 O-RAN.WG11.O-CLOUD-Security-TR.0-R004-v07.00

Technical Report

O-RAN Work Group 11 (Security Work Group)

Study on Security for O-Cloud

Copyright © 2024 by the O-RAN ALLIANCE e.V.

The copying or incorporation into any other work of part or all of the material available in this specification in any form without the prior written permission of O-RAN ALLIANCE e.V. is prohibited, save that you may print or download extracts of the material of this specification for your personal use, or copy the material of this specification for the purpose of sending to individual third parties for their information provided that you acknowledge O-RAN ALLIANCE as the source of the material and that you inform the third party that these conditions apply to them and that they must comply with them.

O-RAN ALLIANCE e.V., Buschkauler Weg 27, 53347 Alfter, Germany

Register of Associations, Bonn VR 11238, VAT ID DE321720189

Contents

List of figures 4

List of tables 4

Foreword 5

Modal verbs terminology 5

Introduction 5

1 Scope 6

2 References 6

2.1 Informative references 6

3 Definition of terms, symbols and abbreviations 10

3.1 Terms 10

3.2 Symbols 11

3.3 Abbreviations 11

4 O-Cloud Architecture 14

4.1 Components 14

4.1.1 SMO 14

4.1.2 O-Cloud management components 15

4.1.3 Hardware resources 15

4.1.4 Operating System (OS) 15

4.1.5 Virtualization Layer 15

4.1.6 NFs Layer 15

4.1.7 O-Cloud images repository 15

4.2 Interfaces 16

4.3 Critical services 18

4.3.1 SERV#01 SMO-O-Cloud O2 services 18

4.3.2 SERV#02 O-Cloud images service 19

4.3.3 SERV#03 O-Cloud monitoring service 19

4.3.4 SERV#04 O-Cloud provisioning service 20

4.3.5 SERV#05 O-Cloud software management service 20

4.3.6 SERV#06 O-Cloud fault management service 20

4.3.7 SERV#07 O-Cloud performance service 21

4.3.8 VOID 21

4.3.9 SERV#009 O-Cloud Inventory 21

5 Cloud deployment scenarios 22

5.1 Main actors 22

5.2 Cloud service models 22

5.2.1 Infrastructure as a Service (IaaS) 22

5.2.2 Platform as a Service (PaaS) 23

5.2.3 Software as a Service (SaaS) 23

5.3 Cloud deployment types 24

5.3.1 Private cloud 24

5.3.2 Community Cloud 25

5.3.3 Public Cloud 27

5.3.4 Hybrid Cloud 27

5.4 High-Level risk assessment 29

6 Roles and responsibilities 34

7 Security Problem Definition 35

7.1 Assets 35

7.2 Threats 38

7.2.1 Threat and impact types 38

7.2.2 Attack surface 39

7.2.3 Vulnerabilities 40

7.2.4 Threat events 41

8 Recommendations and best practices 69

8.1 REC-CM Certificate management 70

8.2 REC-NS Network Segmentation & Filter Network Traffic 70

8.3 REC-IAM Identity, Authentication and Access Management 72

8.4 REC-VHPM Vulnerability Handling and Patch Management 75

8.5 REC-SCONF Security Configuration 75

8.6 REC-SDLC Secure Development Lifecycle 76

8.7 REC-SNFLC Security App/VNF/CNF lifecycle 77

8.8 REC-IMGP Image Protection 78

8.9 REC-LOG Logging, Monitoring and Alerting 80

8.10 REC-SB Secure Boot 81

8.11 REC-ISO Strong Isolation 81

8.12 REC-AUD Security Audit 83

8.13 REC-SS Secure Storage 84

8.14 REC-PHY Physical Security Protection 84

8.15 REC-RA Remote Attestation 86

8.16 REC-SDD Secure data deletion 86

8.17 REC-IDM O-Cloud ID secure management 87

8.18 REC-IDM Identity management for O-Cloud elements 88

8.19 REC-ADMC Admission Control 90

9 Risk Assessment 91

Annex A (informative): Best practices from some of existing main security guidance 92

A.1 CISA/NSA Kubernetes security hardening best practices 92

A.2 CIS Docker security best practices 96

A.3 ONAP VNFs security best practices 98

Annex: Change history/Change request (history) 105

# List of figures

**Figure 4‑1 : O-CLOUD architecture** 14

**Figure 4‑2 : AAL Architecture and interfaces (Source [i.52])** 18

**Figure 4‑3 : Parallel reporting & Alarm correlation [i.1]** 20

**Figure 5‑1 : IaaS cloud service model** 23

**Figure 5‑2 : PaaS cloud service model** 23

**Figure 5‑3 : SaaS cloud service model** 24

**Figure 5‑4 : On-site private cloud** 24

**Figure 5‑5 : Outsourced private cloud** 25

**Figure 5‑6 : On-site community cloud** 26

**Figure 5‑7 : Outsourced community cloud** 26

**Figure 5‑8 : Public cloud** 27

**Figure 5‑9 : Hybrid cloud** 28

**Figure 7‑1 : Cartography of assets** 36

**Figure 7‑2 : Attack vectors** 44

**Figure 7‑3 : Vulnerabilities within O-Cloud** 44

**Figure 7‑4 : Illustration of the VM/Container escape attack** 53

**Figure 7‑5 : Illustration of the migration flooding attack** 55

**Figure 7‑6 : Illustration of the false resource advertising attack** 55

**Figure 7‑7 : Illustration of the migration MITM attack** 55

**Figure 7‑8 : Illustration of the Theft-of-Service/DoS Attack** 57

**Figure 7‑9 : Illustration of the VM/Container hyperjacking attack** 61

**Figure 7‑10 : Illustration of a cross VM/Container side channel attack** 65

# List of tables

[**Table 4‑1 : O-Cloud interfaces** 19](#_Toc148307578)

[**Table 5‑1 : High level security risk assessment of cloud deployment models** 34](#_Toc148307579)

[**Table 6‑1 : List of Users** 36](#_Toc148307580)

[**Table 7‑1 : List of Assets** 42](#_Toc148307581)

# Foreword

This Technical Report (TR) has been produced by the O-RAN Alliance.

# Modal verbs terminology

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

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

# Introduction

This technical report provides the threat model for the O-Cloud. The report identifies threats, security requirements and recommends potential security controls to protect against identified threats.

# Scope

The steps of the threat modelling process are as follows:

1. Identify assets: Identify the valuable assets that the O-Cloud must protect.
2. Identify the threats: Identify the threats that could affect O-Cloud
3. Document the threats: Document each threat using a common threat template that defines a core set of attributes to capture for each threat.
4. Rate the threats: Rate the threats to prioritize and address the most significant threats first. The rating process weighs the probability of the threat against damage that could result should an attack occur.
5. Define potential mitigations to counter the identified threats and reduce their risks.

The structure of the document is divided into clauses and annexure(s):

* Clause 4 describes the O-Cloud architecture in terms of components, interfaces, and critical services.
* Clause 5 explores the different deployment types and models. In addition, it provides a high-level risk assessment of those cloud deployment models.
* Clause 6 highlights the main actors of the O-Cloud in terms of roles and responsibilities.
* Clause 7 outlines the security problem definition in terms of assets, attack vectors, vulnerabilities and threats.
* Clause 8 defines recommendations and best practices to mitigate the identified threats.
* Clause 9 figures out the risk assessment of the identified threats in terms of impact and likelihood.
* Annex A provides best practices from some of the main security guidance on cloud, virtualization and containerization.

# References

## Informative references

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

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

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

1. O-RAN ALLIANCE TS: “O-RAN O2 Interface General Aspects and Principles”
2. O-RAN ALLIANCE TS: “O-RAN Cloudification and Orchestration Use Cases and Requirements for O-RAN Virtualized RAN”
3. CISA/NSA - Kubernetes Hardening Guidance – August 2021

<https://media.defense.gov/2021/Aug/03/2002820425/-1/-1/1/CTR_KUBERNETESHARDENINGGUIDANCE.PDF>

1. MITRE ATT&CK containers matrix

<https://attack.mitre.org/matrices/enterprise/containers/>

1. ENISA Threat Landscape for 5G Networks: Threat assessment for the fifth generation of mobile telecommunications networks (5G); November 2019
2. ETSI GS NFV-SEC 025 "work in progress." Network Functions Virtualisation (NFV) Release 4; Security; Secure End-to-End VNF and NS management specification
3. O-RAN ALLIANCE TS: “O-RAN O2dms Interface Specification”
4. O-RAN ALLIANCE TS: “O-RAN O2ims Interface Specification”
5. IETF RFC 3647: "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework"
6. ETSI GR NFV-SEC 005 Network Functions Virtualisation (NFV); Trust; Report on Certificate Management
7. IETF RFC 6749: "The OAuth 2.0 Authorization Framework".
8. ETSI GS NFV-SEC 022 Network Functions Virtualisation (NFV) Release 2; Security; Access Token Specification for API Access
9. ETSI GR NFV-SEC 018 Network Functions Virtualisation (NFV); Security; Report on NFV Remote Attestation Architecture
10. Ludovic Jacquin, Antonio Lioy, Diego R. Lopez, Adrian L. Shaw, and Tao Su “The trust problem in modern network infrastructures”

<https://security.polito.it/doc/public/trust_modern_network_2015.pdf>

1. ETSI GR NFV-SEC 007 Network Functions Virtualisation (NFV); Trust; Report on Attestation Technologies and Practices for Secure Deployments
2. 3GPP TR 33.848 “Study on security impacts of virtualisation”
3. OWASP Container Security Verification Standard

<https://owasp.org/www-project-container-security-verification-standard/migrated_content>

1. CIS Docker Benchmark Securing Docker

<https://www.cisecurity.org/benchmark/docker/>

1. NIST Special Publication 800-190 Application Container Security Guide
2. OWASP Docker Security Cheat Sheet

<https://cheatsheetseries.owasp.org/cheatsheets/Docker_Security_Cheat_Sheet.html>

1. OWASP Kubernetes Security Cheat Sheet

<https://cheatsheetseries.owasp.org/cheatsheets/Kubernetes_Security_Cheat_Sheet.html>

1. NIST SP 800-145: The NIST Definition of Cloud Computing
2. NIST SP 500-322: Evaluation of Cloud Computing Services Based on NIST 800-145
3. ONAP VNF Development Requirements – VNF Security

<https://docs.onap.org/projects/onap-vnfrqts-requirements/en/latest/Chapter4/Security.html>

1. GSMA NG.126 - Cloud Infrastructure Reference Model

<https://www.gsma.com/newsroom/wp-content/uploads//NG.126-v1.0-2.pdf>

1. Aqua Top 20 Docker Security Best Practices: Ultimate Guide

<https://blog.aquasec.com/docker-security-best-practices>

1. “Security Impacts of Virtualization on a Network Testbed”, Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference

<https://www.researchgate.net/publication/261059755_Security_Impacts_of_Virtualization_on_a_Network_Testbed>

1. Beniel Dennyson W, Dr. S. Prabakaran “Detecting Hyperjacking in cloud based virtual environment”

<https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwi92Zeznun0AhULHxoKHXO6AkgQFnoECAgQAQ&url=http%3A%2F%2Fsersc.org%2Fjournals%2Findex.php%2FIJAST%2Farticle%2Fdownload%2F15440%2F7789%2F&usg=AOvVaw3THwuT_S_WyKr6miOiJ7GS>

1. Muhammad Kazim “Security Aspects of Virtualization in Cloud Computing”

<https://www.researchgate.net/publication/273950406_Security_Aspects_of_Virtualization_in_Cloud_Computing>

1. Anita Choudhary “ A critical survey of live virtual machine migration techniques “

<https://journalofcloudcomputing.springeropen.com/track/pdf/10.1186/s13677-017-0092-1.pdf>

1. Svetlana Kolesnikova, Roman Kulikov, Yuriy Gatchin, Daniil Melnik “Hypervisor Security Analyses Based on Ishikawa Methodology”

<https://www.researchgate.net/publication/323838205_Hypervisor_Security_Analyses_Based_on_Ishikawa_Methodology?enrichId=rgreq-bf37d66002988b1ad9ef24fb09198313-XXX&enrichSource=Y292ZXJQYWdlOzMyMzgzODIwNTtBUzo2MDg0MzY3NTI0ODIzMDVAMTUyMjA3NDAzNDM4NQ%3D%3D&el=1_x_3&_esc=publicationCoverPdf>

1. Mehiar Dabbagh, Ammar Rayes “Internet of Things Security and Privacy”

<https://www.researchgate.net/publication/309375790_Internet_of_Things_Security_and_Privacy?enrichId=rgreq-69e8c33eb9fc28fccec64fbdad0a91d0-XXX&enrichSource=Y292ZXJQYWdlOzMwOTM3NTc5MDtBUzo1NTYxNDY2NDI2OTAwNDhAMTUwOTYwNzEwMDM0OA%3D%3D&el=1_x_3&_esc=publicationCoverPdf>

1. Changwei Liu, Anoop Singhal, Duminda Wijesekera “A Layered Graphical Model for Cloud Forensic Mission Attack Impact Analysis”

<https://www.researchgate.net/publication/327314423_A_Layered_Graphical_Model_for_Cloud_Forensic_Mission_Attack_Impact_Analysis_14th_IFIP_WG_119_International_Conference_New_Delhi_India_January_3-5_2018_Revised_Selected_Papers?enrichId=rgreq-b9ce99f09429d1a36256442e04006d0f-XXX&enrichSource=Y292ZXJQYWdlOzMyNzMxNDQyMztBUzo3NTM0MDM2Mzg1ODMzMDFAMTU1NjYzNjgzMzI4Nw%3D%3D&el=1_x_3&_esc=publicationCoverPdf>

1. Docker Best practices for scanning images

<https://docs.docker.com/develop/scan-images/>

1. Shankar Lal, Tarik Taleb, and Ashutosh Dutta “NFV: Security Threats and Best Practices”

<http://anastacia-h2020.eu/publications/NFV_Security_Threats_and_Best_Practices.pdf>

1. NSA/CISA Mitigating Cloud Vulnerabilities

<https://media.defense.gov/2020/Jan/22/2002237484/-1/-1/0/CSI-MITIGATING-CLOUD-VULNERABILITIES_20200121.PDF>

1. Fraunhofer AISEC report: Threat analysis of container-as-a-service for network function virtualization

<https://www.aisec.fraunhofer.de/content/dam/aisec/Dokumente/Publikationen/Studien_TechReports/englisch/caas_threat_analysis_wp.pdf>

1. CSA. The treacherous 12: Cloud computing top threats in 2016. Technical report, Cloud Security Alliance, 2016. 12, 14, 15, 16, 17, 27
2. ETSI GS NFV-SOL 013 Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; Specification of common aspects for RESTful NFV MANO APIs
3. CISA Cloud Security Technical Reference Architecture

<https://www.cisa.gov/sites/default/files/publications/CISA%20Cloud%20Security%20Technical%20Reference%20Architecture_Version%201.pdf>

1. Guide ANSSI - Hardware security requirements for x86 platforms

<https://www.ssi.gouv.fr/uploads/2019/11/anssi-guide-hardware_security_requirements.pdf>

1. O-RAN ALLIANCE TS: “O-RAN Security Requirements and Controls Specifications”
2. O-RAN ALLIANCE TS: “O-RAN Operations and Maintenance Architecture”
3. NSA/CISA Security Guidance for 5G Cloud Infrastructures

<https://www.cisa.gov/news/2021/10/28/nsa-and-cisa-provide-cybersecurity-guidance-5g-cloud-infrastructures>

1. O-RAN ALLIANCE TS: “O-RAN Security Protocols Specifications”
2. ETSI GS NFV-SOL 004 Network Functions Virtualisation (NFV) Release 3; Protocols and Data Models; VNF Package and PNFD Archive specification
3. ETSI GS NFV-SEC 021 Network Functions Virtualisation (NFV) Release 2; Security; VNF Package Security Specification
4. NIST SP 800-88 ”Guidelines for Media Sanitization”

<https://csrc.nist.gov/publications/detail/sp/800-88/rev-1/final>

1. DoD 5220.22-M wiping standard
2. O-RAN ALLIANCE TR: “O-RAN Security Threat Modeling and Risk Assessment”
3. O-RAN ALLIANCE TS: “O-RAN Acceleration Abstraction Layer Common API”
4. O-RAN ALLIANCE TS: “O-RAN Acceleration Abstraction Layer General Aspects and Principles”
5. Solving the Bottom Turtle - a SPIFFE Way to Establish Trust in Your Infrastructure via Universal Identity

<https://spiffe.io/pdf/Solving-the-bottom-turtle-SPIFFE-SPIRE-Book.pdf>

1. SPIRE Concepts: An overview of SPIRE’s architecture and fundamentals

<https://spiffe.io/docs/latest/spire-about/spire-concepts/>

1. Kubernetes Documentation: PKI certificates and requirements

[<https://kubernetes.io/docs/setup/best-practices/certificates/>](https://kubernetes.io/docs/setup/best-practices/certificates/)

1. Agent plugin: WorkloadAttestor "k8s"

<https://github.com/spiffe/spire/blob/v1.7.0/doc/plugin_agent_workloadattestor_k8s.md>

1. [SPIFFE-ID](file:///C:\Users\krishna.adharapurapu\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\CHB34F6Y\SPIFFE-ID)

<https://github.com/spiffe/spiffe/blob/master/standards/SPIFFE-ID.md>

1. NIST Special Publication 800-207A

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207A.pdf>

1. CWE-524, "Use of Cache Containing Sensitive Information"

<https://cwe.mitre.org/data/definitions/524.html>

1. CAPEC-204, “Lifting Sensitive Data Embedded in Cache”

<https://capec.mitre.org/data/definitions/204.html>

1. Infiltrating Corporate Intranet Like NSA

<https://i.blackhat.com/USA-19/Wednesday/us-19-Tsai-Infiltrating-Corporate-Intranet-Like-NSA.pdf>

1. O-RAN ALLIANCE TS: “O2 Interface General Aspects and Principles”
2. O-RAN ALLIANCE TS: “O-RAN Cloud Architecture and Deployment Scenarios for O-RAN Virtualized RAN”
3. NIST glossary: <https://csrc.nist.gov/glossary/term/sensitive>
4. O-RAN ALLIANCE TS: “O2dms Interface Specification: Kubernetes Native API Profile for Containerized NFs”

# Definition of terms, symbols and abbreviations

## Terms

For the purposes of the present document, the terms and definitions given in 3GPP TR 21.905 and the following apply:

**A1**: Interface between non-RT RIC and Near-RT RIC to enable policy-driven guidance of Near-RT RIC applications/functions, and support AI/ML workflow.

**A1 Enrichment information:** Information utilized by near-RT RIC that is collected or derived at SMO/non-RT RIC either from non-network data sources or from network functions themselves.

**A1 policy:** Type of declarative policies expressed using formal statements that enable the non-RT RIC function in the SMO to guide the near-RT RIC function, and hence the RAN, towards better fulfilment of the RAN intent.

**Deployment ID**: Correlation Identity created by the O-Cloud for the SMO to relate to its inventory and manage.

**E2**: Interface connecting the Near-RT RIC and one or more O-CU-CPs, one or more O-CU-UPs, and one or more O-DUs.

**E2 Node**: A logical node terminating E2 interface. In this version of the specification, O-RAN nodes terminating E2 interface are:

- for NR access: O-CU-CP, O-CU-UP, O-DU or any combination;

- for E-UTRA access: O-eNB.

**FCAPS:** Fault, Configuration, Accounting, Performance, Security.

**Intents**: A declarative policy to steer or guide the behavior of RAN functions, allowing the RAN function to calculate the optimal result to achieve stated objective.

**Near-RT RIC:** O-RAN near-real-time RAN Intelligent Controller: a logical function that enables real-time control and optimization of RAN elements and resources via fine-grained data collection and actions over E2 interface.

**NF Deployment**: An O-Cloud NF Deployment is a deployment of a cloud native Network Function (all or partial), resources shared within a NF Function, or resources shared across network functions. The NF Deployment configures and assembles user-plane resources required for the cloud native construct used to establish the NF Deployment and manage its life cycle from creation to deletion (Refer to clause 1.3.1 O-RAN WG6.GA&P [i.1]).

**NF Deployment Descriptor**: A completed data model which provides an O-Cloud the necessary information to create a deployment.

**Non-RT RIC:** O-RAN non-real-time RAN Intelligent Controller: a logical function that enables non-real-time control and optimization of RAN elements and resources, AI/ML workflow including model training and updates, and policy-based guidance of applications/features in Near-RT RIC.

**O-Cloud**:O-RAN cloud is a distributed cloud composed of Cloud Resource Pools where each pool is a collection of Cloud Resources within a Cloud Site with a specific location. [i.62]

**O-Cloud instance ID:** The O-Cloud instance ID is a unique identifier assigned to components within the O-Cloud platform, including VMs, pods, containers, nodes, and compute pools (i.e., a cluster in Kubernetes). This ensures uniqueness across the entire O-Cloud environment, irrespective of the component type. For instance, a VM, a pod, a container, a node, and a cluster will each have a distinct O-Cloud instance ID within the platform, ensuring that there is no ambiguity in identification.

