VNF/CNF) onboarding.

- Openness: All networking is based on open source or standardized APIs (North Bound Interfaces (NBI) and South Bound Interfaces (SBI)) and should enable integration of open source networking components such as SDN controllers.
- Programmability: Network model enables a programmable forwarding plane controlled from a separately deployed control plane.
- Scalability: Network model enables scalability to handle all traffic traverse North-South and East-West enabling small up to large deployments in a non-blocking manner.
- Workload agnostic: Network model is capable of providing connectivity to any type of workloads, including VNF, CNF and BareMetal workloads.
- Carrier Grade: Network model is capable of supporting deployments of the carrier grade workloads.
- Future proof: Network model is extendible to support known and emerging technology trends including SmartNICs, FPGAs and Programmable Switches, integrated for multi-clouds, and Edge related technologies.

3.6 Storage

The general function of storage subsystem is to provide the needed data store to various virtual and physical resources required for the delivery of a network service. In cloud infrastructure such storage may manifest itself in various ways like storage endpoints being exposed over network from software defined storage dedicated clusters or hyperconverged nodes (combining storage and other functions like compute or networking). Storage also follows the alignment of separated virtual and physical resources of SW Virtualization Layer and HW infrastructure. Reasons for such alignment are described more in section 3.5. The following principles apply to Storage scope for the Reference Model, Reference Architectures, Reference Implementations and Reference Conformance test suites:

- Abstraction: A standardized storage abstraction layer between the Virtualisation Layers and the Storage Physical Resources Layer that hides (or abstracts) the details of the Storage Physical resources from the Virtualisation Layers.
- Agnosticism: Define Storage subsystem concepts and models that can provide various storage types and performance requirements (more in Virtual Resources section 3.2.3 Storage).
- Automation: Enable end-to-end automation, from Physical Storage installation and provisioning to automation of workloads (VNF/CNF) onboarding.
- Openness: All storage is based on open source or standardized APIs (North Bound Interfaces (NBI) and South Bound Interfaces (SBI)) and should enable integration of storage components such as Software Defined Storage controllers.
- Scalability: Storage model enables scalability to enable small up to large deployments.
- Workload agnostic: Storage model can provide storage functionality to any type of workloads, including VNF, CNF and BareMetal workloads.
- Future proof: Storage model is extendible to support known and emerging technology trends covering spectrum of memory-storage technologies including Software Defined Storage with mix of SATA- and NVMe-based SSDs, DRAM and Persistent Memory, integrated for multi-clouds, and Edge related technologies.

3.7 Sample reference model realization

The following diagram presents an example of the realization of the reference model, where a virtual infrastructure layer contains three coexisting but different types of implementation: a typical IaaS using VMs and a hypervisor for virtualisation, a CaaS on VM/hypervisor, and a CaaS on bare metal. This diagram is presented for illustration purposes only and it does not preclude validity of many other different combinations of implementation types. Note that the model enables several potentially different controllers orchestrating different type of resources (virtual and/or hardware). Management clients can manage virtual resources via Virtual Infrastructure Manager (Container Infrastructure Service Manager for CaaS, or Virtual Infrastructure Manager for IaaS), or alternatively hardware infrastructure resources via hardware infrastructure manager. The latter situation may occur for instance when an orchestrator (an example of a management client) is involved in provisioning the physical network resources with the assistance of the controllers. Also, this realization example would enable implementation of a programmable fabric.

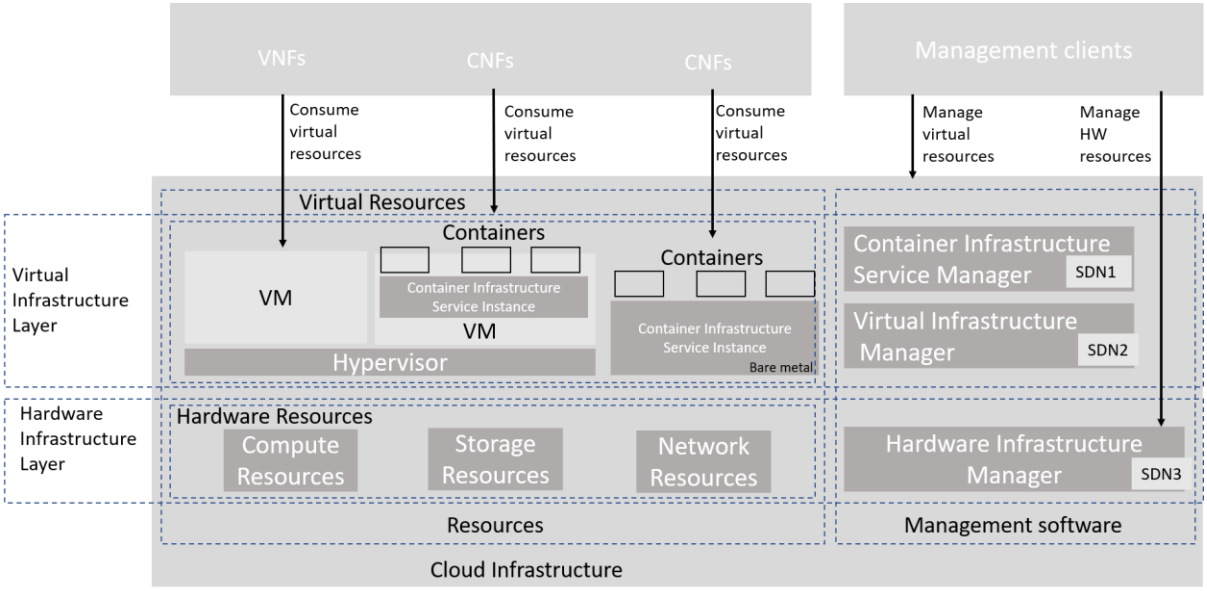

Figure 12: Reference model realization example

The terms Container Infrastructure Service Instance and Container Infrastructure Service Manager should be understood as defined in ETSI GR NFV-IFA 029 V3.3.1 [4]. More detailed deployment examples can be found in section 4.3 of this Reference Model document.

4 Infrastructure Capabilities, Measurements and Catalogue

4.1 Capabilities and Performance Measurements

This section describes and uniquely identifies the Capabilities provided directly by the Infrastructure, as well as Performance Measurements (PMs) generated directly by the Infrastructure (i.e. without the use of external instrumentation).

The Capability and PM identifiers conform to the following schema:

```
a.b.c (Ex. "e.pm.001")
a = Scope <(e)xternal | (i)nternal | (t)hird_party_instrumentation>
b = Type <(cap) capability | (man) management | (pm) performance | (man-
pm)>
c = Serial Number
```

4.1.1 Exposed vs Internal

The following pertains to the context of Cloud Infrastructure Resources, Capabilities and Performance Measurements (PMs) as discussed within this chapter.

Exposed: Refers to any object (e.g., resource discovery/configuration/consumption, platform telemetry, Interface, etc.) that exists in or pertains to, the domain of the Cloud Infrastructure and is made visible (aka “Exposed”) to a workload. When an object is exposed to a given workload, the scope of visibility within a given workload is at the discretion of the specific workload’s designer. From an Infra perspective, the Infra-resident object is simply being exposed to one or more virtual environments (i.e. Workloads). It is then the responsibility of the kernel or supervisor/executive within the VM to control how, when and where the object is further exposed within the VM, with regard to permissions, security, etc. As the object(s) originate with the Infra, they are by definition visible within that domain.

Internal: Effectively the opposite of Exposed; objects Internal to the Cloud Infrastructure, which are exclusively available for use by the Cloud Infrastructure and components within the Cloud Infrastructure.

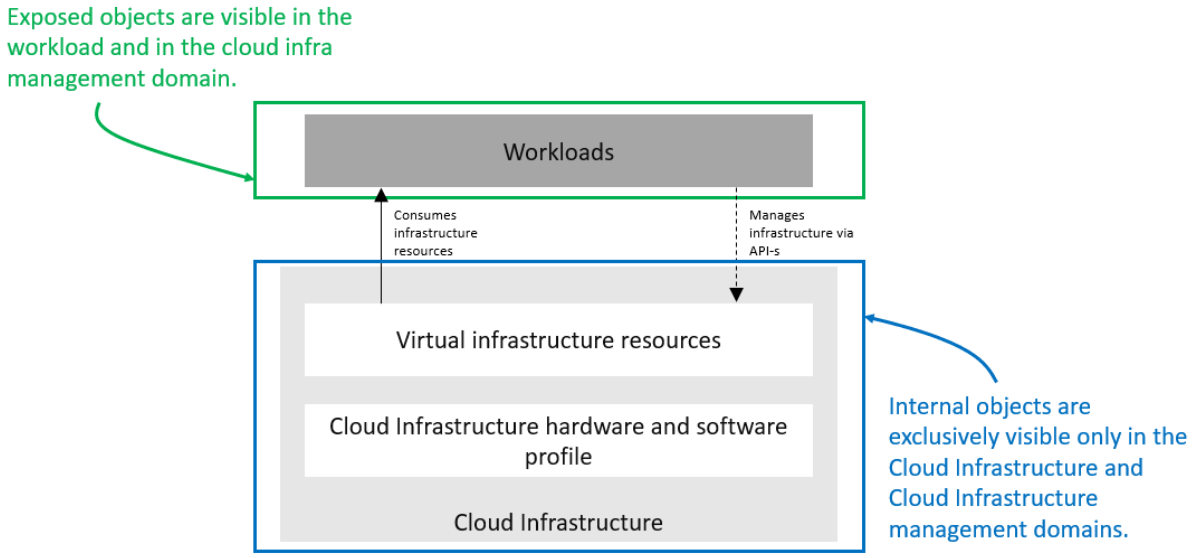

Figure 13: Exposed vs. Internal Scope

As illustrated in the figure above, objects designated as "Internal" are only visible within the area inside the blue oval (the Cloud Infrastructure), and only when the entity accessing the object has the appropriate permissions. Whereas objects designated as "Exposed" are potentially visible from both the area within the green oval (the Workload), as well as from within the Cloud Infrastructure, again provided the entity accessing the object has appropriate permissions.

Note: The figure above indicates the areas from where the objects are visible. It is not intended to indicate where the objects are instantiated. For example, the virtual resources are instantiated within the Cloud Infrastructure (the blue area), but are Exposed, and therefore are visible to the Workload, within the green area.

4.1.2 Exposed Infrastructure Capabilities

This section describes a set of explicit Cloud Infrastructure capabilities and performance measurements that define a Cloud Infrastructure. These capabilities and PMs are well known to workloads as they provide capabilities which workloads rely on.

Note: It is expected that Cloud Infrastructure capabilities and measurements will expand over time as more capabilities are added and technology enhances and matures.

4.1.2.1 Exposed Resource Capabilities

Table 5 below shows resource capabilities of Cloud Infrastructure. Those indicate resources offered to workloads by Cloud Infrastructure.

Ref	Cloud Infrastructure Capability	Unit	Definition/Notes
e.cap.001	# vCPU	number	Max number of vCPU that can be assigned to a single VM or Pod [1)]
e.cap.002	RAM Size	MB	Max memory in MB that can be assigned to a single VM or Pod by the Cloud Infrastructure [2)]
e.cap.003	Total per-instance (ephemeral) storage	GB	Max storage in GB that can be assigned to a single VM or Pod by the Cloud Infrastructure
e.cap.004	# Connection points	number	Max number of connection points that can be assigned to a single VM or Pod by the Cloud Infrastructure
e.cap.005	Total external (persistent) storage	GB	Max storage in GB that can be attached / mounted to VM or Pod by the Cloud Infrastructure

Table 5: Exposed Resource Capabilities of Cloud Infrastructure

Notes: **1)** In a Kubernetes based environment this means the CPU limit of a pod.
2) In a Kubernetes based environment this means the memory limit of a pod.

4.1.2.2 Exposed Performance Optimisation Capabilities

Table 6 shows possible performance optimisation capabilities that can be provided by Cloud Infrastructure. These indicate capabilities exposed to workloads. These capabilities are to be consumed by workloads in a standard way.

Ref	Cloud Infrastructure Capability	Unit	Definition/Notes
e.cap.006	CPU pinning	Yes/No	Indicates if Cloud Infrastructure supports CPU pinning
e.cap.007	NUMA alignment	Yes/No	Indicates if Cloud Infrastructure supports NUMA alignment

e.cap.008	IPSec Acceleration	Yes/No	IPSec Acceleration
e.cap.009	Crypto Acceleration	Yes/No	Crypto Acceleration
e.cap.010	Transcoding Acceleration	Yes/No	Transcoding Acceleration
e.cap.011	Programmable Acceleration	Yes/No	Programmable Acceleration
e.cap.012	Enhanced Cache Management	Yes/No	If supported, L=Lean; E=Equal; X=eXpanded. L and X cache policies require CPU pinning to be active.
e.cap.013	SR-IOV over PCI-PT	Yes/No	Traditional SR-IOV. These Capabilities generally require hardware-dependent drivers be injected into workloads. As such, use of these features shall be governed by the applicable CNTT policy. Consult the relevant RA specification for the usage policy relevant to any needed hardware capability of this type.
e.cap.014	GPU/NPU	Yes/No	Hardware coprocessor. These Capabilities generally require hardware-dependent drivers be injected into workloads. As such, use of these features shall be governed by the applicable CNTT policy. Consult the relevant RA specification for the usage policy relevant to any needed hardware capability of this type.
e.cap.015	SmartNIC	Yes/No	Network Acceleration. SmartNICs that do not utilise PCI-PT are not subject to the CNTT principles, nor any related policies or prohibitions.
e.cap.016	FPGA/other Acceleration H/W	Yes/No	Non-specific hardware. These Capabilities generally require hardware-dependent drivers be injected into workloads. As such, use of these features shall be governed by the applicable CNTT policy. Consult the relevant RA specification for the usage policy relevant to any needed hardware capability of this type.

Table 6: Exposed Performance Optimisation Capabilities of Cloud Infrastructure

Enhanced Cache Management is a compute performance enhancer that applies a cache management policy to the socket hosting a given virtual compute instance, provided the associated physical CPU microarchitecture supports it. Cache management policy can be used to specify the static allocation of cache resources to cores within a socket. The "Equal" policy distributes the available cache resources equally across all of the physical cores in the socket. The "eXpanded" policy provides additional resources to the core pinned to a workload that has the "X" attribute applied. The "Lean" attribute can be applied to workloads which do not realize significant benefit from a marginal cache size increase and are hence willing to relinquish unneeded resources.

In addition to static allocation, an advanced Reference Architecture implementation can implement dynamic cache management control policies, operating with tight (~ms) or standard (10s of seconds) control loop response times, thereby achieving higher overall performance for the socket.

4.1.2.3 Exposed Monitoring Capabilities

Monitoring capabilities are used for the passive observation of workload-specific traffic traversing the Cloud Infrastructure. As with all capabilities, Monitoring may be unavailable or intentionally disabled for security reasons in a given Cloud Infrastructure deployment. If this functionality is enabled, it must be subject to strict security policies. Refer to the Reference Model Security chapter for additional details.

Table 7 shows possible monitoring capabilities available from the Cloud Infrastructure for workloads.

Ref	Cloud Infrastructure Capability	Unit	Definition/Notes
e.cap.017	Monitoring of L2-7 data	Yes/No	Ability to monitor L2-L7 data from workload

Table 7: Exposed Monitoring Capabilities of Cloud Infrastructure

4.1.3 Exposed Infrastructure Performance Measurements

The intent of the following PMs is to be available for and well known to workloads.

4.1.3.1 Exposed Performance Measurements

The following table of exposed Performance Measurements shows PMs per VM or Pod, vNIC or vCPU. Network test setups are aligned with ETSI GS NFV-TST 009 [14]. Specifically exposed PMs use a single workload (PVP) data plane test setup in a single host.

Ref	Cloud Infrastructure Measurement	Unit	Definition/Notes
e.pm.xxx	Place Holder	Units	Concise description

Table 8: Exposed Performance Measurements of Cloud Infrastructure

4.1.4 Internal Infrastructure Capabilities

This section covers a list of implicit Cloud Infrastructure capabilities and measurements that define a Cloud Infrastructure. These capabilities and metrics determine how the Cloud Infrastructure behaves internally. They are hidden from workloads (i.e. workloads may not know about them) but they will impact the overall performance and capabilities of a given Cloud Infrastructure solution.

Note: It is expected that implicit Cloud Infrastructure capabilities and metrics will evolve with time as more capabilities are added as technology enhances and matures.

4.1.4.1 Internal Resource Capabilities

Table 9 shows resource capabilities of Cloud Infrastructure. These include capabilities offered to workloads and resources consumed internally by Cloud Infrastructure.

Ref	Cloud Infrastructure Capability	Unit	Definition/Notes
i.cap.014	CPU cores consumed by the Cloud Infrastructure overhead	%	The ratio of cores consumed by the Cloud Infrastructure components (including host OS) in

	on a worker (compute) node		a compute node to the total number of cores available expressed as a percentage
i.cap.015	Memory consumed by the Cloud Infrastructure overhead on a worker (compute) node	%	The ratio of memory consumed by the Cloud Infrastructure components (including host OS) in a worker (compute) node to the total available memory expressed as a percentage

Table 9: Internal Resource Capabilities of Cloud Infrastructure

4.1.4.2 Internal SLA capabilities

Table 10 below shows SLA (Service Level Agreement) capabilities of Cloud Infrastructure. These include Cloud Infrastructure capabilities required by workloads as well as required internal to Cloud Infrastructure. Application of these capabilities to a given workload is determined by its Cloud Infrastructure Profile.

Ref	Cloud Infrastructure capability	Unit	Definition/Notes
i.cap.016	CPU allocation ratio	N:1	Number of virtual cores per physical core; also known as CPU overbooking ratio
i.cap.017	Connection point QoS	Yes/No	QoS enablement of the connection point (vNIC or interface)

Table 10: Internal SLA capabilities to Cloud Infrastructure

4.1.4.3 Internal Performance Optimisation Capabilities

Table 11 below shows possible performance optimisation capabilities that can be provided by Cloud Infrastructure. These include capabilities exposed to workloads as well as internal capabilities to Cloud Infrastructure. These capabilities will be determined by the Cloud Infrastructure Profile used by the Cloud Infrastructure.

Ref	Cloud Infrastructure capability	Unit	Definition/Notes
i.cap.018	Huge pages	Yes/No	Indicates if the Cloud Infrastructure supports huge pages

Table 11: Internal performance optimisation capabilities of Cloud Infrastructure

4.1.4.4 Internal Performance Measurement Capabilities

Table 12 shows possible performance measurement capabilities available by Cloud Infrastructure. The availability of these capabilities will be determined by the Cloud Infrastructure Profile used by the workloads.

Ref	Cloud Infrastructure Measurement	Unit	Definition/Notes
i.pm.001	Host CPU usage	nanoseconds	Per Compute node. It maps to ETSI GS NFV-TST 008 V3.2.1 [5] clause 6, processor usage metric (Cloud Infrastructure internal).
i.pm.002	Virtual compute	nanoseconds	Per VM or Pod. It maps to ETSI GS NFV-IFA 027

Ref	Cloud Infrastructure Measurement	Unit	Definition/Notes
	resource CPU usage		v2.4.1 [6] Mean Virtual CPU usage and Peak Virtual CPU usage (Cloud Infrastructure external).
i.pm.003	Host CPU utilization	%	Per Compute node. It maps to ETSI GS NFV-TST 008 V3.2.1 [5] clause 6, processor usage metric (Cloud Infrastructure internal).
i.pm.004	Virtual compute resource CPU utilization	%	Per VM or Pod. It maps to ETSI GS NFV-IFA 027 v2.4.1 [6] Mean Virtual CPU usage and Peak Virtual CPU usage (Cloud Infrastructure external).
i.pm.005	Measurement of external storage IOPS	Yes/No	
i.pm.006	Measurement of external storage throughput	Yes/No	
i.pm.007	Available external storage capacity	Yes/No	

Table 12: Internal Measurement Capabilities of Cloud Infrastructure

4.1.5 Cloud Infrastructure Management Capabilities

The Cloud Infrastructure Manager (CIM) is responsible for controlling and managing the Cloud Infrastructure compute, storage, and network resources. Resources allocation is dynamically set up upon workloads requirements. This section covers the list of capabilities offered by the CIM to workloads or service orchestrator.

Table 13 shows capabilities related to resources allocation.

Ref	Cloud Infrastructure Management Capability	Unit	Definition/Notes
e.man.001	Virtual Compute allocation	Yes/No	Capability to allocate virtual compute resources to a workload
e.man.002	Virtual Storage allocation	Yes/No	Capability to allocate virtual storage resources to a workload
e.man.003	Virtual Networking resources allocation	Yes/No	Capability to allocate virtual networking resources to a workload
e.man.004	Multi-tenant isolation	Yes/No	Capability to isolate resources between tenants
e.man.005	Images management	Yes/No	Capability to manage workload software images
e.man.010	Compute Availability Zones	list of strings	The names of each Compute Availability Zone that was defined to separate failure domains
e.man.011	Storage Availability Zones	list of	The names of each Storage Availability Zone

		strings	that was defined to separate failure domains

Table 13: Cloud Infrastructure Management Resource Allocation Capabilities

4.1.6 Cloud Infrastructure Management Performance Measurements

Table 14 shows performance measurement capabilities.

Ref	Cloud Infrastructure Management Capability	Unit	Definition/Notes
e.man.006	Virtual resources inventory per tenant	Yes/No	Capability to provide information related to allocated virtualised resources per tenant
e.man.007	Resources Monitoring	Yes/No	Capability to notify state changes of allocated resources
e.man.008	Virtual resources Performance	Yes/No	Capability to collect and expose performance information on virtualised resources allocated
e.man.009	Virtual resources Fault information	Yes/No	Capability to collect and notify fault information on virtualised resources

Table 14: Cloud Infrastructure Management Performance Measurement Capabilities

4.1.6.1 Resources Management Measurements

Table 15 shows resource management measurements of CIM as aligned with ETSI GR NFV IFA-012 [15]. The intention of this table is to provide a list of measurements to be used in the Reference Architecture specifications, where the concrete values allowed for these measurements in the context of a particular Reference Architecture will be defined.

Ref	Cloud Infrastructure Management Measurement	Unit	Definition/Notes
e.man-pm.001	Time to create Virtual Compute resources (VM/container) for a given workload	Max ms	
e.man-pm.002	Time to delete Virtual Compute resources (VM/container) of a given workload	Max ms	
e.man-pm.003	Time to start Virtual Compute resources (VM/container) of a given workload	Max ms	
e.man-pm.004	Time to stop Virtual Compute resources (VM/container) of a given workload	Max ms	
e.man-pm.005	Time to pause Virtual Compute resources (VM/container) of a given workload	Max ms	
e.man-pm.006	Time to create internal virtual network	Max ms	
e.man-pm.007	Time to delete internal virtual network	Max ms	
e.man-pm.008	Time to update internal virtual network	Max ms	
e.man-pm.009	Time to create external virtual network	Max ms	
e.man-pm.010	Time to delete external virtual network	Max ms	

Ref	Cloud Infrastructure Management Measurement	Unit	Definition/Notes
e.man-pm.011	Time to update external virtual network	Max ms	
e.man-pm.012	Time to create external storage ready for use by workload	Max ms	

Table 15: Cloud Infrastructure management Resource Management Measurements

4.2 Infrastructure Profiles Catalogue

Infrastructure exposes compute Flavours with options, virtual interface options, storage extensions, and acceleration extensions to workloads. These Cloud Infrastructure Profiles are offered to workloads with their corresponding options and extensions.

The idea of the Cloud Infrastructure profiles is to have a predefined set of infrastructure capabilities with a predefined set of compute Flavours which workload vendors use to build their workloads. Each workload can use several Flavours from different Cloud Infrastructure Profiles to build its overall functionality as illustrated in Figure 14.

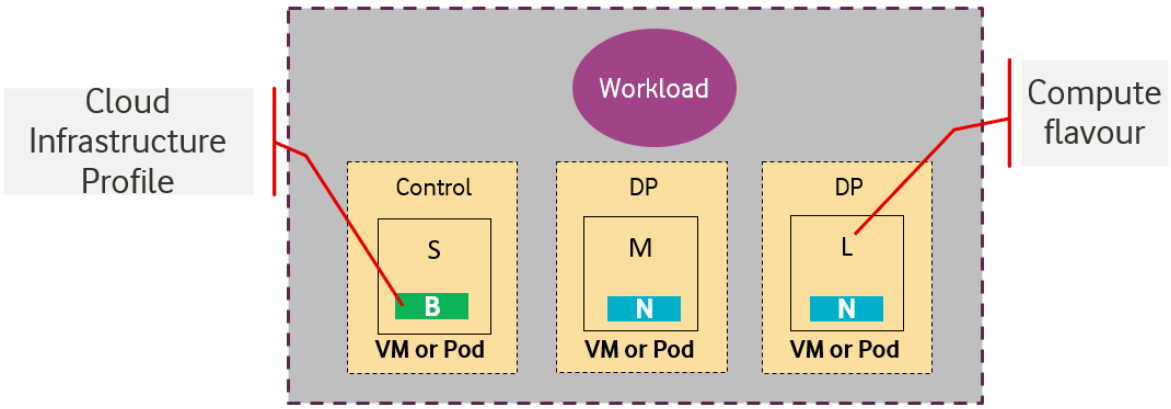

Figure 14: Workloads built against Cloud Infrastructure Profiles and compute Flavours.

4.2.1 Compute Flavours

Compute Flavours represent the compute, memory, storage, and management network resource templates that are used by VMs on the compute hosts. Each VM is given a compute Flavour (resource template), which determines the VMs compute, memory and storage characteristics.

Compute Flavours can also specify secondary ephemeral storage, swap disk, etc. A compute Flavour geometry consists of the following elements:

Element	Description
Compute Flavour Name	A descriptive name
Virtual compute resources (aka vCPUs)	Number of virtual compute resources (vCPUs) presented to the VM instance.

Element	Description
Memory	Virtual compute instance memory in megabytes.
Ephemeral/Local Disk	Specifies the size of an ephemeral data disk that exists only for the life of the instance. Default value is 0.The ephemeral disk may be partitioned into boot (base image) and swap space disks.
Management Interface	Specifies the bandwidth of management interface/s

Table 16: Compute Flavour Geometry Specification.

4.2.1.1 Predefined Compute Flavours

The intent of the following Flavours list is to be comprehensive and yet effective to cover both IT and NFV workloads. The compute Flavours are specified relative to the “large” Flavour. The “large” Flavour configuration consists of 4 vCPUs, 8 GB of RAM and 80 GB of local disk, and the resulting virtual compute instance will have a management interface of 1 Gbps. The “medium” Flavour is half the size of a large and small is half the size of medium. The tiny Flavour is a special sized Flavour.

Note: Customised (Parameterized) Flavours can be used in concession by operators and, if needed, are created using TOSCA, HEAT templates, and/or VIM APIs.