**O-Cloud Node**: An O-Cloud Node is exposed over O2ims as an Abstracted Resource. It is a collection of CPUs, Mem, Storage, NICs, Accelerators, BIOSes, BMCs, etc., and can be thought of as a server (Refer to clause 3.1 in O-RAN WG6.GA&P [i.1]).

**O-Cloud Resource**: Refer to clause 3.4.3.2.1 in O-RAN WG6.GA&P [i.1]

**O-Cloud Software**: The O-Cloud software refers to a collection of Host OS, Hypervisor/Container engine, AAL (Acceleration Abstraction Layer), IMS, DMS, HAM (Hardware Accelerator Manager), and supporting software in O-Cloud to run VNFs/CNFs. [i.63]

**O-CU:** O-RAN Central Unit: a logical node hosting O-CU-CP and O-CU-UP

**O-CU-CP**: O-RAN Central Unit – Control Plane: a logical node hosting the RRC and the control plane part of the PDCP protocol.

**O-CU-UP**: O-RAN Central Unit – User Plane: a logical node hosting the user plane part of the PDCP protocol and the SDAP protocol.

**O-DU**: O-RAN Distributed Unit: a logical node hosting RLC/MAC/High-PHY layers based on a lower layer functional split.

**O-RU**: O-RAN Radio Unit: a logical node hosting Low-PHY layer and RF processing based on a lower layer functional split. This is similar to 3GPP’s “TRP” or “RRH” but more specific in including the Low-PHY layer (FFT/iFFT, PRACH extraction).

**O1:** Interface between management entities (NMS/EMS/MANO) and O-RAN managed elements, for operation and management, by which FCAPS management, Software management, File management shall be achieved.

**RAN**: Generally referred as Radio Access Network. In terms of this document, any component below Near-RT RIC per O-RAN architecture, including O-CU/O-DU/O-RU.

**Sensitive**: A descriptor of information whose loss, misuse, or unauthorized access or modification could adversely affect security [i.64].

**Service Provider**: A network provider who is planning to deploy applications into their network.

**Solution Provider**: An application developer who delivers applications to Service Providers.

**Workload** - Refer to clause 3.4.3.1 in O-RAN WG6.GA&P [i.1]

**NOTE:** For the purposes of the present document, the AAL terms and definitions given in [i.52] apply.

## Symbols

void

## Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 and the following apply:

AAL Acceleration Abstraction Layer

AALI Acceleration Abstraction Layer Interface

AALI-C Acceleration Abstraction Layer Interface-Common

AALI-C-App Acceleration Abstraction Layer Interface-Common-Application

AALI-C-Mgmt Acceleration Abstraction Layer Interface-Common-Management

AALI-P Acceleration Abstraction Layer Interface-Profile

AI/ML Artificial Intelligence/Machine Learning

CNF Cloud-native Network Function (NOTE: same as Containerized Network Function or Cloudified Network Function)

DMS O-Cloud Deployment Management Services

eNB eNodeB (applies to LTE)

FOCOM Federated O-Cloud Orchestration & Management

FTP File Transfer Protocol

FTPS File Transfer Protocol Secure

gNB gNodeB (applies to NR)

IMS O-Cloud Infrastructure Management Services

IPSEC Internet Protocol Security

KPI Key Performance Indicator

KQI Key Quality Indicator

LBT Listen Before Talk

LCM Life Cycle Management

LLS Lower Layer Split

MIMO Multiple Input, Multiple Output

MNO Mobile Network Operator

NETCONF Network Configuration Protocol

NF Network Function

NFO Network Function Orchestration

NFV Network Function Virtualisation

NFVI Network Function Virtualization Infrastructure

O-CU O-RAN Central Unit as defined by O-RAN ALLIANCE

O-CU-CP O-CU Control Plane

O-CU-UP O-CU User Plane

O-DU O-RAN Distributed Unit (uses Lower-level Split)

O-RU O-RAN Radio Unit

PDCP Packet Data Convergence Protocol

PNF Physical Network Function

PRB Physical Resource Block

PTP Precision Timing Protocol

QoE Quality of Experience

RAN Radio Access Network

RASP Runtime Application Self-Protection

RBAC Role-based Access Control

RIC O-RAN RAN Intelligent Controller

SDN Software Defined Network

SIEM Security information and event management

SINR Signal-to-Interference-plus-Noise Ratio

SMOService Management and Orchestration

SSH Secure Shell

TLS Transport Layer Security

UAV Unmanned Aerial Vehicle

V2X Vehicle to Everything

VM Virtual machine

VNF Virtualised Network Function

# O-Cloud Architecture

The O-Cloud architecture is depicted in the following figure:



**Figure 4‑1 : O-CLOUD architecture**

## Components

The following clauses describe the functional blocks identified in Figure 4-1 [i.1], [i.2]:

### SMO

The SMO components managing and orchestrating the O-Cloud software are:

* **Federated O-Cloud Orchestration and Management (FOCOM)**: The FOCOM is responsible for accounting and asset management of the resources in the cloud. The FOCOM is the primary consumer of services provided by the IMS. The FOCOM has information about the O-Cloud resources management. Specifically, the FOCOM needs to know whether the services are within the operator domain or external.
* **Network Function Orchestrator (NFO)**: The NFO is responsible for orchestrating the assembly of the network functions as a composition of NF Deployments in the O-Cloud. It may also utilize OAM Functions in order to access the O1 interface to the NF once it is deployed. Its use of the O1 is not germane to the O2 and is only mentioned here for completeness. The NFO is the primary consumer of the DMS.

### O-Cloud management components

The O-Cloud components providing management services for consumption of SMO are:

* **Infrastructure Management Services (IMS)**: The IMS is responsible for management of the O-Cloud resources and the software which is used to manage those resources. The IMS generally provides services for consumption by the FOCOM.
* **Deployment Management Services (DMS)**: The DMS is responsible for management of NF Deployments into the O-Cloud. It provides the ability to instantiate, monitor, and terminate NF Deployments. The DMS generally provides services for consumption by the NFO.

### Hardware resources

* **A Computer System** is defined to be a physical or composed system capable of performing computations that is Underlay-Network connected. A Computer System can run any major Operating System with or without Virtualization and/or Container support functionality.

NOTE: A computer system in the context of a cloudified network and usage in cluster requires a network connectivity. For example, a server in a data center connected to an underlay network.

* **An underlay network** is a physically connected network enabling Computer Systems to communicate with each other and with the gateway(s) connected to networks outside of the data center.
* **Hardware accelerator manager:** It is an acceleration management function, that provides management capabilities for the HW Accelerator(s) in the O-Cloud Node. Management capabilities include but not limited to lifecycle management, configuration, updates/upgrades and failure handling. Hardware Accelerators include ASIC, FPGA, DSP and GPU.

### Operating System (OS)

An Operating system is a software platform that manages and abstracts the Computer System hardware and software resources as well as provides common services for NFs such as scheduling and network connectivity.

### Virtualization Layer

* **Hypervisor (VMs)**: An hypervisor is an OS that includes the ability to offer multiple Virtual Machines, each acting as a well-separated Computer System.
* **Container Engine**: A Container Engine is an OS that include the ability to offer multiple separated name spaces, quotas, and management for Containers.

### NFs Layer

The following are three deployment scenarios of a NF:

1. **Bare Metal Container Cluster –** A Bare Metal Container Clusteris a set of network-connected computer systems with their individual operating system instances that supports containers in a cluster configuration.
2. **VM-based Container Cluster –** A VM-based Container Clusteris a set of network-connected Virtual Machines with their individual guest operating system instance that supports containers in a clustered configuration.
3. **VM Cluster –** A VM Cluster is a set of network-connected Computer Systems with their individual operating instance that supports virtual machines in a cluster configuration.

### O-Cloud images repository

The repository containing the Software Images of O-RAN Network Functions.

## Interfaces

O-Cloud interfaces are illustrated in the following table:

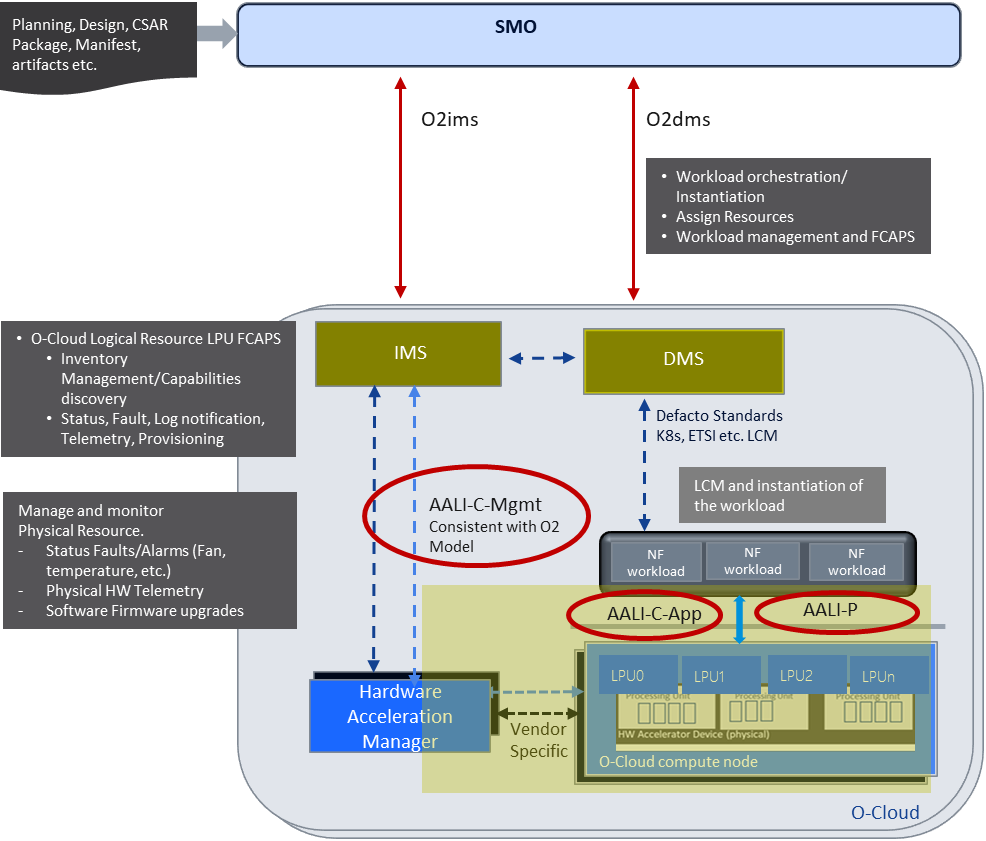
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Interfaces** | **Interface between** | **Exposure** | **Description** | **Security protection** |
| O2dms | SMO and O-Cloud Deployment Management Services (DMS) | External | It is responsible for the deployment management services of the NF Deployment. It offers the following services APIs [i.7]:   * O2dms\_DeploymentInventory Services * O2dms\_DeploymentProvisioning Services * O2dms\_DeploymentFault Services * O2dms\_DeploymentPerformance Services * O2dms\_DeploymentLifecycle Services | TLS 1.2/1.3 |
| O2ims | SMO and O-Cloud Infrastructure Management Services (IMS) | External | It is responsible for infrastructure management services for O-Cloud Nodes. It offers the following services APIs [i.8]:   * O2ims\_InfrastuctureInventory Services: Service for querying the O-Cloud resources and management services. * O2ims\_InfrastructureMonitoring Services: Service for configuring telemetry reporting of O-Cloud infrastructure resources. * O2ims\_InfrastructureProvisioning Services: Service for configuring the O-Cloud infrastructure resources and management services. * O2ims\_InfrastructureSoftwareManagement Services: Services for software inventory and updating the software used for O-Cloud infrastructure resources and management services. * O2ims\_InfrastructureLifecycleManagement Services: Services related to O-Cloud infrastructure lifecycle management and events. | TLS 1.2/1.3 |
| O1 | SMO and the RAN managed functions | External | SMO performs relevant configuration updates in RAN over O1 interface for each network function.  SMO on operator request can perform recover (reset) of an application (in case of failures) within a NF through the O1 interface.  SMO is notified of configuration events, fault events and performance measurements from an O-RAN NF instantiated on an O-Cloud, using the O1 interface.  SMO retrieves configuration data from NFs through O1. | TLS 1.2/1.3 |
| O-Cloud APIs | O-Cloud infrastructure and VNFs/CNFs | Internal | The interface is used to specify interactions between the VNFs/CNFs and O-Cloud. The interfaces can be used by the VNFs/CNFs to register/deregister for receiving events from the O-Cloud. | HTTPS (HTTP over TLS) |
| Management (Tenant level APIs) | External interfaces are use case specific interfaces that are not standardized in O-RAN. Security of these interfaces is outside the scope of this work item and recommended to be outside the scope of the O-RAN Alliance. | External | APIs for the management (creation, deletion, and operation) of the Tenant, software flavours, Operating System and workload images, Identity and Authorization, virtual resources, security, and the workload application. | HTTPS |
| Enabler Service Interfaces | External interfaces are use case specific interfaces that are not standardized in O-RAN. Security of these interfaces is outside the scope of this work item and recommended to be outside the scope of the O-RAN Alliance. | External | An operational O-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 to distribute incoming requests amongst a pool of designated resources. | HTTPS |

**Table 4‑1 : O-Cloud interfaces**

AAL interfaces are as follows [i.51]:

* AALI-C-Mgmt: between the O-Cloud IMS/DMS and the Hardware Accelerator Manager. It is consistent with O2 interface.
* AALI-C-App: common APIs between O-RAN NFs and the hardware accelerator device for initial discovery, AAL initialization, runtime operations, real time telemetry and status, etc.
* AALI-P: Profile specific API – fine grain control over selection and off-loading, etc. The AALI configuration and management APIs are the APIs that an application (O-DU) executes to configure and manage the AAL-LPU(s) that have been allocated to the application by the O-Cloud.
* Vendor specific interface between the hardware accelerator manager and hardware accelerator device.

The transport between a NF and an AAL implementation can be of different types (e.g., based on shared memory, PCIe interconnect, over Ethernet). AALI-C-App shall support abstraction of these various transport mechanisms between a NF and an AAL implementation through a set of common transport APIs, constituting a transport abstraction framework.



**Figure 4‑2 : AAL Architecture and interfaces (Source [i.52])**

## Critical services

The following main critical services are provided by the O-Cloud components [i.1], [i.2] which need to be highly protected and securely maintained. They are identified in this report to assess the negative impact that the identified threats may cause on them.

### SERV#01 SMO-O-Cloud O2 services

Using the O2 interface, the SMO can perform the following operations:

* Provide a boot image for a remote node
* Send a cloud descriptor, (i.e., cloud deployment and configuration files) for initial cloud startup
* Query the O-Cloud for attributes such as SW inventory
* Send a request to O-Cloud to download software sent to it from the SMO, such as request for download of an xAPP deployment.
* Subscribe to notification of configuration events, fault events and performance measurements from the O-Cloud
* Query the O-Cloud for the DMS end points it supports
* Query the O-Cloud for attributes such as capabilities and capacities
* Request the O-Cloud to create a Network Function deployment using cloud resources
* Request the O-Cloud to terminate Network Function deployment(s)
* Request the O-Cloud to reset NF Deployment(s)
* Request the O-Cloud to reset O-Cloud Node(s)
* Issue a subscription to the O-Cloud to receive alarm event notifications
* Query the O-Cloud for alarms on O-Cloud resources with query criteria which define the alarm characteristics that the SMO is interested in
* Query the O-Cloud for state and status information
* Perform an Alarm Subscription query towards the IMS
* Perform a NF deployment (that releases an NF instance) reset through the O2dms interface
* Perform an O-Cloud Node (recovery of the resources within the O-Cloud) reset through the O2ims interface

Using the O2 interface, the O-Cloud can perform the following operations:

* Send asynchronous events to the SMO when available capabilities or capacities are changed, including when new hardware is added
* Asynchronously notify the SMO when a software upgrade completes
* Asynchronously notify the endpoint specified by the SMO (alarm subscriber) of alarm notifications related to O-Cloud resources
* Return alarms in response to the SMO alarm query that match the alarm query criteria

### SERV#02 O-Cloud images service

It provides Add/Delete/Update/Query functions of O-RAN Cloudified Network Functions images with their related information (e.g. SoftwareImageId, Vendor, and Version) from O-Cloud repository.

### SERV#03 O-Cloud monitoring service

When the O-Cloud Infrastructure or the O-RAN cloudified NFs fails, it needs to be fixed immediately, and preferably automatically, to prevent end users from experiencing service disruptions. To avoid this service disruption Network Operations must consider the telemetry information of O-Cloud deployments in the network. The telemetry information serves as a vital resource for analysing the O-Cloud’s state and health, and for delivering on service monitoring goals. The O-Cloud Monitoring Service uses telemetry data to provide monitoring of O-Cloud infrastructures. O-Cloud telemetry shall minimally consist of Fault, Performance, and Configuration Data. There are different types of telemetry:

* **Managed Element Telemetry** to monitor the NFs behavior through O1.
* **Deployment Telemetry** to monitor the number of deployment instances an O-Cloud has at that moment and how many were expected, how the on-progress deployment is going, and health checks. Additional Deployment Telemetry metrics like CPU, network, and memory usage can also be collected. This will be performed through the O2dms interface.
* **Infrastructure Telemetry** to monitor the health of the O-Cloud Infrastructure components. Network Operations are interested in discovering if all the components in the O-Cloud Infrastructure are working properly and at what capacity, how many deployments are running on each node, and the resource utilization of the O-Cloud Infrastructure. This will be performed through the O2ims interface.

The SMO shall be able to collect and correlate telemetries to aggregate problems to a root cause.

The O-Cloud shall be able to report telemetries and make all Configuration Data and any external changes to it available to the SMO.

### SERV#04 O-Cloud provisioning service

O-Cloud Provisioning is the allocation of O-Cloud’s resources and services to an O-RAN Cloudified Network Function. This is one of the key functionalities of the O-Cloud, relating to how an O-RAN Cloudified Network Function procures O-Cloud services and resources. O-Cloud Provisioning shall provide:

* Create/Read/Update/Delete rules for Affinity, Anti-Affinity, and Quorum Diversity
* Query of O-Cloud Capacity & Availability

### SERV#05 O-Cloud software management service

Software management will manage the O-Cloud software. The software management should be a priority as without proper management unnecessary risks may be taken. The software management ensures security, cost management and software support. There are many benefits to software management, of which the main benefits are:

* Prevents unauthorized software from being installed
* Maintains a catalog of authorized software and its versions
* Provides visibility into what software and version is being used
* Provides a better view of which software products and versions are vendor supported

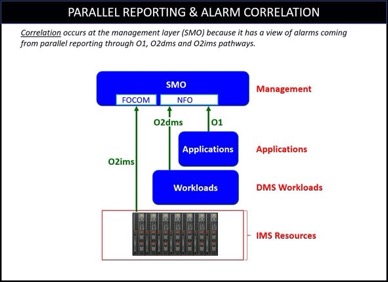
In O-RAN from an O-Cloud perspective there are two types of software which needs to be managed on the O2 Interface:

* The O-Cloud Infrastructure Software
* The O-RAN Cloudified Network Function Software is the software implementation of O-RAN NFs which can run over the O-Cloud

### SERV#06 O-Cloud fault management service

It is related to IMS & DMS fault management. There are three types of fault notifications that originate from O-Cloud and collected by SMO through O1, O2dms and O2ims.

* Fault notifications originate from the O-cloud infrastructure when a condition occurs within IMS on an O-Cloud resource. An event is issued towards the SMO for analysis. The SMO can issue a specific fault query towards to IMS related to the O-Cloud resources and resource pool through O2ims.
* DMS resource faults are associated with a workload. The SMO can issue a specific query related to xApp/NF deployments (e.g. workloads) from the DMS through O2dms.
* NFs faults are reported through O1 (application level).



**Figure 4‑3 : Parallel reporting & Alarm correlation [i.1]**

It is possible that multiple resources can associate with a workload, whereby one fault might trigger a fault notification on IMS, DMS and O1.

For example, it is conceivable that a O-DU application might report a fault on O1, and a related physical server allocated to a workload of a O-DU would raise a DMS fault which might also have an infrastructure fault raised on IMS.

For example, the SMO may receive a fault with a root cause of a network issue among the O-Cloud resources which manifests itself in a deployment and application fault as well. Thus, the SMO might receive three notifications over O2ims, O2dms and O1. It might correlate these faults to a common root cause.

NOTE: Faults issued by the O-Cloud could be stored at the O-Cloud resource for a configured amount of time.

### SERV#07 O-Cloud performance service

The purpose of performance service is to report operational information related to O-cloud resources. Typically, performance information allows an operator or administrator of the O-cloud a sense of how well the system is operating. It is distinct from faults in that it is not about failures in the system but about how well the overall system is performing. Though, faults or alarms may negatively impact performance of the O-Cloud which might be observable in performance measurements.

Performance measurements are typically captured periodically through time. They are collected and stored at regular intervals by the system in order to gauge the performance of a system over a period of time. This allows for analytical operations to be performed on the collected data and statistics to be built over time. This can tell an operator or system analyst whether they have sufficient capacity in a network based on the demands of the network. This can be vital for making business operational decisions such as scaling a network.