.conf	vCPU ("c") [2)]	RAM ("r") [2)]	Local Disk ("d")	Management Interface
.tiny	1	512 MB	1 GB	1 Gbps
.small	1	2 GB	20 GB	1 Gbps
.medium	2	4 GB	40 GB	1 Gbps
.large	4	8 GB	80 GB	1 Gbps
.2xlarge [1)]	8	16 GB	160 GB	1 Gbps
.4xlarge [1)]	16	32 GB	320 GB	1 Gbps
.8xlarge [1)]	32	64 GB	640 GB	1 Gbps

Table 17: Predefined Compute Flavours.

Notes: **1)** These compute Flavours are intended to be used for transitional purposes and workload vendors are expected to consume smaller Flavours and adopt microservices-based designs for their workloads.
2) In Kubernetes based environments these are the resource requests of the containers in the pods. To get guaranteed resources the resource requests should be set to the same values as the resource limits, to get burstable resources the resource limits should be higher than the resource requests while to get best effort resources none of resource requests of resource limits should be set.

4.2.2 Virtual Network Interface Specifications

The virtual network interface specifications extend a Flavour customization with network interface(s), with an associated bandwidth, and are identified by the literal, “n”, followed by the interface bandwidth (in Gbps). Multiple network interfaces can be specified by repeating the “n” option.

Virtual interfaces may be of an Access type, and thereby untagged, or may be of a Trunk type, with one or more 802.1Q tagged logical interfaces. Note, tagged interfaces are encapsulated by the Overlay, such that tenant isolation (i.e. security) is maintained, irrespective of the tag value(s) applied by the workload.

Note, the number of virtual network interfaces, aka vNICs, associated with a virtual compute instance, is directly related to the number of vNIC extensions declared for the environment. The vNIC extension is not part of the base Flavour.

`<network interface bandwidth option> :: <"n"><number (bandwidth in Gbps)>`

Virtual Network Interface Option	Interface Bandwidth
n1, n2, n3, n4, n5, n6	1, 2, 3, 4, 5, 6 Gbps
n10, n20, n30, n40, n50, n60	10, 20, 30, 40, 50, 60 Gbps
n25, n50, n75, n100, n125, n150	25, 50, 75, 100, 125, 150 Gbps
n50, n100, n150, n200, n250, n300	50, 100, 150, 200, 250, 300 Gbps
n100, n200, n300, n400, n500, n600	100, 200, 300, 400, 500, 600 Gbps

Table 18: Virtual Network Interface Specification Examples

4.2.3 Storage Extensions

Persistent storage is associated with workloads via Storage Extensions. The size of an extension can be specified explicitly in increments of 100GB, ranging from a minimum of 100GB to a maximum of 16TB. Extensions are configured with the required performance category, as per Table 19. Multiple persistent Storage Extensions can be attached to virtual compute instances.

Note: CNTT documentation uses GB and GiB to refer to a Gibibyte (2^{30} bytes), except where explicitly stated otherwise.

.conf	Read IO/s	Write IO/s	Read Throughput (MB/s)	Write Throughput (MB/s)	Max Ext Size
.bronze	Up to 3K	Up to 1.5K	Up to 180	Up to 120	16TB
.silver	Up to 60K	Up to 30K	Up to 1200	Up to 400	1TB
.gold	Up to 680K	Up to 360K	Up to 2650	Up to 1400	1TB

Table 19: Storage Extensions

Note: Performance is based on a block size of 256KB or larger.

4.2.4 Cloud Infrastructure Profiles

4.2.4.1 Basic Profile

This Cloud Infrastructure Profile is intended to be used for both IT workloads as well as NFV workloads. It has limited IO capabilities (up to 10Gbps Network interface).

4.2.4.2 Network Intensive Profile

This Cloud Infrastructure Profile is intended to be used for those applications that has high network throughput requirements (up to 50Gbps).

Network Acceleration Extensions

Network Intensive Profile can come with Network Acceleration extensions to assist workloads offloading some of their network intensive operations to hardware. The list below is preliminary and is expected to grow as more network acceleration resources are developed and standardized.

4.2.4.2.1

Note: Interface types are aligned with ETSI GS NFV-IFA 002 [7].

.conf	Interface type	Description
.il-ipsec	virtio-ipsec	In-line IPSec acceleration.
.la-crypto	virtio-crypto	Look-Aside encryption/decryption engine.

Table 20: Acceleration Extensions for Network Intensive Profile

Note: Need to work with relevant open source communities to create missing interfaces.

4.2.5 Cloud Infrastructure Profile Capabilities Mapping

Ref	Basic	Network Intensive	Notes
e.cap.001(#vCPU cores)	Per selected <Flavour>	Per selected <Flavour>	Exposed resource capabilities as per Table 5
e.cap.002(RAM Size (MB))	Per selected <Flavour>	Per selected <Flavour>	
e.cap.003(Total instance (ephemeral) storage (GB))	Per selected <Flavour>	Per selected <Flavour>	
e.cap.004(# Connection points)	Per selected	Per selected	
e.cap.005(Total external (persistent) storage (GB))	Per selected	Per selected	
e.cap.006(CPU pinning)	No	Yes	Exposed performance capabilities as per Table 6
e.cap.007(NUMA alignment)	No	Yes	
e.cap.008(IPSec Acceleration)	No	Yes (if offered)	
e.cap.009(Crypto Acceleration)	No	Yes (if offered)	
e.cap.010(Transcoding Acceleration)	No	No	

Ref	Basic	Network Intensive	Notes
e.cap.011(Programmable Acceleration)	No	Yes/No	
e.cap.012(Enhanced Cache Management)	E	E	
e.cap.013(SR-IOV over PCI-PT)	No	Yes	
e.cap.014(GPU/NPU)	No	Yes / No	Yes : in case of AI and Video Edge use cases
e.cap.015(SmartNIC)	No	Yes / No	
e.cap.016(FPGA/other Acceleration H/W)	No	Yes / No	Yes : in case of vRAN Edge use case
i.cap.014(CPU cores consumed by the Cloud Infrastructure on the worker nodes)	any	any	
i.cap.015(Memory consumed by Cloud Infrastructure on the worker nodes)	any	any	
i.cap.016(CPU allocation ratio)	1:1	1:1	Internal SLA capabilities as per Table 10. ***Note**: This is set to 1:1 for the Basic profile to enable predictable and consistent performance during benchmarking and certification. Operators may choose to modify this for actual deployments if they are willing to accept the risk of performance impact to workloads using the basic profile.*
i.cap.017(Connection point QoS)	No	Yes	
i.cap.018(Huge page support)	No	Yes	Internal performance capabilities as per Table 11
i.pm.001(Host CPU usage)	Yes	Yes	Internal monitoring capabilities as per Table 12
i.pm.002(Virtual compute resource CPU usage)	Yes	Yes	
i.pm.003(Host CPU utilization)	Yes	Yes	
i.pm.004(Virtual compute resource CPU utilization)	Yes	Yes	
i.pm.005(Measurement of external storage IOPS)	Yes	Yes	
i.pm.006(Measurement of external storage throughput)	Yes	Yes	
i.pm.007(Available external	Yes	Yes	

Ref	Basic	Network Intensive	Notes
storage capacity)			

Table 21: Mapping of Capabilities to Cloud Infrastructure Profiles

4.2.6 Void

Note: Reserved for future use

4.2.7 One stop shop

4.2.7.1 Naming convention

An entry in the infrastructure profile catalogue can be referenced using the following naming convention.

```
B/N <I opt> . <Flavour> . <S ext> . <A ext>
```

Whereas:

- **B/N**: specifies the Cloud Infrastructure Profile (Basic or Network Intensive)
- **<I opt>**: specifies the interface option.
- **<Flavour>**: specifies the compute Flavour.
- **<S ext>**: specifies an optional storage extension.
- **<A ext>**: specifies an optional acceleration extension for the Network Intensive Cloud Infrastructure Profile.

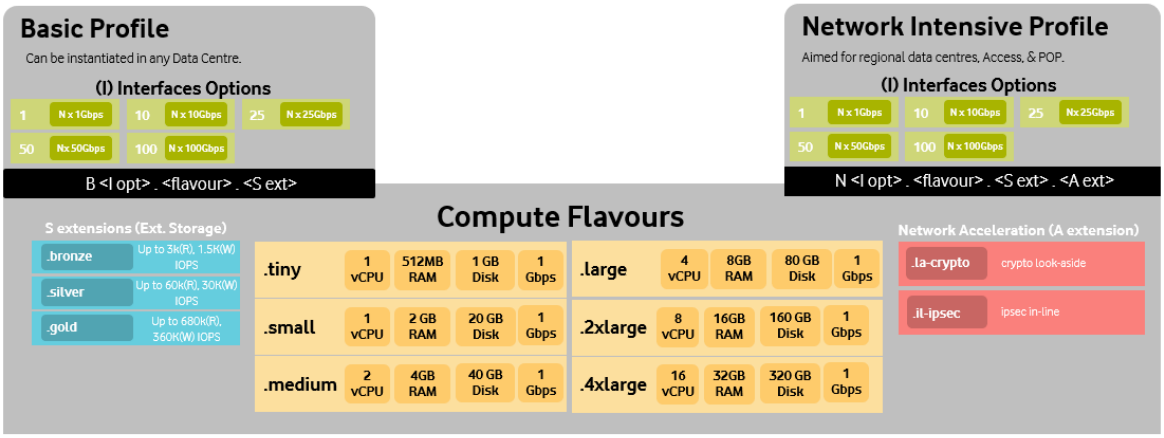

Figure 15: Infrastructure Profiles Catalogue

4.2.7.2 Backwards Compatibility

The Reference Model (RM) specification describes an infrastructure abstraction including a set of cloud infrastructure hardware and software profiles and compute flavours offered to workloads. The set of defined profiles and flavours will evolve along the releases but at the same time the existing workloads need to be supported. This means that any CNTT deployed cloud should be backwards compatible and support profiles and flavours from the latest three CNTT releases (N-2, N-1, N) as presented in Figure 16.

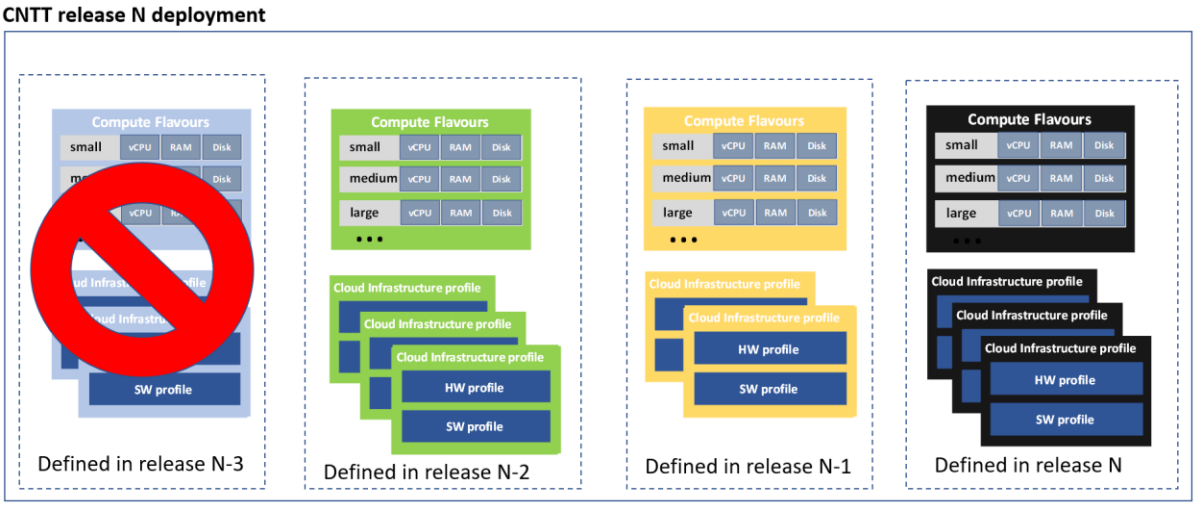

Figure 16: Backwards Compatibility

Cloud Infrastructure profiles that are available in CNTT release N deployment can be divided into two categories:

1. Cloud infrastructure profiles that are part of CNTT release N. These can be either

   - new profiles defined in release N or
   - existing profiles from earlier releases that are incorporated for backward compatibility reasons in release N

2. Cloud infrastructure profiles from releases N-1 and N-2 that are deployed only because of backwards compatibility, these profiles are not part of CNTT release N definition.

Note: a profile defined in previous releases that is modified in release N is considered to be a new profile

Different profile categories described above are presented in Figure 17. In this example profiles that are part of CNTT release N consist of two new profiles (yellow), one profile that is originally defined in release N-1 (green) and one defined in release N-2 (blue). Profiles that were defined in earlier releases but are also supported in release N will be referred to by several names. Existing workloads continue using the profile names from previous releases. New workloads will use release N naming.

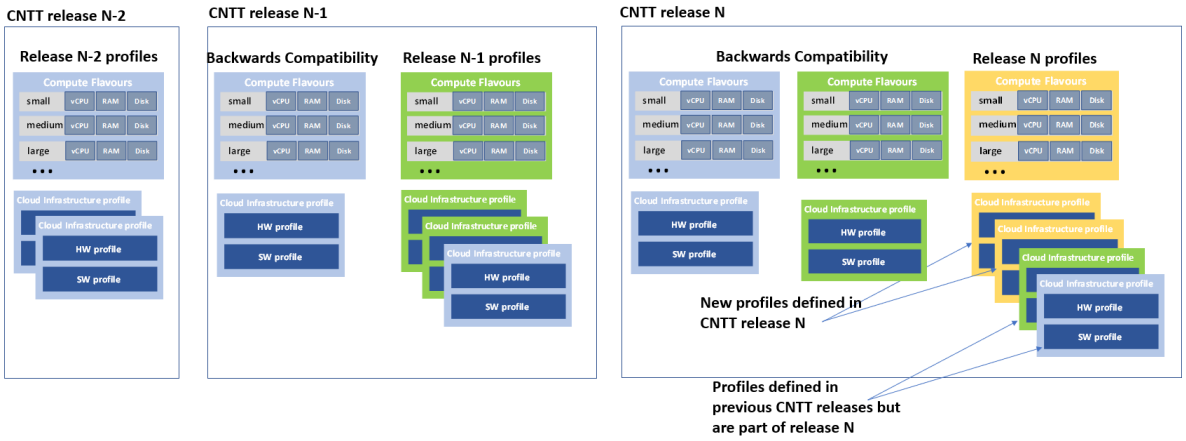

Figure 17: Cloud Infrastructure profiles in CNTT release N

Like predefined cloud infrastructure profiles, predefined compute flavours are also specified per CNTT release. CNTT release N flavours are used when new workloads are deployed into profiles that are part of the CNTT N release. Existing workloads continue using the flavours from previous releases. The difference in flavours can be for example, that newer flavours defined in release N may not have extra-large flavours that are earlier defined for transitional purposes. Workloads that use backwards compatible profiles will use the flavours from the older release (Figure 18).

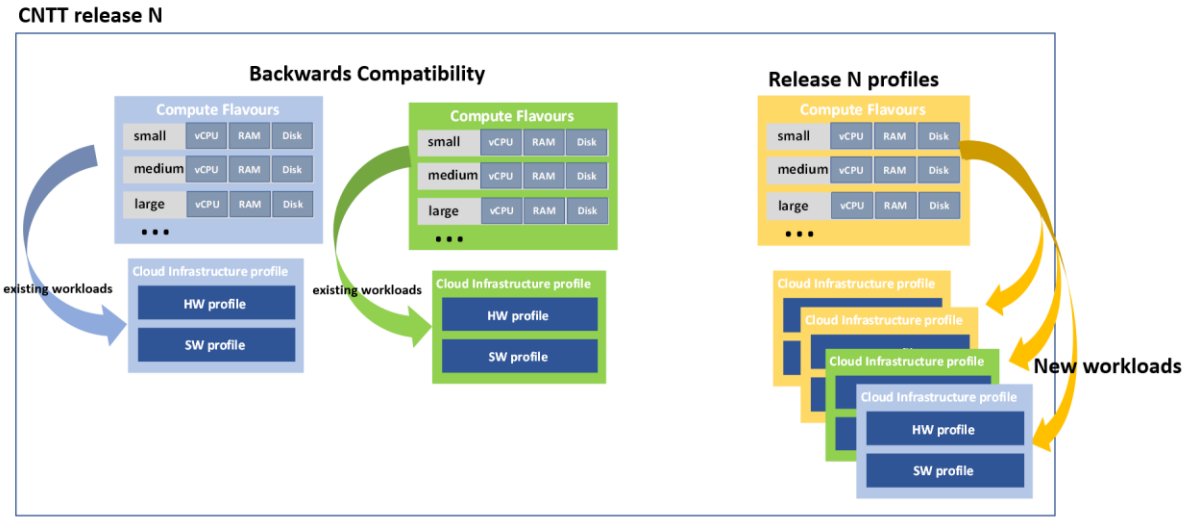

Figure 18: New workloads in Release N would use only Release N profiles

As discussed above backwards compatibility is the reason why cloud infrastructure profiles and flavours from several CNTT releases are configured and used in one CNTT deployment. Therefore, CNTT release number need to be added to each profile:

`B/N<"_Gen"><release #>. <Flavour>`

Flavours are unique only when combined with a profile. For example, CNTT release N small flavour in basic profile has the naming:

B_GenN.small

4.2.7.3 Forward compatibility

Technology specific exceptions are dealt with in the relevant RA specifications. Such exceptions are not part of any Cloud Infrastructure profile defined in CNTT.

4.3 Networking

The Cloud Infrastructure Networking Reference Model is an essential foundation that governs all Reference Architectures and Cloud Infrastructure implementations to enable multiple cloud infrastructure virtualisation technology choices and their evolution. These include:

- the current single Infrastructure as a Service (IaaS) based virtualisation instances with Virtual Machines (VM)
- multi IaaS based virtualisation instances
- Cloud Native Container as a Service (CaaS) based virtualisation instances, and
- hybrid multi IaaS and CaaS based virtualisation instances

To retain the cloud paradigms of automation, scalability and usage of shared hardware resources when introducing CaaS instances it is necessary to enable an ability to co-deploy multiple simultaneous IaaS and CaaS instances on a shared pool of hardware resources.

Compute and Storage resources are rarely shared in between IaaS or CaaS instances, but the underpinning networking, most commonly implemented with Ethernet and IP, must be shared and managed as a shared pool of underlay network resources to enable the pooled usage of Compute and Storage from a managed shared pool.

Throughout this chapter and its figures a number of references to ETSI NFV are made and they explicitly are made towards the ETSI NFV models in the Architectural Framework:

- ETSI GS NFV 002 V1.2.1 [3]
- ETSI GR NFV-IFA 029 V3.3.1 [4]

4.3.1 Network Layering and Concepts

Cloud and Telco networking are layered, and it is very important to keep the dependencies between the layers low to enable security, separation and portability in between multiple implementations and generations.

Before we start developing a deep model we need to agree on some foundational concepts and layering that allow decoupling of implementations in between the layers. We will emphasize four concepts in this section:

- Underlay and Overlay Networking concepts
- Hardware and Virtual Infrastructure Layer concepts
- Software Defined Underlay and Overlay Networking concepts
- Programmable Networking Fabric concept

4.3.1.1 Underlay and Overlay Networking concepts

The ETSI Network Functions Virtualisation Architectural Framework (as referred above) describes how a Virtualization Layer instance abstract the hardware resources and separate

Virtualisation Tenants (Workload) from each other. It does also specifically state that the control and implementation of the hardware layer is out of scope for that specification.

When having multiple Virtualization Layer instances on a shared hardware infrastructure, the networking can be layered in an Underlay and an Overlay Network layer. The purpose with this layering is to ensure separation of the Virtualisation Tenants (Workload) Overlay Networks from each other, whilst allowing the traffic to flow on the shared Underlay Network in between all Ethernet connected hardware (HW) devices.

The Overlay Networking separation is often done through encapsulation of Tenants traffic using overlay protocols e.g. through VxLAN or EVPN on the Underlay Networks e.g. based on L2 (VLAN) or L3 (IP) networks.

In some instances, the Virtualisation Tenants can bypass the Overlay Networking encapsulation to achieve better performance or network visibility/control. A common method to bypass the Overlay Networking encapsulation normally done by the Virtualisation Layer, is the VNF/CNF usage of SR-IOV that effectively take over the Physical and Virtual Functions of the NIC directly into the VNF/CNF Tenant. In these cases, the Underlay Networking must handle the separation e.g. through a Virtual Termination End Point (VTEP) that encapsulate the Overlay Network traffic.

Note: Bypassing the Overlay Networking layer is a violation of the basic CNTT decoupling principles, but in some cases unavoidable with existing technologies and available standards. Until suitable technologies and standards are developed, CNTT have a set of agreed exemptions that forces the Underlay Networking to handle the bypassed Overlay Networking separation.

VTEP could be manually provisioned in the Underlay Networking or be automated and controlled through a Software Defined Networking controller interfaces into the underlying networking in the HW Infrastructure Layer.

4.3.1.2 Hardware and Virtual Infrastructure Layer concepts

The Cloud Infrastructure (based on ETSI NFV Infrastructure with hardware extensions) can be considered to be composed of two distinct layers, here referred to as HW Infrastructure Layer and Virtual Infrastructure Layer. When there are multiple separated simultaneously deployed Virtual Infrastructure domains, the architecture and deployed implementations must enable each of them to be in individual non-dependent administrative domains. The HW Infrastructure must then also be enabled to be a fully separated administrative domain from all of the Virtualisation domains.

For Cloud Infrastructure implementations of multiple well separated simultaneous Virtual Infrastructure Layer instances on a shared HW Infrastructure there must be a separation of the hardware resources i.e. servers, storage and the Underlay Networking resources that interconnect the hardware resources e.g. through a switching fabric.

To allow multiple separated simultaneous Virtual Infrastructure Layer instances onto a shared switching fabric there is a need to split up the Underlay Networking resources into non overlapping addressing domains on suitable protocols e.g. VxLAN with their VNI Ranges. This separation must be done through an administrative domain that could not be

compromised by any of the individual Virtualisation Infrastructure Layer domains either by malicious or unintentional Underlay Network mapping or configuration.

These concepts are very similar to how the Hyperscaler Cloud Providers (HCP) offer Virtual Private Clouds for users of Bare Metal deployment on the HCP shared pool of servers, storage and networking resources.

The separation of Hardware and Virtual Infrastructure Layers administrative domains makes it important that the Reference Architectures do not include direct management or dependencies of the pooled physical hardware resources in the HW Infrastructure Layer e.g. servers, switches and underlay networks from within the Virtual Infrastructure Layer. All automated interaction from the Virtual Infrastructure Layer implementations towards the HW Infrastructure with its shared networking resources in the HW Infrastructure Layer must go through a common abstracted Reference Model interface.

4.3.1.3 Software Defined Underlay and Overlay Networking concepts

A major point with a Cloud Infrastructures is to automate as much as possible. An important tool for Networking automation is Software Defined Networking (SDN) that comes in many different shapes and can act on multiple layers of the networking. In this section we will deal with the internal networking of a datacentre and not how datacentres interconnect with each other or get access to the world outside of a datacentre.

When there are multiple simultaneously deployed instances of the Virtual Infrastructure Layers on the same HW Infrastructure, there is a need to ensure Underlay networking separation in the HW Infrastructure Layer. This separation can be done manually through provisioning of a statically configured separation of the Underlay Networking in the HW Infrastructure Layer. A better and more agile usage of the HW Infrastructure is to offer each instance of the Virtual Infrastructure Layer a unique instance of a SDN interface into the shared HW Infrastructure. Since these SDN instances only deal with a well separated portion (or slice) of the Underlay Networking we call this interface SDN-Underlay (SDNu).

The HW Infrastructure Layer is responsible for keeping the different Virtual Infrastructure Layer instances separated in the Underlay Networking. This can be done through manual provisioning methods or be automated through a HW Infrastructure Layer orchestration interface. The separation responsibility is also valid between all instances of the SDNu interface since each Virtual Infrastructure Layer instance shall not know about, be disturbed by or have any capability to reach the other Virtual Infrastructure instances.

An SDN-Overlay control interface (here denoted SDNo) is responsible for managing the Virtual Infrastructure Layer virtual switching and/or routing as well as its encapsulation and its mapping onto the Underlay Networks.

In cases where the VNF/CNF bypasses the Virtual Infrastructure Layer virtual switching and its encapsulation, as described above, the HW Infrastructure Layer must perform the encapsulation and mapping onto the Underlay Networking to ensure the Underlay Networking separation. This should be a prioritized capability in the SDNu control interface since CNTT currently allow exemptions for bypassing the virtual switching (e.g. through SR-IOV).

SDNo controllers can request Underlay Networking encapsulation and mapping to be done by signalling to an SDNu controller. There are however today no standardized way for this signalling and by that there is a missing reference point and API description in this architecture.

Multiple instances of Container as a Service (CaaS) Virtualization Layers running on an Infrastructure as a Service (IaaS) Virtual Infrastructure Layer could make use of the IaaS layer to handle the required Underlay Networking separation. In these cases, the IaaS Virtualisation Infrastructure Manager (VIM) could include an SDNu control interface enabling automation.

Note: The Reference Model describes a logical separation of SDNu and SDNo interfaces to clarify the separation of administrative domains where applicable. In real deployment cases an Operator can select to deploy a single SDN controller instance that implements all needed administrative domain separations or have separate SDN controllers for each administrative domain. A common deployment scenario today is to use a single SDN controller handling both Underlay and Overlay Networking which works well in the implementations where there is only one administrative domain that owns both the HW Infrastructure and the single Virtual Infrastructure instance. However a shared Underlay Network that shall ensure separation must be under the control of the shared HW Infrastructure Layer. One consequence of this is that the Reference Architectures must not model collapsed SDNo and SDNu controllers since each SDNo must stay unaware of other deployed implementations in the Virtual Infrastructure Layer running on the same HW Infrastructure.

4.3.1.4 Programmable Networking Fabric concept

The concept of a Programmable Networking Fabric pertains to the ability to have an effective forwarding pipeline (a.k.a. forwarding plane) that can be programmed and/or configured without any risk of disruption to the shared Underlay Networking that is involved with the reprogramming for the specific efficiency increase.

The forwarding plane is distributed by nature and must be possible to implement both in switch elements and on SmartNICs (managed outside the reach of host software), that both can be managed from a logically centralised control plane, residing in the HW Infrastructure Layer.

The logically centralised control plane is the foundation for the authoritative separation between different Virtualisation instances or Bare Metal Network Function applications that are regarded as untrusted both from the shared layers and each other.

Although the control plane is logically centralized, scaling and control latency concerns must allow the actual implementation of the control plane to be distributed when required.

All VNF, CNF and Virtualisation instance acceleration as well as all specific support functionality that is programmable in the forwarding plane must be confined to the well separated sections or stages of any shared Underlay Networking. A practical example could be a Virtualisation instance or VNF/CNF that controls a NIC/SmartNIC where the Underlay

Networking (Switch Fabric) ensures the separation in the same way as it is done for SR-IOV cases today.