### VOID

### SERV#009 O-Cloud Inventory

O-Clouds contain physical, software and logical resources. The O-Cloud is the source of information for these assets but is required to provide a mechanism for asset management to account for them. The inventory services provide the read-only interface to its inventory such that accounting can be achieved (see O-RAN O2ims Interface Specification [i.8]). One of the responsibilities of the SMO is for “financial” accounting of the physical and logical inventory. The SMO inquires about the mapping of resources to their use. The O-Cloud is responsible for the assignment of resources to their use. Typically, the O-Cloud inventory is the result of the SMO provisioning requests and assignments of O-Cloud Resources with additions of O-Cloud internal status of the existing internal resources, e.g., if some of the resources are down due to faults or maintenance operations (see O-RAN WG6.GA&P [i.1]).

**Infrastructure Inventory** - Infrastructure inventory plays a pivotal role in the management and allocation of O-Cloud resources. It involves the use of an immutable O-Cloud Resource Identifier to track resources efficiently. This identifier aids in correlating inventory data, which includes provisioning status and operational insights.

**Logical Inventory** - Logical inventory involves managing the Deployment Management Services (DMS) that support various virtual cluster technologies, such as Kubernetes and OpenStack. Each DMS endpoint acts as an interface for deployments, cataloged by the SMO as a Logical Cloud. This inventory segment assists the SMO in selecting appropriate O-Cloud environments for deployments.

**Deployment Inventory** - Focusing on the deployment aspect, the inventory includes Deployment Descriptors that detail the resources allocated for services, such as VMs, Pods, Containers, and Networks. Each descriptor corresponds with specific O-Cloud capabilities and returns a unique Deployment ID, crucial for tracking and managing deployment specifics over the O2dms interface.

# Cloud deployment scenarios

## Main actors

1. Operator
2. CSP (Cloud Service Provider)
3. Vendors
4. Third parties (e.g. service providers, verticals)

## Cloud service models

Different levels of abstraction constitute the platform of the cloud architecture. These abstractions are grouped into the different service levels, depending on what resources are offered as a service for a given abstraction level. According to NIST SP 800-145 the three standard cloud service models are Platform as a Service (PaaS), Software as a Service (SaaS), and Infrastructure as a Service (IaaS).

### Infrastructure as a Service (IaaS)

The capability provided to the Operator is to provision processing, storage, networks, and other fundamental computing resources where the Operators or Vendors are able to deploy and run arbitrary software, which can include Host OS (Operator or Vendor), virtualization platforms (Operator or Vendor) and VNFs/CNFs (Operator). Operators and Vendors do not manage or control the underlying cloud infrastructure but have control over Host OS (Operator or Vendor), virtualization platforms (Operator or Vendor) and VNFs/CNFs (Operator).



**Figure 5‑1 : IaaS cloud service model**

### Platform as a Service (PaaS)

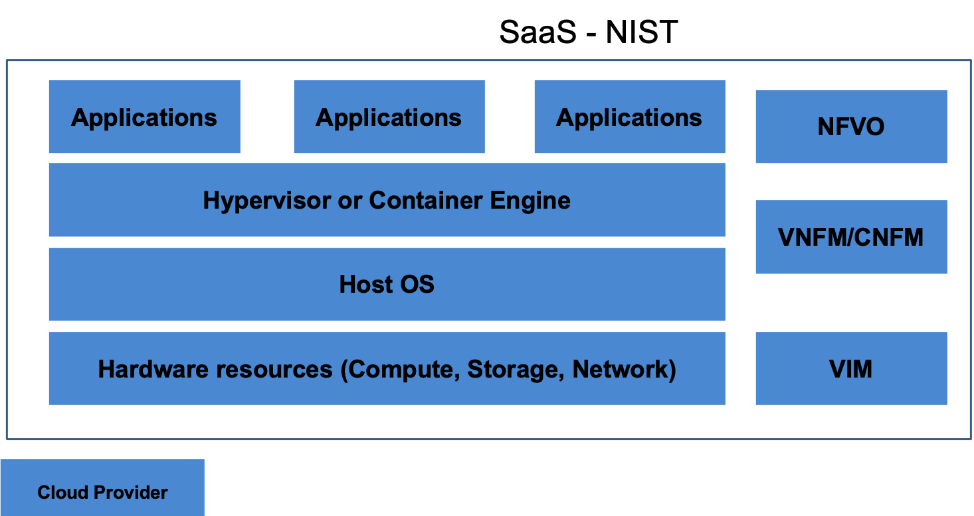
The capability provided to the Operator is to deploy onto the cloud infrastructure VNFs/CNFs created using programming languages, libraries, services, and tools supported by the cloud provider. The Operator does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed VNFs/CNFs and possibly configuration settings for the application-hosting environment. PaaS provides this platform from the cloud, hence allowing operators to develop and run VNFs/CNFs without the overhead cost (CAPEX) of building and maintaining separate platforms.



**Figure 5‑2 : PaaS cloud service model**

### Software as a Service (SaaS)

SaaS as defined by NIST is the capability provided to the Cloud Consumer to use the Cloud Provider’s applications running on a cloud infrastructure. Applications are accessible through web browser or program interface. The operator does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even Applications.



**Figure 5‑3 : SaaS cloud service model**

This service model introduces risk for the Operator, acting as the Cloud Consumer, due to shared resources, reduced control, lack of transparency for data security, secure access configuration, limited visibility to logging at lower layers, malicious insiders, regulatory drift, and lack of due diligence. It is not expected that 5G operators will deploy with the SaaS service model.

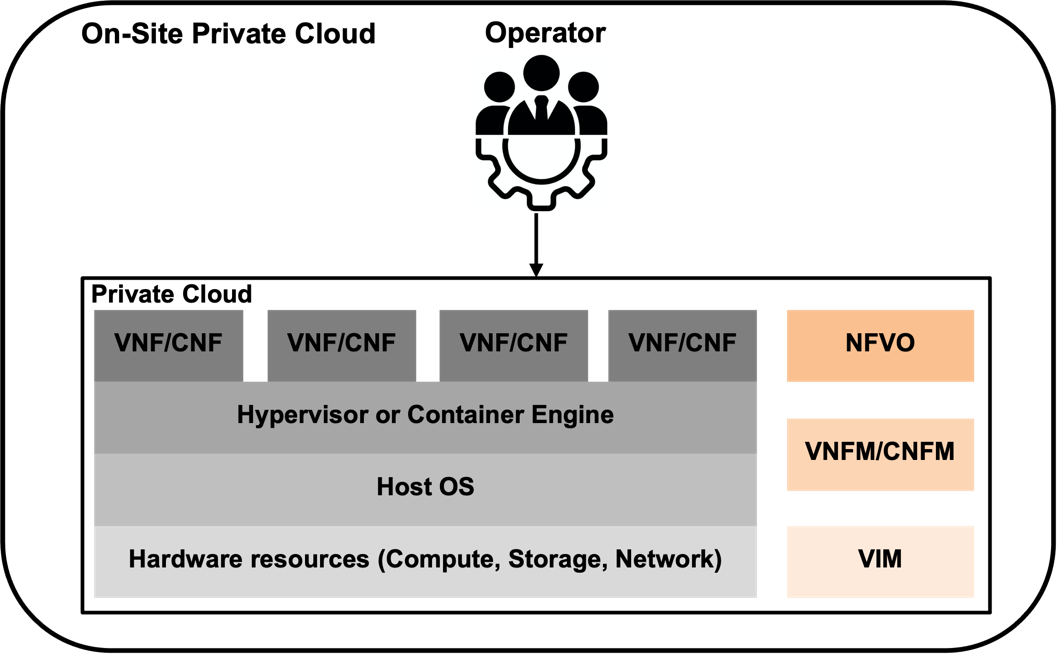
The Cloud Provider may offer 5G private networks directly to its customers in a SaaS service model. In this service model the Cloud Provider’s customer is the Cloud Consumer, which has responsibility for securing its data and access to the applications.

## Cloud deployment types

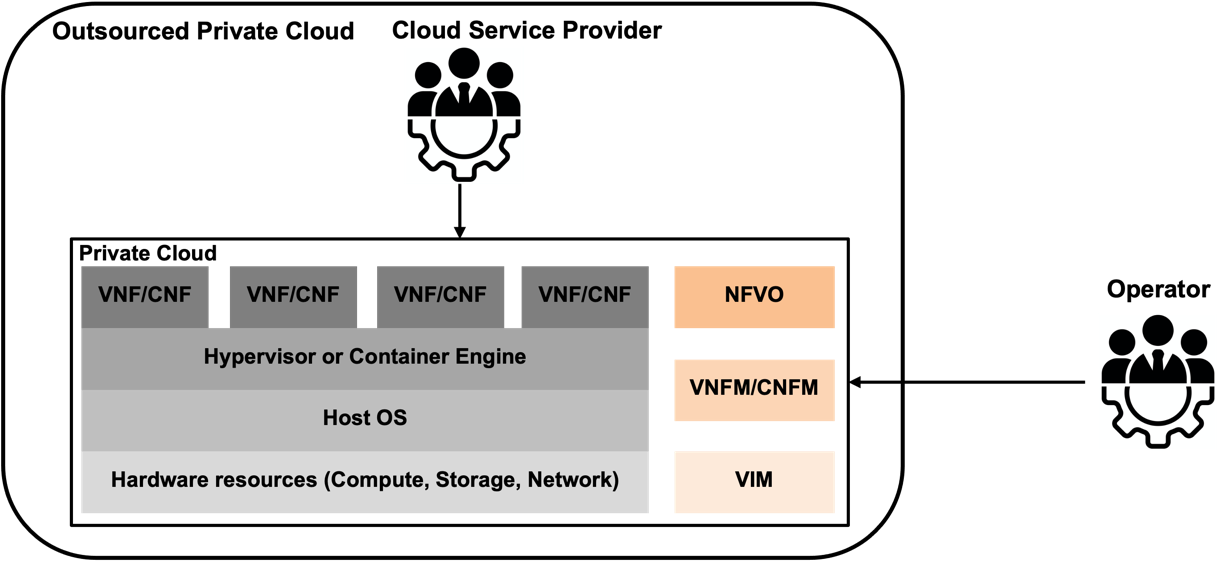
In NIST SP 800-145, cloud deployment models describe how the cloud is operated and who has access to the cloud service resources. The four deployment models are defined in NIST SP 800-145 as follows:

### Private cloud

A private cloud gives a single Operator the exclusive access to and usage of the cloud service and related infrastructure and computational resources. It may be managed either by the Operator or by a third party Cloud Provider and may be hosted on the Operator’s premises (i.e., on-site private clouds) or outsourced to a hosting company (i.e., outsourced private clouds). The following figures present an on-site private cloud and an outsourced private cloud, respectively.



**Figure 5‑4 : On-site private cloud**



**Figure 5‑5 : Outsourced private cloud**

**On-site private cloud**

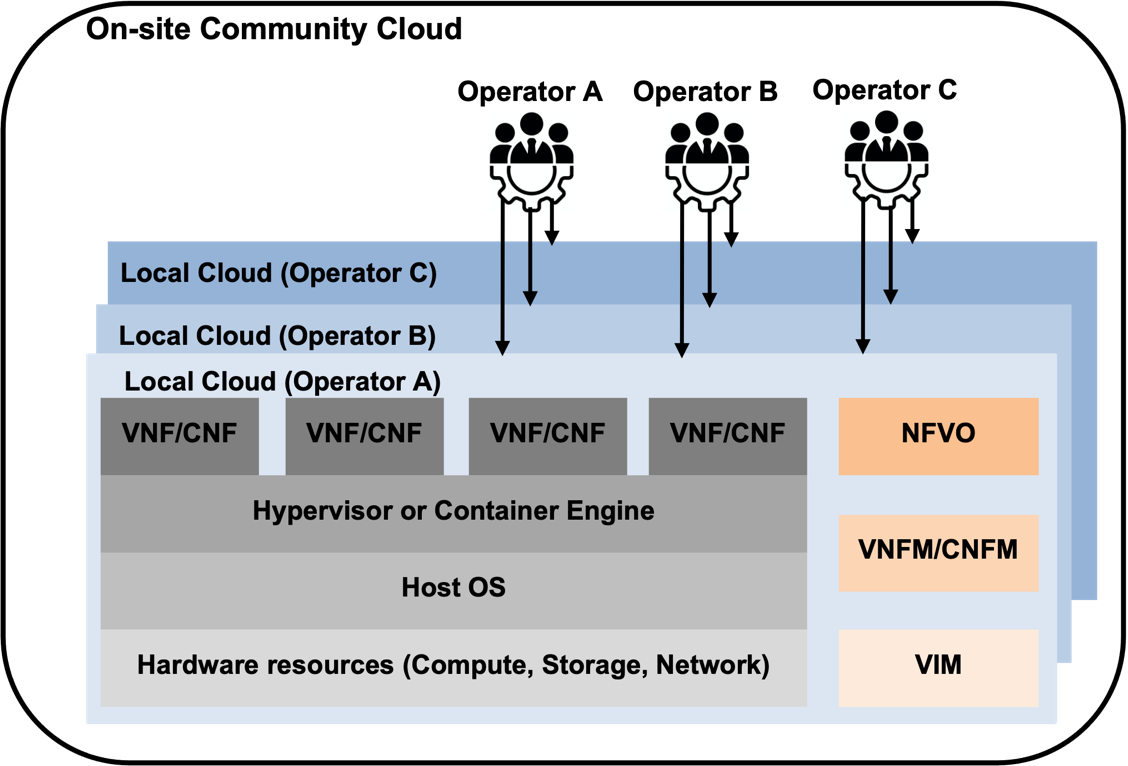
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud type** | **Cloud model** | **Hardware** | **Host OS** | **Container Engine or Hypervisor** | **VNFs/CNFs** |
| On-site private | SaaS On-Premise | Operator | Operator | Operator | Operator |
| On-site private | IaaS | Operator, CSP | CSP, Operator | Operator, Vendors | Operator |
| On-site private | PaaS | Operator, CSP | CSP, Operator | CSP, Operator | Operator |

**Outsourced private cloud**

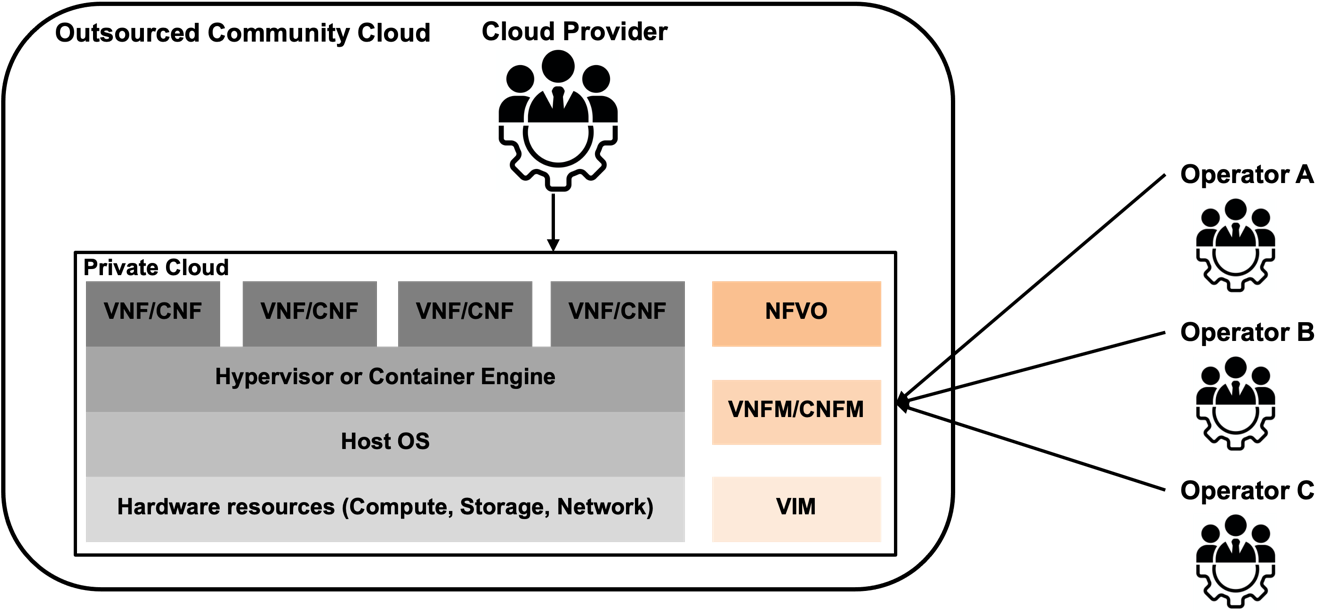
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud type** | **Cloud model** | **Hardware** | **Host OS** | **Container Engine or Hypervisor** | **VNFs/CNFs** |
| Outsourced private | SaaS On-Premise | NA | NA | NA | NA |
| Outsourced private | IaaS | Operator, CSP | CSP, Operator | Operator, Vendors | Operator |
| Outsourced private | PaaS | Operator, CSP | CSP, Operator | CSP, Operator | Operator |

### Community Cloud

A community cloud serves a group of Operators that have shared concerns such as mission objectives, security, privacy and compliance policy, rather than serving a single Operator (e.g., a private cloud). Similar to private clouds, a community cloud may be managed by the Operators or by a third party Cloud Provider and may be implemented on the Operator’s premise (i.e., on-site community cloud) or outsourced to a hosting company (i.e., outsourced community cloud). Figure 5-6 depicts an on-site community cloud comprised of a number of participant Operators. An Operator can access the local cloud resources, and also the resources of other Operators through the connections between the associated Operators. Figure 5-7 shows an outsourced community cloud, where the server side is outsourced to a hosting company. In this case, an outsourced community cloud builds its infrastructure off premise, and serves a set of Operators that request and consume cloud services.



**Figure 5‑6 : On-site community cloud**



**Figure 5‑7 : Outsourced community cloud**

**On-site Community Cloud**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud type** | **Cloud model** | **Hardware** | **Host OS** | **Container Engine or Hypervisor** | **VNFs/CNFs** |
| On-site Community Cloud | On-Premise | Operators | Operators | Operators | Operators |
| On-site Community Cloud | IaaS | CSP, Operators | CSP, Operators, Vendors | CSP, Operators, Vendors | Operators |
| On-site Community Cloud | PaaS | CSP, Operators | CSP, Operators, Vendors | CSP, Operators, Vendors | Operators |

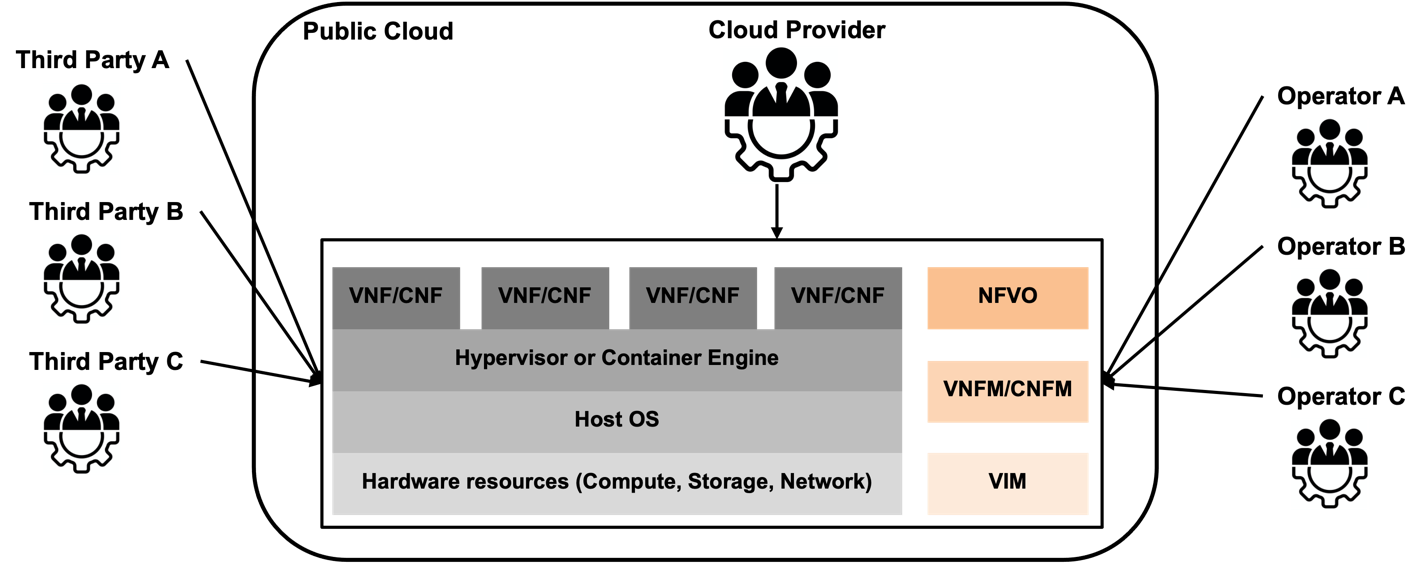
**Outsourced community cloud**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud type** | **Cloud model** | **Hardware** | **Host OS** | **Container Engine or Hypervisor** | **VNFs/CNFs** |
| Outsourced community | On-Premise | NA | NA | NA | NA |
| Outsourced community | IaaS | CSP | CSP, Operators, Vendors | CSP, Operators, Vendors | Operators |
| Outsourced community | PaaS | CSP | CSP, Operators, Vendors | CSP, Operators, Vendors | Operators |

### Public Cloud

A public cloud is one in which the cloud infrastructure and computing resources are made available to the general public over a public network. A public cloud is owned by an organization providing cloud services, and serves a diverse pool of clients (e.g. Operators, Third parties service providers).

Figure 5-8 presents a simple view of a public cloud and its actors.



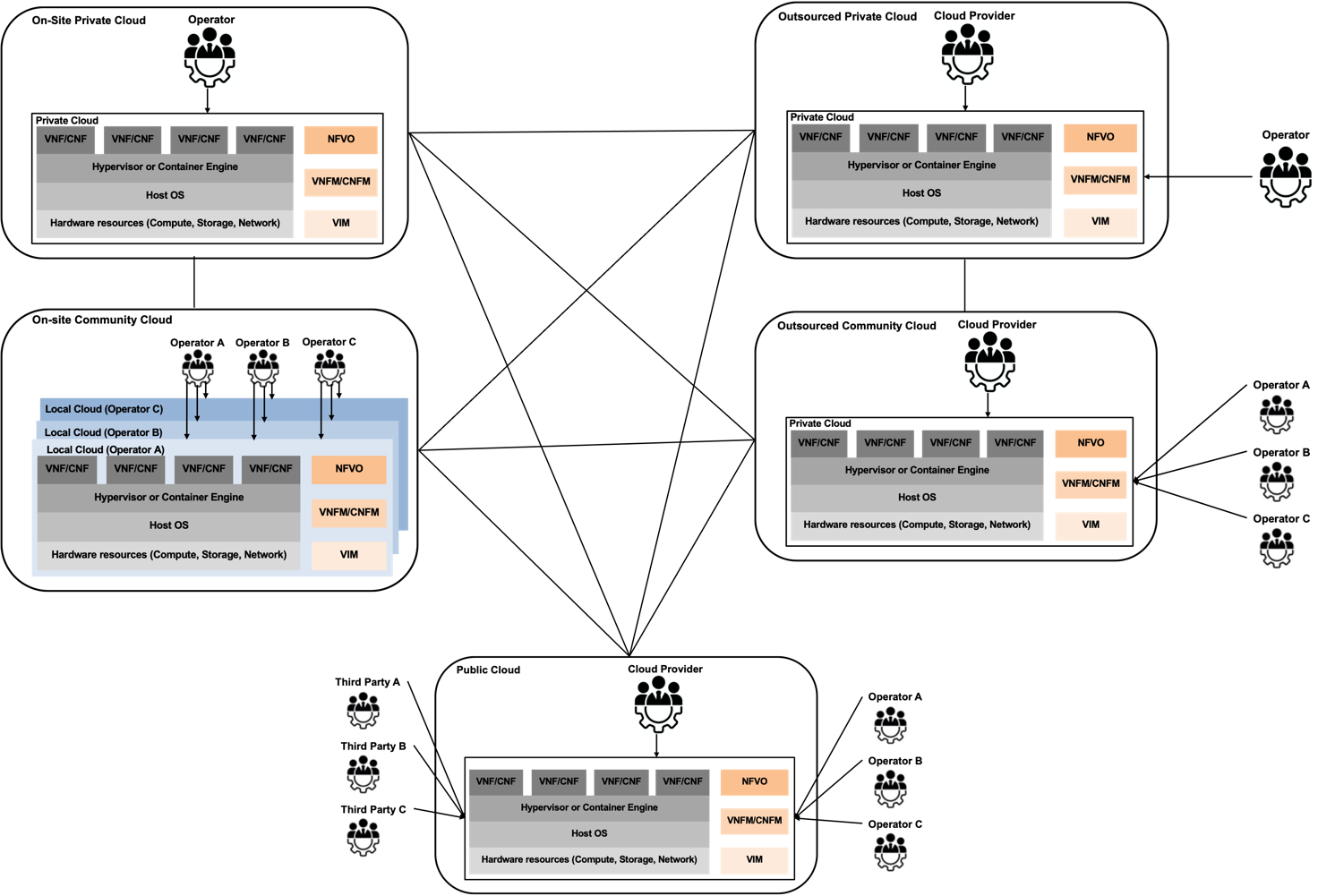
**Figure 5‑8 : Public cloud**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cloud type** | **Cloud model** | **Hardware** | **Host OS** | **Container Engine or Hypervisor** | **VNFs/CNFs** |
| Public | On-Premise | NA | NA | NA | NA |
| Public | IaaS | NA | NA | NA | Operators |
| Public | PaaS | CSP | CSP | CSP | Operators |

### Hybrid Cloud

A hybrid cloud is a composition of two or more clouds (on-site private, on-site community, off-site private, off-site community or public) that remain as distinct entities but are bound together by standardized or proprietary technology that enables data and application portability. Figure 5-9 presents a simple view of a hybrid cloud that could be built with a set of clouds in the five deployment model variants.

The idea is to combine the benefits of multiple deployment models. The deployment in hybrid clouds could be either a combination of private, public or community clouds. One of the popular use cases of hybrid clouds is in enhancing security and privacy on the cloud without incurring the overhead costs (CAPEX) of building a private cloud. In this case, non-critical resources like test workloads can be hosted in the public cloud, while critical resources like user data and workloads are hosted internally.



**Figure 5‑9 : Hybrid cloud**

## High-Level risk assessment

The following table illustrates the high-level security risk assessment of the cloud deployment models (Private, Community, Public and Hybrid). Colors red, yellow, and green are used on the security risk assessment to indicate high, medium, and low levels of risk. The Cloud Consumer (operator) is accountable for the security posture of the deployment for all cloud deployment models [i.44].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risks** | **Private Cloud** | **Community Cloud** | **Public Cloud** | **Hybrid Cloud** |
| **Regulatory Compliance**: Risk of non-compliance with regulations on data ownership, geographic location and privacy. Regulatory drift.  Data security and regulatory risk can be associated with loss, leakage, or unavailability of data. This can cause business interruption, loss of revenue, loss of reputation, or regulatory incompliance (e.g. GDPR, HIPAA).  The Cloud Consumer (operator) is accountable for data, regulatory compliance, and configuration of security controls. | **Low Risk**: Similar to traditional computing, as operators have better control and understanding on how various government rules, laws, and regulations apply to them. The Cloud Consumer (operator) has control of data and regulatory compliance. | **Moderate Risk**: Cloud Consumer (operator) has better control and understanding on how various government rules, laws, and regulations apply to them. Data leakage or data access risks due to multitenancy/shared infrastructure between different operators. | **High Risk**: Lack of flexibility over data protections mechanisms, such as encryption and implementation of specific controls by data type. Different operators or service providers might have different encryption and control requirements, and a public cloud provider may not be able to customize their infrastructure or provide customers the control over encryption keys.  In some situations, it may be determined that a Cloud Service Provider's cybersecurity tools, processes and methods are insufficient for protecting highly sensitive data. The Cloud Consumer (operator) is responsible for the security configuration of CSP provided applications, tools and services.  The Cloud Consumer (operator) must practice due diligence to assess the regulatory compliance of the Cloud Service Provider's environment. | **Moderate Risk**: Cloud Consumer (operator) has better control and understanding on how various government rules, laws, and regulations apply to them.  Administrators have more control and flexibility when implementing security. The Cloud Consumer can architect the Hybrid Cloud deployment to ensure regulatory compliance of most sensitive data, while less sensitive data is accessed, stored, and processed in the Public Cloud.  The Cloud Consumer (operator) must practice due diligence to assess the regulatory compliance of the Cloud Service Provider's environment. |
| **Multi-Tenancy**: Risk due to Multi-Tenancy with shared resources. Risks include data leakage, VM Escape, Host Escape, and Hyperjacking. | **Low Risk**: Private Networks avoid shared resources. | **High Risk**: Data leakage or access risks due to multitenancy/shared resources for multiple Cloud Consumers (operators) that may be competitors. Cloud Consumer (operator) must ensure security configurations to protect access to network functions, confidentiality and integrity of data in motion, and confidentiality, integrity, and access controls for data at rest. | **High Risk**: Data leakage or access risks due to multitenancy/shared resources. Cloud Consumer (operator) must ensure security configurations to protect access to network functions, confidentiality and integrity of data in motion, and confidentiality, integrity, and access controls for data at rest. | **High Risk**: Data leakage or access risks due to multitenancy/shared infrastructure. Cloud Consumer (operator) must ensure security configurations to protect access to network functions, confidentiality and integrity of data in motion, and confidentiality, integrity, and access controls for data at rest. |
| **Third-Party Administration**: Operational risk can be associated with O-RAN network services, lack of control, visibility, etc. | **Low Risk**: With private cloud, the operator gains full control and visibility over its cybersecurity posture and can customize it to fit its specific needs.  Operator has complete visibility, monitoring and control, analyze logs, alerts and other data down to the packet level. | **Moderate Risk**: Security administrators may deal with differing methods and tools to monitor and act on threats depending on where vulnerable resources reside.  Lack of customized security and service level for different O-RAN network services, which might require the operator to choose a proximate acceptable security and service level, including those related to availability and disaster recovery. | **High Risk**: Lack of customized security and service level for different O-RAN network services, which might require the operator to choose a compensating security control and security baseline.  The underlying infrastructure is not managed by the operator, security information may be not accessible.  Reduced control on critical network services can impact availability and disaster recovery. Disaster Recovery with a second CSP, which may not have the same security posture due to use of different security tools and configurations. | **High Risk**: Varying security tools and configurations may not prevent a consistent security baseline across the Hybrid Cloud deployment. The Cloud Consumer may have reduced visibility with the CSP controlled part of the service.  Disaster Recovery should be provided with a second CSP, which may not have the same security posture due to use of different security tools and configurations.  Security roles and responsibilities must be clearly defined and assigned to the Cloud Consumer (operator) and Cloud Service Provider (CSP). The Cloud Consumer (operator) must practice due diligence to ensure a consistent security posture across the Hybrid Cloud deployment. |
| **Technology Evolution**: Technology risk can be associated with constantly evolving technologies and lack of standardization in how they integrate or interoperate. Technology risks could lead to costly rearchitecture efforts for adoption or integration with new technology. | **High Risk**: A constantly evolving technology landscape that might require the operator to upgrade or rearchitect its computing resources and retrain its technology support staff.  A potential for human error due to the number of configurable points and frequency of deployments.  Lack of qualified talent. | **Moderate Risk**: Operators share a common set of security tools, for which each is responsible for properly configuring the security of its tenancy.  Operators can share cloud complexity to manage constantly evolving technology landscape that might require operators to upgrade or rearchitect its computing resources and retrain its technology support staff. | **Low Risk**: Larger CSPs often invest heavily in top-end cybersecurity tools, as well as staff who are highly knowledgeable in their field. This makes offloading cybersecurity tools and tasks from in-house to a third party highly appealing.  Operators can transfer cloud complexity to the CSP, which already has the necessary cloud expertise, infrastructure and systems. For example, faster time-to-market and improved operations efficiency are achieved by taking advantage of CSP-trained talent and their existing CI/CD DevOps, automation, orchestration and analytics capabilities. Operators can then focus on service differentiation, not the underlying cloud platforms. | **Moderate Risk**: Hybrid Cloud deployments provide flexible management of legacy and proprietary applications while enabling use of technologies associated with cloud-based services and applications.  A potential for human error due to the number of configurable points and frequency of deployments. The operator is responsible for properly configuring the security across the Hybrid Cloud. The Cloud Consumer (operator) must practice due diligence to ensure a consistent security posture across the Hybrid Cloud deployment. |
| **Financial costs & operation** | **High Risk**: Operating private clouds is often a more expensive endeavor than public cloud options. Businesses pay a premium for granular cloud control and visibility.  Designing and maintaining cybersecurity tools inside private clouds dramatically increase management responsibilities.  Lack of Operator-trained administrators. | **Moderate Risk**: Designing and maintaining cybersecurity tools inside community clouds increase management responsibilities.  Lack of Operator-trained administrators.  Operators can share financial cost and the operation of community cloud. | **Low Risk**: Operating public clouds is often a less expensive endeavor than private and community cloud options.  Operators can transfer cloud operation to the CSP, which already has the necessary cloud expertise, infrastructure, and systems. | **Moderate Risk**: Designing and maintaining cybersecurity tools inside hybrid clouds increase management responsibilities.  Operators can transfer part of the cloud operation to the CSP, which already has the necessary cloud expertise, infrastructure, and systems. |

Risk levels: High, Moderate, Low

**Table 5‑1 : High level security risk assessment of cloud deployment models**

# Roles and responsibilities

The list of users likely to interact with the O-CLOUD as well as their roles are presented in the table below. The last column distinguishes between stakeholders acting:

* Under the direct responsibility of an operator or
* Under the responsibility of a third party (e.g. equipment supplier) acting through a contract with the operator.

The table shows many roles in which the operator and cloud service provider are both stakeholders.  This highlights the need to have:

* Clearly defined and agreed upon roles and responsibilities in the cloud service agreement.
* Separation of Duties enforced by the cloud service provider to limit insider threats.
* Principles of least privilege access control configured and enforced by the operator.

|  |  |  |
| --- | --- | --- |
| **Roles** | **Responsibilities** | **Stakeholders** |
| O-Cloud Planner | Operator planning O-Cloud instance. | Operator, CSP |
| O-Cloud Installation Manager | Operator designing each O-Cloud instance.  Cloud Installation Manager issues a “service request” to the SMO to update its Inventory, i.e., indicate that a new O-Cloud is to be inventoried. The Cloud Installation Project Manager also registers to the SMO the basic software for the O-Cloud. | Operator, CSP |
| Network Function Install Manager | Operator designing each Network Function instance. | Operator, Vendor |
| O-Cloud Installer | Operator installing O-Cloud Node. He notifies SMO of the new O-Cloud Build  NOTE: Cloud Installer must provide SMO with information to connect with the new O-Cloud Build, including the target O-Cloud Node in the O-Cloud and identity of the basic software for O-Cloud to be loaded on the O-Cloud | Operator, CSP |
| O-Cloud Maintainer | Operator that operates and manages O-Cloud Nodes | Operator |
| Security administrator | They manage component security configuration: configuration of access rights, password and cryptographic key storage system, IPSec parameters, configuring remote access systems, etc.  They hold the root password and the secret encryption root key. | Operator |
| System administrator | They manage “system” components (e.g. back-up, update, configuring system logs, etc.). | Operator |
| O-RAN NF Administrator | They can manage and configure one or more VNFs/CNFs (e.g. Near RT RIC, O-CU, O-DU). They can perform the daily operational tasks (notably provisioning) of NFs. | Operator |
| IMS Administrator | They manage IMS activities. They can perform the daily operational tasks and configure the IMS. They control functions providing access to the IMS. They are responsible for VNFs/CNFs hardware configuration. | Operator, CSP |
| DMS Administrator | They manage DMS activities. They can perform the daily operational tasks (notably provisioning) and configure the DMS. They control functions providing access to DMS. | Operator CSP |
| Virtualization layer administrator (Hypervisor/Container Engine) | They manage the virtualization/containerization layer (Hypervisor/Container Engine). They can perform the daily operational tasks (notably provisioning) and configuration of the virtualization layer. They control functions providing access to this layer. | Operator, CSP |
| Compute nodes administrator (physical servers) | They manage compute nodes (physical servers) They can access the node’s graphic console, start, stop, restart the node. They can activate/deactivate the node network interfaces and check its status. They manage local storage in compute nodes. | Operator, CSP |
| Storage administrator | They manage shared storage (e.g. images repository) | Operator, CSP |
| Maintainer (equipment supplier or third party) | They can remotely (e.g. via VPN) or locally access equipment to update, configure, remove or add network applications. They have access to compute node configuration interfaces to update firmware and configure BIOS, CPU, MMU, etc.  The maintainer works on the O-Cloud platform under the control of the operator who (e.g. via IAM: Identity Access Management) creates specific accounts for a limited period. | Vendor, Third Party |
| Internal validation laboratory | They are responsible for vulnerability scanning, testing, qualifications, security/compliance of network applications before their deployment. | Operator, Vendor, CSP |
| Security Incident Response Team | Setup a vulnerability management process of monitoring, identifying, evaluating, treating, and reporting on security vulnerabilities and incidents.  Maintenance of the O-Cloud components that includes monitoring for use of open source software with known vulnerabilities and patching for vulnerabilities.  Provide a process for users, including security researchers, to submit bug reports (e.g., using an issue tracker or a mailing list). | Operator, Vendor, CSP |

**Table 6‑1 : List of Users**

# Security Problem Definition

Before analyzing the way to protect the O-Cloud, it is important to identify the threats affecting the different O-Cloud components and data.

For this identification of threats, it is essential first to know the critical assets, consisting of anything that has value for within O-Cloud and needs to be protected, and secondly to identify the threat agents, the entities that can adversely act on the asset.

## Assets

The assets listed below are associated with O-Cloud. They are described in Clause 6.3 of the O-RAN Security Threat Modeling and Risk Assessment [i.50]:

* ASSET-D-12: O-Cloud Inventory information of hardware resources and software resources
* ASSET-D-13: Functional telemetry: deployment, infrastructure
* ASSET-D-14: O-Cloud Provisioning information, O-Cloud software management information, List of authorized VNF/CNF, VNF/CNF description files
* ASSET-D-15: O-RAN Cloudified Network Function Software Image and metadata, secrets, configuration files
* ASSET-D-16 & ASSET-D-17: Private keys, certificates support for authentication, encryption, signing (e.g. for TLS and similar protocols, image signing)
* ASSET-D-18: Software components at runtime and their associated data, data of VNF/CNF hosted in the shared storage system, RAM and the disk in the compute node
* ASSET-D-20: Credentials (Administrators): account information and passwords
* ASSET-D-29: Security event log files generated by O-Cloud components
* ASSET-D-31: Security telemetry from the NFV system for detecting threats and anomalies
* ASSET-D-32: Cryptographic keys used during secure boot, for encryption/decryption, etc.
* ASSET-D-33: Data in transit AALI-C-Mgmt interface
* ASSET-D-34: Data in transit AALI-C-App & AALI-P interfaces
* ASSET-D-35: Data in transit vendor specific interface
* ASSET-D-36: AAL profiles. It specifies one or more Accelerated Functions that an accelerator processes on behalf of an AAL Application within an O-RAN Network Function in an O-Cloud [i.52].
* ASSET-D-37: AAL-LPU. It is a logical representation of resources within an instance of a HW Accelerator [i.52].
* ASSET-D-38: Stored AAL data (e.g., logs, configuration data)
* ASSET-C-08: O-Cloud software including the OS kernel and the virtualisation layer.
* ASSET-C-23: O2, including protocol stack
* ASSET-C-29: AAL Implementation. It is a realization of an AAL including but not limited to the software libraries and drivers [i.52].
* ASSET-C-30: The hardware accelerator device firmware and hardware.