The nature of a shared Underlay Network that shall ensure separation and be robust is that all code in the forwarding plane and in the control plane must be under the scrutiny and life cycle management of the HW Infrastructure Layer.

This also imply that programmable forwarding functions in a Programmable Networking Fabric are shared resources and by that will have to get standardised interfaces over time to be useful for multiple VNF/CNF and multi-vendor architectures such as ETSI NFV. Example of such future extensions of shared functionality implemented by a Programmable Networking Fabric could be L3 as a Service, Firewall as a Service and Load Balancing as a Service.

Note: Appliance-like applications that fully own its infrastructure layers (share nothing) could manage and utilize a Programmable Networking Fabric in many ways, but that is not a Cloud Infrastructure implementation and falls outside the use cases for these specifications.

4.3.2 Networking Reference Model

The Cloud Infrastructure Networking Reference Model depicted in Figure 19 is based on the ETSI NFV model enhanced with Container Virtualisation support and a strict separation of the HW Infrastructure and Virtualization Infrastructure Layers in NFVI. It includes all above concepts and enables multiple well separated simultaneous Virtualisation instances and domains allowing a mix of IaaS, CaaS on IaaS and CaaS on Bare Metal on top of a shared HW Infrastructure.

It is up to any deployment of the Cloud Infrastructure to decide what Networking related objects to use, but all Reference Architectures have to be able to map into this model.

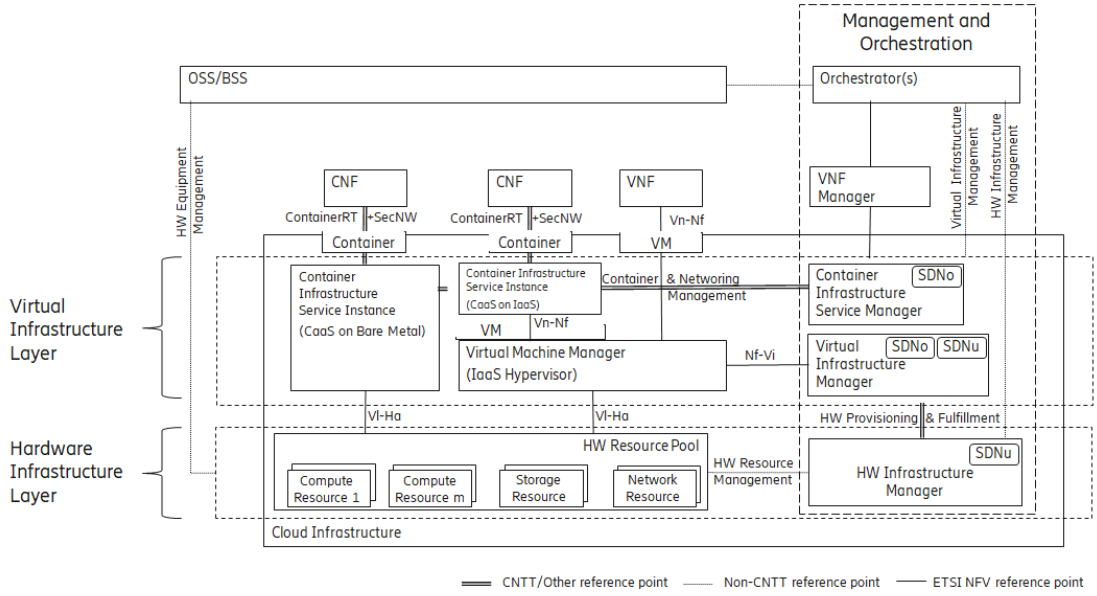

Figure 19: Networking Reference Model based on the ETSI NFV

4.3.3 Deployment examples based on the Networking Reference Model

4.3.3.1 Switch Fabric and SmartNIC examples for Underlay Networking separation

The HW Infrastructure Layer can implement the Underlay Networking separation in any type of packet handling component. This may be deployed in many different ways depending on target use case requirements, workload characteristics and available platforms. Two of the most common ways are: (1) within the physical Switch Fabric and (2) in a SmartNIC connected to the Server CPU being controlled over a management channel that is not reachable from the Server CPU and its host software. In either way the Underlay Networking separation is controlled by the HW Infrastructure Manager.

In both cases the Underlay Networking can be externally controlled over the SDNu interface that must be instantiated with appropriate Underlay Networking separation for each of the Virtualization administrative domains.

Note: The use of SmartNIC in this section is only pertaining to Underlay Networking separation of Virtual instances in separate Overlay domains in much the same way as AWS do with their Nitro SmartNIC. This is the important consideration for the Reference Model that enables multiple implementation instances from one or several Reference Architectures to be used on a shared Underlay Network. The use of SmartNIC components from any specific Virtual instance e.g. for internal virtual switching control and acceleration must be regulated by each Reference Architecture without interfering with the authoritative Underlay separation laid out in the Reference Model.

Two exemplifications of different common HW realisations of Underlay Network separation in the HW Infrastructure Layer can be seen in Figure 20.

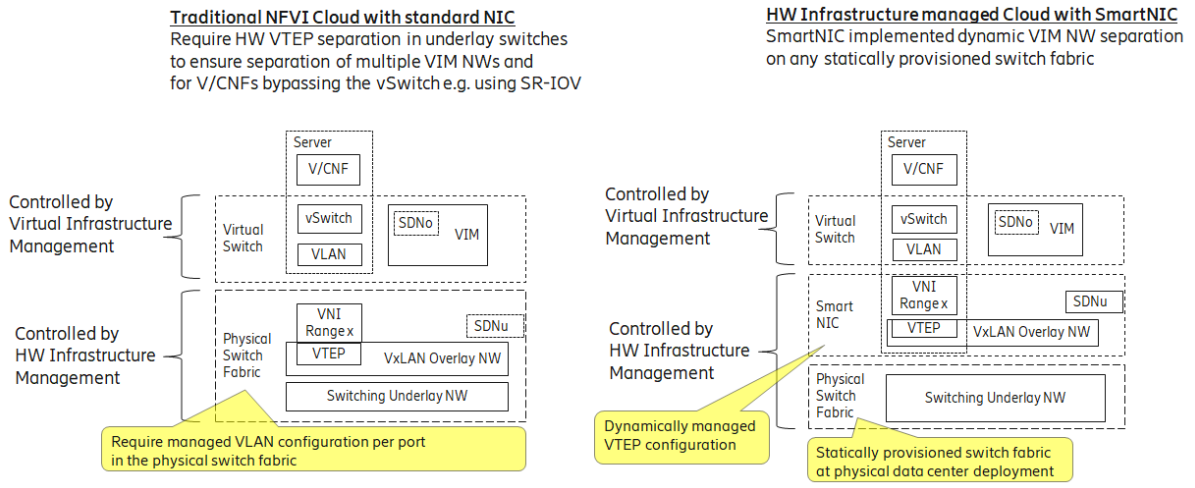

Figure 20: Underlay Networking separation examples

4.3.3.2 SDN Overlay and SDN Underlay layering and relationship example

Two use case examples with both SDNo and SDNu control functions depicting a software based virtual switch instance in the Virtual Infrastructure Layer and another high performance oriented Virtual Infrastructure instance (e.g. enabling SR-IOV) are described in Figure 21. The examples are showing how the encapsulation and mapping could be done in the virtual switch or in a SmartNIC on top of a statically provisioned underlay switching fabric, but another example could also have been depicted with the SDNu controlling the underlay switching fabric without usage of SmartNICs.

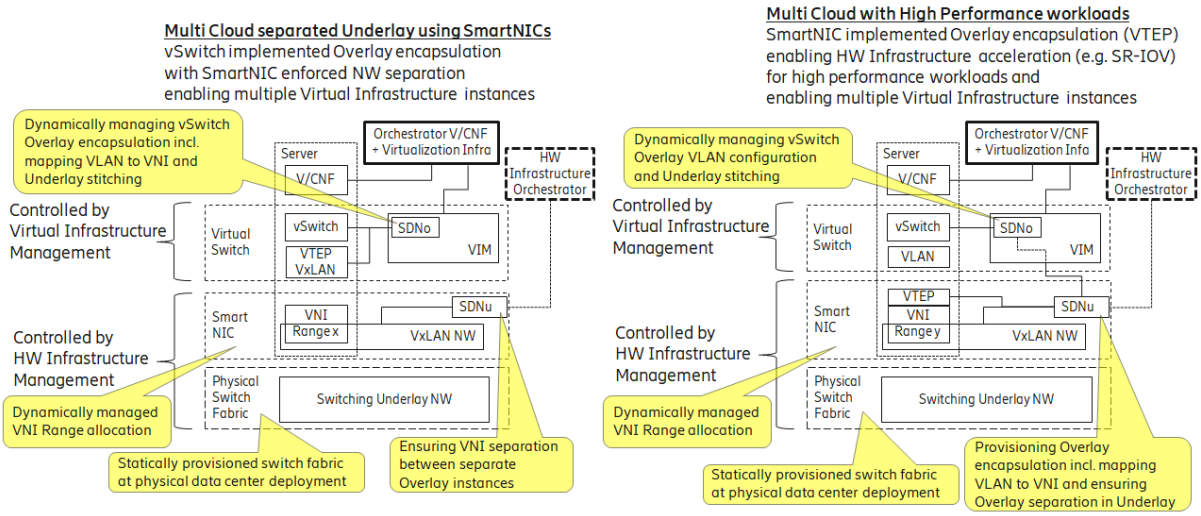

Figure 21: SDN Controller relationship examples

4.3.3.3 Example of IaaS and CaaS Virtualization Infrastructure instances on a shared HW Infrastructure with SDN

A Networking Reference Model deployment example is depicted in Figure 22 to demonstrate the mapping to ETSI NFV reference points with additions of packet flows through the infrastructure layers and some other needed reference points. The example illustrates

individual responsibilities of a complex organization with multiple separated administrative domains represented with separate colours.

The example is or will be a common scenario for operators that modernise their network functions during a rather long period of migration from VNFs to Cloud Native CNFs. Today the network functions are predominantly VNFs on IaaS environments and the operators are gradually moving a selection of these into CNFs on CaaS that either sit on top of the existing IaaS or directly on Bare Metal. It is expected that there will be multiple CaaS instances in most networks, since it is not foreseen any generic standard of a CaaS implementation that will be capable to support all types of CNFs from any vendor. It is also expected that many CNFs will have dependencies to a particular CaaS version or instances which then will prohibit a separation of Life Cycle Management in between individual CNFs and CaaS instances.

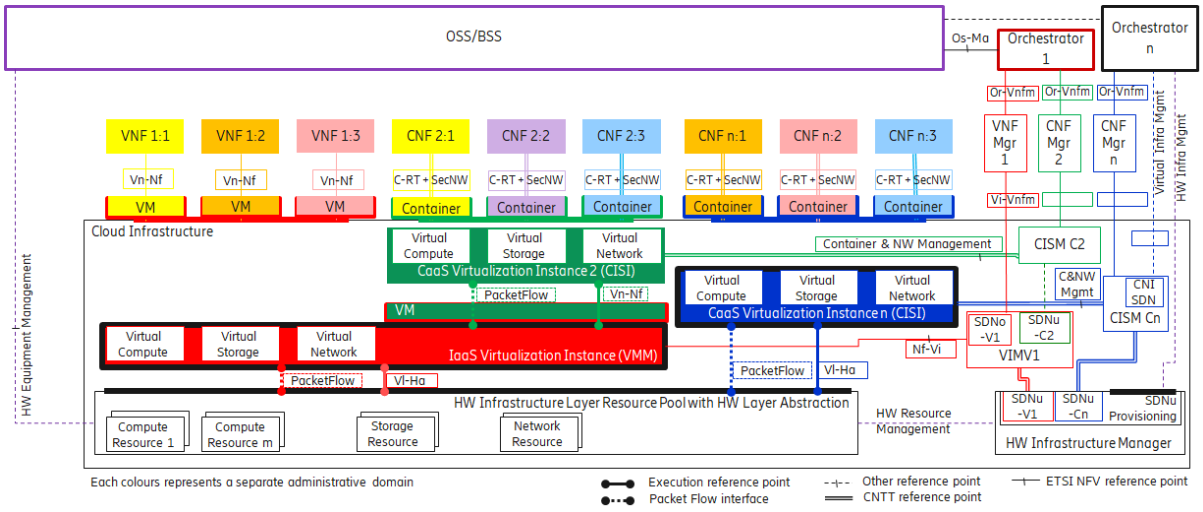

Figure 22: Networking Reference Model deployment example

5 Feature set and Requirements from Infrastructure

5.1 Cloud Infrastructure Software profile description

Cloud Infrastructure Software layer is composed of 2 layers, Figure 23:

- The virtualisation Infrastructure layer, which is based on hypervisor virtualisation technology or container-based virtualisation technology. Container virtualisation can be nested in hypervisor-based virtualisation
- The host OS layer

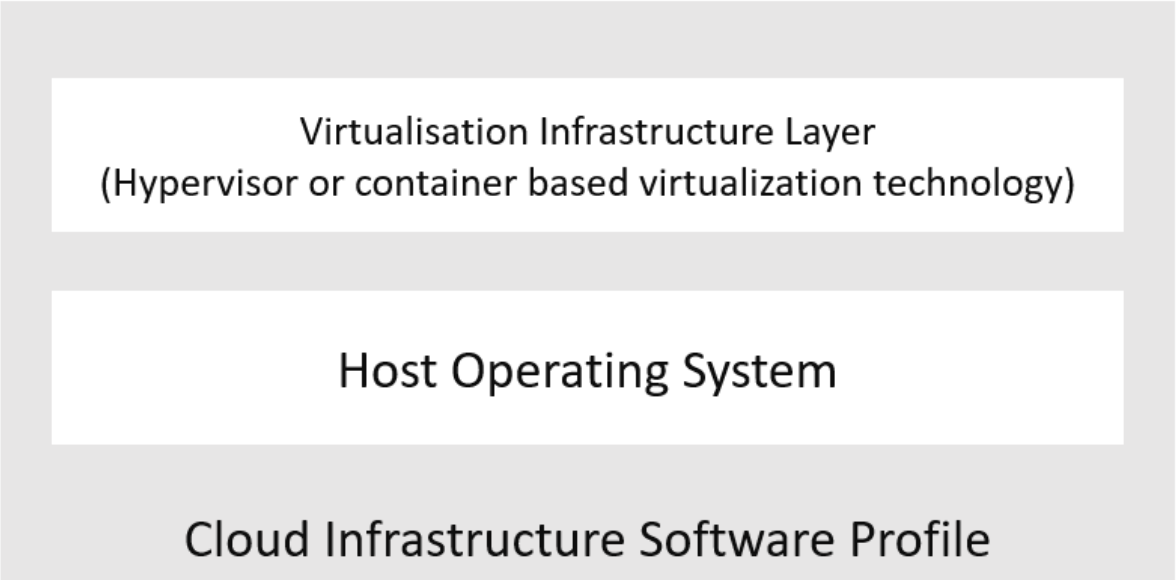

Figure 23: Cloud Infrastructure software layers

For a host (compute node or physical server), the virtualisation layer is an abstraction layer between hardware components (compute, storage, and network resources) and virtual resources allocated to a VM or a Pod. Figure 24 represents the virtual resources (virtual compute, virtual network, and virtual storage) allocated to a VM or a Pod and managed by the Cloud Infrastructure Manager.

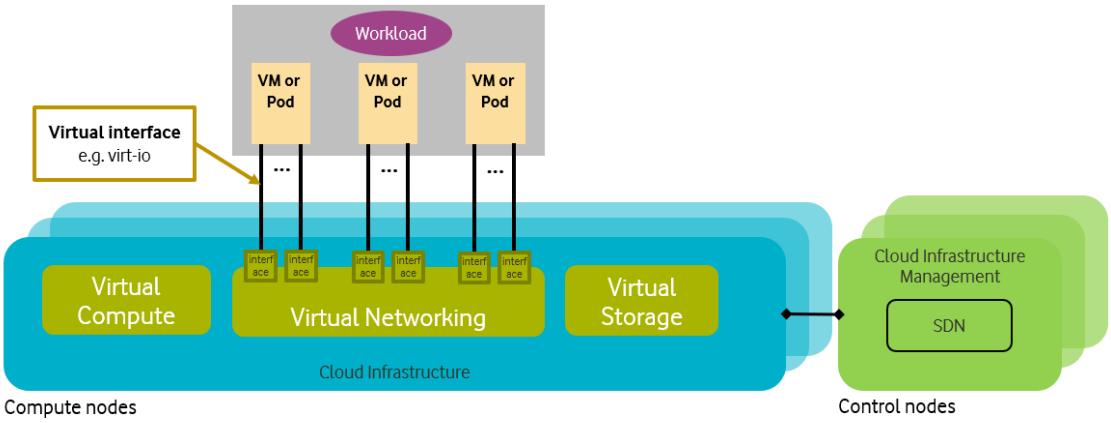

Figure 24: Cloud Infrastructure Virtual resources

Depending on the requirements of the workloads, a VM or a Pod will be deployed with a Cloud Infrastructure Profile and an appropriate compute flavour. A Cloud Infrastructure Profile is defined by a Cloud Infrastructure Software Profile and a Cloud Infrastructure Hardware Profile. A Cloud Infrastructure Software Profile is a set of features, capabilities, and metrics offered by a Cloud Infrastructure software layer. Figure 25 depicts a high level view of the Basic and Network Intensive Cloud Infrastructure Profiles.

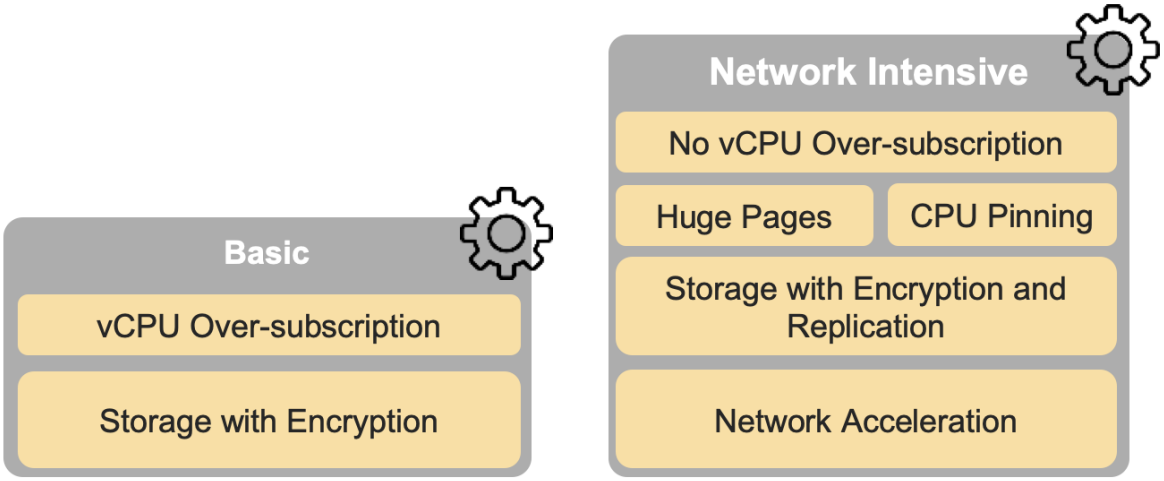

Figure 25: Cloud Infrastructure Profiles

The following sections detail the Cloud Infrastructure Software Profile features per type of virtual resource. The list of these features will evolve over time.

5.1.1 Virtual Compute

Table 22 and Table 23 depict the features related to virtual compute.

Reference	Feature	Type	Description
infra.com.cfg.001	CPU allocation ratio	Value	Number of virtual cores per physical core
infra.com.cfg.002	NUMA alignment	Yes/No	Support of NUMA at the virtualization layer
infra.com.cfg.003	CPU pinning	Yes/No	Binds a process/vCPU to a physical core or SMT thread
infra.com.cfg.004	Huge Pages	Yes/No	Ability to manage huge pages of memory

Table 22: Virtual Compute features.

Reference	Feature	Type	Description
infra.com.acc.cfg.001	Left for RA specifications		

Table 23: Virtual Compute Acceleration features.

5.1.2 Virtual Storage

Table 24 and Table 25 depict the features related to virtual storage.

Reference	Feature	Type	Description
infra.stg.cfg.001	Catalogue Storage Types	Yes/No	Support of Storage types described in the catalogue
infra.stg.cfg.002	Storage Block	Yes/No	
infra.stg.cfg.003	Storage with replication	Yes/No	
infra.stg.cfg.004	Storage with encryption	Yes/No	

Table 24: Virtual Storage features.

Reference	Feature	Type	Description
infra.stg.acc.cfg.001	Storage IOPS oriented	Yes/No	
infra.stg.acc.cfg.002	Storage capacity oriented	Yes/No	

Table 25: Virtual Storage Acceleration features.

5.1.3 Virtual Networking

Table 26 and Table 27 depict the features related to virtual networking.

Reference	Feature	Type	Description
infra.net.cfg.001	Connection Point interface	IO virtualisation	e.g. virtio1.1
infra.net.cfg.002	Overlay protocol	Protocols	The overlay network encapsulation protocol needs to enable ECMP in the underlay to take advantage of the scale-out features of the network fabric.
infra.net.cfg.003	NAT	Yes/No	Support of Network Address Translation
infra.net.cfg.004	Security Groups	Yes/No	Set of rules managing incoming and outgoing network traffic
infra.net.cfg.005	Service Function Chaining	Yes/No	Support of Service Function Chaining (SFC)
infra.net.cfg.006	Traffic patterns symmetry	Yes/No	Traffic patterns should be optimal, in terms of packet flow. North-south traffic shall not be concentrated in specific elements in the architecture, making those critical choke-points, unless strictly necessary (i.e. when NAT 1:many is required).

Table 26: Virtual Networking features.

Reference	Feature	Type	Description
infra.net.acc.cfg.001	vSwitch optimisation	Yes/No and SW Optimisation	e.g. DPDK.
infra.net.acc.cfg.002	Support of HW offload	Yes/No	e.g. support of SmartNic.
infra.net.acc.cfg.003	Crypto acceleration	Yes/No	
infra.net.acc.cfg.004	Crypto Acceleration Interface	Yes/No	

Table 27: Virtual Networking Acceleration features.

5.1.4 Security

See Chapter 7 Security.

5.1.5 Platform Services

This section details the services that may be made available to workloads by the Cloud Infrastructure.

Reference	Feature	Type	Description
infra.svc.stg.001	Object Storage	Yes/No	Object Storage Service (e.g. S3-compatible)

Table 28: Cloud Infrastructure Platform services.

Minimum requirements	Example
Database as a service	Cassandra
Queue	Rabbit MQ
LB and HA Proxy	

Table 29:Service examples

5.2 Cloud Infrastructure Software Profiles features and requirements

This section will detail Cloud Infrastructure Software Profiles and associated configurations for the 2 types of Cloud Infrastructure Profiles: Basic and Network intensive.

5.2.1 Virtual Compute

Table 30 depicts the features and configurations related to virtual compute for the 2 types of Cloud Infrastructure Profiles.

Reference	Feature	Type	Basic	Network Intensive	Notes
infra.com.cfg.001	CPU allocation ratio	value	1:1	1:1	This is set to 1:1 for the Basic profile to enable predictable and consistent performance during benchmarking and certification. Operators may choose to modify this for actual deployments if they are willing to accept the risk of performance impact to workloads using the basic profile.
infra.com.cfg.002	NUMA alignment	Yes/No	N	Y	
infra.com.cfg.003	CPU pinning	Yes/No	N	Y	
infra.com.cfg.004	Huge Pages	Yes/No	N	Y	

Table 30: Virtual Compute features and configuration for the 2 types of Cloud Infrastructure Profiles.

Note: Capability nfvi.com.cfg.001 is set to 1:1 for the Basic profile to enable predictable and consistent performance during benchmarking, certification, and deployment. Operators may choose to modify this for actual

deployments if they are willing to accept the risk of performance impact to these workloads.

Table 31 will gather virtual compute acceleration features. It will be filled over time.

Reference	Feature	Type	Basic	Network Intensive
infra.com.acc.cfg.001	***Note:*** *for further study*			

Table 31: Virtual Compute Acceleration features.

5.2.2 Virtual Storage

Table 32 and Table 33 depict the features and configurations related to virtual storage for the 2 types of Cloud Infrastructure Profiles.

Reference	Feature	Type	Basic	Network Intensive
infra.stg.cfg.001	Catalogue storage Types	Yes/No	Y	Y
infra.stg.cfg.002	Storage Block	Yes/No	Y	Y
infra.stg.cfg.003	Storage with replication	Yes/No	N	Y
infra.stg.cfg.004	Storage with encryption	Yes/No	Y	Y

Table 32: Virtual Storage features and configuration for the 2 types of SW profiles.

Table 33 depicts the features related to Virtual storage Acceleration

Reference	Feature	Type	Basic	Network Intensive
infra.stg.acc.cfg.001	Storage IOPS oriented	Yes/No	N	Y
infra.stg.acc.cfg.002	Storage capacity oriented	Yes/No	N	N

Table 33: Virtual Storage Acceleration features.

5.2.3 Virtual Networking

Table 34 and Table 35 depict the features and configurations related to virtual networking for the 2 types of Cloud Infrastructure Profiles.

Reference	Feature	Type	Basic	Network Intensive
infra.net.cfg.001	Connection Point interface	IO virtualisation	virtio1.1	virtio1.1*
infra.net.cfg.002	Overlay protocol	Protocols	VXLAN, MPLSoUDP, GENEVE, other	
infra.net.cfg.003	NAT	Yes/No	Y	Y
infra.net.cfg.004	Security Group	Yes/No	Y	Y
infra.net.cfg.005	Service Function Chaining	Yes/No	N	Y
infra.net.cfg.006	Traffic patterns symmetry	Yes/No	Y	Y

Table 34: Virtual Networking features and configuration for the 2 types of SW profiles.

Note: * might have other interfaces (such as SR-IOV VFs to be directly passed to a VM or a Pod) or NIC-specific drivers on guest machines transiently allowed until mature enough solutions are available with a similar efficiency level (for example regarding CPU and energy consumption).

Reference	Feature	Type	Basic	Network Intensive
infra.net.acc.cfg.001	vSwitch optimisation	Yes/No and SW Optimisation	N	Y, DPDK
infra.net.acc.cfg.002	Support of HW offload	Yes/No	N	Y, support of SmartNic
infra.net.acc.cfg.003	Crypto acceleration	Yes/No	N	Y
infra.net.acc.cfg.004	Crypto Acceleration Interface	Yes/No	N	Y

Table 35: Virtual Networking Acceleration features.

5.3 Cloud Infrastructure Hardware Profile description

The support of a variety of different workload types, each with different (sometimes conflicting) compute, storage, and network characteristics, including accelerations and optimizations, drives the need to aggregate these characteristics as a hardware (host) profile and capabilities. A host profile is essentially a “personality” assigned to a compute host (physical server, also known as compute host, host, node, or pServer). The host profiles and related capabilities consist of the intrinsic compute host capabilities (such as #CPUs (sockets), # of cores/CPU, RAM, local disks and their capacity, etc.), and capabilities enabled in hardware/BIOS, specialised hardware (such as accelerators), the underlay networking, and storage.

This chapter defines a simplified host, host profile and related capabilities model associated with each of the different Cloud Infrastructure Hardware Profile and related capabilities; some of these profiles and capability parameters are shown in Figure 26.

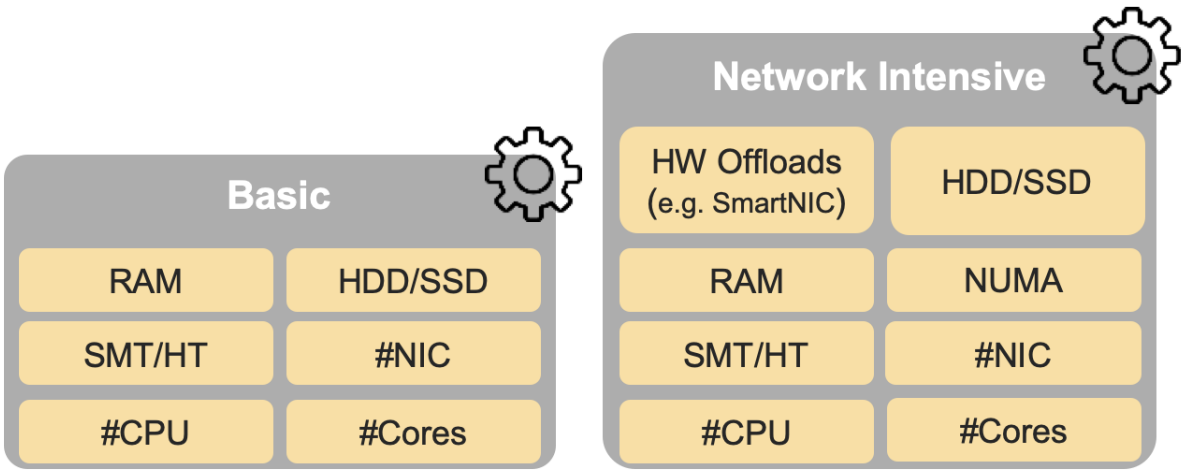

Figure 26: Cloud Infrastructure Hardware Profiles and host associated capabilities.

The host profile model and configuration parameters (hereafter for simplicity simply "host profile") will be used in the **Reference Architecture** to define different hardware profiles. The host profiles can be considered to be the set of EPA-related (Enhanced Performance Awareness) configurations on Cloud Infrastructure resources.

Note: In this chapter we shall not list all of the EPA-related configuration parameters.

A software profile (see **Chapter 4**, **5.1 and 5.2**) defines the characteristics of Cloud Infrastructure SW of which Virtual Machines or Containers will be deployed on. A many to many relationship exists between software profiles and host profiles. A given host can only be assigned a single host profile; a host profile can be assigned to multiple hosts. Different Cloud Service Providers (CSP) may use different naming standards for their host profiles.

The following naming convention is used in this document:

`<host profile name>:: <"hp"><numeral host profile sequence #>`
When a software profile is associated with a host profile, a qualified name can be used as specified below.

`<qualified host profile>:: <software profile><"-"><"hp"><numeral host profile sequence #>`
For Example: *for software profile "n" (network intensive) the above host profile name would be "n-hp1".*

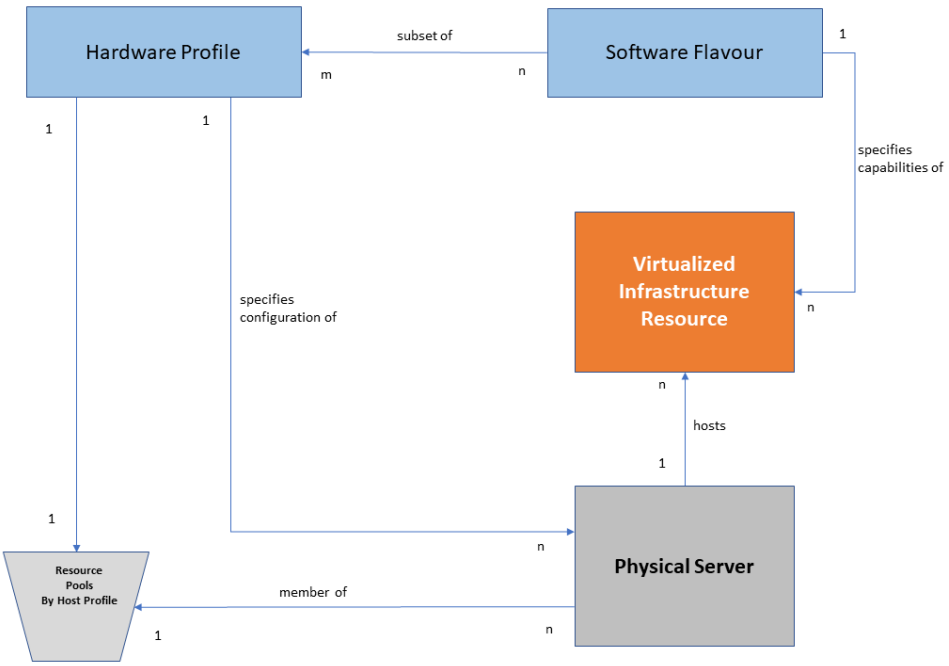

Figure 27: Generic Hardware Profile, Software Flavour, Physical server relationship.

Figure 27 shows a simplistic depiction of the relationship between Hardware profile, Software Profile, Physical server, and virtual compute. In the diagram the resource pool, a logical construct, depicts all physical hosts that have been configured as per a given host profile; there is one resource pool for each hardware profile.

Note: resource pools are not OpenStack host aggregates.

The host profile and capabilities include:

1. **# of CPUs (sockets)**: is the #of CPUs installed on the physical server.
2. **# of cores/CPU**: is the number of cores on each of the CPUs of the physical server.
3. **RAM (GB)**: is the amount of RAM installed on the physical server.
4. **Local Disk Capacity**: is the # of local disks and the capacity of the disks installed on the physical server.
5. **SMT (Simultaneous Multithreading)**: Enabled on all physical servers. Gets multiple threads per physical core. Always ON. Configured in the host.
6. **NUMA (Non-Uniform Memory Access)**: Indicates that vCPU will be on a Socket that is aligned with the associated NIC card and memory. Important for performance optimized workloads. Configured in the host.
7. **SR-IOV (Single-Root Input/Output Virtualisation)**: Configure PCIe ports to enable SR-IOV.
8. **smartNIC (aka Intelligent Server Adaptors)**: Accelerated virtual switch using smartNIC
9. **Cryptography Accelerators**: such as AES-NI, SIMD/AVX and QAT.
10. **Security features**: such as Trusted Platform Module (TPM).

The following model, Figure 28, depicts the essential characteristics of a host that are of interest in specifying a host profile. The host (physical server) is composed of compute, network, and storage resources. The compute resources are composed of physical CPUs (aka CPU sockets or sockets) and memory (RAM). The network resources and storage resources are similarly modelled.

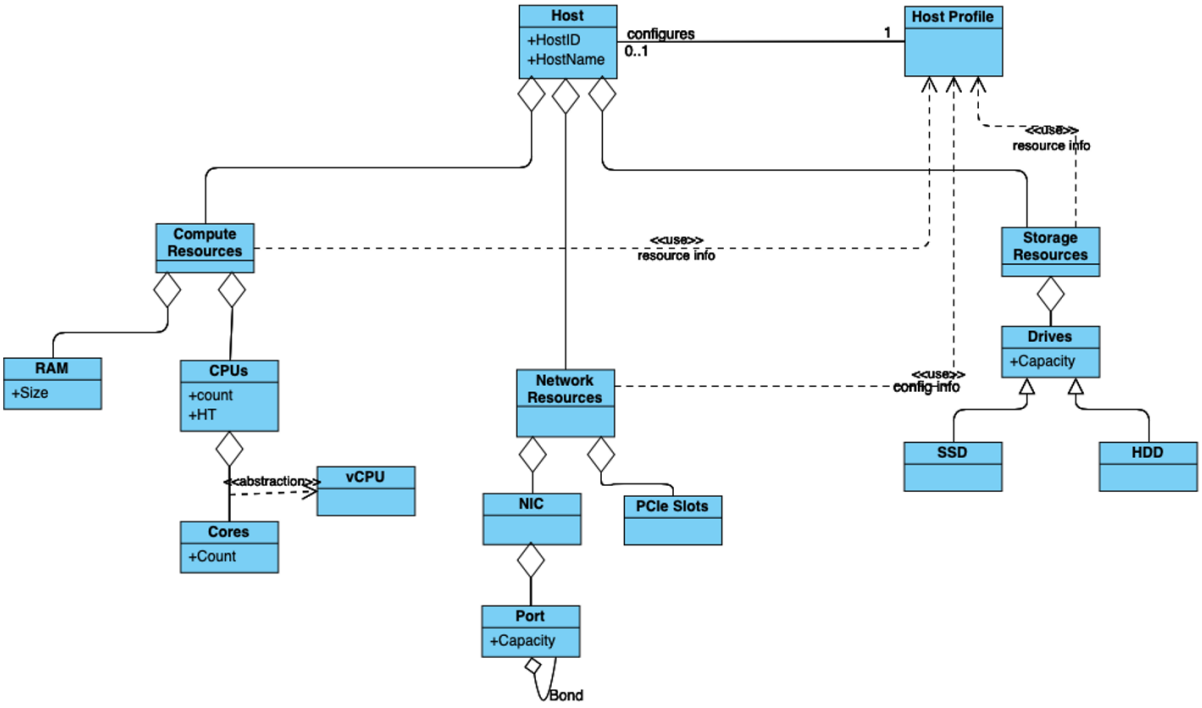

Figure 28: Generic model of a compute host for use in Host Profile configurations.

The hardware (host) profile properties are specified in the following sub-sections. The following diagram (Figure 29) pictorially represents a high-level abstraction of a physical server (host).

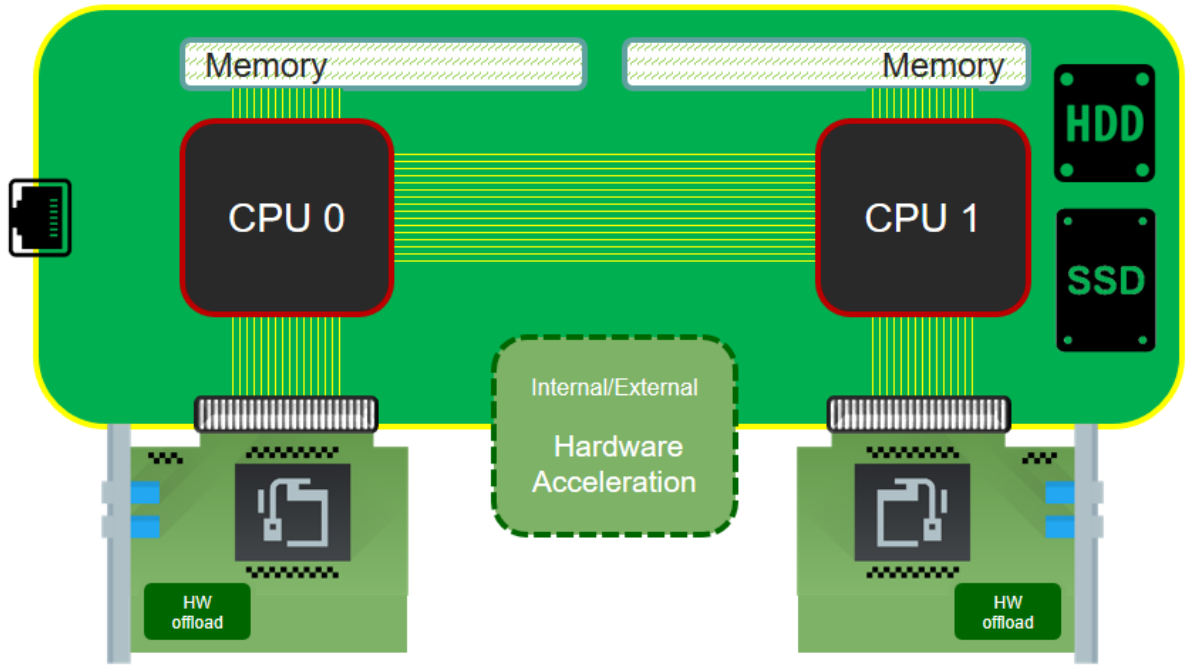

Figure 29: Generic model of a compute host for use in Host Profile configurations.

5.4 Cloud Infrastructure Hardware Profiles features and requirements.

The configurations specified in here will be used in specifying the actual hardware profile configurations for each of the Cloud Infrastructure Hardware Profiles depicted in Figure 26.

5.4.1 Compute Resources

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.cpu.cfg.001	Minimum Number of CPU sockets	This determines the minimum number of CPU sockets within each host	2	2
infra.hw.cpu.cfg.002	Minimum Number of cores per CPU	This determines the number of cores needed per CPU.	20	20
infra.hw.cpu.cfg.003	NUMA alignment	NUMA alignment support and BIOS configured to enable NUMA	N	Y
infra.hw.cpu.cfg.004	Simultaneous Multithreading (SMT)	This allows a CPU to work multiple streams of data simultaneously	Y	Y

Table 36: Minimum Compute resources configuration parameters.

5.4.1.1 Compute Acceleration Hardware Specifications

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.cac.cfg.001	GPU	GPU	N	N

Table 37: Compute acceleration configuration specifications.

5.4.2 Storage Configurations

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.stg.hdd.cfg.001*	Local Storage HDD	Hard Disk Drive		
infra.hw.stg.ssd.cfg.002*	Local Storage SSD	Solid State Drive	Recommended	Recommended

Table 38: Storage configuration specification.

Note: * This specified local storage configurations including # and capacity of storage drives.

5.4.3 Network Resources

5.4.3.1 NIC configurations

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.nic.cfg.001	NIC Ports	Total Number of NIC Ports available in the host	4	4
infra.hw.nic.cfg.002	Port Speed	Port speed specified in Gbps (minimum values)	10	25

Table 39: Minimum NIC configuration specification.

5.4.3.2 PCIe Configurations

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.pci.cfg.001	PCIe slots	Number of PCIe slots available in the host	8	8
infra.hw.pci.cfg.002	PCIe speed		Gen 3	Gen 3
infra.hw.pci.cfg.003	PCIe Lanes		8	8

Table 40: PCIe configuration specification.

5.4.3.3 Network Acceleration Configurations

Reference	Feature	Description	Basic Type	Network Intensive
infra.hw.nac.cfg.001	Crypto	IPSec, Crypto	N	Optional

	Acceleration			
infra.hw.nac.cfg.002	SmartNIC	A SmartNIC that is used to offload network functionality to hardware	N	Optional
infra.hw.nac.cfg.003	Compression			Optional

Table 41: Network acceleration configuration specification.

6 External Interfaces

6.1 Introduction

In this document's earlier chapters, the various resources and capabilities of the Cloud Infrastructure have been catalogued and the workloads have been profiled with respect to those capabilities. The intent behind this chapter and an "API Layer" is to similarly provide a single place to catalogue and thereby codify, a common set of open APIs to access (i.e. request, consume, control, etc.) the aforementioned resources, be them directly exposed to the workloads, or purely internal to the Cloud Infrastructure.

It is a further intent of this chapter and this document to ensure the APIs adopted for CNTT Cloud Infrastructure implementations are open and not proprietary, in support of compatibility, component substitution, and ability to realize maximum value from existing and future test heads and harnesses.

While it is the intent of this chapter to catalogue the APIs, it is not the intent of this chapter to reprint the APIs, as this would make maintenance of the chapter impractical and the length of the chapter disproportionate within the Reference Model document. Instead, the APIs selected for CNTT Cloud Infrastructure implementations and specified in this chapter, will be incorporated by reference and URLs for the latest, authoritative versions of the APIs, provided in the References section of this document.

Although the document does not attempt to reprint the APIs themselves, where appropriate and generally where the mapping of resources and capabilities within the Cloud Infrastructure to objects in APIs would be otherwise ambiguous, this chapter shall provide explicit identification and mapping.

In addition to the raw or base-level Cloud Infrastructure functionality to API and object mapping, it is further the intent to specify an explicit, normalized set of APIs and mappings to control the logical interconnections and relationships between these objects, notably, but not limited to, support of SFC (Service Function Chaining) and other networking and network management functionality.

This chapter specifies the abstract interfaces (API, CLI, etc.) supported by the Cloud Infrastructure Reference Model. The purpose of this chapter is to define and catalogue a common set of open (not proprietary) APIs, of the following types:

- Cloud Infrastructure APIs: These APIs are provided to the workloads (i.e. exposed), by the infrastructure in order for workloads to access (i.e. request, consume, control, etc.) Cloud Infrastructure resources.

- Intra-Cloud Infrastructure APIs: These APIs are provided and consumed directly by the infrastructure. These APIs are purely internal to the Cloud Infrastructure and are not exposed to the workloads.
- Enabler Services APIs: These APIs are provided by non-Cloud Infrastructure services and provide capabilities that are required for a majority of workloads, e.g. DHCP, DNS, NTP, DBaaS, etc.

6.2 Cloud Infrastructure APIs

The Cloud Infrastructure APIs consist of set of APIs that are externally and internally visible. The externally visible APIs are made available for orchestration and management of the execution environments that host workloads while the internally visible APIs support actions on the hypervisor and the physical resources. The ETSI NFV Reference MANO Architecture (Figure 30) shows a number of Interface points where specific or sets of APIs are supported. For the scope of the reference model the relevant interface points are shown in Table 42.

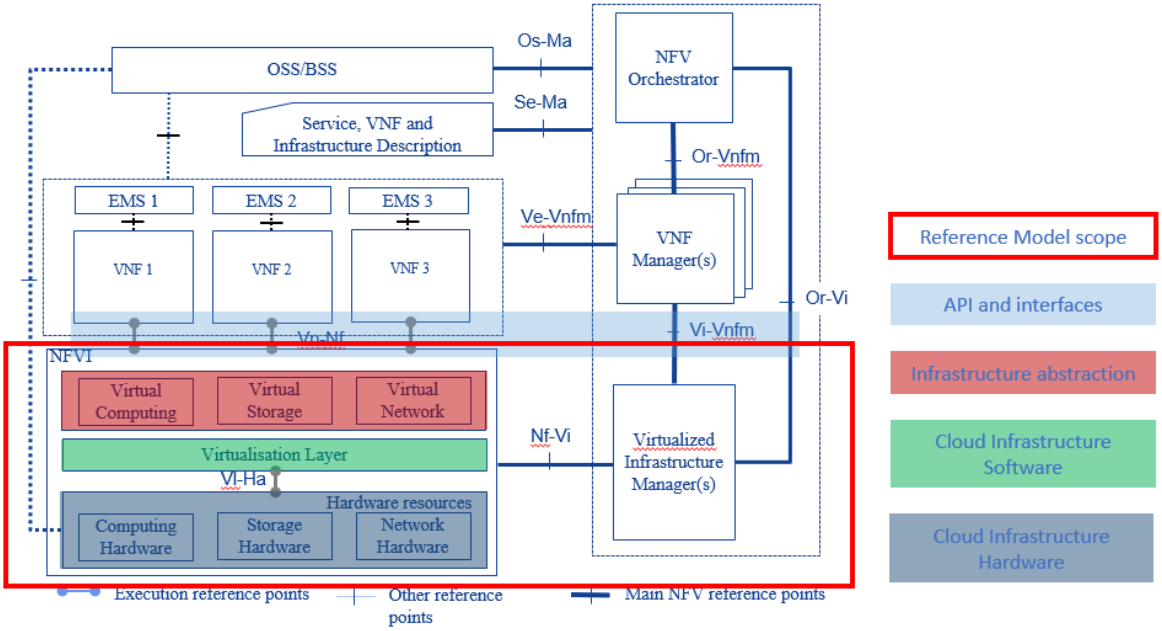

Figure 30: ETSI NFV architecture mapping

Interface Point	Cloud Infrastructure Exposure	Interface Between	Description
Vi-Ha	Internal NFVI	Software Layer and Hardware Resources	1. Discover/collect resources and their configuration information 2. Create execution environment (e.g., VM) for workloads (VNF)
Vn-Nf	External	NFVI and VM (VNF)	Here VNF represents the execution environment. The interface is used to specify interactions between the VNF and abstract NFVI accelerators. The interfaces can be used to discover, configure, and manage these accelerators and for the VNF to

			register/deregister for receiving accelerator events and data.
NF-Vi	External	NFVI and VIM	1. Discover/collect physical/virtual resources and their configuration information2. Manage (create, resize, (un) suspend, reboot, etc.) physical/virtualised resources3. Physical/Virtual resources configuration changes4. Physical/Virtual resource configuration.
Or-Vi	External	VNF Orchestrator and VIM	See below
Vi-Vnfm	External	VNF Manager and VIM	See below

Table 42: NFVI and VIM Interfaces with Other System Components in the ETSI NFV architecture

The Or-Vi and Vi-Vnfm are both specifying interfaces provided by the VIM and therefore are related. The Or-Vi reference point is used for exchanges between NFV Orchestrator and VIM, and supports the following interfaces; virtualised resources refers to virtualised compute, storage, and network resources:

- Software Image Management
- Virtualised Resources Information Management
- Virtualised Resources Capacity Management (only VNF Orchestrator and VIM (Or-Vi))
- Virtualised Resources Management
- Virtualised Resources Change Management
- Virtualised Resources Reservation Management
- Virtualised Resources Quota Management
- Virtualised Resources Performance Management
- Virtualised Resources Fault Management
- Policy Management
- Network Forwarding Path (NFP) Management (only VNF Orchestrator and VIM (Or-Vi))

6.2.1 Tenant Level APIs

In the abstraction model of the Cloud Infrastructure (**Chapter 3**) a conceptual model of a Tenant (Figure 8) represents the slice of a cloud zone dedicated to a workload. This slice, the Tenant, is composed of virtual resources being utilized by workloads within that Tenant. The Tenant has an assigned quota of virtual resources, a set of users can perform operations as per their assigned roles, and the Tenant exists within a Cloud Zone. The APIs will specify the allowed operations on the Tenant including its component virtual resources and the different APIs can only be executed by users with the appropriate roles. For example, a Tenant may only be allowed to be created and deleted by Cloud Zone administrators while virtual compute resources could be allowed to be created and deleted by Tenant administrators.

For a workload to be created in a Tenant also requires APIs for the management (creation, deletion, and operation) of the Tenant, software flavours (Chapter 5), Operating System and workload images (“Images”), Identity and Authorization (“Identity”), virtual resources, security, and the workload application (“stack”).

A virtual compute resource is created as per the flavour template (specifies the compute, memory, and local storage capacity) and is launched using an image with access and security credentials; once launched, it is referred to as a virtual compute instance or just “Instance”). Instances can be launched by specifying the compute, memory, and local storage capacity parameters instead of an existing flavour; reference to flavours covers the situation where the capacity parameters are specified. IP addresses and storage volumes can be attached to a running Instance.

Resource	Create	List	Attach	Detach	Delete	Notes
Flavour	+	+			+	
Image	+	+			+	Create/delete by appropriate administrators
Key pairs	+	+			+	
Privileges						Created and managed by Cloud Service Provider(CSP) administrators
Role	+	+			+	Create/delete by authorized administrators where roles are assigned privileges and mapped to users in scope
Security Groups	+	+			+	Create and delete only by VDC administrators
Stack	+	+			+	Create/delete by VDC users with appropriate role
Virtual Storage	+	+	+	+	+	Create/delete by VDC users with appropriate role
User	+	+		+	+	Create/delete only by VDC administrators
Tenant	+	+		+	+	Create/delete only by Cloud Zone administrators
Virtual compute	+	+		+	+	Create/delete by VDC users with appropriate role. Additional operations would include suspend/unsuspend
Virtual network	+	+	+	+	+	Create/delete by VDC users with appropriate role

Table 43: API types for a minimal set of resources.

Table 43 specifies a minimal set of operations for a minimal set of resources that are needed to orchestrate workloads. The actual APIs for the listed operations will be specified in the Reference Architectures; each listed operation could have a number of associated APIs with a different set of parameters. For example, create virtual resource using an image or a device.

6.2.2 Hardware Acceleration Interfaces

Acceleration Interface Specifications ETSI GS NFV-IFA 002 [7] defines a technology and implementation independent virtual accelerator, the accelerator interface requirements and specifications that would allow a workload to leverage a Virtual Accelerator. The virtual accelerator is modelled on extensible para-virtualised devices (EDP). ETSI GS NFV-IFA 002 [7] specifies the architectural model in Chapter 4 and the abstract interfaces for management, configuration, monitoring, and Data exchange in Chapter 7.

ETSI NFV-IFA 019 3.1.1 [8] has defined a set of technology independent interfaces for acceleration resource life cycle management. These operations allow: allocation, release, and querying of acceleration resource, get and reset statistics, subscribe/unsubscribe (terminate) to fault notifications, notify (only used by NFVI), and get alarm information.

These acceleration interfaces are summarized here in Table 44 only for convenience.