The figure below presents a cartography of the assets within the O-Cloud architecture:



**Figure 7‑1 : Cartography of O-Cloud assets**

## Threats

The complexity and the extension of attack surface of a virtualized environment increase the difficulty to list, in an exhaustive manner, the security threats to which the O-Cloud assets are exposed. For this activity of threat analysis, some reports have been used to help the identification of a large number of these threats:

* CISA/NSA - Kubernetes Hardening Guidance [i.3]
* MITRE ATT&CK containers matrix [i.4]
* ENISA Threat Landscape for 5G Networks: Threat assessment for the fifth generation of mobile telecommunications networks (5G) [i.5]
* ETSI Secure End-to-End VNF and NS management specification Secure End-to-End VNF and NS management specification [i.6]

The following table is the template used to present the threat characteristics:

|  |  |
| --- | --- |
| **Threat ID** | Unique identification per Threat (e.g. T-XX-01) |
| **Threat title** | Title of the threat |
| **Threat description** | Description of the Threat |
| **Threat type** | Spoofing  Tampering  Repudiation  Information disclosure  Denial of Service  Elevation of Privilege |
| **Vulnerabilities** | What vulnerabilities can the threat exploits? |
| **Impact type** | Authenticity  Integrity  Non-repudiability  Confidentiality  Availability  Authorization |
| **Threatened Assets** | Impacted Asset(s) (Data & Component) |
| **Affected Services** | What services (from section 4.3) could be affected by this Threat |
| **Potential mitigations** | Mitigation measures to address the threat |

### Threat and impact types

For identifying threats, we are using STRIDE:

1. S - Spoofing identity. An application or program can masquerade as another to gain advantages not typically allowed for that program.
2. T - Tampering with data. This involves the malicious modification of data, including making unauthorized changes to a database and alteration of data as it flows between computers.
3. R - Repudiation. A user or program refuses the authenticity of a good or reasonable command or action.
4. I - Information disclosure. This involves the exposure of information to individuals with unauthorized access to it. For example, users gain the ability to read a file that they normally would not have been granted access to, or an intruder can read data in transit between computers.
5. D - Denial of service. These attacks deny service to valid users, such as making a website unavailable or unusable by flooding it with illegitimate requests to keep legitimate users without access.
6. E - Elevation of privileges. An unauthorized user gains privileged rights to access previously no granted to compromise or destroy the system, such as a change in membership.

|  |  |
| --- | --- |
| **Threat types** | **Impact types** |
| Spoofing | Authenticity |
| Tampering | Integrity |
| Repudiation | Non-repudiability |
| Information disclosure | Confidentiality |
| Denial of Service | Availability |
| Elevation of Privilege | Authorization |

### Attack surface

An attack surface of a system refers to the set of various entry points that can be exploited. The various components that compose the attack surface of the O-Cloud are depicted in the following figure. They are as follows:

1. VNFs/CNFs: O-DU, O-CU, O-RU, Near RT-RIC/xApps
2. Images repository with its interface to O-Cloud
3. Virtualization layer: Hypervisor and/or Container Engine, Host OS
4. Hardware resources including compute, storage, network, and hardware accelerator manager
5. O-Cloud API
6. O2dms and O2ims interfaces
7. NFO and Federated O-Cloud O&M (FOCOM) within the SMO
8. AAL

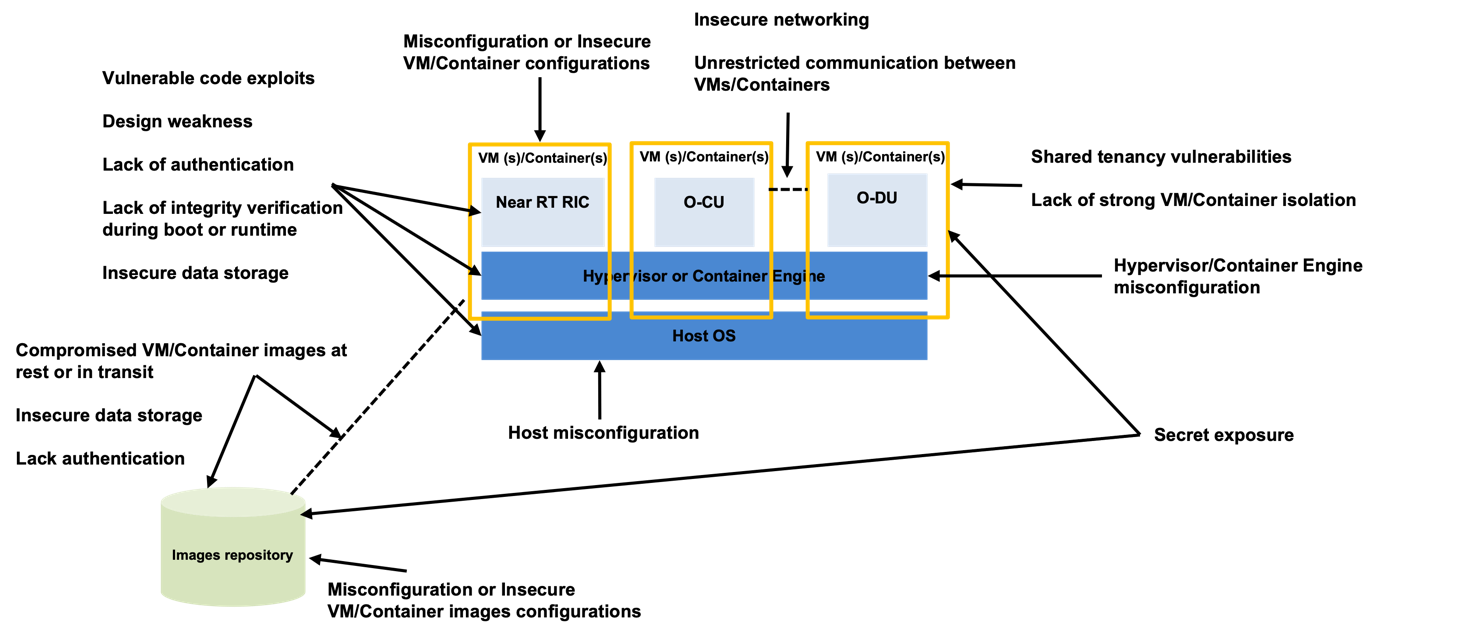
The attack vectors for exploiting vulnerabilities of these components are discussed in detail in the threat events sections.



**Figure 7‑2 : Attack vectors**

### Vulnerabilities

The following figure highlights the main vulnerabilities that may emerge within the attack vectors (see Figure 7-2).



**Figure 7‑3 : Vulnerabilities within O-Cloud**

### Threat events

The following threats to O-Cloud have been identified and are analyzed in this Clause. Other threats within the attack vectors ④, ⑥, ⑦ ⑧ will be added in future versions of the report.

|  |  |
| --- | --- |
| **Threat** | **Threat Title** |
| **Generic Threats** | |
| T-GEN-01 | Software flaw attack |
| T-GEN-02 | Malicious access to exposed services using valid accounts |
| T-GEN-03 | Untrusted binding between the different O-Cloud layers |
| T-GEN-04 | Lack of Authentication & Authorization in interfaces between O-Cloud components |
| T-GEN-05 | Unsecured credentials and keys |
| T-GEN-06 | Sensitive application data cache exploitation |
| **Threats concerning VMs/Containers** | |
| T-VM-C-01 | Abuse of a privileged VM/Container |
| T-VM-C-02 | VM/Container escape attack |
| T-VM-C-03 | VM/Container data theft |
| T-VM-C-04 | VM/Container migration attacks |
| T-VM-C-05 | Changing virtualization resource without authorization |
| T-VM-C-06 | Failed or incomplete VNF/CNF termination or releasing of resources |
| **Threats concerning VM/Container images** | |
| T-IMG-01 | VM/Container images tampering |
| T-IMG-02 | Insecure channels with images repository |
| T-IMG-03 | Secrets disclosure in VM/Container images |
| T-IMG-04 | Build image on VL |
| **Threats concerning the virtualization layer (Host OS-Hypervisor/Container engine)** | |
| T-VL-01 | VM/Container hyperjacking attack |
| T-VL-02 | Boot tampering |
| T-VL-03 | Attack internal network services |
| **Threats concerning O-Cloud interfaces** | |
| T-O2-01 | MitM attacks on O2 interface between O-Cloud and SMO |
| T-OCAPI-01 | MitM attacks on O-Cloud interface between VNFs/CNFs and the virtualization layer |
| **Threats concerning hardware resources** | |
| T-HW-01 | Cross VM/Container side channel attacks |
| T-HW-02 | MitM attacks on the interface between virtualization layer and hardware |
| **Threats concerning O-Cloud management (SMO, NFO, FOCOM)** | |
| T-ADMIN-01 | Denial of service against NFO/FOCOM |
| T-ADMIN-02 | Abuse an O-Cloud administration service |
| **Threats concerning Acceleration Abstraction Layer (AAL)** | |
| T-AAL-01 | Attacker exploits insecure API to gain access to hardware accelerator resources |
| T-AAL-02 | Internal Overload DoS attack targeting AAL services |
| T-AAL-03 | Fail to clear resources |
| T-AAL-04 | HAM compromise |
| T-AAL-05 | Malicious memory accesses |
| T-AAL-06 | Firmware attacks |
| **Threats concerning O-Cloud instance ID** | |
| T-O-CLOUD-ID-01 | ID reuse in O-Cloud's object lifecycle |
| T-O-CLOUD-ID-02 | Node redundancy in O-Cloud deployments |
| T-O-CLOUD-ID-03 | O-Cloud ID mismanagement |
| **Threats concerning time synchronization** | |
| T-TS-01 | Time synchronization disruption in O-Cloud |

#### Generic Threats

|  |  |
| --- | --- |
| **Threat ID** | T-GEN-01 |
| **Threat title** | Software flaw attack |
| **Threat description** | Code of host OS, Hypervisor/Container Engine and VNF/CNF can include flaws that an attacker can exploit if they are present.  O-RAN software components rely on opensource software, opensource libraries, 3rd party components. Vulnerability in any of these software components likely to allow attacker to exploit O-CLOUD environment. This could lead attacker to carry out to malicious activities, such as:   * Compromise of the underlying VM/Container * Exploit host access via Escape to Host * Take advantage of weak identity and access management policies to attempt to elevate privileges * Execute adversary-controlled code * Enable an adversary to move from a virtualized environment, such as within a virtual machine or container, onto the underlying host |
| **Threat type** | Spoofing, Tampering, Information disclosure, Elevation of Privilege |
| **Vulnerabilities** | Vulnerable code exploits, Design Weakness |
| **Impact type** | Authenticity, Integrity, Confidentiality, Authorization |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | All services |
| **Potential mitigations** | REC-IMGP Image protection  REC-VHPM Vulnerability Handling and Patch Management  REC-SDLC Secure Development Lifecycle  REC-LOG Logging, Monitoring and Alerting  REC-ISO Strong Isolation  REC-AUD Security Audit |

|  |  |
| --- | --- |
| **Threat ID** | T-GEN-02 |
| **Threat title** | Malicious access to exposed services using valid accounts |
| **Threat description** | Access to valid accounts to use the O-Cloud services is often a requirement, which could be obtained through credential pharming or by obtaining the credentials from users after compromising the network.  Adversaries may obtain and abuse credentials of existing accounts as a means of gaining initial access, persistence, privilege escalation, or defense evasion. Compromised credentials may be used to bypass access controls placed on various resources on O-Cloud.  Compromised credentials may also grant an adversary increased privilege to specific O-Cloud services or access to restricted areas of the O-Cloud network.  Access may be also gained through an exposed service that doesn’t require authentication. In containerized environments, this may include an exposed Docker API, Kubernetes API server, kubelet, or web application such as the Kubernetes dashboard. |
| **Threat type** | Spoofing, Tampering, Information disclosure, Elevation of Privilege |
| **Vulnerabilities** | Lack of authentication |
| **Impact type** | Authenticity, Integrity, Confidentiality, Authorization |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | All services |
| **Potential mitigations** | REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-LOG Logging, Monitoring and Alerting  REC-NS Network Segmentation & Filter Network Traffic  REC-AUD Security Audit |

|  |  |
| --- | --- |
| **Threat ID** | T-GEN-03 |
| **Threat title** | Untrusted binding between the different O-Cloud layers |
| **Threat description** | One major challenge in virtualized architectures and especially in O-Cloud is to prove that a particular VM/Container runs on top of a specific Hypervisor/Container Engine. More specifically, it is necessary to assure that a trusted VM/Container is executed on a particular trusted Hypervisor/Container Engine, whereas the Hypervisor/Container Engine's trust state relies on an attestation that considers the entire corresponding hard and software stack. More precisely, this includes all hardware chips, firmware, OS and Hypervisor/Container Engine components that are relevant for the Hypervisor/Container Engine's trust state determination.  If it is not possible to establish a correlation between VM/Container and Hypervisor/Container Engine, an attacker is able to make use of a trusted VM/Container that runs on top of an untrusted Hypervisor/Container Engine and it would be impossible to detect any interference made by the malicious Hypervisor/Container Engine, e.g. intercepting communication, replacing strong or using weak cryptographic keys, etc. Similarly, trustworthiness in the service-layer might only be established if there is a mechanism to determine that only trusted VNFs/CNFs, w.r.t trusted VM/Container's, are running on specific trusted Hypervisors/Container Engines that are part of the service-provisioning-chain. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Lack of integrity verification during boot or runtime |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | All services |
| **Potential mitigations** | REC-IMGP Image protection  REC-LOG Logging, Monitoring and Alerting  REC-RA Remote Attestation  REC-SB Secure Boot  REC-SNFLC Security VNF/CNF lifecycle |

|  |  |
| --- | --- |
| **Threat ID** | T-GEN-04 |
| **Threat title** | Lack of Authentication & Authorization in interfaces between O-Cloud components |
| **Threat description** | O-Cloud deploys CNF applications as containers in a cluster of physical nodes which may be spanned across geographical locations. Owing to the Service Based Architecture of CNFs, this introduces several service endpoints communicating across each other over the network (container to container, container to cloud infrastructure component) and it is difficult to distinguish between a service terminating an external interface and a service exposing only an internal interface.  Multi-tenant deployments and deployments in public cloud also require the CNF applications to run alongside unknown entities. In such deployment scenarios, CNF service endpoints with no authentication/weak authentication expose risk of attack that can impact the availability of service and the CNF.  Lack of proper authentication in interfaces exposed by CNF services, introduces threats of lateral movement where a compromised container/rogue container:   * Can compromise the availability of internal service by bringing down the internal service and perform lateral movement of attack by exploiting the availability of other such services * Can compromise the confidentiality of the internal service by extracting critical application data |
| **Threat type** | Information disclosure, Denial of Service |
| **Vulnerabilities** | Lack of authentication, Insecure interfaces |
| **Impact type** | Availability, Confidentiality |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | All services |
| **Potential mitigations** | REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management |

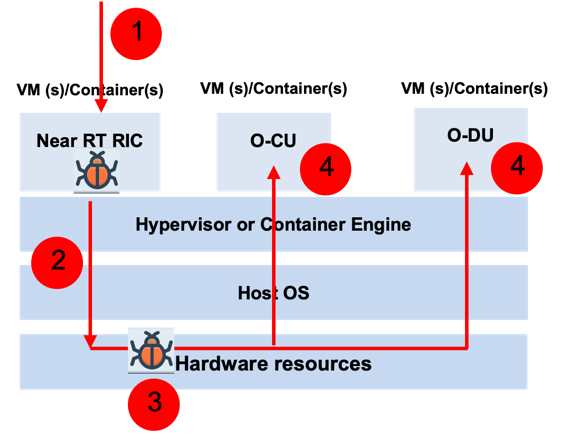
|  |  |
| --- | --- |
| **Threat ID** | T-GEN-05 |
| **Threat title** | Unsecured credentials and keys |
| **Threat description** | Adversaries may search compromised O-RAN NFs, VL, orchestration layer or hardware to find and obtain insecurely stored credentials. These credentials can be stored and/or misplaced in many locations on the O-cloud platform, including plaintext files (e.g. Bash History), operating system or application-specific repositories (e.g. Credentials in Registry), or other specialized files/artifacts (e.g. Private Keys) [i.4].  **Bash History**[[1]](#footnote-2): Adversaries may search the bash command history on compromised systems for insecurely stored credentials.  **Credentials in registry**[[2]](#footnote-3): Adversaries may search the Registry on compromised systems for insecurely stored credentials.  **Private Keys**[[3]](#footnote-4): Adversaries may search for private key certificate files on compromised systems for insecurely stored credentials. Private cryptographic keys and certificates are used for authentication, encryption/decryption, and digital signatures; Common key and certificate file extensions include: .key, .pgp, .gpg, .ppk., .p12, .pem, .pfx, .cer, .p7b, .asc.  Adversary tools have been discovered that search compromised systems for file extensions relating to cryptographic keys and certificates. |
| **Threat type** | Tampering, Information disclosure, and Elevation of privilege |
| **Vulnerabilities** | Lack of authentication, secret exposure (insufficient safeguarding of credentials) |
| **Impact type** | Integrity, Confidentiality, Authorization |
| **Threatened Assets** | Private keys and credentials |
| **Affected Services** | All services |
| **Potential mitigations** | REC-SS Secure Storage  REC-PHY Physical Security Protection  REC-SDD Secure data deletion |

|  |  |
| --- | --- |
| **Threat ID** | T-GEN-06 |
| **Threat title** | Sensitive application data cache exploitation |
| **Threat description** | Most of the applications use data that is sensitive in nature which needs to be secured. And it is common for these applications to cache such sensitive data, after retrieving the data from a secure storage. This caching occurs in various forms: within application memory, in persistent file systems, or in ephemeral file systems (For example, non-persistent container file system).  For a VNF, the application can cache the sensitive data in its memory which is non-persistent (erased when the application ceases to exist) or in the persistent virtual machine file system.  For a CNF, the application can store the sensitive data in its memory which is non-persistent (erased when the application/microservice ceases to exist) or in the non-persistent container file system, or in the persistent host file system, provided the application has the necessary privileges to access host file system.  The storage of sensitive data (For example, certificates and private keys or session keys) in the application cache is used to improve performance of the applications and keeping the sensitive information readily available to the applications for faster initialization, re-initialization or recovery.  An example of faster recovery/re-initialization is a scenario where a TLS client establishes a TLS session with a server and stores the client certificate and corresponding private key in the application cache. If the TLS session is terminated unexpectedly (For example, due to network error conditions), the TLS client uses the cached client certificates and private key for recovering / re-establishing the TLS session, instead of retrieving the information from a more secure, but slower, storage solution.  While leveraging application cache is indispensable for significant performance gains, it also presents a notable security risk [i.59][i.60]. The sensitive data stored in application cache would be a primary target from attackers/threat actors [i.61], who can exploit this information to penetrate deeper into O-RAN network functions.  To counter this threat, a “Defense in depth” approach is essential. This approach encompasses multiple security layers designed to mitigate risks associated with application caches, focusing on strong encryption, strict access management and anomaly detection to protect sensitive data effectively. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Secret exposure (insufficient safeguarding of credentials, etc.) |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | Certificates, private keys, credentials and other application specific sensitive data |
| **Affected Services** | All services |
| **Potential mitigations** | REC-NS Network Segmentation & Filter Network Traffic  REC-LOG Logging, Monitoring and Alerting  REC-SDD Secure data deletion  REC-SS Secure Storage |

#### Threats concerning VMs/Containers

|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-01 |
| **Threat title** | Abuse of a privileged VM/Container |
| **Threat description** | It’s possible to run VMs/Containers with unintended configurations. Such misconfigurations can help the adversaries to compromise even strongest of VM/Container isolation measures.  Such misconfigurations scenarios include:   * VMs/Containers can be configured to have more privileges than what is actually required (e.g. settings that give it unnecessary, and perhaps unplanned, privileges). For example, an attacker with access to such a container, can use it to gain higher privileges on host, perform un-authorized operations and get to anything that the host, or any of the containers running on that host, can reach. * VMs/Containers have unintended read/write access to a directory on host filesystem. This could allow an attacker to perform unauthorized modifications to the contents, create symbolic links to any directories or files not directly exposed by the hostPath, install SSH keys, read secrets mounted to the host, and take other malicious actions. |
| **Threat type** | Spoofing, Tampering, Information disclosure, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Misconfiguration or Insecure VM/Container configurations |
| **Impact type** | Authenticity, Integrity, Confidentiality, Availability and Authorization |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-SCONF Security Configuration  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-ISO Strong Isolation  REC-SNFLC Security VNF/CNF lifecycle  REC-NS Network Segmentation & Filter Network Traffic  REC-SS Secure Storage  REC-ADMC Admission Control Service |

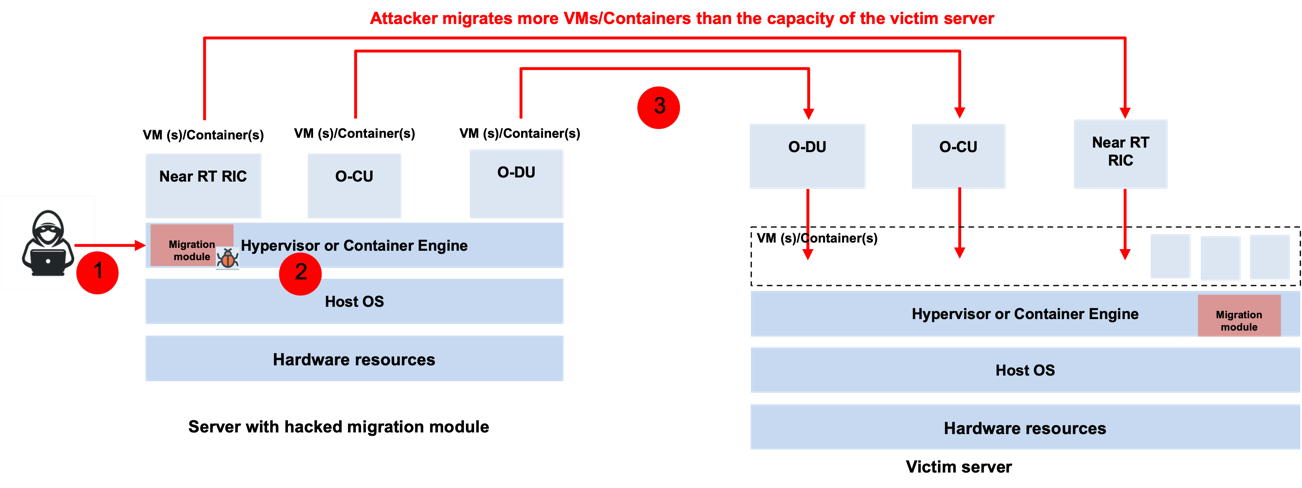
|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-02 |
| **Threat title** | VM/Container escape attack |
| **Threat description** | VNF/CNF deployed on the same physical machine as tenants share the same host kernel and host OS resources. Lack of strong isolation between the VMs/Containers and the host allows for a potential risk of a rogue VM/Container escaping the VM/Container confinement and impacting other co-hosted VMs/Containers. In others, an attacker may deploy a new malicious VM/Container configured without network rules, user limitations, etc. to bypass existing defenses within O-Cloud infrastructure.  Attacker deploys malicious VM/Container to escape the host (Hypervisor/Container Engine/Host OS) and reach the server’s hardware, then the malicious VM/Container can gain root access to the whole server where it resides. This gives the malicious VM/Container full control on all the VMs/Containers hosted on the same hacked server. This could allow an attacker to undermine the confidentiality, integrity and/or availability of VNFs/CNFs resources.  Containers can be deployed by various means, such as via Docker's create and start APIs or via a web application such as the Kubernetes dashboard or Kubeflow. Adversaries may deploy containers based on retrieved or built malicious images or from benign images that download and execute malicious payloads at runtime.  When a malicious VM/Container escapes isolation, it can gain full control over the underlying host and cause any of the below serious threats:   * Attacker would gain the ability to mount attacks on the host or compromise the host functionalities * Compromise the confidentiality & integrity of co-hosted VMs/Containers and tenants * Launch DDOS attacks on co-hosted VMs/Containers and host services thereby degrading their performance * Introduce new vulnerabilities in host to be used for future attacks * Lack of network segmentation could potentially expose other VMs/Containers in the environment to attack. An example of this could be reconnaissance, exploitation and subsequent lateral movement to another host within the cluster. |
| **Threat type** | Spoofing, Tampering, Information disclosure, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Shared tenancy vulnerabilities (multitenant environment), Lack of strong VM/Container isolation, lack of authentication, Insecure networking, Unrestricted communication between VMs/Containers |
| **Impact type** | Authenticity, Integrity, Confidentiality, Availability and Authorization |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-ISO Strong Isolation  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-NS Network Segmentation & Filter Network Traffic  REC-LOG Logging, Monitoring and Alerting  REC-SCONF Security Configuration  REC-SNFLC Security VNF/CNF lifecycle  REC-SS Secure Storage |