Request	Response	From, To	Type	Parameter	Description
InitAccRequest	InitAccResponse	VNF → NFVI	Input	accFilter	the accelerator sub-system(s) to initialize and retrieve their capabilities.
			Filter	accAttributeSelector	attribute names of accelerator capabilities
			Output	accCapabilities	acceleration sub-system capabilities
RegisterForAccEventRequest	RegisterForAccEventResponse	VNF → NFVI	Input	accEvent	event the VNF is interested in
			Input	vnfEventHandlerId	the handler for NFVI to use when notifying the VNF of the event
AccEventNotificationRequest	AccEventNotificationResponse	NFVI → VNF	Input	vnfEventHandlerId	Handler used by VNF registering for this event
			Input	accEventMetaData	
DeRegisterForAccEventRequest	DeRegisterForAccEventResponse	VNF → NFVI	Input	accEvent	Event VNF is deregistering from
ReleaseAccRequest	ReleaseAccResponse	VNF → NFVI			
ModifyAccConfigurationRequest	ModifyAccConfigurationResponse	VNF → NFVI	Input	accConfigurationData	Config data for accelerator
			Input	accSubSysConfigurationData	Config data for accelerator sub-system
GetAccConfigsRequest	GetAccConfigsResponse	VNF → NFVI	Input	accFilter	Filter for subsystems from which config data requested
			Input	accConfigSelector	attributes of config types
			Output	accComfigs	Config info (only for the specified attributes) for specified subsystems
ResetAccConfigsRequest	ResetAccConfigsResponse	VNF → NFVI	Input	accFilter	Filter for subsystems for which config is to be reset

Request	Response	From, To	Type	Parameter	Description
			Input	accConfigSelector	attributes of config types whose values will be reset
AccDataRequest	AccDataResponse	VNF → NFVI	Input	accData	Data (metadata) sent too accelerator
			Input	accChannel	Channel data is to be sent to
			Output	accData	Data from accelerator
AccSendDataRequest	AccSendDataResponse	VNF → NFVI	Input	accData	Data (metadata) sent too accelerator
			Input	accChannel	Channel data is to be sent to
AccReceiveDataRequest	AccReceiveDataResponse	VNF → NFVI	Input	maxNumberOfDataItems	Max number of data items to be received
			Input	accChannel	Channel data is requested from
			Output	accData	Data received form Accelerator
RegisterForAccDataAvailableEventRequest	RegisterForAccDataAvailableEventResponse	VNF → NFVI	Input	regHandlerId	Registration Identifier
			Input	accChannel	Channel where event is requested for
AccDataAvailableEventNotificationRequest	AccDataAvailableEventNotificationResponse	NFVI → VNF	Input	regHandlerId	Reference used by VNF when registering for the event
DeRegisterForAccDataAvailableEventRequest	DeRegisterForAccDataAvailableEventResponse	VNF → NFVI	Input	accChannel	Channel related to the event
AllocateAccResourceRequest	AllocateAccResourceResponse	VIM → NFVI	Input	attachTargetInfo	the resource the accelerator is to be attached to (e.g., VM)
			Input	accResourceInfo	Accelerator Information
			Output	accResourceId	Id if successful

Request	Response	From, To	Type	Parameter	Description
ReleaseAccResourceRequest	ReleaseAccResourceResponse	VIM → NFVI	Input	accResourceId	Id of resource to be released
QueryAccResourceRequest	QueryAccResourceResponse	VIM → NFVI	Input	hostId	Id of specified host
			Input	Filter	Specifies the accelerators for which query applies
			Output	accQueryResult	Details of the accelerators matching the input filter located in the selected host.
GetAccStatisticsRequest	GetAccStatisticsResponse	VIM → NFVI	Input	accFilter	Accelerator subsystems from which data is requested
			Input	accStatSelector	attributes of AccStatistics whose data will be returned
			Output	accStatistics	Statistics data of the accelerators matching the input filter located in the selected host.
ResetAccStatisticsRequest	ResetAccStatisticsResponse	VIM → NFVI	Input	accFilter	Accelerator subsystems for which data is to be reset
			Input	accStatSelector	attributes of AccStatistics whose data will be reset
SubscribeRequest	SubscribeResponse	VIM → NFVI	Input	hostId	Id of specified host
			Input	Filter	Specifies the accelerators and related alarms. The filter could include accelerator information, severity of the alarm, etc.
			Output	SubscriptionId	Identifier of the successfully created subscription.
UnsubscribeRequest	UnsubscribeResponse	VIM → NFVI	Input	hostId	Id of specified host

Request	Response	From, To	Type	Parameter	Description
			Input	SubscriptionId	Identifier of the subscription to be unsubscribed.
Notify		NFVI → VIM			NFVI notifies an alarm to VIM
GetAlarmInfoRequest	GetAlarmInfoResponse	VIM → NFVI	Input	hostId	Id of specified host
			Input	Filter	Specifies the accelerators and related alarms. The filter could include accelerator information, severity of the alarm, etc.
			Output	Alarm	Information about the alarms if filter matches an alarm.
AccResourcesDiscoveryRequest	AccResourcesDiscoveryResponse	VIM → NFVI	Input	hostId	Id of specified host
			Output	discoveredAccResourceInfo	Information on the acceleration resources discovered within the NFVI.
OnloadAccImageRequest	OnloadAccImageResponse	VIM → NFVI	Input	accResourceId	Identifier of the chosen accelerator in the NFVI.
			Input	accImageInfo	Information about the acceleration image.
			Input	accImage	The binary file of acceleration image.

Table 44: Hardware Acceleration Interfaces in the ETSI NFV architecture

6.3 Intra-Cloud Infrastructure Interfaces

6.3.1 Hypervisor Hardware Interface

Table 42 lists a number of NFVI and VIM interfaces, including the internal VI-Ha interface. The VI-Ha interface allows the hypervisor to control the physical infrastructure; the hypervisor acts under VIM control. The VIM issues all requests and responses using the NF-VI interface; requests and responses include commands, configuration requests, policies, updates, alerts, and response to infrastructure results. The hypervisor also provides information about the health of the physical infrastructure resources to the VM. All these activities, on behalf of the VIM, are performed by the hypervisor using the VI-Ha interface. While no abstract APIs have yet been defined for this internal VI-Ha interface, ETSI GS NFV-INF 004 [9] defines a set of requirements and details of the information that is required by the VIM from the physical infrastructure resources. Hypervisors utilize various programs to get this data including BIOS, IPMI, PCI, I/O Adapters/Drivers, etc.

6.4 Enabler Services Interfaces

An operational cloud needs a set of standard services to function. Services such as NTP for time synchronization, DHCP for IP address allocation, DNS for obtaining IP addresses for domain names, and LBaaS (version 2) to distribute incoming requests amongst a pool of designated resources.

7 Security

7.1 Introduction

Security vulnerabilities and attack vectors are everywhere. The Telecom industry and its cloud infrastructures are even more vulnerable to potential attacks due to the ubiquitous nature of the infrastructures and services combined with the vital role Telecommunications play in the modern world. The attack vectors are many and varied, ranging from the potential for exposure of sensitive data, both personal and corporate, to weaponized disruption to the global telecommunications networks. The threats can take the form of a physical attack on the locations the infrastructure hardware is housed, to network attacks such as denial of service and targeted corruption of the network service applications themselves. Whatever the source, any Cloud Infrastructure built needs to be able to withstand attacks in whatever form they take.

This chapter examines multiple aspects of security as it relates to Cloud Infrastructure and security aspects for workloads. After discussing security attack vectors, this chapter delves into security requirements. Regarding security requirements and best practices, specifications and documents are published by standards organizations. A selection of standards of interest for Cloud Infrastructure security is listed in a dedicated section. The chapter culminates with a consolidated set of “must” requirements and desired (should) recommendations; it is suggested that operators carefully evaluate the recommendations for possible implementation.

7.2 Potential attack vectors

Previously attacks designed to place and migrate workload outside the legal boundaries were not possible using traditional infrastructure, due to the closed nature of these systems. However, using Cloud Infrastructure, violation of regulatory policies and laws becomes possible by actors diverting or moving an application from an authenticated and legal location to another potentially illegal location. The consequences of violating regulatory policies may take the form of a complete banning of service and/or an exertion of a financial penalty by a governmental agency or through SLA enforcement. Such vectors of attack may well be the original intention of the attacker in an effort to harm the service provider. One possible attack scenario can be when an attacker exploits the insecure VNF API to dump the records of personal data from the database in an attempt to violate user privacy. Cloud Infrastructure operators should ensure that the applications APIs are secure, accessible over a secure network (TLS) under very strict set of security best practices, and RBAC policies to limit exposure of this vulnerability.

7.3 Security Scope

7.3.1 In-scope and Out-of-Scope definition

The scope of the security controls requirements maps to the scope of the Reference Model architecture.

Cloud Infrastructure requirements must cover the virtual infrastructure layer and the hardware infrastructure layer, including virtual resources, hardware resources, virtual infrastructure manager and hardware infrastructure manager, as described in Chapter 3.

7.3.2 High level security Requirements

The following diagram shows the different security domains that impact the Reference Model:

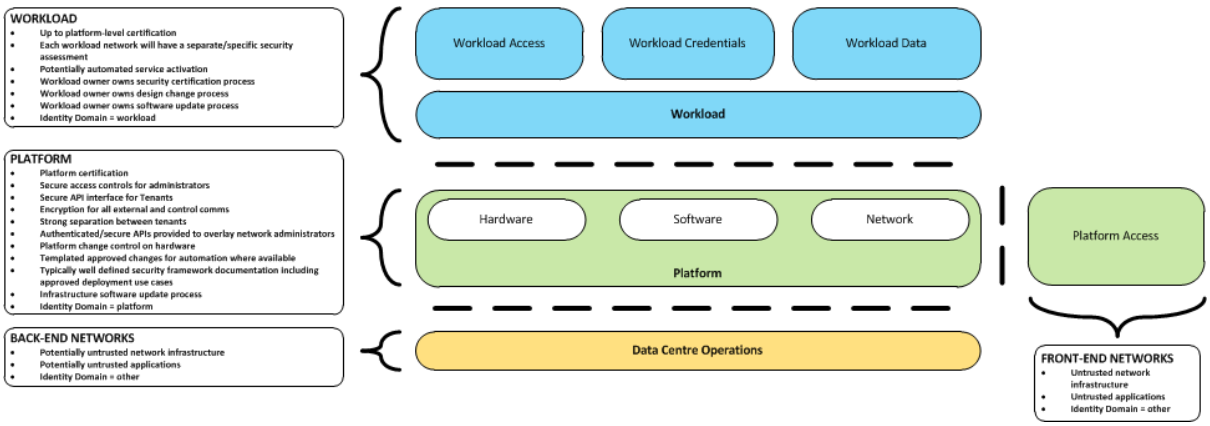

Figure 31: Reference Model Security Domains

Note: "Platform" refers to the Cloud Infrastructure with all its hardware and software components.

7.3.2.1 Platform security requirements

At a high level, the following areas/requirements cover platform security for a particular deployment:

- Platform certification
- Secure access controls for administrators
- Secure API interface for tenants
- Encryption for all external and control communications
- Strong separation between tenants - ensuring network, data, memory and runtime process (CPU running core) isolation between tenants
- Authenticated/secure APIs provided to overlay network administrators
- Platform change control on hardware
- Templated approved changes for automation where available
- Typically well-defined security framework documentation including approved deployment use cases
- Infrastructure software update process

7.3.2.2 Workload security requirements

At a high level, the following areas/requirements cover workload security for a particular deployment:

- Up to platform-level certification
- Each workload network will need to undertake its own security self-assessment and accreditation, and not inherit a security accreditation from the platform
- Potentially automated service activation
- Workload owner owns workload security certification process
- Workload owner owns workload design change process
- Workload owner owns workload software update process

7.4 Cloud Infrastructure Security

7.4.1 General Platform Security

The security certification of the platform will typically need to be the same, or higher, than the workload or VNF requirements.

The platform supports the workload, and in effect controls access to the workload from and to external endpoints such as carriage networks used by workloads, or by Data Centre Operations staff supporting the workload, or by tenants accessing workloads. From an access security perspective, the following diagram shows where different access controls will operate within the platform to provide access controls throughout the platform:

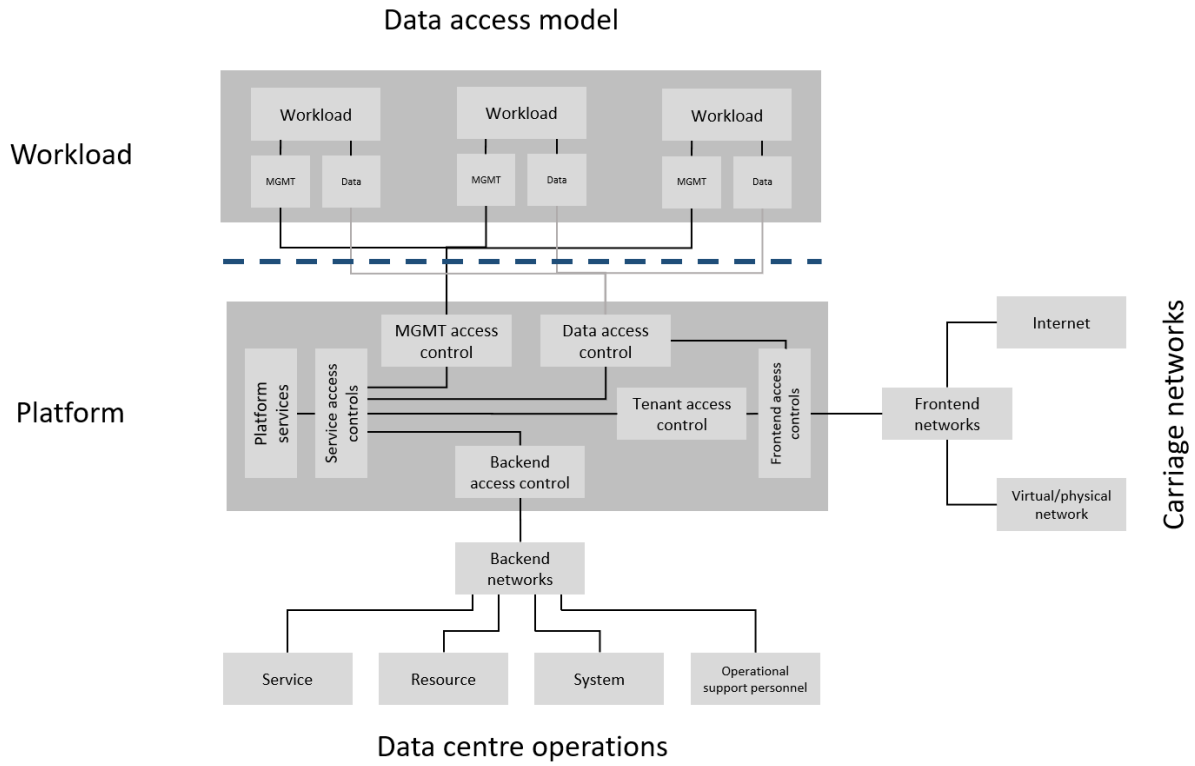

Figure 32: Reference Model Access Controls

7.4.1.1 The high-level functions of these different access controls

- **MGNT ACCESS CONTROLS** - Platform access to workloads for service management. Typically all management and control-plane traffic is encrypted.
- **DATA ACCESS CONTROLS** - Control of east-west traffic between workloads, and control of north-south traffic between the VNF and other platform services such as front-end carriage networks and platform services. Inherently strong separation between tenants is mandatory.
- **SERVICES ACCESS CONTROLS** - Protects platform services from any platform access
- **BACK-END ACCESS CONTROLS** - Data Centre Operations access to the platform, and subsequently, workloads. Typically stronger authentication requirements such as (Two-Factor Authentication) 2FA, and using technologies such as Role-Based Access Control (RBAC) and encryption. Application Programming Interface (API) gateways may be required for automated/script-driven processes.
- **FRONT-END ACCESS CONTROLS** - Protects the platform from malicious carriage network access, and provides connectivity for specific workloads to specific carriage networks. Carriage networks being those that are provided as public networks and operated by carriers, and in this case with interfaces that are usually sub, or virtual networks.
- **TENANT ACCESS CONTROLS** - Provides appropriate tenant access controls to specific platform services, and tenant workloads - including Role-Based Access Control (RBAC), authentication controls as appropriate for the access arrangement, and Application Programming Interface (API) gateways for automated/script-driven processes.

7.4.1.2 The following general security requirements apply to the Cloud Infrastructure

System Hardening

- Adhering to the principle of least privilege, no login to root on any platform systems (platform systems are those that are associated with the platform and include systems that directly or indirectly affect the viability of the platform) when root privileges are not required.
- Ensure that all the platform's components (including hypervisors, VMs, etc.) are kept up to date with the latest patch.
- In order to tightly control access to resources and protect them from malicious access and introspection, Linux Security Modules such as SELinux should be used to enforce access rules.

Platform access

- Restrict traffic to only traffic that is necessary, and deny all other traffic, including traffic from and to 'Back-end'.
- Provide protections between the Internet and any workloads including web and volumetric attack preventions.
- All host to host communications within the cloud provider network are to be cryptographically protected in transit.
- Use cryptographically-protected protocols for administrative access to the platform.
- Data Centre Operations staff and systems must use management protocols that limit security risk such as SNMPv3, SSH v2, ICMP, NTP, syslog, and TLS v1.2 or higher.
- Processes for managing platform access control filters must be documented, followed, and monitored.
- Role-Based Access Control (RBAC) must apply for all platform systems access.
- All APIs access must use TLS protocol, including back-end APIs.

Workload security

- Restrict traffic to (and from) the workload to only traffic that is necessary, and deny all other traffic.
- Support zoning within a tenant workload - using application-level filtering.
- Not expose tenant internal IP address details to another tenant.
- All production workloads must be separated from all non-production workloads including separation between non-hosted non-production external networks.

Confidentiality and Integrity

- All data persisted to primary, replica, or backup storage is to be encrypted.

Monitoring and security audit

- All platform security logs are to be time synchronised.
- Logs are to be regularly scanned for events of interest.
- The cloud services must be regularly vulnerability and penetration tested.

Platform provisioning and LCM

- A platform change management process that is documented, well communicated to staff and tenants, and rigorously followed.
- A process to check change management adherence that is implemented, and rigorously followed.
- An approved system or process for last resort access must exist for the platform.
- Where there are multiple hosting facilities used in the provisioning of a service, network communications between the facilities for the purpose of backup, management, and workload communications are cryptographically protected in transit between data centre facilities.
- Continuous Cloud security compliance is mandatory.
- An incident response plan must exist for the platform.

7.4.2 Platform 'back-end' access security

- Validate and verify the integrity of resources management requests coming from a higher orchestration layer to the Cloud Infrastructure manager.

7.4.3 Platform 'front-end' access security

- Front-end network security at the application level will be the responsibility of the workload, however the platform must ensure the isolation and integrity of tenant connectivity to front-end networks.
- The front-end network may provide (Distributed Denial Of Service) DDOS support.

7.5 Workload Security - Vendor Responsibility

7.5.1 Software Hardening

- No hard-coded credentials or clear text passwords. Software should support configurable, or industry standard, password complexity rules.
- Software should be independent of the infrastructure platform (no OS point release dependencies to patch).
- Software is code signed and all individual sub-components are assessed and verified for EULA violations.
- Software should have a process for discovery, classification, communication, and timely resolution of security vulnerabilities (i.e.; bug bounty, Penetration testing/scan findings, etc.).
- Software should support recognized encryption standards and encryption should be decoupled from software.
- Software should have support for configurable banners to display authorized use criteria/policy.

7.5.2 Port Protection

- Unused software and unused network ports should be disabled by default

7.5.3 Software Code Quality and Security

- Vendors should use industry recognized software testing suites
    - Static and dynamic scanning
    - Automated static code review with remediation of Medium/High/Critical security issues. The tool used for static code analysis and analysis of code being released must be shared.
    - Dynamic security tests with remediation of Medium/High/Critical security issues. The tool used for Dynamic security analysis of code being released must be shared
    - Penetration tests (pen tests) with remediation of Medium/High/Critical security issues.
    - Methodology for ensuring security is included in the Agile/DevOps delivery lifecycle for ongoing feature enhancement/maintenance.

7.5.4 Alerting and monitoring

- Security event logging: all security events must be logged, including informational.
- Privilege escalation must be detected.

7.5.5 Logging

- Logging output should support customizable Log retention and Log rotation.

7.5.6 VNF images

- Image integrity – fingerprinting/validation.
- Container Images.
    - Container Management.
    - Immutability.

7.5.7 Vulnerability Management

- Security defect must be reported.
- Cadence should aligned with Cloud Infrastructure vendors (OSSA for OpenStack).
- Components should be analysed: mechanisms to validate components of the platform stack by checking libraries and supporting code against the Common Vulnerabilities and Exposures (CVE) databases to determine whether the code contains any known vulnerabilities must be embedded into the NFVI architecture itself. Some of the components required include tools for checking common libraries against CVE databases integrated into the deployment and orchestration pipelines.

7.6 Workload Security - Cloud Infrastructure Operator Responsibility

The Operator’s responsibility is to not only make sure that security is included in all the vendor supplied infrastructure and NFV components, but it is also responsible for the maintenance of the security functions from an operational and management perspective. This includes but is not limited to securing the following elements:

- Maintaining standard security operational management methods and processes.
- Monitoring and reporting functions.

- Processes to address regulatory compliance failure.
- Support for appropriate incident response and reporting.
- Methods to support appropriate remote attestation certification of the validity of the security components, architectures, and methodologies used.

7.6.1 Remote Attestation/openCIT

Cloud Infrastructure operators must ensure that remote attestation methods are used to remotely verify the trust status of a given Cloud Infrastructure platform. The basic concept is based on boot integrity measurements leveraging the Trusted Platform Module (TPM) built into the underlying hardware. Remote attestation can be provided as a service, and may be used by either the platform owner or a consumer/customer to verify that the platform has booted in a trusted manner. Practical implementations of the remote attestation service include the Open Cloud Integrity Tool (Open CIT). Open CIT provides 'Trust' visibility of the Cloud Infrastructure and enables compliance in Cloud Datacentres by establishing the root of trust and builds the chain of trust across hardware, operating system, hypervisor, VM, and container. It includes asset tagging for location and boundary control. The platform trust and asset tag attestation information is used by Orchestrators and/or Policy Compliance management to ensure workloads are launched on trusted and location/boundary compliant platforms. They provide the needed visibility and auditability of infrastructure in both public and private cloud environments.

7.6.2 Workload Image Scanning / Signing

It is easy to tamper with workload images. It requires only a few seconds to insert some malware into a workload image file while it is being uploaded to an image database or being transferred from an image database to a compute node. To guard against this possibility, workload images can be cryptographically signed and verified during launch time. This can be achieved by setting up a signing authority and modifying the hypervisor configuration to verify an image's signature before they are launched. To implement image security, the VNF operator must test the image and supplementary components verifying that everything conforms to security policies and best practices.

Use of Image scanners such as OpenSCAP to determine security vulnerabilities is strongly recommended.

7.6.3 Networking Security Zoning

Network segmentation is important to ensure that applications can only communicate with the applications they are supposed to. To prevent a workload from impacting other workloads or hosts, it is a good practice to separate workload traffic and management traffic. This will prevent attacks by VMs or containers breaking into the management infrastructure. It is also best to separate the VLAN traffic into appropriate groups and disable all other VLANs that are not in use. Likewise, workloads of similar functionalities can be grouped into specific zones and their traffic isolated. Each zone can be protected using access control policies and a dedicated firewall based on the needed security level.

Recommended practice to set network security policies following the principle of least privileged, only allowing approved protocol flows. For example, set 'default deny' inbound and add approved policies required for the functionality of the application running on the NFV Infrastructure.

7.6.4 Volume Encryption

Virtual volume disks associated with workloads may contain sensitive data. Therefore, they need to be protected. Best practice is to secure the workload volumes by encrypting them and storing the cryptographic keys at safe locations. Be aware that the decision to encrypt the volumes might cause reduced performance, so the decision to encrypt needs to be dependent on the requirements of the given infrastructure. The TPM module can also be used to securely store these keys. In addition, the hypervisor should be configured to securely erase the virtual volume disks in the event of application crashes or is intentionally destroyed to prevent it from unauthorized access.

7.6.5 Root of Trust for Measurements (RTM)

The sections that follow define mechanisms to ensure the integrity of the infrastructure pre-boot and post-boot (running). The following defines a set of terms used in those sections.

- The hardware root of trust helps with the pre-boot and post-boot security issues.
- Unified Extensible Firmware Interface (UEFI) adheres to standards defined by an industry consortium. Vendors (hardware, software) and solution providers collaborate to define common interfaces, protocols and structures for computing platforms.
- Platform Configuration Register (PCR) is a memory location in the TPM used to store TPM Measurements (hash values generated by the SHA-1 standard hashing algorithm). PCRs are cleared only on TPM reset. UEFI defines 24 PCRs of which the first 16, PCR 0 - PCR 15, are used to store measures created during the UEFI boot process.
- Root of Trust for Measurement (RTM) is a computing engine capable of making integrity measurements.
- Core Root of Trust for Measurements (CRTM) is a set of instructions executed when performing RTM.
- Platform Attestation provides proof of validity of the platform’s integrity measurements. Please see Section 7.6.1 Remote Attestation/openCIT.

Values stored in a PCR cannot be reset (or forged) as they can only be extended. Whenever a measurement is sent to a TPM, the hash of the concatenation of the current value of the PCR and the new measurement is stored in the PCR. The PCR values are used to encrypt data. If the proper environment is not loaded which will result in different PCR values, the TPM will be unable to decrypt the data.

7.6.5.1 Static Root of Trust for Measurement (SRTM)

Static RTM (SRTM) begins with measuring and verifying the integrity of the BIOS firmware. It then measures additional firmware modules, verifies their integrity, and adds each component’s measure to an SRTM value. The final value represents the expected state of boot path loads. SRTM stores results as one or more values stored in PCR storage. In SRTM, the CRTM resets PCRs 0 to 15 only at boot.

Using a Trusted Platform Module (TPM), as a hardware root of trust, measurements of platform components, such as firmware, bootloader, OS kernel, can be securely stored and verified. Cloud Infrastructure operators should ensure that the TPM support is enabled in the platform firmware, so that platform measurements are correctly recorded during boot time.

A simple process would work as follows;

1. The BIOS CRTM (Bios Boot Block) is executed by the CPU and used to measure the BIOS firmware.
2. The SHA1 hash of the result of the measurement is sent to the TPM.
3. The TPM stores this new result hash by extending the currently stored value.
4. The hash comparisons can validate settings as well as the integrity of the modules.

Cloud Infrastructure operators should ensure that OS kernel measurements can be recorded by using a TPM-aware bootloader (e.g. tboot, see https://sourceforge.net/projects/tboot/ or shim, see https://github.com/rhboot/shim), which can extend the root of trust up to the kernel level.

The validation of the platform measurements can be performed by TPM's launch control policy (LCP) or through the remote attestation server.

7.6.5.2 Dynamic Root of Trust for Measurement (DRTM)

In Dynamic Root of Trust for Measurement (DRTM), the RTM for the running environment are stored in PCRs starting with PCR 17.

If a remote attestation server is used to monitor platform integrity, the operators should ensure that attestation is performed periodically or in a timely manner. Additionally, platform monitoring can be extended to monitor the integrity of the static file system at run-time by using a TPM aware kernel module, such as Linux IMA (Integrity Measurement Architecture), see https://sourceforge.net/p/linux-ima/wiki/Home, or by using the trust policies (see https://github.com/opencit/opencit/wiki/Open-CIT-3.2-Product-Guide#88-trust-policies) functionality of OpenCIT.

The static file system includes a set of important files and folders which do not change between reboots during the lifecycle of the platform. This allows the attestation server to detect any tampering with the static file system during the runtime of the platform.

7.7 Common security standards

The Cloud Infrastructure Reference Model and the supporting architectures are not only required to optimally support networking functions, but they must be designed with common security principles and standards from inception. These best practices must be applied at all layers of the infrastructure stack and across all points of interconnections with outside networks, APIs and contact points with the NFV network functions overlaying or interacting with that infrastructure. Standards organizations with recommendations and best practices, and certifications that need to be taken into consideration include the following examples. However this is by no means an exhaustive list, just some of the more important standards in current use.

- Center for Internet Security - https://www.cisecurity.org/
- Cloud Security Alliance - https://cloudsecurityalliance.org/
- Open Web Application Security Project https://www.owasp.org
- The National Institute of Standards and Technology (NIST) (US Only)
- FedRAMP Certification https://www.fedramp.gov/ (US Only)

- ETSI Cyber Security Technical Committee (TC CYBER) - https://www.etsi.org/committee/cyber
- ETSI Industry Specification Group Network Functions Virtualisation (ISG NFV) - https://www.etsi.org/technologies/nfv
    - ETSI NFV ISG SEC WG specifications
- ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) - www.iso.org. The following ISO standards are of particular interest for NFVI
    - ISO/IEC 27002:2013 - ISO/IEC 27001 are the international Standard for best-practice information security management systems (ISMSs)
    - ISO/IEC 27032 - ISO/IEC 27032 is the international Standard focusing explicitly on cybersecurity
    - ISO/IEC 27035 - ISO/IEC 27035 is the international Standard for incident management
    - ISO/IEC 27031 - ISO/IEC 27031 is the international Standard for ICT readiness for business continuity

A good place to start to understand the requirements is to use the widely accepted definitions developed by the OWASP – Open Web Application Security Project. These include the following core principles:

- Confidentiality – Only allow access to data for which the user is permitted.
- Integrity – Ensure data is not tampered with or altered by unauthorized users.
- Availability – ensure systems and data are available to authorized users when they need it.

Additional Cloud Infrastructure security principles that need to be incorporated:

- Authenticity – The ability to confirm the users are in fact valid users with the correct rights to access the systems or data.

7.8 Testing & certification

7.8.1 Testing demarcation points

It is not enough to just secure all potential points of entry and hope for the best, any Cloud Infrastructure architecture must be able to be tested and validated that it is in fact protected from attack as much as possible. The ability to test the infrastructure for vulnerabilities on a continuous basis is critical for maintaining the highest level of security possible. Testing needs to be done both from the inside and outside of the systems and networks. Below is a small sample of some of the testing methodologies and frameworks available.

- OWASP testing guide
- Penetration Testing Execution Standard, PTES
- Technical Guide to Information Security Testing and Assessment, NIST 800-115
- VULCAN, Vulnerability Assessment Framework for Cloud Computing, IEEE 2013
- Penetration Testing Framework, VulnerabilityAssessment.co.uk

- Information Systems Security Assessment Framework (ISSAF)
- Open Source Security Testing Methodology Manual (OSSTMM)
- FedRAMP Penetration Test Guidance (US Only)
- CREST Penetration Testing Guide

Insuring that the security standards and best practices are incorporated into the Cloud Infrastructure and architectures must be a shared responsibility, among the Telecommunications operators interested in building and maintaining the infrastructures in support of their services, the application vendors developing the network services that will be consumed by the operators, and the Cloud Infrastructure vendors creating the infrastructures for their Telecommunications customers. All of the parties need to incorporate security and testing components, and maintain operational processes and procedures to address any security threats or incidents in an appropriate manner. Each of the stakeholders need to contribute their part to create effective security for the Cloud Infrastructure.

7.8.2 Certification requirements

Security certification should encompass the following elements:

- Security test cases executed and test case results.
- Industry standard compliance achieved (NIST, ISO, PCI, FedRAMP Moderate etc.).
- Output and analysis from automated static code review, dynamic tests, and penetration tests with remediation of Medium/High/Critical security issues. Tools used for security testing of software being released must be shared.
- Details on un-remediated low severity security issues must be shared.
- Threat models performed during design phase. Including remediation summary to mitigate threats identified.
- Details on un-remediated low severity security issues.
- Any additional Security and Privacy requirements implemented in the software deliverable beyond the default rules used security analysis tools.
- Resiliency tests run (such as hardware failures or power failure tests)

7.9 Consolidated Security Requirements

7.9.1 System Hardening

Ref	Requirement	Definition/Note
req.sec.gen.001	The Platform **must** maintain the specified configuration.	
req.sec.gen.002	All systems part of Cloud Infrastructure **must** support password hardening as defined in CIS Password Policy Guide https://www.cisecurity.org/white-papers/cis-password-policy-guide.	Hardening: CIS Password Policy Guide
req.sec.gen.003	All servers part of Cloud Infrastructure **must** support a root of trust and secure boot.	
req.sec.gen.004	The Operating Systems of all the servers part of Cloud Infrastructure **must** be hardened by removing or disabling unnecessary services, applications and network protocols, configuring operating system user authentication, configuring resource controls, installing and configuring	NIST SP 800-123

Ref	Requirement	Definition/Note
	additional security controls where needed, and testing the security of the Operating System.	
req.sec.gen.005	The Platform **must** support Operating System level access control.	
req.sec.gen.006	The Platform **must** support Secure logging. Logging with root account must be prohibited when root privileges are not required.	
req.sec.gen.007	All servers part of Cloud Infrastructure **must** be Time synchronized with authenticated Time service.	
req.sec.gen.008	All servers part of Cloud Infrastructure **must** be regularly updated to address security vulnerabilities.	
req.sec.gen.009	The Platform **must** support Software integrity protection and verification.	
req.sec.gen.010	The Cloud Infrastructure **must** support encrypted storage, for example, block, object and file storage, with access to encryption keys restricted based on a need to know. Controlled Access Based on the Need to Know https://www.cisecurity.org/controls/controlled-access-based-on-the-need-to-know.	
req.sec.gen.011	The Cloud Infrastructure **should** support Read and Write only storage partitions (write only permission to one or more authorized actors).	
req.sec.gen.012	The Operator **must** ensure that only authorized actors have physical access to the underlying infrastructure.	It is mandatory for a Cloud Infrastructure Operator, but this requirement's verification goes beyond CNTT testing scope
req.sec.gen.013	The Platform **must** ensure that only authorized actors have logical access to the underlying infrastructure.	
req.sec.gen.014	All servers part of Cloud Infrastructure **should** support measured boot and an attestation server that monitors the measurements of the servers.	
req.sec.gen.015	Any change to the Platform must be logged as a security event, and the logged event must include the identity of the entity making the change, the change, the date and the time of the change.	

Table 45: System hardening requirements

7.9.2 Platform and Access

Ref	Requirement	Definition/Note
req.sec.sys.001	The Platform must support authenticated and secure access to API, GUI and command line interfaces.	
req.sec.sys.002	The Platform **must** support Traffic Filtering for workloads (for example, Fire Wall).	
req.sec.sys.003	The Platform **must** support Secure and encrypted communications, and confidentiality and integrity of network traffic.	
req.sec.sys.004	The Cloud Infrastructure **must** support authentication, integrity and confidentiality on all network channels.	A secure channel enables transferring of data that is resistant to overhearing and tampering.
req.sec.sys.005	The Cloud Infrastructure **must** segregate the underlay and overlay networks.	
req.sec.sys.006	The Cloud Infrastructure must be able to utilize the Cloud Infrastructure Manager identity lifecycle management capabilities.	
req.sec.sys.007	The Platform **must** implement controls enforcing separation of duties and privileges, least privilege use and least common mechanism (Role-Based Access Control).	
req.sec.sys.008	The Platform **must** be able to assign the Entities that comprise the tenant networks to different trust domains.	Communication between different trust domains is not allowed, by default
req.sec.sys.009	The Platform **must** support creation of Trust Relationships between trust domains.	These maybe uni-directional relationships where the trusting domain trusts anther domain (the “trusted domain”) to authenticate users for them or to allow access to its resources from the trusted domain. In a bidirectional relationship both domain are “trusting” and “trusted”.
req.sec.sys.010	For two or more domains without existing trust relationships, the Platform **must not** allow the effect of an attack on one domain to impact the other domains either directly or indirectly.	
req.sec.sys.011	The Platform **must not** reuse the	

Ref	Requirement	Definition/Note
	same authentication credential (e.g., key-pair) on different Platform components (e.g., on different hosts, or different services).	
req.sec.sys.012	The Platform **must** protect all secrets by using strong encryption techniques, and storing the protected secrets externally from the component.	(e.g., in OpenStack Barbican).
req.sec.sys.013	The Platform **must** provide secrets dynamically as and when needed.	
req.sec.sys.014	The Platform **should** use Linux Security Modules such as SELinux to control access to resources.	

Table 46: Platform and access requirements

7.9.3 Confidentiality and Integrity

Ref	Requirement	Definition/Note
req.sec.ci.001	The Platform **must** support Confidentiality and Integrity of data at rest and in transit.	
req.sec.ci.002	The Platform **should** support self-encrypting storage devices.	
req.sec.ci.003	The Platform **must** support Confidentiality and Integrity of data related metadata.	
req.sec.ci.004	The Platform **must** support Confidentiality of processes and restrict information sharing with only the process owner (e.g., tenant).	
req.sec.ci.005	The Platform **must** support Confidentiality and Integrity of process-related metadata and restrict information sharing with only the process owner (e.g., tenant).	
req.sec.ci.006	The Platform **must** support Confidentiality and Integrity of workload resource utilization (RAM, CPU, Storage, Network I/O, cache, hardware offload) and restrict information sharing with only the workload owner (e.g., tenant).	
req.sec.ci.007	The Platform **must not** allow Memory Inspection by any actor other than the authorized actors for the Entity to which Memory is assigned (e.g., tenants owning the workload), for Lawful Inspection, and by secure monitoring services.	Admin access must be carefully regulated.
req.sec.ci.008	The Cloud Infrastructure **must** support tenant networks segregation.	

Table 47: Confidentiality and integrity requirements

7.9.4 Workload Security

Ref	Requirement	Definition/Note
req.sec.wl.001	The Platform **must** support Workload placement policy.	
req.sec.wl.002	The Cloud Infrastructure **must** provide methods to ensure the platform's trust status and integrity (e.g. remote attestation, Trusted Platform Module).	
req.sec.wl.003	The Platform **must** support secure provisioning of workloads.	
req.sec.wl.004	The Platform **must** support Location assertion (for mandated in-country or location requirements).	
req.sec.wl.005	The Platform **must** support the separation of production and non-production Workloads.	This requirement's verification goes beyond CNTT testing scope.
req.sec.wl.006	The Platform **must** support the separation of Workloads based on their categorisation (for example, payment card information, healthcare, etc.).	
req.sec.wl.007	The Operator **should** implement processes and tools to verify VNF authenticity and integrity.	

Table 48: Workload security requirements

7.9.5 Image Security

Ref	Requirement	Definition/Note
req.sec.img.001	Images from untrusted sources **must not** be used.	
req.sec.img.002	Images **must** be maintained to be free from known vulnerabilities.	
req.sec.img.003	Images **must not** be configured to run with privileges higher than the privileges of the actor authorized to run them.	
req.sec.img.004	Images **must** only be accessible to authorized actors.	
req.sec.img.005	Image Registries **must** only be accessible to authorized actors.	
req.sec.img.006	Image Registries **must** only be accessible over secure networks that enforce authentication, integrity and confidentiality.	
req.sec.img.007	Image registries **must** be clear of vulnerable and out of date versions.	

Table 49: Image security requirements

7.9.6 Security LCM

Ref	Requirement	Definition/Note
req.sec.lcm.001	The Platform **must** support Secure Provisioning, Availability, and Deprovisioning (Secure Clean-Up) of workload resources where Secure Clean-Up includes tear-down,	Secure clean-up: tear-down, defending against virus or other attacks, or observing of cryptographic or user service

Ref	Requirement	Definition/Note
	defence against virus or other attacks.	data.
req.sec.lcm.002	Operational **must** use management protocols limiting security risk such as SNMPv3, SSH v2, ICMP, NTP, syslog and TLS v1.2 or higher.	
req.sec.lcm.003	The Cloud Operator **must** implement and strictly follow change management processes for Cloud Infrastructure, Cloud Infrastructure Manager and other components of the cloud, and Platform change control on hardware.	
req.sec.lcm.004	The Cloud Operator **should** support automated templated approved changes.	Templated approved changes for automation where available.
req.sec.lcm.005	Platform **must** provide logs and these logs must be regularly monitored for anomalous behaviour.	
req.sec.lcm.006	The Platform **must** verify the integrity of all Resource management requests.	
req.sec.lcm.007	The Platform **must** be able to update newly instantiated, suspended, hibernated, migrated and restarted images with current time information.	
req.sec.lcm.008	The Platform **must** be able to update newly instantiated, suspended, hibernated, migrated and restarted images with relevant DNS information.	
req.sec.lcm.009	The Platform **must** be able to update the tag of newly instantiated, suspended, hibernated, migrated and restarted images with relevant geolocation (geographical) information.	
req.sec.lcm.010	The Platform **must** log all changes to geolocation along with the mechanisms and sources of location information (i.e. GPS, IP block, and timing).	
req.sec.lcm.011	The Platform **must** implement Security life cycle management processes including the proactive update and patching of all deployed Cloud Infrastructure software.	

Table 50: Security LCM requirements

7.9.7 Monitoring and Security Audit

The Platform is assumed to provide configurable alerting and notification capability and the operator is assumed to have automated systems, policies and procedures to act on alerts and notifications in a timely fashion. In the following the monitoring and logging capabilities can trigger alerts and notifications for appropriate action.

Ref	Requirement	Definition/Note
req.sec.mon.001	Platform **must** provide logs and these logs must be regularly	

Ref	Requirement	Definition/Note
	monitored for events of interest.	
req.sec.mon.002	Security logs **must** be time synchronised.	
req.sec.mon.003	The Platform **must** log all changes to time server source, time, date and time zones.	
req.sec.mon.004	The Platform **must** secure and protect Audit logs (containing sensitive information) both in-transit and at rest.	
req.sec.mon.005	The Platform **must** Monitor and Audit various behaviours of connection and login attempts to detect access attacks and potential access attempts and take corrective actions accordingly.	
req.sec.mon.006	The Platform **must** Monitor and Audit operations by authorized account access after login to detect malicious operational activity and take corrective actions.	
req.sec.mon.007	The Platform **must** Monitor and Audit security parameter configurations for compliance with defined security policies.	
req.sec.mon.008	The Platform **must** Monitor and Audit externally exposed interfaces for illegal access (attacks) and take corrective security hardening measures.	
req.sec.mon.009	The Platform **must** Monitor and Audit service for various attacks (malformed messages, signalling flooding and replaying, etc.) and take corrective actions accordingly.	
req.sec.mon.010	The Platform **must** Monitor and Audit running processes to detect unexpected or unauthorized processes and take corrective actions accordingly.	
req.sec.mon.011	The Platform **must** Monitor and Audit logs from infrastructure elements and workloads to detected anomalies in the system components and take corrective actions accordingly.	
req.sec.mon.012	The Platform **must** Monitor and Audit Traffic patterns and volumes to prevent malware download attempts.	
req.sec.mon.013	The monitoring system **must not** affect the security (integrity and confidentiality) of the infrastructure, workloads, or the user data (through back door entries).	
req.sec.mon.014	The Monitoring systems **should not** impact IAAS, PAAS, and SAAS SLAs including availability SLAs.	
req.sec.mon.015	The Platform **must** ensure that the Monitoring systems are never starved of resources.	
req.sec.mon.016	The Platform Monitoring components **should** follow security best practices for auditing, including secure logging and tracing.	
req.sec.lcm.017	The Platform **must** audit systems for any missing security patches and take appropriate actions.	

Table 51: Monitoring and security audit requirements

7.9.8 Compliance with Standards

Ref	Requirement	Definition/Note
req.sec.std.001	The Cloud Operator **should** comply with Center for Internet Security CIS Controls.	Center for Internet Security - https://www.cisecurity.org/
req.sec.std.002	The Cloud Operator, Platform and Workloads **should** follow the guidance in the CSA Security Guidance for Critical Areas of Focus in Cloud Computing (latest version).	Cloud Security Alliance - https://cloudsecurityalliance.org/
req.sec.std.003	The Platform and Workloads **should** follow the guidance in the OWASP Cheat Sheet Series (OCSS) https://github.com/OWASP/CheatSheetSeries.	Open Web Application Security Project https://www.owasp.org
req.sec.std.004	The Cloud Operator, Platform and Workloads **should** ensure that their code is not vulnerable to the OWASP Top Ten Security Risks https://owasp.org/www-project-top-ten/.	
req.sec.std.005	The Cloud Operator, Platform and Workloads **should** strive to improve their maturity on the OWASP Software Maturity Model (SAMM) https://owaspsamm.org/blog/2019/12/20/version2-community-release/.	
req.sec.std.006	The Cloud Operator, Platform and Workloads **should** utilize the OWASP Web Security Testing Guide https://github.com/OWASP/wstg/tree/master/document.	
req.sec.std.007	The Cloud Operator, and Platform **should** satisfy the requirements for Information Management Systems specified in ISO/IEC 27001 https://www.iso.org/obp/ui/#iso:std:iso-iec:27001:ed-2:v1:en.	ISO/IEC 27002:2013 - ISO/IEC 27001 is the international Standard for best-practice information security management systems (ISMSs).
req.sec.std.008	The Cloud Operator, and Platform **should** implement the Code of practice for Security Controls specified ISO/IEC 27002:2013 (or latest) https://www.iso.org/obp/ui/#iso:std:iso-iec:27002:ed-2:v1:en.	
req.sec.std.009	The Cloud Operator, and Platform **should** implement the ISO/IEC 27032:2012 (or latest) Guidelines for Cybersecurity techniques https://www.iso.org/obp/ui/#iso:std:iso-iec:27032:ed-1:v1:en.	ISO/IEC 27032 - ISO/IEC 27032is the international Standard focusing explicitly on cybersecurity.
req.sec.std.010	The Cloud Operator **should** conform to the ISO/IEC 27035 standard for incidence management.	ISO/IEC 27035 - ISO/IEC 27035 is the international Standard for incident management.
req.sec.std.011	The Cloud Operator **should** conform to the ISO/IEC 27031 standard for business continuity ISO/IEC	

Ref	Requirement	Definition/Note
	27031 - ISO/IEC 27031 is the international Standard for ICT readiness for business continuity.	
req.sec.std.012	The Public Cloud Operator **must**, and the Private Cloud Operator **may** be certified to be compliant with the International Standard on Awareness Engagements (ISAE) 3402 (in the US: SSAE 16).	International Standard on Awareness Engagements (ISAE) 3402. US Equivalent: SSAE16.

Table 52: Compliance with standards requirements

7.10 Security References

Network Functions Virtualisation (NFV);NFV Security; Problem Statement, ETSI GS NFV-SEC 001 V1.1.1 (2014-10)

Network Functions Virtualisation (NFV);NFV Security; Security and Trust Guidance, ETSI GS NFV-SEC 003 V1.1.1 (2014-12)

Network Functions Virtualisation (NFV) Release 3; Security; Security Management and Monitoring specification, ETSI GS NFV-SEC 013 V3.1.1 (2017-02)

Network Functions Virtualisation (NFV) Release 3; NFV Security; Security Specification for MANO Components and Reference points, ETSI GS NFV-SEC 014 V3.1.1 (2018-04)

Network Functions Virtualisation (NFV) Release 2; Security; VNF Package Security Specification, ETSI GS NFV-SEC 021 V2.6.1 (2019-06)

ETSI Industry Specification Group Network Functions Virtualisation (ISG NFV) - https://www.etsi.org/committee/1427-nfv

ETSI Cyber Security Technical Committee (TC CYBER) - https://www.etsi.org/committee/cyber

NIST Documents

NIST SP 800-53 Security and Privacy Controls for Federal Information Systems and Organizations https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf

NIST SP 800-53A Assessing Security and Privacy Controls in Federal Information Systems and Organizations: Building Effective Assessment Plans https://www.serdp-estcp.org/content/download/47513/453118/file/NIST%20SP%20800-53A%20Rev%204%202013.pdf

NIST SP 800-63B Digital Identity Guidelines https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf

NIST SP 800-115 Technical Guide to Information Security Testing and Assessment https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf

NIST SP 800-123 Guide to General Server Security https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-123.pdf

NIST SP 800-125 Guide to Security for Full Virtualization Technologies
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-125.pdf

NIST SP 800-125a Security Recommendations for Server-based Hypervisor Platforms
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-125Ar1.pdf

NIST SP 800-125b Secure Virtual Network Configuration for Virtual Machine (VM) Protection
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-125B.pdf

NIST SP 800-137 Information Security Continuous Monitoring for Federal Information Systems and Organizations
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-137.pdf

NIST SP 800-145 The NIST Definition of Cloud Computing
https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

NIST SP 800-190 Application Container Security Guide
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf

8 Chapter reserved for future use

This chapter is intentionally blank and is reserved for future use.

9 Infrastructure Operations and Lifecycle Management

9.1 Introduction

The purpose of this chapter is to define the capabilities required of the infrastructure to ensure it is effectively supported, maintained and otherwise lifecycle-managed by Operations teams. This includes requirements relating to the need to be able to maintain infrastructure services "in-service" without impacting the applications and VNFs, whilst minimising human labour. It shall also capture any exceptions and related assumptions.

There are two main business operating frameworks that are commonly known and used across the Telecommunications industry related to the topics in this chapter:

- FCAPS (ISO model for network management)
- eTOM (TM Forum Business Process Framework (eTOM))

The chapters below roughly map to these frameworks as follows:

Chapter Name	FCAPS	eTOM
Configuration and Lifecycle Management	Configuration	Fulfilment
Assurance	Performance	Assurance
	Fault	
Capacity Management	Configuration	Fulfilment

Table 53: Operating Frameworks

9.2 Configuration and Lifecycle Management

Configuration management is concerned with defining the configuration of infrastructure and its components, and tracking (observing) the running configuration of that infrastructure, and any changes that take place. Modern configuration management practices such as desired state configuration management also mean that any changes from the desired state that are observed (aka the delta) are rectified by an orchestration / fulfilment component of the configuration management system. This "closed loop" mitigates against configuration drift in the infrastructure and its components. Our recommendation is to keep these closed loops as small as possible to reduce complexity and risk of error. Figure 33 shows the configuration management "loop" and how this relates to lifecycle management.

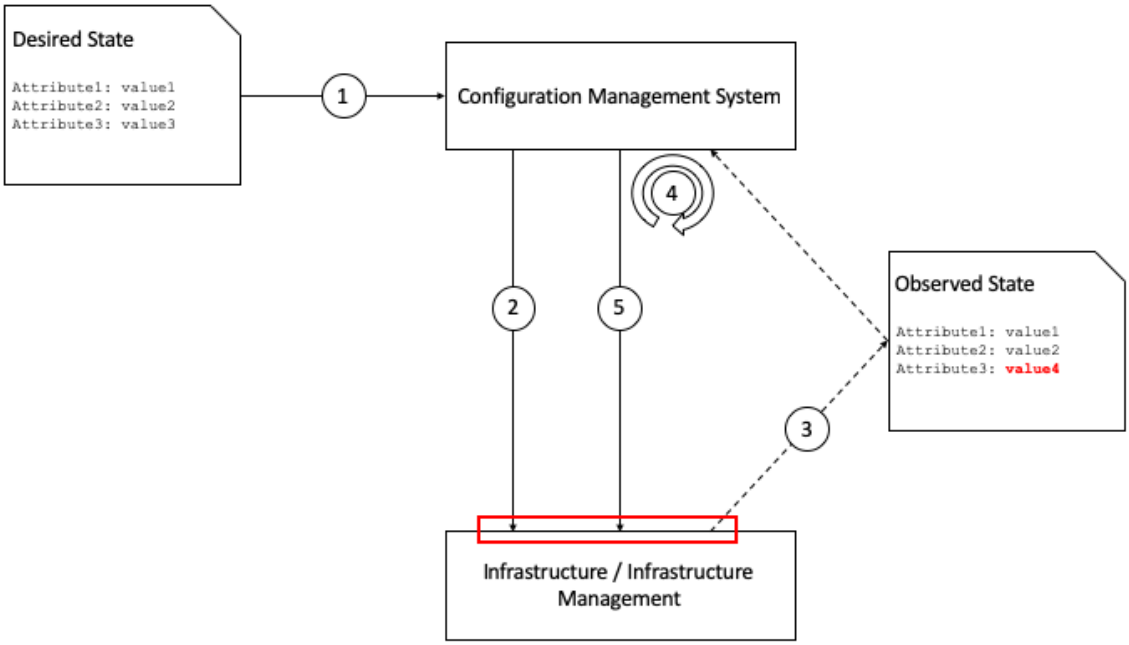

Figure 33: Configuration and Lifecycle Management

The initial desired state might be for 10 hosts with a particular set of configuration attributes, including the version of the hypervisor and any management agents. The configuration management system will take that as input (1) and configure the infrastructure as required (2). It will then observe the current state periodically over time (3) and in the case of a difference between the desired state and the observed state it will calculate the delta (4) and re-configure the infrastructure (5). For each lifecycle stage (create, update, delete) this loop takes place - for example if an update to the hypervisor version is defined in the desired state, the configuration management system will calculate the delta (e.g. v1 --> v2) and re-configure the infrastructure as required.

However, the key requirements for the infrastructure and infrastructure management are those interfaces and reference points in the red box - where configuration is **set**, and where it is **observed**. Table 54 lists the main components and capabilities required in order to manage the configuration and lifecycle of those components.

Component	set / observe	Capability	Example
Cloud Infrastructure Management Software	Set	Target software / firmware version	Software: v1.2.1
		Desired configuration attribute	dhcp_lease_time: 86400
		Desired component quantities	# hypervisor hosts: 10
	Observe	Observed software / firmware version	Software: v1.2.1
		Observed configuration attribute	dhcp_lease_time: 86400
		Observed component quantities	# hypervisor hosts: 10
Cloud Infrastructure Software	Set	Target software version	Hypervisor software: v3.4.1
		Desired configuration attribute	management_int: eth0
		Desired component quantities	# NICs for data: 6
	Observe	Observed software / firmware version	Hypervisor software: v3.4.1
		Observed configuration attribute	management_int: eth0
		Observed component quantities	# NICs for data: 6
Infrastructure Hardware	Set	Target software / firmware version	Storage controller firmware: v10.3.4
		Desired configuration attribute	Virtual disk 1: RAID1 [HDD1, HDD2]
	Observe	Observed software / firmware version	Storage controller firmware: v10.3.4
		Observed configuration attribute	Virtual disk 1: RAID1 [HDD1, HDD2]

Table 54: Configuration and Lifecycle Management Capabilities

This leads to the following table (Table 55) which defines the standard interfaces that should be made available by the infrastructure and Cloud Infrastructure Management components to allow for successful Configuration Management.

Component	Interface Standard	Link
Infrastructure Management	Defined in RA specifications	RA-1, RA-2
Infrastructure Software	Defined in RA specifications	RA-1, RA-2
Infrastructure Hardware	Redfish API	DMTF RedFish specification [11]

Table 55: Interface Standards for Configuration Management

9.3 Assurance

Assurance is concerned with:

- The proactive and reactive maintenance activities that are required to ensure infrastructure services are available as per defined performance and availability levels.
- Continuous monitoring of the status and performance of individual components and of the service as a whole.
- Collection and analysis of performance data, which is used to identify potential issues including the ability to resolve the issue with no customer impact.

There are the following requirement types:

1. Data collection from all components, e.g.

    - The ability to collect data relating to events (transactions, security events, physical interface up/down events, warning events, error events, etc.)
    - The ability to collect data relating to component status (up/down, physical temperature, disk speed, etc.)
    - The ability to collect data relating to component performance (used CPU resources, storage throughput, network bandwidth in/out, API transactions, transaction response times, etc.)