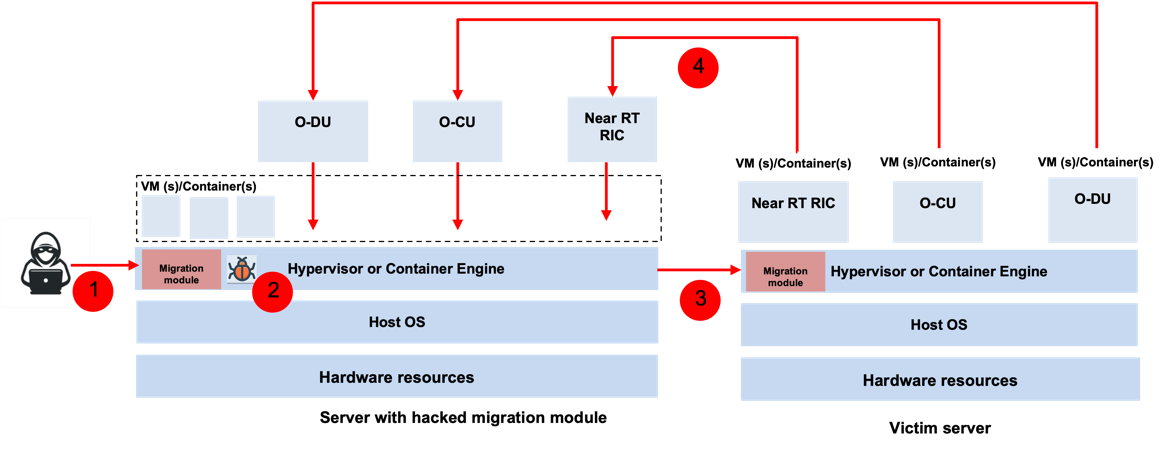
**Figure 7‑4 : Illustration of the VM/Container escape attack**

|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-03 |
| **Threat title** | VM/Container data theft |
| **Threat description** | The VNF/CNF remotely stores sensitive data (e.g. passwords, private keys, subscription data, logs) on the logical volume that the IMS/DMS allocates to the VNF/CNF. An attacker can retrieve/manipulate these data if they have been stored in an insecure way (e.g. clear text, unsalted hashes) or a malware is installed on the logical volume that the VIM allocates to the VNF/CNF.  Container example: Adversaries may attempt to discover containers and other resources that are available locally within O-Cloud. Other resources may include images, deployments, pods, nodes, and other information such as the status of a cluster. These resources can be viewed within web applications such as the Kubernetes dashboard or can be queried via the Docker and Kubernetes APIs. In Docker, logs may leak information about the environment, such as the environment’s configuration, which services are available, and what cloud provider the victim may be utilizing. The discovery of these resources may inform an adversary’s next steps in the environment, such as how to perform lateral movement and which methods to utilize for execution. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Lack of authentication, insecure data storage |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | Images, environment’s configuration, and other information such as the status of a cluster |
| **Affected Services** | SERV#01, SERV#02 |
| **Potential mitigations** | REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-NS Network Segmentation & Filter Network Traffic  REC-LOG Logging, Monitoring and Alerting  REC-PHY Physical Security Protection  REC-SS Secure Storage  REC-SDD Secure data deletion |

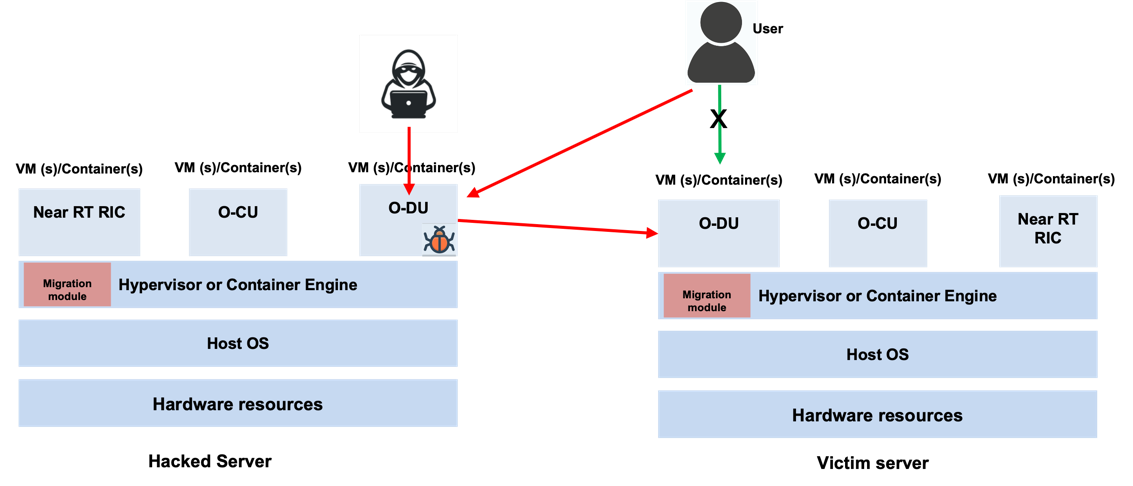
|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-04 |
| **Threat title** | VM/Container migration attacks |
| **Threat description** | The attacks that exploit VM/Container migration can be divided into two subcategories based on the target plane:   1. **Control Plane Attacks**: These attacks target the module that is responsible for handling the migration process on a server which is called the migration module that is found in the host. By exploiting a bug in the migration module software, the attacker can hack the server and take full control over the migration module. This gives the attacker the ability to launch malicious activities including the following:    1. **Migration Flooding**: The attacker moves all the VMs/Containers that are hosted on the hacked server to a victim server that does not have enough resource capacity to host all the moved VMs/Containers. This causes a denial of service for the VNFs/CNFs running in the VMs/Containers of the victim server as there will not be enough resources to satisfy the demands of all the hosted VMs/Containers leading into VM/Container performance degradation and VM/Container crashes.    2. **False Resource Advertising**: The hacked server claims that it has a large resource slack (a large amount of free resources). This attracts other servers to off-load some of their VMs/Containers to the hacked server so that the O-Cloud workload gets distributed over the O-Cloud servers. After moving VMs/Containers from other servers to the hacked server, the attacker can exploit other vulnerabilities to break into the offloaded VMs/Containers as now these VMs/Containers are placed on a server that is under the control of the attacker. 2. **Data Plane Attacks**: These constitute the second type of VM/Container migration attacks and those attacks target the network links over which the VM/Container is moved from a server to another. Such data plane attacks include the MitM where an attacker sniffs the packets that are exchanged between the source and destination servers and reads the migrated memory pages. The attacker can monitor and/or modify the received packets while continuing to forward them to victim VM/Container resides so that the victim does not detect that any malicious activity is going on. |
| **Threat type** | Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Host misconfiguration, lack of authentication, memory pages copied in clear, vulnerable code exploits |
| **Impact type** | Integrity, Confidentiality, Availability |
| **Threatened Assets** | VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-CM Certificate management  REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication and Access Management  REC-SNFLC Security VNF/CNF lifecycle  REC-LOG Logging, Monitoring and Alerting  REC-RA Remote Attestation |



**Figure 7‑5 : Illustration of the migration flooding attack**

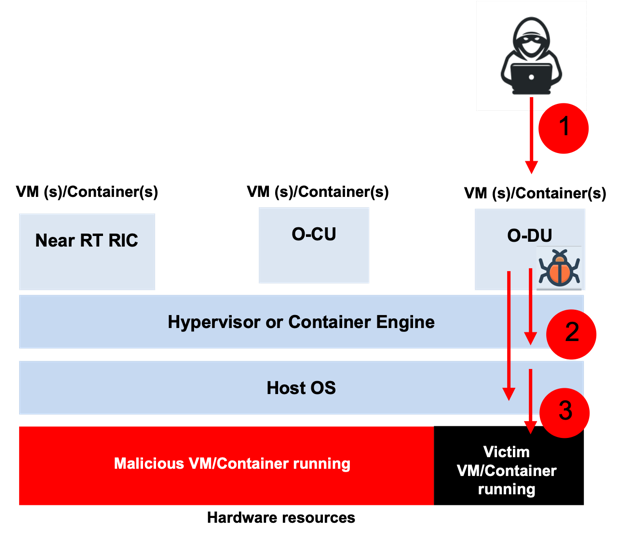


**Figure 7‑6 : Illustration of the false resource advertising attack**



**Figure 7‑7 : Illustration of the migration MITM attack**

|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-05 |
| **Threat title** | Changing virtualization resource without authorization |
| **Threat description** | IMS/DMS which manage the Virtualization layer is responsible for assigning virtualized resource as requested.  There are several ways to cause a DoS attack for the VNFs/CNFs:   * If IMS/DMS are compromised or the O2 interface is not securely protected, an attacker who compromised the IMS/DMS or breached the O2 interface can change the virtualized resource used by a VNF/CNF by manipulating the allocation of virtualized resource. For example, when an instantiated VNF/CNF is running, adversaries having access to a compromised IMS/DMS or adversaries breaching the insecure O2 interface can misguide the Virtualization layer to reduce the resource of or delete a VM/Container on which a VNF/CNF is running. This can result in the reliability, availability or even illegal termination of a VNF/CNF and hence the denial of service. * Hardware resource configuration and state information (e.g. events) exchange is performed through O2 interface. If the IMS is compromised or the O2 interface is not securely protected, an attacker who compromised the IMS or breached the O2 interface can tamper the hardware configuration and state information so that the virtualized resource supported by the hardware layer becomes unreliable. * Adversaries having access to a compromised virtualization layer can change the virtualization resource used by the instantiated VNF/CNF without authorization, * A malicious VM/Container deployed for one instance of a VNF/CNF on a host can illegally occupy the resources of the instantiated VNF/CNF deployed on the same host, resulting in resource limitation of the instantiated VNF/CNF   In this type of attacks, the extra allocation of resources for the malicious VM/Container comes at the expense of the other VMs/Containers that share the same server as the malicious VM/Container, where these victim VMs/Containers get allocated less share of resources than what they should actually obtain, which in turn degrades their performance. |
| **Threat type** | Denial of Service |
| **Vulnerabilities** | Insecure O1/O2 interfaces, Lack of authentication/access control on IMS/DMS |
| **Impact type** | Availability |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01 |
| **Potential mitigations** | REC-LOG Logging, Monitoring and Alerting  REC-CM Certificate management  REC-IAM Identity, Authentication and Access Management  REC-ISO Strong Isolation  REC-PHY Physical Security Protection  REC-AUD Security Audit |



**Figure 7‑8 : Illustration of the Theft-of-Service/DoS Attack**

|  |  |
| --- | --- |
| **Threat ID** | T-VM-C-06 |
| **Threat title** | Failed or incomplete VNF/CNF termination or releasing of resources |
| **Threat description** | A misconfigured VNF/CNF is instantiated in the O-Cloud infrastructure to access data not erased from a terminated VNF/CNF or any VNF/CNF that has released resources. Data could include application data, cryptographic keys, etc.  Abuse of resources allocation in the O-Cloud infrastructure to allocate to a malicious VNF/CNF the virtual resources released from a terminated VNF/CNF or from a VNF/CNF that has released resources after a move or a scaling process.  Inclusion of concealed software in the O-Cloud infrastructure to prevent the deletion/erasure of data and states of the VNF/CNF that has been terminated. |
| **Threat type** | Information disclosure |
| **Vulnerabilities** | Lack of authentication, misconfigurations (VNF/CNF, Host OS, Hypervisor/Container Engine) |
| **Impact type** | Confidentiality |
| **Threatened Assets** | Sensitive data (e.g. passwords, private keys, subscription data, logs), VNFs/CNFs (Near RT RIC, O-CU, O-DU) software and data |
| **Affected Services** | No affected services |
| **Potential mitigations** | REC-SNFLC Security VNF/CNF lifecycle  REC-LOG Logging, Monitoring and Alerting  REC-SS Secure Storage  REC-SCONF Security Configuration |

#### Threats concerning VM/Container images

|  |  |
| --- | --- |
| **Threat ID** | T-IMG-01 |
| **Threat title** | VM/Container images tampering |
| **Threat description** | An attacker can inject malicious code or tamper the information inside the unprotected image during on boarding. Then after the instantiation of the VNF/CNF, the tampered code can cause DoS, information stealing, fraud and so on. There are several attacks categories belonging to this threat. Such attacks include:   * Build machine attacks: If an attacker can modify or influence the way a VM/Container image is built, they could insert malicious code that will subsequently get run in the production environment. * Supply chain attacks: Once the VM/Container image is built, it gets stored in a registry, and it gets retrieved or “pulled” from the registry at the point where it’s going to be run. An attacker who can replace an image or modify an image between build and deployment could run arbitrary code on your deployment. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Compromised VM/Container images (Build machine attacks, Supply chain attacks) at rest, lack of authentication, misconfiguration or Insecure VM/Container images configurations |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | VM/Container images |
| **Affected Services** | SERV#02 |
| **Potential mitigations** | REC-AUD Security Audit  REC-LOG Logging, Monitoring and Alerting  REC-IMGP Image Protection  REC-VHPM Vulnerability Handling and Patch Management  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-SDLC Secure Development Lifecycle  REC-ADMC Admission Control Service |

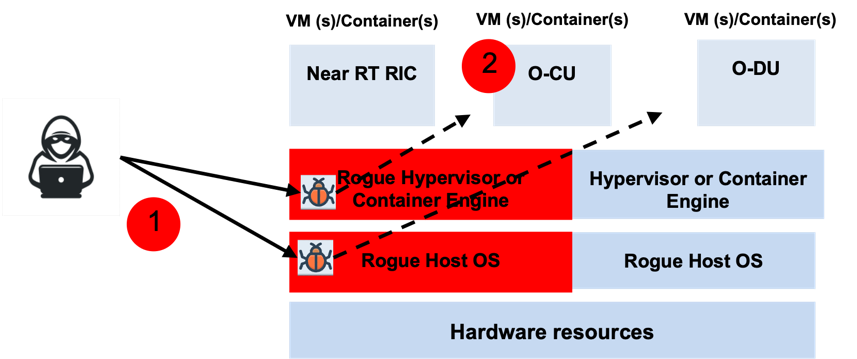
|  |  |
| --- | --- |
| **Threat ID** | T-IMG-02 |
| **Threat title** | Insecure channels with images repository |
| **Threat description** | Images often contain sensitive components like an organization’s proprietary software, and embedded secrets and administrator credentials. If connections to registries are performed over insecure channels, man-in-the-middle attacks could intercept network traffic and therefore the contents integrity and confidentiality of images may be compromised. There is also an increased risk of man-in-the-middle attacks that could intercept network traffic intended for registries and steal developer or administrator credentials within that traffic. Thus, could be used to provide fraudulent or outdated images to orchestrators, etc. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Compromised VM/Container images in transit |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | VMs/Containers images |
| **Affected Services** | SERV#02 |
| **Potential mitigations** | REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-IMGP Image Protection  REC-LOG Logging, Monitoring and Alerting |

|  |  |
| --- | --- |
| **Threat ID** | T-IMG-03 |
| **Threat title** | Secrets disclosure in VM/Container images |
| **Threat description** | There are scenarios which benefit from including configuration and secrets, such as passwords or credentials in VNFs/CNFs images. For e.g. VMs/Containers require to be able to connect to other VMs/Containers within the deployment as well as with external entities. All these connections need to be authenticated and secured. One way of achieving this is to provide the requisite secrets or keys to the VMs/Containers which allow them to authenticate, be authenticated, secure the communication channel and signature. A common but in-secure means of providing secrets to the VMs/Containers is by packaging the secrets or the keys with the image itself. There is the risk that the same can be extracted, read or manipulated before the VM/Container is deployed and the secret used.  With a long supply chain, VM/Container images are vulnerable to outside scrutiny. With VM/Container images containing secrets or keys, this becomes a serious threat vector. Adversaries can extract them by obtaining a copy of the image and they can be potentially shared with third parties for illicit gain.   * Secrets embedded within a VM/Container image can be stolen. * Secrets embedded within a VM/Container image can be modified   Compromised private keys and algorithms used for image signing due to poor key protection/management/design could undermine the security of image signing process. |
| **Threat type** | Spoofing, Tampering, Information disclosure |
| **Vulnerabilities** | Secret exposure in VNF/CNF images |
| **Impact type** | Authenticity, Integrity, Confidentiality |
| **Threatened Assets** | VM/Container images |
| **Affected Services** | SERV#02 |
| **Potential mitigations** | REC-AUD Security Audit  REC-IMGP Image Protection  REC-SS Secure Storage  REC-SDLC Secure Development Lifecycle |

|  |  |
| --- | --- |
| **Threat ID** | T-IMG-04 |
| **Threat title** | Build image on VL |
| **Threat description** | Adversaries may build a VM/Container image directly on the VL to bypass defenses that monitor for the retrieval of malicious images from a registry.  Container example: A remote build request may be sent to the Docker API that includes a Dockerfile that pulls a vanilla base image, such as alpine, from a public or local registry and then builds a custom image upon it.  An adversary may take advantage of that build API to build a custom image on the host that includes malware downloaded from their C2 server, and they then may deploy container using that custom image. If the base image is pulled from a public registry, defenses will likely not detect the image as malicious since it’s a vanilla image. If the base image already resides in a local registry, the pull may be considered even less suspicious since the image is already in the environment. |
| **Threat type** | Spoofing, Tampering, Information disclosure, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Host misconfiguration, lack of authentication |
| **Impact type** | Authenticity, Integrity, Confidentiality, Availability and Authorization |
| **Threatened Assets** | VM/Container images |
| **Affected Services** | SERV#02, SERV#05 |
| **Potential mitigations** | REC-AUD Security Audit  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-LOG Logging, Monitoring and Alerting  REC-IMGP Image Protection  REC-SB Secure Boot  REC-SS Secure Storage  REC-RA Remote Attestation |

#### Threats concerning the virtualization layer (Host OS-Hypervisor/Container engine)

|  |  |
| --- | --- |
| **Threat ID** | T-VL-01 |
| **Threat title** | VM/Container hyperjacking attack |
| **Threat description** | VMs/Containers run on host machines, and it is needed to ensure that those hosts (Hypervisor/Container Engine/Host OS- are not running vulnerable code (for example, old versions of components with known vulnerabilities).  Hyperjacking is an attack in which adversaries gain control over the host of a server or install a malicious Hypervisor/Container Engine/Host OS and exploit that to run malicious applications on the VM/Container that run on top of the host. This would enable the attacker to control all the VMs/Containers running on the host.  Hyperjacking involves installing a malicious, fake the Hypervisor/Container Engine/Host OS that can manage the entire server system. If the attacker gains access to the Hypervisor/Container Engine/Host OS, everything that is connected to that server can be manipulated. The Hypervisor/Container Engine/Host OS represents a single point of failure when it comes to the security and protection of sensitive information.  For a hyperjacking attack to succeed, an attacker would have to take control of the Hypervisor/Container Engine/Host OS by the following methods:   * Injecting a rogue Hypervisor/Container Engine or Host OS beneath the original hypervisor or on top of an existing Hypervisor/Container Engine/Host OS * Directly obtaining control of the original Hypervisor/Container Engine or Host OS * Running a rogue hypervisor on top of an existing hypervisor |
| **Threat type** | Spoofing, Tampering, Information disclosure, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Host misconfiguration, lack of authentication |
| **Impact type** | Authenticity, Integrity, Confidentiality, Availability and Authorization |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-SCONF Security Configuration  REC-SB Secure Boot  REC-VHPM Vulnerability Handling and Patch Management  REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-ISO Strong Isolation  REC-SS Secure Storage |



**Figure 7‑9 : Illustration of the VM/Container hyperjacking attack**

|  |  |
| --- | --- |
| **Threat ID** | T-VL-02 |
| **Threat title** | Boot tampering |
| **Threat description** | The bootloader of the virtualization layer (Host OS, Hypervisor, Container Engine) for VNF/CNF may be maliciously tampered by an attacker, e.g. the attacker compromises hypervisor or host OS to tamper the bootloader of guest OS (in case of VM) or Container.  In a O-Cloud environment any failure during the boot sequence can result in a number of situations that need to be handled by the NFO/FOCOM:  • failure of the physical machine to start at all  • physical machine entering a safe-mode  • physical machine continuing boot regardless of the integrity measurements |
| **Threat type** | Tampering |
| **Vulnerabilities** | Host misconfigurations, lack of authentication |
| **Impact type** | Integrity |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-SB Secure Boot  REC-PHY Physical Security Protection  REC-RA Remote Attestation  REC-LOG Logging, Monitoring and Alerting |

|  |  |
| --- | --- |
| **Threat ID** | T-VL-03 |
| **Threat title** | Attack internal network services |
| **Threat description** | In addition to attacking the network between containers, adversaries can also attack supporting services such as Kubernetes DNS service, which is only reachable from within the cluster network. The highly distributed nature of containers requires shared services for example for coordination and service discovery. An attacker can target these services to degrade services. For example, a denial-of-service against the service discovery infrastructure could prevent O-Cloud to react to changing resource requirements properly. Thus, O-Cloud may no longer be able to scale appropriately to sudden demand spikes [i.37]. |
| **Threat type** | Denial of Service |
| **Vulnerabilities** | Lack of authentication, vulnerable code exploits, design weakness, misconfiguration |
| **Impact type** | Availability |
| **Threatened Assets** | Software components at runtime and their associated data |
| **Affected Services** | SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-ISO Strong Isolation  REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication and Access Management  REC-ADMC Admission Control Service |

#### Threats concerning O-Cloud interfaces

##### O2 interface

Two main interfaces are defined in O-RAN WG6 specification and identified as critical assets of O-Cloud, i.e. interfaces O2 between O-Cloud and SMO. The threats on these interfaces are as follows.