2. Capabilities of the Infrastructure Management Software to allow for in-service maintenance of the Infrastructure Software and Hardware under its management, e.g.

    - The ability to mark a physical compute node as being in some sort of "maintenance mode" and for the Infrastructure Management Software to ensure all running workloads are moved off or rescheduled on to other available nodes (after checking that there is sufficient capacity) before marking the node as being ready for whatever maintenance activity needs to be performed
    - The ability to co-ordinate, automate, and allow the declarative input of in-service software component upgrades - such as internal orchestration and scheduler components in the Infrastructure Management Software

Note that the above only refers to components - it is expected that any "service" level assurance doesn't add any further requirements onto the infrastructure, but rather takes the data extracted and builds service models based on the knowledge it has of the services being offered.

9.4 Capacity Management

Capacity Management is a potentially wide ranging process that includes taking demand across lines of business, analysing data about the infrastructure that is running, and calculating when additional infrastructure might be required, or when infrastructure might need to be decommissioned.

As such the requirements for Capacity Management on the infrastructure are covered by the Assurance and Configuration and Lifecycle Management sections above. The Assurance section deals with the collection of data - there is no reason to consider that this would be done by a different mechanism for Capacity Management as it is for Assurance - and the Configuration and Lifecycle Management section deals with the changes being made to the infrastructure hardware, software, and management components (e.g. changing of number of hypervisor hosts from 10 to 12).

10 Challenges and Gaps

10.1 Introduction

This chapter is dedicated to identifying the challenges and gaps found in the course of the development of the reference model to ensure that it continues to be of strategic and tactical value intended over time. Should a challenge or gap not be identified that is not already addressed in the model itself, the community may assume it will remain an unknown and, therefore, the community is encouraged to engage with and raise an issue with the appropriate working group(s) to close the gap. In this manner, the Reference Model can continuously improve.

10.2 Challenges

The continuous challenge is finalizing a stable version from which all stakeholders in the application value-chain can derive the intended value of a Common Cloud Infrastructure. This maturity level is reached when the released Reference Model version is adopted by stakeholders into their application development and deployment cycles.

10.3 Gaps

This section addresses major open issues identified in the development of the Reference Model, Reference Architecture and Reference Implementation of the Common Cloud Infrastructure Lifecycle Framework.

10.3.1 Discovery

The workloads (VNFs/CNFs) and Cloud Infrastructure should be able to discover each other and exchange their capabilities required or offered. One of the key pain points for most of the operators is the VNF/CNF onboarding - both in terms of time and complexity. It could take weeks or months to on board a VNF/CNF. There are lots of static and laborious checks performed to ensure the compatibility of the workloads with the corresponding Cloud Infrastructure. The onboarding of the workloads (network functions) should be automated as much as possible. The workloads and Cloud Infrastructure should be able to discover and negotiate their capabilities. Following should be supported:

- Capabilities Discovery and Advertising
    - Cloud Infrastructure should be able to publish the capabilities it offers to workloads (network functions)
    - workloads should be able to query the Cloud Infrastructure for specific capabilities - such as number of cores, performance parameters

- Capabilities Negotiation/Hand Shake API:
    - workloads and Cloud Infrastructure should be able to negotiate on certain capabilities. For instance, workload desires HW acceleration for high throughput, but should be able to fall back to high throughput offered by Cloud Infrastructure via DPDK offering, and vice-a-versa.

10.3.2 Support Load Balance of VNF/CNFs

The ability to dynamically scale a network function by load balancing across multiple instances/replicas of the same VNF or CNF is essential. New architectures and application patterns such as micro services is making this even more crucial. It must not only be possible to load balance and scale each service layer independently, support to chain the different layers together through "Service Function Chaining" is also needed.

The load balancing and scaling needed for typical enterprise applications is well supported in OpenStack by the Octavia v2 API, the Octavia v2 API is a backwards compatible superset of the old neutron LBaaS v2 API that it is replacing.

The built in mechanism in Kubernetes for scaling enterprise type of services and PODs is also sufficient for applications that only use one interface.

What is not supported in either OpenStack or Kubernetes is to scale and load balance a typical VNF and CNF. There is no support in OpenStack to scale stateful L3 applications such as SCTP, QUIC. mTCP, and gRPC. In Kubernetes it is even worse. The built in Kubernetes network support is tied to the first POD/container interface. Support for secondary interfaces is managed through the Container Network Interface, CNI, and by CNI plugins, such as Multus, that support the "Kubernetes Network Customs Resource Definition" specified by the Kubernetes Network Plumbing Group. This specification supports attachment of network endpoints to PODs, IP address management and the ability of define interface specific static routes. There is no support for network orchestration and functions such as load balancing, routing, ACL and firewalls.

10.3.3 Service Function Chain

Over the past few years there has been a significant move towards decomposing network functions into smaller sub-functions that can be independently scaled and potentially reused across multiple network functions. A service function chain allows composition of network functions by passing selected packets through an ordered list of services. In order to support this capability in a sustainable manner, there is a need to have the capability to model service chains as a high level abstraction. This is essential to ensure that the underlying connection setup, and (re-)direction of traffic flows can be performed in an automated manner. There is also a need to provide specialized tools aid troubleshooting of individual services and the communication between them in order to investigate issues in the performance of composed network functions.

10.3.4 Packet Acceleration Request (e.g. Hardware Acceleration)

While generic server hardware capabilities can be exclusively used for handling networking related workloads, this strategy is neither performant nor energy efficient. There are several forms of accelerators such as smart NICs, programmable networking fabrics/switches, and

GPUs that can offload some of these workloads in order to provide higher throughput, energy efficiency and lower latency. The acceleration hardware is typically optimized for specific kinds of workloads and some form of disaggregation might be required to separate control and user plane responsibilities between generic server hardware and accelerators. Additionally, there might also be a need for disaggregation between user plane functions that may require different types of accelerators. There is also a need for workload orchestration to be able to understand the packet acceleration needs of specific workloads and schedule such workloads on infrastructure with the requisite capabilities. This will require that there be some form of discovery mechanism that would allow the workload orchestration to discover the presence of acceleration hardware.

10.3.5 Multi-cloud architecture directions for network workloads

There is a growing interest in using a multi-cloud environment for the deployment of network functions. The industry investigates and starts to experiment with deploying and operating network functions across several private and/or public clouds to reuse services and capabilities available in these cloud environments instead of investing in a duplications of such capabilities. 5G and Edge deployments seem to be some of the catalysts of the growing interest. The Reference Model will need to provide in its future releases relevant guidelines for such multi-cloud architectures.

Annex A Cloud iNfrastructure Telco Taskforce

A.1 Overview

Initially organized early in 2019, the Cloud iNfrastructure Telco Taskforce (CNTT) was created in response to rapid changes in how networking applications are being designed, built and managed, plus a growing recognition of a perceived functional gap between the previous standard infrastructure models and the architectures needed to support Network Function Virtualisation (NFV) applications. Organizationally, the Cloud iNfrastructure Telco Taskforce, jointly hosted by GSMA and the Linux Foundation, operates as an open committee responsible for creating and documenting an industry aligned Common Cloud Infrastructure Framework. The CNTT was created with the intent that it would create the cloud infrastructure framework, and eventually morph into an on-going project under the auspices of the GSMA and the Linux Foundation umbrellas. The final on-going operational form of the Taskforce will be determined as the project evolves.

A.1.1 Terminology and Glossary

The definition and intent of the terminology used throughout the documents is defined in the Reference Model Glossary, see Annex B.

A.1.2 Problem Statement

Based on informal conversations with many operators and developers, there is a realisation that there are significant technical, operational and business challenges to the development and deployment of VNF/CNF applications related to the lack of a common cloud infrastructure platform. These include but are not limited to the following:

- Higher development costs due to the need to develop Virtual Network Functions (VNF) on multiple custom platforms for each operator
- Increased complexities due to the need to maintain multiple versions of applications to support each custom environment
- Lack of testing and validation commonalities, leading to inefficiencies and increased time to market. While the operators will still do internal testing, but using an industry driven verification program based on a common cloud infrastructure would provide a head start.
- Slower adoption of cloud-native applications and architectures. A Common Telco Cloud may provide an easier path to methodologies that will drive faster cloud-native development.
- Increased operational overhead due to the need for operators to integrate diverse and sometime conflicting cloud platform requirements.

One of major challenges holding back the more rapid and widespread adoption of VNF is that the traditional telecom ecosystem vendors, while building or designing their virtualised services (whether it be Voice over LTE (VoLTE), Evolved Packet Core (EPC), or popular customer facing enterprise services such as SD-WAN (Software Defined Wide Area Network), are making their own infrastructure assumptions and requirements, often with custom design parameters. This leaves the operators being forced to build complex integrations of various vendor/function specific silos which are incompatible with each other and might possibly have different and conflicting operating models. In addition, this makes

the onboarding and conformance processes of VNFs/CNFs (coming from different vendors) hard to automate and standardise.

To put this effort into perspective, over the past few years, the telecom industry has been going through a massive technology revolution by embracing software defined networking and cloud architecture principles, in pursuit of the goal of achieving more flexibility, agility and operational efficiency. At a high level, the main objective of NFV (Network Function Virtualisation) is the ability to use general purpose standard COTS (Commercial off the Shelf) compute, memory and storage hardware platforms to run multiple Virtualised Network Functions. Earlier common infrastructure models built on the previous assumption that networking applications are typically built on discrete hardware, do not offer the level of flexibility and agility needed for the support of newer networking technologies such as 5G, intelligent networks and Edge computing. By running network applications as software rather than on purpose-built hardware, as it has been done since the early 1990's, the operators aspire to realize operational efficiencies, and capital expense savings. These Software Defined Network (SDN) applications are increasingly being used by telecom operators to support their internal and customer facing network infrastructures. The need for a common model across the industry to facilitate more rapid adoption is clear.

A.1.3 Project Goals and Purpose

The goal of the task force is to develop a robust infrastructure model and a limited discrete set of architectures built on that model that can be validated for use across the entire member community. The community, which is made up of a cross section of global operators and supporting vendors alike, was created to support the development, deployment and management of NFV applications faster and more easily.

All of this had led to a growing awareness of the need to develop more open models and validation mechanisms to bring the most value to telco operators as well as vendors, by agreeing on a standard set of infrastructure profiles to use for the underlying infrastructure to support VNF/CNF applications across the industry and telecom community at large. To achieve this goal, the cloud environment needs to be fully abstracted via APIs and other mechanisms to the VNFs/CNFs so that both developers of the VNF/CNF applications and the operators managing the environments can benefit from the flexibility that the disaggregation of the underlying infrastructure offers.

The next step after the Reference Model has been identified and developed is to take the general model, which is purposely designed to be able to be applied to a number of technologies, and apply it to a discrete number of concrete and ultimately deployable Reference Architecture platforms. The intention is to choose the reference architectures carefully so that there will only be a small set of architectures that meets the specific requirements for supporting NFV and Telecom specific applications. Per the principles laid out later in this document, the Reference Architectures need to meet the following criteria as much as is practical:

- Initially should be based on widely established technology and systems used in the Telecom Industry. This will help ensure a faster adoption rate because the operators are already familiar with the technology and might even have systems in production. Another advantage to this approach is a project faster development cycle.

- Subsequent architectures should be based on either additional established or promising emerging technologies that are chosen by the community members.

A.1.4 Common Cloud Infrastructure Benefits

By providing a pre-defined environment with common capabilities, applications are able to be developed and deployed more rapidly. In addition, the common infrastructure can be optimized for various workloads, such as IT (Information Technology), VNF, AI (Artificial Intelligence), and other future workload types as new technologies emerge. The benefits of this approach are:

- Configuration automation over customisation
    - By abstracting the infrastructure capabilities as much as possible, operators are able to use common infrastructure platforms across all VNF/CNF vendors.
    - Maintaining a consistent infrastructure allows for higher levels of automation due to a reduced need for customisation of the various components.
    - Overall, the intention is to reduce the total cost of ownership for operators and development costs for vendors
- Onboarding and conformance
    - By defining abstracted infrastructure capabilities, and the metrics by which they are measured, the onboarding and conformance process for both cloud infrastructure and VNFs/CNFs can be standardized, reducing development time for the VNF/CNF developers and deployment and operational management costs for the operators standing up the cloud environments.
    - Supply chain, procurement and assurance teams can then use these metrics to more accurately assess the most efficient / best value vendor for a given environment and network services requirement.
- Better utilisation
    - Properly mapping VNFs/CNFs to flavours to the underlying infrastructure, brings the potential for more efficient utilisation, than needing to create specific configurations for each type of application in the infrastructure.

In conclusion, to serve the stated objective building a common cloud infrastructure that is able to take advantage of true cloud models for the more rapid development and deployment of SDN NFV applications, the CNTT is documentation of a reference model, a select set of architectures, a set of reference implementations, and a set of conformance suites, so that there is a more consistent model infrastructure for developers and vendors of SDN software and applications to build to.

A.2 Principles

Any specification work created within CNTT **must** conform to the following principles:

A.2.1 Overall Principles

1. A top-level objective is to build a single, overarching Reference Model with the smallest number of Reference Architectures tied to it as is practical. Two principles are introduced in support of these objectives:

    - **Minimise Architecture proliferation by stipulating compatible features be contained within a single Architecture as much as possible:**

        - Features which are compatible, meaning they are not mutually exclusive and can coexist in the same cloud infrastructure instance, shall be incorporated into the same Reference Architecture. For example, IPv4 and IPv6 should be captured in the same Architecture, because they don't interfere with each other
        - Focus on the commonalities of the features over the perceived differences. Seek an approach that allows small differences to be handled at either the low-level design or implementation stage. For example, assume the use of existing common APIs over new ones.

    - **Create an additional Architecture only when incompatible elements are unavoidable:**

        - Creating additional Architectures is limited to when incompatible elements are desired by Taskforce members. For example, if one member desires KVM be used as the hypervisor, and another desires ESXi be used as the hypervisor, and no compromise or mitigation* can be negotiated, the Architecture could be forked, subject to community consensus, such that one Architecture would be KVM-based and the other would be ESXi-based.

    Note: *Depending on the relationships and substitutability of the component(s) in question, it may be possible to mitigate component incompatibility by creating annexes to a single Architecture, rather than creating an additional Architecture. With this approach, the infrastructure architecture designers might implement the Architecture as described in the reference document, however when there is a potential for incompatibility for particular component, they would select their preferred option from one of the relevant annexes. For example, if one member wanted to use Software-Defined storage (SDS) as CEPH, and another member wanted to use Storage Attached Network (SAN), assuming the components are equally compatible with the rest of the Architecture, there could be one annex for the CEPH implementation and one annex for the SAN implementation.

2. Cloud Infrastructure provides abstract and physical resources corresponding to:

    - Compute resources
    - Storage resources

- Memory resources
- Networking resources (Limited to connectivity services only)
- Acceleration resources

3. Vendor independence of Cloud Infrastructure exposed resources.
4. Cloud Infrastructure Application Programming Interfaces (APIs) ensure Interoperability (multi-vendor, components substitution), drive Simplification, and open source implementations that have an open governance model (e.g. come from Open Communities or Standards Development Organisations). • These APIs support, for example, cloud infrastructure resources discovery, monitoring by management entities, configuration on behalf of workloads and consumption by workloads
5. Workloads are modular and designed to utilise the minimum resources required for the service.
6. Workloads consume only the resources, capabilities and features provided by the Cloud infrastructure.
7. Workload functional capabilities independence from Cloud Infrastructure (hardware and software) accelerations.
8. Workload independence from Cloud Infrastructure (hardware and software) hardware-dependent software

    - This is in support of workload abstraction, enabling portability across the Infra and simplification of workload design
    - Use of critical features in this category are governed by technology specific policies and exceptions in the RA specifications.

9. Abstraction of specific internal hardware details above the Infrastructure Cloud Management layers unless managed through Hardware Infrastructure Manager

    - This is in support of workload abstraction, enabling portability across the Infra and simplification of workload design
    - Use of critical features in this category are governed by technology specific policies and exceptions in the RA specifications.

A.2.2 Requirements Principles

The agreed upon rules and recommendations to which a compliant workload or cloud infrastructure must adhere.

- All requirements will be hosted and maintained in the RM or relevant RA
- All requirements must be assigned a requirements ID and not be embedded in narrative text. This is to ensure that readers do not have to infer if a requirement exists and is applicable
- Requirements must have a unique ID for tracking and reference purposes
- The requirement ID should include a prefix to delineate the source project
- Requirements must state the level of compliance (ex: MUST, SHOULD, MAY) per RFC 2119[2]
- Mandatory requirements must be defined in such a way that they are unambiguously verifiable via automated testing

- Requirements should be publishable or extractable into a machine readable format such as JSON
- Requirements should include information about the impact of non-conformance and the rationale for their existence

A.2.3 Architectural Principles

Following are a number of key architectural principles that apply to all Reference Architectures produced by CNTT:

1. **Open source preference:** To ensure, by building on technology available in open source projects, that suppliers' and operators' investment have a tangible pathway towards a standard and production ready Cloud Infrastructure solution portfolio.
2. **Open APIs:** To enable interoperability and component substitution, and minimize integration efforts by using openly published API definitions.
3. **Separation of concerns:** To promote lifecycle independence of different architectural layers and modules (e.g. disaggregation of software from hardware).
4. **Automated lifecycle management:** To minimize costs of the end-to-end lifecycle, maintenance downtime (target zero downtime), avoid errors and discrepancies resulting from manual processes.
5. **Automated scalability:** To minimize costs and operational impacts through automated policy-driven scaling of workloads by enabling automated horizontal scalability of workloads.
6. **Automated closed loop assurance:** To minimize operational costs and simplify Cloud Infrastructure platform operations by using automated fault resolution and performance optimization.
7. **Cloud nativeness:** To optimise the utilization of resources and enable operational efficiencies.
8. **Security compliance:** To ensure the architecture follows the industry best security practices and is at all levels compliant to relevant security regulations.
9. **Resilience and Availability:** To allow High Availability and Resilience for hosted VNFs, and to avoid Single Point of Failure.

A.3 Scope

Within the framework of the common Telecom cloud infrastructure vision, there are four levels of documents needed to describe the components, realize the practical application of the systems and qualify the resulting cloud infrastructure. They are, as highlighted in Figure 34: **Reference Model**, **Reference Architecture**, **Reference Implementation**, and **Reference Conformance**.

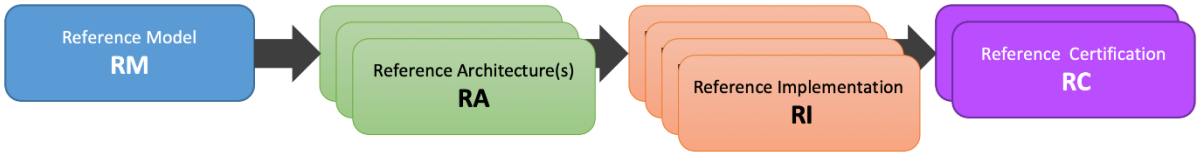

Figure 34: Documentation Scope of CNTT

A.3.1 Functional Scope

In terms of the functional scope of the CNTT documentation, in order to target the project goals as described above, we are focussed on:

- Functional capabilities of the cloud infrastructure and the infrastructure management
- Functional interfaces between infrastructure and infrastructure management
- Functional interfaces between workloads and workload management

Due to the close alignment with ETSI GR NFV 002 [3], those ETSI interfaces that are considered relevant (with notes where required) are included in the figure below.

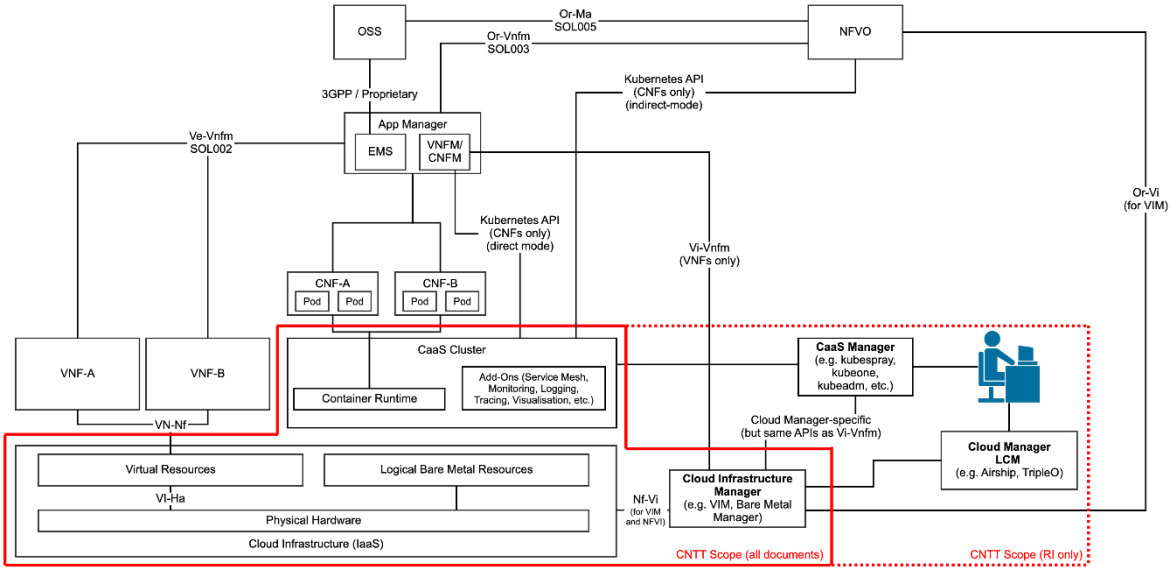

Figure 35: Functional Scope of CNTT

A.3.2 Out of Scope Components

While the nature of the CNTT might seem quite broad, the following areas are not at this time part of the scope of this effort.

- Hardware specifications: beyond the abstracted high-level CPU, memory, network interface and storage elements. The intention is to write the documents so they are general enough that any vendor hardware can be used in a conformant implementation without making significant changes to the model.
- Workload specifications: Other than the API interfaces when they directly need to touch the workloads themselves, the intention is to assume the workload application is a black box that the cloud infrastructure is providing resources to. The majority of interactions for lifecycle management of the workloads will be through the cloud infrastructure whenever possible.
- Lifecycle Management of the CaaS Clusters: whilst a complete NFV-MANO solution would need to provide lifecycle management for the Kubernetes clusters it is using to deploy its CNFs, the CNTT doesn't describe the NFVO and VNFM parts, and therefore the management of the cluster(s) is not in scope, while the VIM and the lifecycle management of containers (by Kubernetes) is in scope.

- Company specific requirements: The CNTT documents are designed to be general enough that most operators and others in the Open Source communities will be able to adapt and extend them to their own non-functional requirements.

A.3.3 Specification Types

- **Reference Model (RM)**: focuses on the **Infrastructure Abstraction** and how services and resources are exposed to VNFs/CNFs. It needs to be written at a high enough level that as new **Reference Architectures** and **Reference Implementations** are added, the model document should require few or no changes. Additionally, the Reference Model is intended to be neutral towards VMs or Containers.
- **Reference Architecture (RA)**: Reference Architectures defines all infrastructure components and properties which have effect on the VNF/CNF run time, deployment time, and design time. It is expected that at least one, but not more than a few Reference Architectures will be created, and they will conform to the Reference Model. The intention is, whenever possible, to use existing elements, rather than specify entirely new architectures in support of the high-level goals specified in the **Reference Model**.
- **Reference Implementation (RI)**: Builds on the requirements and specifications developed in RM, RAs and adds details so that it can be implemented. Each Reference Architecture is expected to be implemented by at least one Reference Implementation.
- **Reference Conformance (RC)**: Builds on the requirements and specifications developed in the other documents and adds details on how an implementation will be verified, tested and certified. Both infrastructure verification and conformance as well as VNFs/CNFs verifications and conformance will be covered.

Figure 36 below illustrates how each type of specifications relate to different element of a typical cloud platform stack.

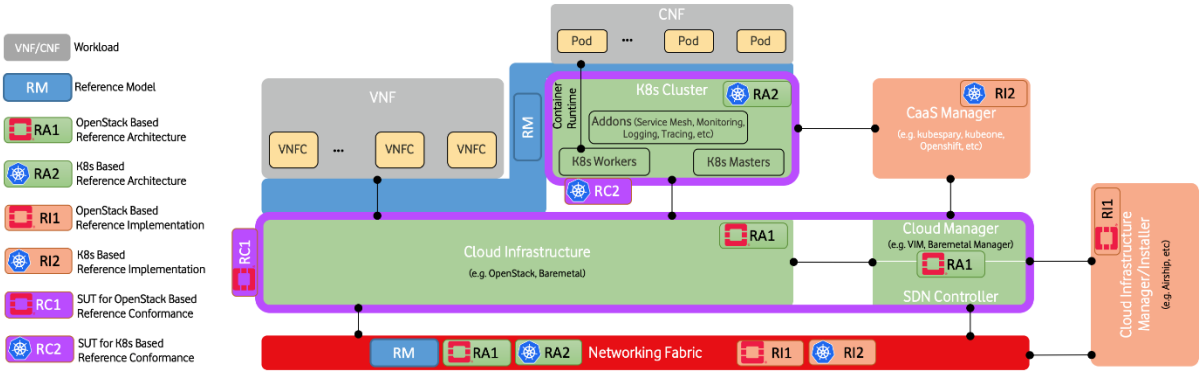

Figure 36: Documentation Scope of CNTT

The following diagram shows the different artefacts that will need to be created to support the implementation of the abstract concepts presented in the **Reference Model**, which are then applied to create the **Reference Architecture** that will be deployed using the requirements spelled out in the **Reference Implementation**.