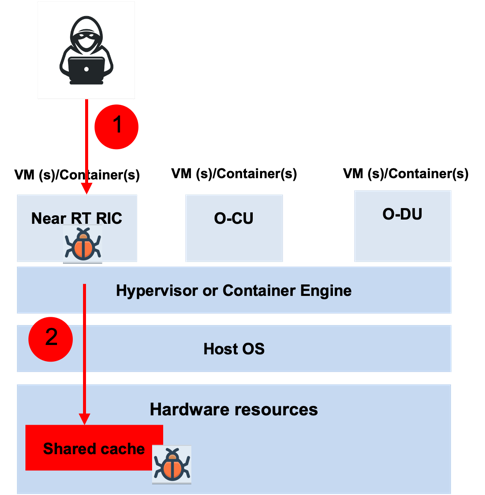
|  |  |
| --- | --- |
| **Threat ID** | T-O2-01 |
| **Threat title** | MitM attacks on O2 interface between O-Cloud and SMO |
| **Threat description** | If the interface O2 interface is not protected, an attacker can attack all the requests/responses sent between the O-Cloud and the SMO (FOCOM and NFO).  For example, the attacker can tamper/alter/disclose requests and services (See ‘Critical services’ in 2.3) sent over O2 between O-Cloud and SMO, hence the virtualized resource or relevant status information is not as requested. This affects the normal operation of the O-Cloud, and even causes DoS attacks, information leakage.  An attacker can tamper the specific assignment of virtualized resources to cause resource assignment errors or an attacker can intercept virtualized resources state information leading to information disclosure.  An attacker can compromise IMS to tamper with the hardware state information (e.g. deleting hardware alarm information) to affect the hardware's operation or to result in information disclosure (e.g. an attacker can get the hardware configuration from the compromised IMS. Then, the attacker can attack the hardware according to the configuration such as CPU type, memory size etc.). An attacker can also tamper or intercept the hardware resource configuration and state information if the configuration and state information are transmitted using an insecure protocol on the O2 interface. |
| **Threat type** | Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Insecure O2 interface, lack authentication |
| **Impact type** | Integrity, Confidentiality, Availability |
| **Threatened Assets** | Telemetry, provisioning, logs, software management, performance information |
| **Affected Services** | SERV#01 |
| **Potential mitigations** | REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-SCONF Security Configuration |

##### O-Cloud API

|  |  |
| --- | --- |
| **Threat ID** | T-OCAPI-01 |
| **Threat title** | MitM attacks on O-Cloud interface between VNFs/CNFs and the virtualization layer |
| **Threat description** | An attacker can attack an instantiated VNF/CNF through a compromised virtualization layer. For example, cryptographic keys or other security critical data of an instantiated VNF/CNF could be stolen by an attacker with access to the virtualization layer, or the virtualized resource provided by the Virtualization layer to the instantiated VNF/CNF can be manipulated or the bootloader of Guest OS (in case of VM) or Container of an instantiated VNF/CNF can be tampered by an attacker via a compromised virtualization layer. |
| **Threat type** | Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Insecure O-Cloud APIs, lack of authentication |
| **Impact type** | Integrity, Confidentiality, Availability |
| **Threatened Assets** | Host OS, Hypervisor/Container Engine and VNFs/CNFs (Near RT RIC, O-CU, O-DU) |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-SCONF Security Configuration  REC-SS Secure Storage  REC-PHY Physical Security Protection |

#### Threats concerning hardware resources

|  |  |
| --- | --- |
| **Threat ID** | T-HW-01 |
| **Threat title** | Cross VM/Container side channel attacks |
| **Threat description** | In a typical cross-VM/Container side channel attack scenario, an adversary places a malicious VM/Container co-resident to the target VM/Container so that they share the same hardware resources. Then, the attacker extracts useful information such as cryptographic keys from the target VM/Container to use them for traffic eavesdropping and man-in-the-middle attacks. Through the side channel attack, an attacker sharing the same cache as the victim can monitor the cache access behavior of the victim. For example, the attacker is able to monitor cache timing information by measuring the execution of different operations on the victim’s VM/Container. Generally, the attacker exploits timings in the shared high-level cache memory. However, power consumption or electromagnetic leaks can also be used as a vector to launch side channel attacks.  In the virtual environment, prior to the cross-VM/Container side channel attack, the attacker needs to identify the target VM/Container’s location and place a malicious VM/Container co-resident with the target. Later, that attacker may use the maliciously placed VM/Container to extract information from the target VM/Container with the side channel attack.  Hardware vulnerabilities in processors can also have a large impact on O-Cloud security. Flaws in chip design can result in the compromise of tenant information in the cloud through side-channel attacks[[4]](#footnote-5). |
| **Threat type** | Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Flaws in chip design, use of shared hardware, Lack of isolation, lack of authentication |
| **Impact type** | Integrity, Confidentiality, Availability |
| **Threatened Assets** | Sensitive data (e.g. passwords, private keys, subscription data, logs), VNFs/CNFs (Near RT RIC, O-CU, O-DU) related data |
| **Affected Services** | SERV#01, SERV#03, SERV#04, SERV#05, SERV#06, SERV#07 |
| **Potential mitigations** | REC-ISO Strong Isolation  REC-PHY Physical Security Protection  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-SS Secure Storage |



**Figure 7‑10 : Illustration of a cross VM/Container side channel attack**

|  |  |
| --- | --- |
| **Threat ID** | T-HW-02 |
| **Threat title** | MitM attacks on the interface between virtualization layer and hardware |
| **Threat description** | An attacker can utilize the vulnerabilities of hardware (e.g. Meltdown and Specter of CPU in host) to attack virtualization layer and/or VNFs/CNFs through this interface, resulting in tampering, information disclosure or DoS. |
| **Threat type** | Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Insecure interfaces between HW and VL layers, lack of authentication, misconfiguration |
| **Impact type** | Integrity, Confidentiality, Availability |
| **Threatened Assets** | All assets |
| **Affected Services** | All services |
| **Potential mitigations** | REC-PHY Physical Security Protection  REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management |

#### Threats concerning O-Cloud management (SMO, NFO, FOCOM)

|  |  |
| --- | --- |
| **Threat ID** | T-ADMIN-01 |
| **Threat title** | Denial of service against NFO/FOCOM |
| **Threat description** | A denial-of-service attack against the NFO/FOCOM can interfere with the ability of operators to control and maintain their deployments. This can lead to the inability to react to changing resource requirements. In addition, the NFO/FOCOM is the external API to interact with the O-Cloud platform. Thus, other services may become inaccessible as well. For example, operators may be unable to retrieve logs, telemetry data. An attacker could use this opportunity to hide additional attacks on VM/Container instances.  In addition, an attacker on the NFO/FOCOM could prevent the O-Cloud software update to exploit a known security flaw in the O-Cloud software. |
| **Threat type** | Denial of Service |
| **Vulnerabilities** | Lack of authentication, vulnerable code exploits, design weakness, insecure O2 interface |
| **Impact type** | Availability |
| **Threatened Assets** | VM/Container images, VNF/CNF, host OS, Hypervisor/Container Engine software |
| **Affected Services** | SERV#02, SERV#05 |
| **Potential mitigations** | REC-IAM Identity, Authentication and Access Management  REC-CM Certificate management  REC-AUD Security Audit  REC-SCONF Security Configuration |

|  |  |
| --- | --- |
| **Threat ID** | T-ADMIN-02 |
| **Threat title** | Abuse an O-Cloud administration service |
| **Threat description** | Usually, the SMO including NFO/FOCOM is exposed to the tenant in a web front-end or REST API. In case these interfaces contain software vulnerabilities or implement authentication and authorization insufficiently, an adversary would be able to gain access to the VM/Container management and pose as a tenant. It is also possible that an adversary gains the ability to submit requests without prior authentication and authorization.  The NFO/FOCOM interfaces encompass a great deal of privileges because anyone gaining sufficient access is able to deploy new instances and disrupt existing O-Cloud services. It may also be possible for an adversary to submit compromised VM/Container images that unsuspecting tenants then use to initiate O-Cloud services. Moreover, adversaries can use the same access to extract business data, configuration data, user data and possibly credentials. For example, they may be able to create backups of VM/Container instances or they can export VM/Container images. The impact of compromised credentials is exacerbated by the fact that weak and insufficient safeguarding of credentials is recognized as one of the top threats in cloud computing [i.38].  Container example: Adversaries may abuse a container administration service to execute commands within a container. A container administration service such as the Docker daemon, the Kubernetes API server, or the kubelet may allow remote management of containers within an environment.  Container example: In Docker, adversaries may specify an entrypoint during container deployment that executes a script or command, or they may use a command such as *docker exec* to execute a command within a running container. In Kubernetes, if an adversary has sufficient permissions, they may gain remote execution in a container in the cluster via interaction with the Kubernetes API server, the kubelet, or by running a command such as *kubectl exec*. |
| **Threat type** | Tampering, Information disclosure, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Lack of authentication, secret exposure (insufficient safeguarding of credentials), vulnerable code exploits, design weakness |
| **Impact type** | Integrity, Confidentiality, Availability, Authorization |
| **Threatened Assets** | VNF/CNF, host OS, Hypervisor/Container Engine software and related information |
| **Affected Services** | SERV#03, SERV#04, SERV#06, SERV#07 |
| **Potential mitigations** | REC-NS Network Segmentation & Filter Network Traffic  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-AUD Security Audit  REC-SCONF Security Configuration  REC-VHPM Vulnerability Handling and Patch Management |

#### Threats concerning Acceleration Abstraction Layer (AAL)

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-01 |
| **Threat title** | Attacker exploits insecure API to gain access to hardware accelerator resources |
| **Threat description** | Insecure AAL API allows an attacker to tamper the requests/responses sent between the AAL components, the O-Cloud platform and O-RAN APPs/VNFs/CNFs.  For example, the attacker can tamper requests and services sent over AALI-C-Mgmt between IMS and the hardware accelerator manager, hence capability of the hardware accelerator device, fault information, logs, performance information and others are not as requested. This affects the normal operation of the O-Cloud, and even causes tampering.  An attacker can tamper application (e.g. O-DU) requests sent over AALI-C-App to an AAL implementation for allocation of buffers. This affects the normal operation of the applications, and even causes tampering.  An attacker can tamper with application (e.g. O-DU) requests sent over AALI-P for configuring and managing the AAL-LPU(s). This affects the normal operation of the applications, and even causes tampering. |
| **Threat type** | Tampering |
| **Vulnerabilities** | Insecure AAL APIs and interfaces, lack of authentication and authorization |
| **Impact type** | Integrity |
| **Threatened Assets** | Data in transit AALI-C-App & AALI-P interfaces, Data in transit vendor specific interface, AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#04, SERV#06, SERV#07 |
| **Potential mitigations** | REC-IAM Identity, Authentication and Access Management  REC-LOG Logging, Monitoring and Alerting |

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-02 |
| **Threat title** | Internal Overload DoS attack targeting AAL services |
| **Threat description** | Overload situation could appear in the case of DoS attack or increased traffic on AAL interfaces. Inability to mitigate traffic volumetric attacks on AAL affects availability of AAL data and services.  DoS attacks on the AALI-C interface affect the different services provided by the hardware accelerator manager and the transport abstraction framework.  DoS attacks on the AALI-P interface affect the configuration and management of AAL-LPU (Acceleration Abstraction Layer Logical Processing Unit) by an application (e.g. O-DU) in addition to acceleration functionality. |
| **Threat type** | Denial of Service |
| **Vulnerabilities** | Insecure AAL APIs and interfaces, lack of authentication and authorization |
| **Impact type** | Availability |
| **Threatened Assets** | AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#04, SERV#06, SERV#07 |
| **Potential mitigations** | REC-IAM Identity, Authentication and Access Management  REC-LOG Logging, Monitoring and Alerting |

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-03 |
| **Threat title** | Fail to clear resources |
| **Threat description** | Failure to clear accelerator resources after a process termination can cause an information leakage and incorrect results for computations. Further, failure to release accelerator resources may prevent other processes from running.  This threat is relevant to accelerator resources either inside the hardware accelerator device (internal memories, registers, cache) or in the O-Cloud memories used by accelerators. |
| **Threat type** | Information disclosure, Denial of service |
| **Vulnerabilities** | Lack of secure deletion of data after process termination, Insecure AAL APIs, Flaws in AAL design |
| **Impact type** | Confidentiality, Availability |
| **Threatened Assets** | AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#06, SERV#07 |
| **Potential mitigations** | REC-LOG, REC-SS, REC-PHY, REC-SDD |

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-04 |
| **Threat title** | HAM compromise |
| **Threat description** | A malicious actor can gain access to HAM to gain unauthorized access and control of the hardware accelerator device. This can result in the disruption of services (DoS) and tampering of accelerator components, such as firmware, drivers which can cause the accelerator to behave abnormally or crash altogether. |
| **Threat type** | Tampering, Denial of service |
| **Vulnerabilities** | Lack of access control, Insecure HAM APIs, Flaws in HAM design |
| **Impact type** | Integrity, Availability |
| **Threatened Assets** | Data in transit AALI-C-App & AALI-P interfaces, Data in transit vendor specific interface, AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#06, SERV#07 |
| **Potential mitigations** | REC-LOG, REC-SS, REC-PHY, REC-SDD |

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-05 |
| **Threat title** | Malicious memory accesses |
| **Threat description** | AAL that allows one process running on the hardware accelerator device to access memory owned by another process running on the hardware accelerator device can leak information (impact on confidentiality).  Similarly, AAL allowing concurrently executing processes to write to one another’s memory may have correctness errors (impact on integrity).  If multiple processes are running concurrently and one is allowed to dominate accelerator resources, the other may suffer from degraded performance. For example, if one process can evict all cache entries belonging to the other, the victim will suffer performance penalties (impact on availability). |
| **Threat type** | Information disclosure, Tampering, Denial of service |
| **Vulnerabilities** | Unrestricted memory access, Lack of access control, Insecure AAL APIs, Flaws in AAL design |
| **Impact type** | Confidentiality, Integrity, Availability |
| **Threatened Assets** | AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#06, SERV#07 |
| **Potential mitigations** | REC-LOG, REC-SS, REC-PHY, REC-SDD, REC-ISO |

|  |  |
| --- | --- |
| **Threat ID** | T-AAL-06 |
| **Threat title** | Firmware attacks |
| **Threat description** | Hardware accelerators often have their own firmware, which can be targeted by attackers. This could include modifying the firmware to introduce vulnerabilities (e.g., malware) or installing malicious firmware to extract/modify sensitive information or execute unauthorized actions (e.g., control the device remotely). |
| **Threat type** | Information disclosure, Tampering |
| **Vulnerabilities** | Weak accelerator design, Misconfiguration, Insecure AAL/HAM APIs |
| **Impact type** | Confidentiality, Integrity |
| **Threatened Assets** | AAL profiles, AAL-LPUs, Stored AAL data, AAL software, Device firmware |
| **Affected Services** | SERV#03, SERV#06, SERV#07 |
| **Potential mitigations** | REC-LOG, REC-SS, REC-PHY, REC-SDD, REC-ISO |

#### Threats concerning O-Cloud instance ID

|  |  |
| --- | --- |
| **Threat ID** | T-O-CLOUD-ID-01 |
| **Threat title** | ID reuse in O-Cloud's object lifecycle |
| **Threat description** | In O-Cloud, objects such as Containers, Pods, Nodes, and Services are identified by their IDs within a given compute pool (e.g., cluster in Kubernetes). When an object is deleted, its ID becomes available for reuse. This means that a new object can be created with the same ID as a previously deleted object. If an object gets deleted but all its associated data isn't properly isolated or cleaned, the ID, if reused, could lead to unintended data associations or leaks.  Potential consequences:   * Data Residue: A new object, reusing an ID, may inherit residual data or configurations from its predecessor, leading to potential misconfigurations and incorrect data associations. This can result in sensitive data exposure. * Data Overwrite: Automated processes unaware of the deletion and subsequent recreation might mistakenly write or read data from the new object, thinking it's the old one. * Monitoring Ambiguities: Monitoring tools might combine metrics from the old and new objects, resulting in confusing data. * Operational Disruptions: The new object might operate based on the residual configurations of the old object, potentially leading to system inefficiencies or failures. |
| **Threat type** | Tampering, Information disclosure |
| **Vulnerabilities** | Predictable ID generation, Insufficient logging or auditing, Insufficient ID validation |
| **Impact type** | Integrity, Confidentiality |
| **Threatened Assets** | O-Cloud Provisioning information, Software components at runtime and their associated data, Security event logs |
| **Affected Services** | SERV#04, SERV#05 |
| **Potential mitigations** | REC-IDM O-Cloud ID secure management |

|  |  |
| --- | --- |
| **Threat ID** | T-O-CLOUD-ID-02 |
| **Threat title** | Node redundancy in O-Cloud deployments |
| **Threat description** | Nodes in O-Cloud often represent physical or virtual machines. If a machine fails and is replaced without deleting its corresponding Node object, and the new machine is given the same ID or the hostname, O-Cloud might treat the new machine as if it were the original.  Potential consequences:   * Resource Mismatch: The new host might have different resources (CPU, memory, storage) than the old one, leading to scheduling issues or resource constraints. * Stale Data: The new node might inherit data or configurations from the old node, leading to potential security or operational risks. * Network Issues: Network configurations or IP address assignments might be inconsistent or conflicting. |
| **Threat type** | Spoofing, Tampering, Information disclosure, Denial of Service |
| **Vulnerabilities** | Lack of node validation, Inadequate resource allocation and monitoring, Lack of network configuration verification |
| **Impact type** | Authenticity, Integrity, Confidentiality, Availability |
| **Threatened Assets** | O-Cloud Provisioning information, Software components at runtime and their associated data, Security event logs |
| **Affected Services** | SERV#04, SERV#05 |
| **Potential mitigations** | REC-IDM O-Cloud ID secure management |

|  |  |
| --- | --- |
| **Threat ID** | T-O-CLOUD-ID-03 |
| **Threat title** | O-Cloud ID mismanagement |
| **Threat description** | IDs are crucial for uniquely identifying objects within the O-Cloud. Mismanagement occurs when these IDs are not properly assigned, tracked, or validated, leading to potential overlaps or inconsistencies.  Potential consequences:   * ID Collision: Due to system glitches or bugs, two distinct objects could inadvertently be allocated the same ID. Such an occurrence is termed an ID collision. This can result in operations meant for one object inadvertently affecting the other. * Resource Overwrite: If two objects share the same ID, updates or modifications intended for one might overwrite the data of the other, leading to data inconsistencies or loss. * ID-Based Permissions: Many security protocols and access controls in O-Cloud can be tied to object IDs. If an attacker can predict, guess, or manipulate the ID generation process, they might gain unauthorized access to resources. * Log Merging: Monitoring tools and logging systems use IDs to track events and operations related to specific objects. If two objects share an ID, their logs might get merged, making it challenging to trace events back to their source. * RBAC Anomalies: Role-Based Access Control (RBAC) regulations attached to specific object IDs could unintentionally approve or restrict access to the novel object due to misidentification. |
| **Threat type** | Spoofing, Tampering, Information Disclosure, Repudiation, Elevation of Privilege, Denial of Service |
| **Vulnerabilities** | Inefficient ID assignment algorithm, Lack of validation checks during the ID assignment or object creation process, Weak or absent monitoring of ID lifecycle, Insufficient randomization in ID generation, Lack of timely updates to RBAC policies, Lack of ID collision detection |
| **Impact type** | Authenticity, Integrity, Non-repudiability, Confidentiality, Availability, Authorization |
| **Threatened Assets** | O-Cloud Provisioning information, Software components at runtime and their associated data, Security event logs |
| **Affected Services** | SERV#04, SERV#05 |
| **Potential mitigations** | REC-IDM O-Cloud ID secure management |