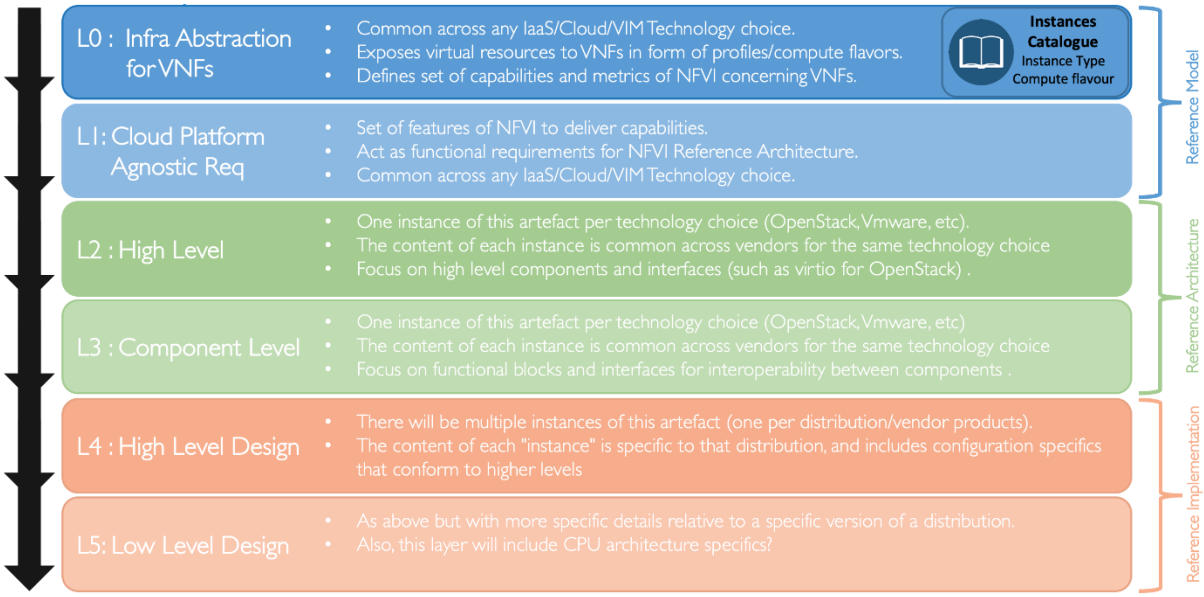

Figure 37: Description of the possible different levels of CNTT artefacts

A.3.4 Relationship to other industry projects

The CNTT work is not done in a vacuum. The intention from the beginning was to utilize the work from other Open Source and standards bodies within the industry. Some of the projects, but by no means all, that are related in some way to the CNTT efforts include:

- ETSI NFV ISG
- OpenStack
- OPNFV
- ONAP
- CNCF
- MEF
- TM Forum
- OSM (ETSI Open Source MANO project)
- ODIM (Open Distributed Infrastructure Management)
- VMware (While not an Open Source project, VMware is a commonly used platform used for VNF deployments in the telecom industry)

A.3.4.1 Relationship to ETSI-NFV

The ETSI NFV ISG is very closely related to the CNTT, in that it is a group that is working on supporting technologies for NFV applications (Figure 38 illustrates the scope of ETSI-NFV). To facilitate more collaboration as the project matures, the CNTT scope (Figure 35) purposely references certain ETSI NFV reference points, as specified by ETSI GR NFV 002[3].

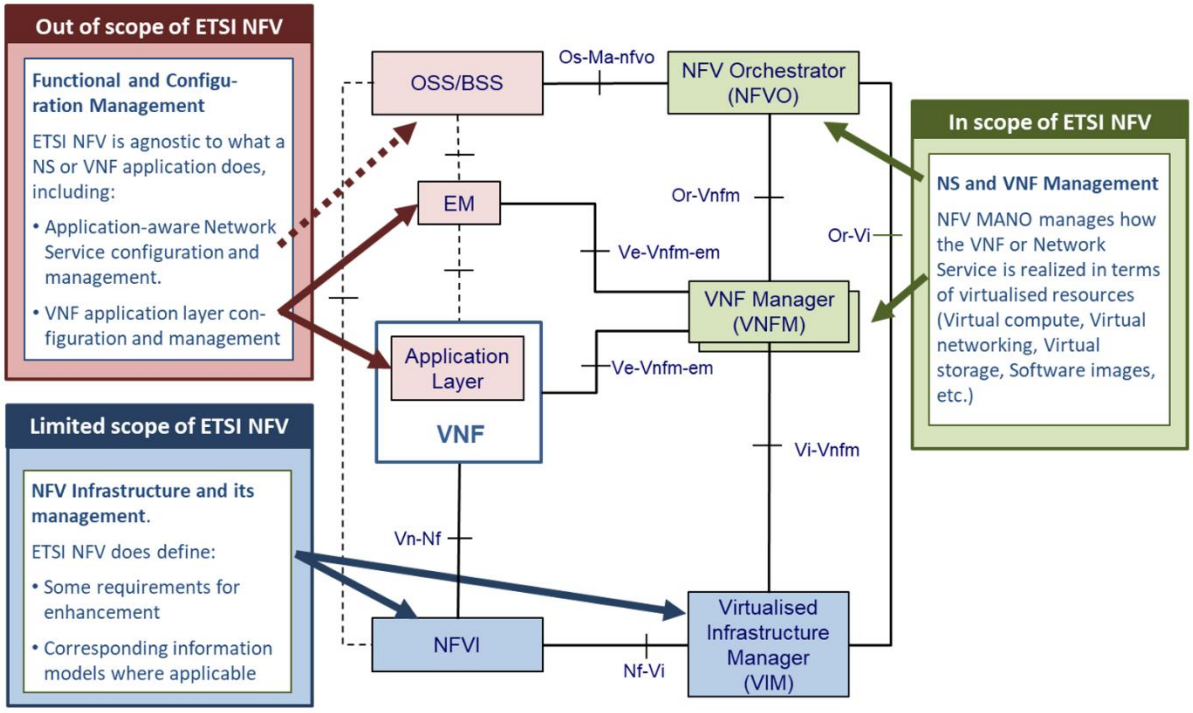

Figure 38: Scope ETSI NFV

A.3.4.2 Relationship to OPNFV and OVP

The CNTT is also closely aligned with OVP, an open source, community-led compliance and verification program that demonstrates the readiness and availability of commercial NFV products and services including **Vendor's Implementation (VI)** of cloud infrastructure and VNFs, using OPNFV. OVP combines open source-based automated compliance and verification testing for multiple parts of the NFV stack specifications established by ONAP, multiple SDOs such as ETSI and GSMA, and the LF Networking End User Advisory Group (EUAG).

Once the CNTT specifications are on place, OPNFV is expected to create a reference implementation that is adherent to requirements set in CNTT (**Reference Implementations**). Additionally, Commercial products adhering to CNTT specifications will be able to undergo an enhanced OVP's program (based on CNTT **Reference Conformance** specification) using OPNFV testing framework and tools. Figure 39 below illustrates the relationship with OPNFV and OVP in more details (specific to OpenStack based specifications)

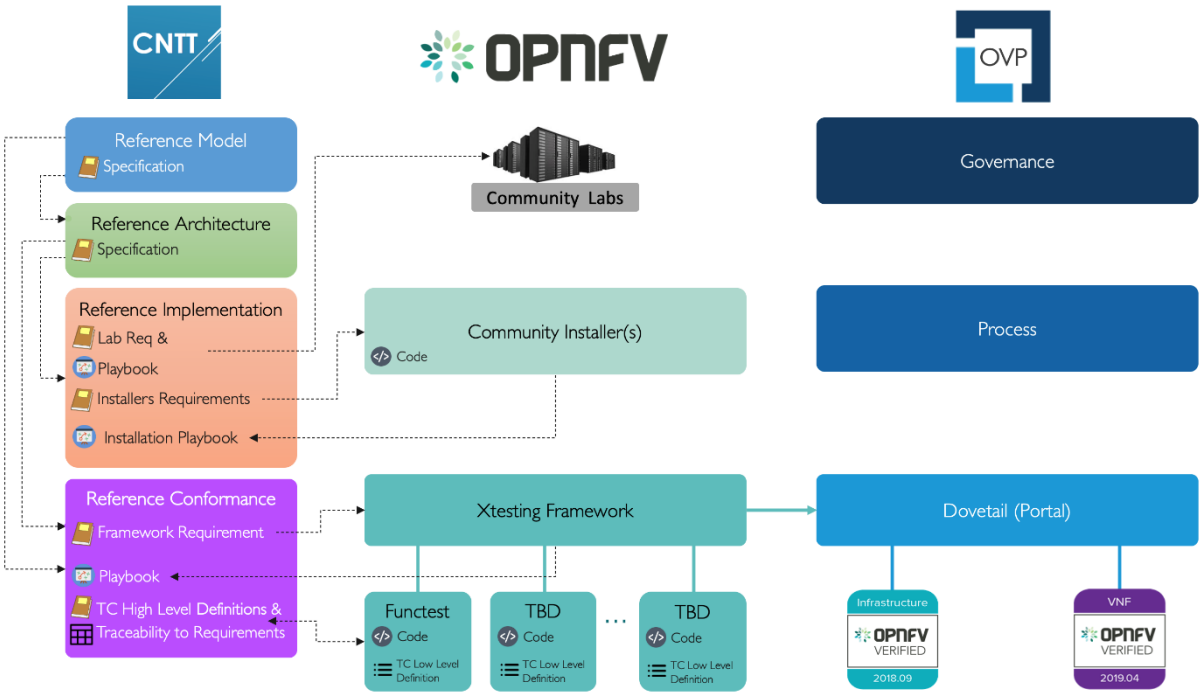

Figure 39: CNTT Relationship with OPNFV and OVP

As can be seen from the above figure, roles and responsibilities are as follows:

- CNTT specifies lab requirements in the **Reference Implementation** document which will be used by **OPNFV** to define what labs can be used within the community for the purpose of installing and testing CNTT conformant cloud infrastructure implementations.
- CNTT includes a lab Playbook in its **Reference Implementation** detailing available suitable labs to run and test cloud infrastructure implementations with access details and processes.
- CNTT specifies installer requirements in the **Reference Implementation** document will be used by **OPNFV** to determine a suitable installer to be used to install a Reference Implementation of cloud infrastructure that are conformant to CNTT specifications.
- CNTT includes an installation Playbook in its **Reference Implementation** detailing instructions of how to install an OPNFV reference implementation using community installers. The **Reference Implementation** created by **OPNFV** is expected to be fully conformant to CNTT specifications and must pass all test suites defined in the CNTT **Reference Conformance** document.
- CNTT specifies testing framework requirements in the **Reference Conformance** document will be used by OPNFV to determine a suitable testing framework and portals to be used for the purpose of running test suites and tools, and carry out badging processes.
- CNTT defines high level test cases in the **Reference Conformance** document that relates to requirements coming from both the **Reference Model** and **Reference Architecture** to be used by **OPNFV** to determine what testing projects within the community are suitable to deliver those tests and convert them into low level test cases that can be implemented within those **OPNFV** testing Projects.

- CNTT includes a traceability matrix in its **Reference Conformance** document detailing every test case (or group of test cases) available in the community and map them to the high level test case definition and the requirements they are fulfilling.
- CNTT includes a testing Playbook in its **Reference Conformance** document detailing instructions of how to run **OPNFV** testing framework and test cases against commercial NFV products (infrastructure and workload) to check conformance to CNTT specifications. CNTT testing Playbook will also details instructions of how to submit testing results for the **OVP** badging process.

A.3.4.3 Relationship to CNCF

A close relationship between CNTT and CNCF is maintained around the contents development for RA-2, RI-2, and RC-2.

A.3.4.4 Relationship to other communities

The CNTT collaborates with relevant API workgroups of SDOs (such as MEF, TM Forum, 3GPP, TIP, etc.) where applicable to align with their specification work and utilise their efforts.

Annex B Reference Model Glossary

B.1 Terminology

To help guide the reader, this glossary provides an introduction to the terminology used within this document. These definitions are, with a few exceptions, based on the ETSI GR NFV 003 V1.5.1 [1] definitions. In a few cases, they have been modified to avoid deployment technology dependencies only when it seems necessary to avoid confusion.

B.2 Software Layer Terminology

- **Cloud Infrastructure:** A generic term covering **NFVI**, **IaaS** and **CaaS** capabilities - essentially the infrastructure on which a **Workload** can be executed.

Note*:* **NFVI**, **IaaS** and **CaaS** layers can be built on top of each other. In case of CaaS some cloud infrastructure features (e.g.: HW management or multitenancy) are implemented by using an underlying **IaaS** layer.

- **Cloud Infrastructure Profile:** The combination of the Cloud Infrastructure Software Profile and the Cloud Infrastructure Hardware Profile that defines the capabilities and configuration of the Cloud Infrastructure resources available for the workloads.
- **Cloud Infrastructure Software Configuration:** a set of settings (Key:Value) that are applied/mapped to **cloud infrastructure** SW deployment.
- **Cloud Infrastructure Software Profile:** defines the behaviour, capabilities and metrics provided by a Cloud Infrastructure Software Layer on resources available for the workloads.
- **Cloud Native Network Function (CNF):** A cloud native network function (CNF) is a cloud native application that implements network functionality. A CNF consists of one or more microservices. All layers of a CNF is developed using Cloud Native Principles including immutable infrastructure, declarative APIs, and a “repeatable deployment process”.

Note: This definition is derived from the Cloud Native Thinking for Telecommunications Whitepaper (https://github.com/cncf/telecom-user-group/blob/master/whitepaper/cloud_native_thinking_for_telecommunications.md#1.4) which also includes further detail and examples.

- **Compute flavour:** defines the sizing of the virtualised resources (compute, memory, and storage) required to run a workload.

Note: used to define the configuration/capacity limit of a virtualised container.

- **Hypervisor:** a software that abstracts and isolates workloads with their own operating systems from the underlying physical resources. Also known as a virtual machine monitor (VMM).
- **Instance:** is a virtual compute resource, in a known state such as running or suspended, that can be used like a physical server.

Note: Can be used to specify VM Instance or Container Instance.

- **Network Function (NF):** functional block or application that has well-defined external interfaces and well-defined functional behaviour.
    - Within **NFV**, a **Network Function** is implemented in a form of **Virtualised NF** (VNF) or a **Cloud Native NF** (CNF).
- **Network Function Virtualisation (NFV):** The concept of separating network functions from the hardware they run on by using a virtual hardware abstraction layer.
- **Network Function Virtualisation Infrastructure (NFVI):** The totality of all hardware and software components used to build the environment in which a set of virtual applications (VAs) are deployed; also referred to as cloud infrastructure.

Note: The NFVI can span across many locations, e.g. places where data centres or edge nodes are operated. The network providing connectivity between these locations is regarded to be part of the cloud infrastructure. **NFVI** and **VNF** are the top-level conceptual entities in the scope of Network Function Virtualisation. All other components are sub-entities of these two main entities.

- **Network Service (NS):** composition of **Network Function**(s) and/or **Network Service**(s), defined by its functional and behavioural specification, including the service lifecycle.
- **Software Defined Storage (SDS):** An architecture which consists of the storage software that is independent from the underlying storage hardware. The storage access software provides data request interfaces (APIs) and the SDS controller software provides storage access services and networking.
- **Virtual Application (VA):** A general term for software which can be loaded into a Virtual Machine.

Note: a **VNF** is one type of VA.

- **Virtual CPU (vCPU):** Represents a portion of the host's computing resources allocated to a virtualised resource, for example, to a virtual machine or a container. One or more vCPUs can be assigned to a virtualised resource.
- **Virtual Machine (VM):** virtualised computation environment that behaves like a physical computer/server.

Note: A **VM** consists of all of the components (processor (CPU), memory, storage, interfaces/ports, etc.) of a physical computer/server. It is created using sizing information or Compute Flavour.

- **Virtual Network Function (VNF)**: a software implementation of a **Network Function**, capable of running on the **Cloud Infrastructure**.
    - **VNF**s are built from one or more VNF Components (**VNFC**) and, in most cases, the VNFC is hosted on a single VM or Container.
- **Virtual resources:**
    - **Virtual Compute resource (a.k.a. virtualisation container):** partition of a compute node that provides an isolated virtualised computation environment.
    - **Virtual Storage resource:** virtualised non-volatile storage allocated to a virtualised computation environment hosting a **VNFC**.
    - **Virtual Networking resource:** routes information among the network interfaces of a virtual compute resource and physical network interfaces, providing the necessary connectivity.
- **Workload:** an application (for example **VNF**, or **CNF**) that performs certain task(s) for the users. In the Cloud Infrastructure, these applications run on top of compute resources such as **VMs** or **Containers**. Most relevant workload categories in context of the CNTT Cloud Infrastructure are:
    - **Data Plane Workloads:** that perform tasks related to packet handling of the end-to-end communication between applications. These tasks are expected to be very I/O and memory read/write operations intensive.
    - **Control Plane Workloads:** that perform tasks related to any other communication between NFs that is not directly related to the end-to-end data communication between applications. For example, this category includes session management, routing or authentication.
    - **Storage Workloads:** that perform tasks related to disk storage (either SSD or HDD or other). Examples range from non-intensive router logging to more intensive database read/write operations.

B.3 Hardware Layer Terminology

- **Cloud Infrastructure Hardware Configuration:** a set of settings (Key:Value) that are applied/mapped to **Cloud Infrastructure** HW deployment.

- **Cloud Infrastructure Hardware Profile:** defines the behaviour, capabilities, configuration, and metrics provided by a cloud infrastructure hardware layer resources available for the workloads.
    - Host Profile: is another term for a Cloud Infrastructure Hardware Profile.
- **Hardware resources:** Compute/Storage/Network hardware resources on which the cloud infrastructure platform software, virtual machines and containers run on.
- **Physical Network Function (PNF):** Implementation of a network function via tightly coupled dedicated hardware and software system.

Note: This is a physical cloud infrastructure resource with the NF software.

- **Simultaneous Multithreading:** Simultaneous multithreading (SMT) is a technique for improving the overall efficiency of superscalar CPUs with hardware multithreading. SMT permits multiple independent threads of execution on a single core to better utilise the resources provided by modern processor architectures.

B.4 Operational and Administrative Terminology

- **Cloud service user:** Natural person, or entity acting on their behalf, associated with a cloud service customer that uses cloud services.

Note: Examples of such entities include devices and applications.

- **Compute Node:** An abstract definition of a server.

Note: A compute node can refer to a set of hardware and software that support the VMs or Containers running on it.

- **External Network:** External networks provide network connectivity for a cloud infrastructure tenant to resources outside of the tenant space.
- **Fluentd (https://www.fluentd.org/):** An open source data collector for unified logging layer, which allows data collection and consumption for better use and understanding of data. **Fluentd** is a CNCF graduated project.
- **Kibana:** An open source data visualisation system.
- **Multi-tenancy:** feature where physical, virtual or service resources are allocated in such a way that multiple tenants and their computations and data are isolated from and inaccessible by each other.
- **Prometheus:** An open-source monitoring and alerting system.
- **Quota:** An imposed upper limit on specific types of resources, usually used to prevent excessive resource consumption by a given consumer (tenant, VM, container).
- **Resource pool:** A logical grouping of cloud infrastructure hardware and software resources. A resource pool can be based on a certain resource type (for example, compute, storage and network) or a combination of resource types. A **Cloud Infrastructure** resource can be part of none, one or more resource pools.
- **Service Assurance (SA):** collects alarm and monitoring data. Applications within SA or interfacing with SA can then use this data for fault correlation, root cause analysis, service impact analysis, SLA management, security, monitoring and analytic, etc.

- **Tenant:** cloud service users sharing access to a set of physical and virtual resources, ITU (https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.3500-201408-I!!PDF-E&type=items).

Note: Tenants represent an independently manageable logical pool of compute, storage and network resources abstracted from physical hardware.

- **Tenant Instance:** refers to a single **Tenant**.
- **Tenant (Internal) Networks:** Virtual networks that are internal to **Tenant Instances**.

B.5 Container Related Terminology

Note: Relevant terms are added here from RA2. Most of these term definitions are taken from Kubernetes glossary (https://kubernetes.io/docs/reference/glossary) but in some cases should be made independent from Kubernetes as a specific container orchestration engine.

- **CaaS Manager:** A management plane function that manages the lifecycle (instantiation, scaling, healing, etc.) of one or more CaaS instances, including communication with VIM for master/node lifecycle management.
- **Container:** A lightweight and portable executable image that contains software and all of its dependencies.

Note: OCI defines **Container** as "An environment for executing processes with configurable isolation and resource limitations. For example, namespaces, resource limits, and mounts are all part of the container environment." A **Container** provides operating-system-level virtualisation by abstracting the “user space”. One big difference between **Containers** and **VMs** is that unlike VMs, where each **VM** is self-contained with all the operating systems components are within the **VM** package, containers "share" the host system’s kernel with other containers.

- **Container Engine:** Software components used to create, destroy, and manage containers on top of an operating system.
- **Container Image:** Stored instance of a container that holds a set of software needed to run an application.
- **Container Runtime:** The software that is responsible for running containers.

Note: as explained in OCI Glossary (https://github.com/opencontainers/runtime-spec/blob/master/glossary.md) it reads the configuration files for a **Container** from a directory structure, uses that information to create a container, launches a process inside the container, and performs other lifecycle actions.

- **Container-as-a-Service (CaaS)**: A complete set of technologies to enable the management of containerised software, including a Kubernetes cluster, container networking, storage, routing, service mesh, etc.

- **Kubernetes Cluster:** A set of machines, called nodes and master, that run containerised applications managed by Kubernetes. A cluster has at least one worker node and at least one master.

Note: adapted from Kubernetes Glossary (https://kubernetes.io/docs/reference/glossary/?all=true#term-cluster).

- **Kubernetes Control Plane:** The container orchestration layer that exposes the API and interfaces to define, deploy, and manage the lifecycle of containers.
- **Kubernetes Master:** Master(s) manage the worker nodes and the Pods in the cluster. The master may run on a **VM** or a physical machine. Multiple masters can be used to provide a cluster with failover and high availability.
- **Kubernetes Node:** A node is a worker machine in Kubernetes. A worker node may be a **VM** or physical machine, depending on the cluster. It has local daemons or services necessary to run Pods and is managed by the control plane.
- **Kubernetes Service:** An abstract way to expose an application running on a set of Pods as a network service.

Note: This definition from Kubernetes Glossary (https://kubernetes.io/docs/reference/glossary/?all=true#term-service) uses the term "network service" differently than in ETSI NFV.

- **Pod:** The smallest and simplest Kubernetes object. A Pod represents a set of running containers on your cluster. A Pod is typically set up to run a single primary container. It can also run optional sidecar containers that add supplementary features like logging.

B.6 OpenStack Related Terminology

Note: The official OpenStack Glossary (https://docs.openstack.org/image-guide/common/glossary.html) is an extensive list of OpenStack-related concepts. Some additional terms used in the Reference Architecture RA-1 or used to relate RA-1 terms with terms defined elsewhere.

- **Core (physical):** An independent computer processing unit that can independently execute CPU instructions and is integrated with other cores on a multiprocessor (chip, integrated circuit die). Please note that the multiprocessor chip is also referred to as a CPU that is placed in a socket of a computer motherboard.
- **Flavor Capability:** The capability of the Cloud Infrastructure Profile, such as CPU Pinning, NUMA or huge pages.
- **Flavor Geometry:** Flavor sizing such as number of vCPUs, RAM, disk, etc.
- **Hugepages:** Physical memory is partitioned and accessed using the basic page unit (in Linux default size of 4 KB). Hugepages, typically 2 MB and 1GB size, allows large amounts of memory to be utilised with reduced overhead. In an NFV environment, huge pages are critical to support large memory pool allocation for data packet buffers. This results in fewer Translation Lookaside Buffers (TLB) lookups, which reduces the virtual to physical pages address translations. Without huge pages enabled high TLB miss rates would occur thereby degrading performance.

B.7 Cloud Platform Abstraction Related Terminology:

- **Abstraction:** Process of removing concrete, fine-grained or lower level details or attributes or common properties in the study of systems to focus attention on topics of greater importance or general concepts. It can be the result of decoupling.
  Adapted from Wikipedia:Abstraction (https://en.wikipedia.org/wiki/Abstraction_(computer_science)), Wikipedia:Generalization(https://en.wikipedia.org/wiki/Generalization)
- **Appliance deployment model:** Application has tight coupling with underlying Platform even if the application is virtualized or containerized.
- **Application Control:** Any method or system of controlling applications (VNFs). Depending on RA and technologies used, this can be a VNF Manager or NFV Orchestrator provided as a VNF or Platform capability.
- **Cloud deployment model:** Applications are decoupled from the platform provided by Cloud operator.
- **Decomposition:** Decomposition (also known as factoring) is breaking a complex system into parts that are easier to program and maintain.
  Adapted from Wikipedia:Decomposition (https://en.wikipedia.org/wiki/Decomposition_(computer_science)
- **Decoupling, Loose Coupling:** Loosely coupled system is one in which each of its components has, or makes use of, little or no knowledge of the implementation details of other separate components. Loose coupling is the opposite of tight coupling.
  Adapted from Wikipedia:Loose Coupling https://en.wikipedia.org/wiki/Loose_coupling.
- **Encapsulation:** Restricting of direct access to some of an object's components.
  Adapted from Wikipedia:Encapsulation https://en.wikipedia.org/wiki/Encapsulation_(computer_programming)
- **Observability:** Observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.
  Adapted from Wikipedia:Observability https://en.wikipedia.org/wiki/Observability
- **Resilience:** Resilience is the ability to provide and maintain an acceptable level of service in the face of various faults and challenges to normal operation.
  Adapted from Wikipedia:Resilience https://en.wikipedia.org/wiki/Resilience_(network)

B.8 Other Referenced Terminology

- **Carrier Grade:** Carrier grade refers to network functions and infrastructure that are characterised by all or some of the following attributes: High reliability allowing near 100% uptime, typically measured as better than “five nines”; Quality of Service (QoS) allowing prioritization of traffic; High Performance optimized for low latency/packet loss, and high bandwidth; Scalability to handle demand growth by adding virtual and/or physical resources; Security to be able to withstand natural and man-made attacks.
- **Monitoring (Capability):** Monitoring capabilities are used for the passive observation of workload-specific traffic traversing the Cloud Infrastructure. Note, as with all capabilities, Monitoring may be unavailable or intentionally disabled for security reasons in a given cloud infrastructure instance.

- **NFV Orchestrator (NFVO):** Manages the VNF lifecycle and **Cloud Infrastructure** resources (supported by the **VIM**) to ensure an optimised allocation of the necessary resources and connectivity.
- **Platform:** A cloud capabilities type in which the cloud service user can deploy, manage and run customer-created or customer-acquired applications using one or more programming languages and one or more execution environments supported by the cloud service provider.
  Adapted from ITU https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.3500-201408-I!!PDF-E&type=items

Note: For CNTT, this includes the physical infrastructure, Operating Systems, virtualisation/containerisation software and other orchestration, security, monitoring/logging and life-cycle management software.

- **PM / Performance Measurement / Measurement:** The procedure or set of operations having the object of determining a Measured Value or Measurement Result. In this context, PMs reflect data generated and collected within the cloud infrastructure that reflects the performance of the infrastructure. For example, a count of frames or packets traversing an interface, memory usage information, other resource usage and availability, etc. These data may be instantaneous or accumulated, and made available (i.e. exposed) based on permissions and contexts (e.g., workload vs. infra).
- **PVP: Physical-Virtual-Physical:** PVP represents a workload test topology where a measurement is taken across two physical test points (e.g., physical NICs on a host), with traffic traversing a virtualized workload that is logically connected between the physical points. PVP is an ETSI term, defined in ETSI GS NFV-TST 009 [14].
- **Virtualised Infrastructure Manager (VIM):** Responsible for controlling and managing the **Network Function Virtualisation Infrastructure** compute, storage and network resources.

Annex C Document Management

C.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
1.0	11 Nov 2020	Initial version		Kozlowski, Walter/ Telstra

Other Information

Type	Description
Document Owner	Network Group
Editor / Company	Kozlowski, Walter / Telstra

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

comments or suggestions & questions are always welcome.