#### Threats concerning time synchronization

|  |  |
| --- | --- |
| **Threat ID** | T-TS-01 |
| **Threat title** | Time synchronization disruption in O-Cloud |
| **Threat description** | Disruption or manipulation of the time synchronization process within the O-Cloud can lead to severe security and operational issues. An attacker may target the time synchronization mechanisms to alter time settings across O-Cloud nodes, leading to:   * **Impersonation of time synchronization server:** An attacker may impersonate a legitimate time synchronization server, tricking nodes into accepting incorrect time data. This can cause nodes to synchronize with a malicious time source, leading to widespread time discrepancies. * **Data integrity loss:** Inconsistent timestamps can corrupt logs and audit trails within the O-Cloud, making it difficult to trace and analyze security events accurately. * **Operational disruption:** Time-sensitive operations within the O-Cloud may be disrupted, leading to data loss and service outages. * **Authentication and authorization bypass:** Many security mechanisms in the O-Cloud rely on accurate time. Disrupted time synchronization can undermine token expiration, certificate validity, and other time-based access controls, potentially allowing unauthorized access to O-Cloud resources.   **EXAMPLE:** Time-sensitive operations such as automated backups, scheduled tasks (e.g., resource scaling, patch management), and certificate renewals.  NOTE: This threat is linked to the operation of CronJob resources as described in clause 4.3.1.1.6 of the O-RAN specification [i.65]. Any disruption or manipulation of the time synchronization process can severely impact scheduled operations managed by CronJobs within the O-Cloud. |
| **Threat type** | Spoofing, Tampering, Denial of Service and Elevation of privilege |
| **Vulnerabilities** | Insecure time synchronization protocols  Insufficient authentication mechanisms and access controls for the time synchronization settings  Misconfiguration of time synchronization settings on O-Cloud nodes |
| **Impact type** | Authenticity, Integrity, Availability and Authorization |
| **Threatened Assets** | Log and audit trails  O-Cloud time synchronization settings |
| **Affected Services** | SERV#03, SERV#06, SERV#07 |
| **Potential mitigations** | REC-SCONF Security Configuration  REC-IAM Identity, Authentication, and Access Management  REC-CM Certificate management  REC-LOG Logging, Monitoring, and Alerting |

# Recommendations and best practices

A large number of useful and relevant security guidelines and best practices exist that are relevant for VM/Container-based virtualization and cloud computing in general. Therefore, we provide them here collectively and highly recommend that readers consult them for additional references:

* 3GPP TS 33.848 Study on security impacts of virtualisation
* MITRE containers matrix
* CIS Critical Security Controls
* CIS Docker Benchmark
* CIS VMWARE Benchmark
* ONAP master documentation
* CISA/NSA Kubernetes security guidance
* OpenStack security guide
* Cloud native wiki by aqua
* ANSSI “Recommandations de sécurité relatives aux déploiements de conteneur docker “
* CISA/NSA “Security guidance for 5g cloud infrastructures” parts 1 to 4
* GSMA NG.126 “Cloud Infrastructure Reference Model”
* Docker best practices
* FFTelecoms “Référentiel d'objectifs de sécurité en matière de fonctions réseau virtualisées “
* Fraunhofer AISEC Threat analysis of container-as-a-service for network function virtualization

## REC-CM Certificate management

**Recommendation**: O-Cloud should support Public Key Cryptography for the purpose of distributing Public Key Certificates (PKC) for authenticating, authorizing, and encrypting links between components in O-Cloud. Each operator should develop a Certificate Policy in accordance with their regional and national requirements. In addition, Operators should setup a renewal procedure (preferably automatically) of certificates prior to their expiration.

In O-Cloud, the following scenarios are considered for certificate usage:

* O-Cloud and VNFs/CNFs should employ certificates which can be used for images signing and verification during onboarding and registration.
* SMO and VNF/CNF should employ certificates which can be used in order to establish secure connections between them.
* SMO employs certificates in order to establish secure management connections with NFO and FOCOM.
* O-Cloud infrastructure employs certificate(s) in order to establish secure connections with NFO and FOCOM through O2 interface.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-CM-BP-1 | The certificate policy should be consistent with the Internet X.509 Certificate Policy and Certification Practices Framework as defined in IETF RFC 3647 [i.9]. |
| REC-CM-BP-2 | In order to eliminate or mitigate risks against attacks such as spoofing, tampering and information disclosure, secure connection can be established on all the interfaces within O-Cloud. IPsec and TLS mechanisms are widely deployed to protect the communication between O-Cloud components using certificates. |
| REC-CM-BP-3 | Certificate management in O-Cloud should be automated and manual operations should be avoided as much as possible. Automated certificate lifecycle management should be supported, i.e. creation, update and revocation of certificate. |
| REC-CM-BP-4 | It’s important to continuously monitor and audit certificates that are issued and active, with the ability to generate audits and keep on top of expirations and renewals to avoid any disruption in O-Cloud services. Unknown, rogue or non-complaint certificates can result in an unexpected outage, or worse, misuse that allows unintended access to O-Cloud components. |
| REC-CM-BP-5 | Securing the CAs that issue the certificates is critical. If a CA is compromised, attackers can issue their own identities that will be trusted by default across O-Cloud, and this can be extremely costly to remediate, as it effectively invalidates every identity issued by that CA. Hardware devices such as HSMs should be used to secure CA certificates and keys, ensure robust protection and complete visibility, policy enforcement, and automation for all certificates. |

For more details see ETSI GR NFV-SEC 005 [i.10].

**This recommendation can help to mitigate**: T-GEN-02, T-GEN-04, T-VM-C-01, T-VM-C-02, T-VM-C-03, T-VM-C-04, T-VM-C-05, T-IMG-01, T-IMG-02, T-IMG-04, T-VL-01, T-O2-01, T-OCAPI-01, T-HW-01, T-HW-02, T-ADMIN-01, T-ADMIN-02, T-TS-01

## REC-NS Network Segmentation & Filter Network Traffic

**Recommendation**: Physical and logical segmentation to prevent access to potentially sensitive O-Cloud components and information should be implemented. In addition, network policies should be defined to ensure a secure communication between VMs/Containers and limit the communication between VMs/Containers as much as possible to limit potential damage if a VM/Container is compromised.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-NS-BP-1 | Isolate O-Cloud infrastructure components that do not require broad network access. |
| REC-NS-BP-2 | Configure access controls, network proxies, gateways, and firewalls to limit access to:   * Critical O-Cloud components. Most cloud environments support separate virtual private cloud (VPC) instances that enable further segmentation of cloud systems. * Systems used to create and manage accounts. |
| REC-NS-BP-3 | Follow best practices for network firewall and proxies configurations to allow only necessary ports and legitimate traffic to enter and exit the network. |
| REC-NS-BP-4 | Operate intrusion detection, analysis, and response systems on a separate network from the production environment to lessen the chances that an adversary can see and interfere with critical response functions. |
| REC-NS-BP-5 | Consider filtering DNS requests to unknown, untrusted, or known bad domains and resources. Resolving DNS requests with on-premise/proxy servers may also disrupt adversary attempts to conceal data within DNS packets. |
| REC-NS-BP-6 | Enforce proxies and use dedicated servers for services such as DNS and only allow those systems to communicate over respective ports/protocols, instead of all systems within a network. |
| REC-NS-BP-7 | Consider using IP allowlisting along with user account management to ensure that data access is restricted not only to valid users but only from expected IP ranges to mitigate the use of stolen credentials to access data. |
| REC-NS-BP-8 | Apply extended ACLs to block unauthorized protocols outside the trusted network. |
| REC-NS-BP-9 | Upon identifying a compromised network device being used to bridge a network boundary, block the malicious packets using an unaffected network device in path, such as a firewall or a router that has not been compromised. Continue to monitor for additional activity and to ensure that the blocks are indeed effective. |
| REC-NS-BP-10 | When flood volumes exceed the capacity of the network connection being targeted, it is typically necessary to intercept the incoming traffic upstream to filter out the attack traffic from the legitimate traffic. Such defenses can be provided by the hosting Internet Service Provider (ISP) or by a 3rd party such as a Content Delivery Network (CDN) or providers specializing in DoS mitigations. |
| REC-NS-BP-11 | Filter network traffic to prevent use of protocols across the network boundary that are unnecessary. |
| REC-NS-BP-12 | Use of network intrusion detection and prevention systems that can   * Identify traffic patterns and use network signatures to identify traffic for specific adversary malware, can be used to mitigate malicious activity at the network level. * Detect and prevent remote service scans. |

**This recommendation can help to mitigate**: T-GEN-02, T-GEN-06, T-VM-C-01, T-VM-C-02, T-VM-C-03, T-VM-C-04, T-IMG-02, T-VL-01, T-O2-01, T-OCAPI-01, T-HW-02, T-ADMIN-02

## REC-IAM Identity, Authentication and Access Management

**Recommendation**: A strong IAM framework should be in place to ensure authentication, authorization, accounting and access control. It should be used to initiate, capture, record and manage user identities and their related access permissions to O-Cloud assets in an automated way.

The IAM framework should be set up to manage and secure access to NFO/FOCOM, VNFs/CNFs, Host OS, Hypervisor, Container Engine, data and services that reside in the O-Cloud. This framework should protect credentials and accounts associated with O-Cloud platforms. Specifically, IAM protects root accounts for servers in the O-Cloud, limits privileged access to the O-Cloud control panel and governs ongoing access to privileged resources in the O-Cloud.

IAM should be based on Authentication, Authorization, and Accounting (AAA) systems and least privileges approach to limit actions administrators can perform and provide a history of user actions to detect unauthorized use and abuse.

Protocols without encryption/authentication mechanisms should not be used. Access to administrative and management interfaces from untrusted network sources should be limited.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-IAM-BP-1 | OAuth2.0 with access token defined by IETF RFC 6749 [i.11] should be used for the authorization of API request and notifications. There are several existing Access Token solutions. Such solutions are Openstack Keystone[[5]](#footnote-6), OpenID Connect[[6]](#footnote-7), 3GPP authorization framework ETSI TS 133 501[[7]](#footnote-8), IETF TLS-Based AccessToken Binding[[8]](#footnote-9) [[9]](#footnote-10).  For more details see ETSI GR NFV-SEC 022 [i.12]. |
| REC-IAM-BP-2 | O-Cloud components must support authenticated and secure access to API, GUI and command line interfaces. In addition, they must support Traffic Filtering (for example, FireWall, IDS/IPS). |
| REC-IAM-BP-3 | O-Cloud interfaces should support secure and encrypted communications, and confidentiality and integrity of network traffic on all channels. All APIs access must use TLS protocol, including back-end APIs [i.45]. |
| REC-IAM-BP-4 | O-Cloud components should implement controls enforcing separation of duties and privileges, least privilege use and least common mechanism (e.g. Role-Based Access Control).  Communication between different trust domains is not allowed, by default.  The O-Cloud infrastructure must not allow the effect of an attack on one trust domain to impact the other domains either directly or indirectly. |
| REC-IAM-BP-5 | The O-Cloud should not reuse the same authentication credential (e.g., key-pair) on different components (e.g., on different hosts, or different services).  The O-Cloud should protect all secrets by using strong encryption techniques and storing the protected secrets externally from the component.  The O-Cloud should provide secrets dynamically as and when needed. |
| REC-IAM-BP-6 | Data Centre Operations staff and O-Cloud components should use management protocols that limit security risk. Management protocols such as SNMPv3, SSH v2, ICMP, NTP, syslog, and TLS v1.2 or higher. |
| REC-IAM-BP-7 | Remotely available services that may be unnecessary should be disabled or blocked. Remote access to internal O-Cloud resources should be controlled using centrally managed concentrators such as VPNs and other managed remote access systems of network, proxies, gateways, and firewalls.  Strong two-factor or multi-factor authentication for remote service accounts to mitigate an adversary's ability to leverage stolen credentials should be used. |
| REC-IAM-BP-8 | Ensure VM/Containers are not running as root by default. |
| REC-IAM-BP-9 | Do not allow domain administrator accounts (of Active directory) to be used for day-to-day operations that may expose them to potential adversaries on unprivileged systems.  Consider using temporary credentials for accounts that are only valid for a certain period of time to reduce the effectiveness of compromised accounts. |
| REC-IAM-BP-10 | Limit access to the root account and prevent users from loading kernel modules and extensions through proper privilege separation and limiting Privilege Escalation opportunities. |
| REC-IAM-BP-11 | Container: Run the microservice with the least privilege possible; First, never use the --privileged flag. It gives all so-called capabilities to the container, and it can access host devices (/dev) including disks, and also has access to the /sys and /proc filesystem. And with a little work the container can even load kernel modules on the host[[10]](#footnote-11). The good thing is that containers are per default unprivileged. You would have to configure them explicitly to run privileged. |
| REC-IAM-BP-12 | When Administrative Shell is necessary, restrict Administrative Shell execution policy to administrators. Be aware that there are methods of bypassing the Administrative Shell execution policy, depending on environment configuration. |
| REC-IAM-BP-13 | Least privilege principles:   * Use the principal of least privilege and protect administrative access to domain trusts such as the service account in Kubernetes, Openstacks, etc. to limit impact of exploitation. * Restrict administrator accounts to as few individuals as possible, following least privilege principles. * Limit permissions associated with creating and modifying images or VMs/Containers based on the principle of least privilege. |
| REC-IAM-BP-14 | Prevent credential overlap across systems of administrator and privileged accounts. |
| REC-IAM-BP-15 | Ensure critical system files as well as those known to be abused by adversaries have restrictive permissions and are owned by an appropriately privileged account, especially if access is not required by users nor will inhibit system functionality. |
| REC-IAM-BP-16 | Audit domain and local accounts as well as their permission levels routinely to look for situations that could allow an adversary to gain wide access by obtaining credentials of a privileged account. These audits should also include if default accounts have been enabled, or if new local accounts are created that have not been authorized. |
| REC-IAM-BP-17 | Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. |
| REC-IAM-BP-18 | Do not allow remote access to services as a privileged account unless necessary. |
| REC-IAM-BP-19 | Do not allow remote access via SSH as root or other privileged accounts. |
| REC-IAM-BP-20 | Restrict execution of particularly vulnerable binaries to privileged accounts or groups that need to use it to lessen the opportunities for malicious usage. |
| REC-IAM-BP-21 | Ensure that any accounts used by third-party providers to access these systems are traceable to the third-party and are not used throughout the network or used by other third-party providers in the same environment. Ensure there are regular reviews of accounts provisioned to these systems to verify continued business need and ensure there is governance to trace de-provisioning of access that is no longer required. Ensure proper system and access isolation for critical network systems through use of account privilege separation. |
| REC-IAM-BP-22 | A Cloud Access Security Broker (CASB) can be used to set usage policies and manage user permissions on cloud applications to prevent access to application access tokens. |
| REC-IAM-BP-23 | Password policy   * O-Cloud components that utilize default username and password should be changed immediately after the installation, and before deployment to a production environment. When possible, passwords should be updated periodically and properly secured. (Refer to NIST guidelines when creating password policies NIST Special Publication 800-63B[[11]](#footnote-12), CIS Password Policy Guide[[12]](#footnote-13)). * Use strong passwords to increase the difficulty of credential hashes from being cracked if they are obtained. * Do not reuse local administrator account passwords across systems. Ensure password complexity and uniqueness such that the passwords cannot be cracked or guessed. * In case SSH is used with password (not recommended), Ensure SSH key pairs have strong passwords and refrain from using key-store technologies such as ssh-agent unless they are properly protected. * Consider rotating access keys within a certain number of days to reduce the effectiveness of stolen credentials. * Use strong passphrases for private keys to make cracking difficult. Do not store credentials within the Registry. Establish an organizational policy that prohibits password storage in files. * Ensure that cloud accounts, particularly privileged accounts, have complex, unique passwords across all systems on the network. Passwords and access keys should be rotated regularly. This limits the amount of time credentials can be used to access resources if a credential is compromised without your knowledge. Cloud service providers may track access key age to help audit and identify keys that may need to be rotated. |

**This recommendation can help to mitigate**: T-GEN-02, T-GEN-04, T-VM-C-01, T-VM-C-02, T-VM-C-03, T-VM-C-04, T-VM-C-05, T-IMG-01, T-IMG-02, T-IMG-04, T-VL-01, T-O2-01, T-OCAPI-01, T-HW-01, T-HW-02, T-ADMIN-01, T-ADMIN-02, T-TS-01

## REC-VHPM Vulnerability Handling and Patch Management

**Recommendation**: A vulnerability handling process should be in place to find potentially exploitable software vulnerabilities, to determine what types and levels of threat may use software exploits and 0-days against O-Cloud and to remediate them.

A patch management process should be implemented to check unused dependencies, unmaintained and/or previously vulnerable dependencies, unnecessary features, components, files, and documentation.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-VHPM-BP-1 | Regularly scan O-Cloud software components for vulnerabilities and establish procedures to rapidly patch systems when critical vulnerabilities are discovered through scanning and through public disclosure. |
| REC-VHPM-BP-2 | Regularly scan the internal network for available services to identify new and potentially vulnerable services. |
| REC-VHPM-BP-3 | Continuous monitoring of vulnerability sources and the use of automatic and manual code review tools should also be implemented as well. |
| REC-VHPM-BP-3 | Patch strategy shall be applied for each component to maintain the O-Cloud software.  The patch strategy should handle regular and emergency patches. It is also needed to be prepared for testing patches and rollback procedures. The patch strategy should be automated. It's also recommended to make use of plugins for monitoring software and notifying of pending patches.  Keep in mind that some patches require a restart of their service, a new deployment (VM/Container image) or even a reboot (host) to become effective. If this won't be done, patching otherwise could be as effective as just not to patch. Technical details when to restart a service, a host or initiate a new deployment need to be defined in the patch strategy too.  There are four different patch domains:   * Images: the VM/Container distribution * Hypervisor/Container Engine software: e.g., VMware vSphere, Docker * Orchestration software (Kubernetes, Mesos, OpenShift, ...) * Host operating system |
| REC-VHPM-BP-4 | Maintain SBOMs for O-RAN software components in a centralized repository. This will make possible to quickly scan and search the SBOM for any Zero-Day vulnerability when get disclosed, which allows O-Cloud operator to respond quickly to the Zero-day vulnerability to mitigate potential attacks. For more details about SBOM see Clause 4 ‘SBOM Guidelines for O-RAN’ in [i.42]. |

**This recommendation can help to mitigate**: T-GEN-01, T-IMG-01, T-VL-01, T-ADMIN-02, T-O2-01, T-OCAPI-01

## REC-SCONF Security Configuration

**Recommendation**: The O-Cloud virtualization layer shall

* Be configured to ensure conformance to industry standard benchmarks and requirements with respect to security configurations.
* Use automated or manual mechanisms to detect configuration drifts/misconfigurations from industry standard benchmarks and requirements.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-SCONF-BP-1 | O-Cloud virtualization layer should be configured to adhere/conform to industry standard benchmarks and guidelines for securing the virtualization layer. For example, the CIS benchmark guidelines enforce many security configurations related to the container deployment like:   * Avoid running containers as privileged, apply the principle of least privilege access controls for privileged containers, and restrict functionality of privileged containers. * Restrict containers from using hostPath volume mounts write permissions from within container. * Enforce non-root default user in containers (using Kubernetes pod spec parameters or USER directive in Dockerfile). |
| REC-SCONF-BP-2 | O-Cloud operator & vendor should perform automated or manual scans to check security posture adherence to industry standard benchmarks. For example, Cloud Security Posture Management tools can be used to check conformance to industry standard benchmarks like CIS, NIST, etc. |
| REC-SCONF-BP-3 | O-Cloud CNF vendors should define the CPU and Memory requirements of their CNF applications, ie, the CPU and Memory requirements for the CNF applications to perform its functions under normal operating scenarios and the threshold limit value of CPU & Memory requirements beyond which the CNF should not be allowed to use. O-Cloud virtualization platform should consider the CPU & Memory resource requirements & limits associated to each CNF provided by O-Cloud CNF vendors during onboarding of the CNFs. |
| REC-SCONF-BP-4 | O-Cloud virtualization platform should consider enforcing the O-Cloud CNF vendors to specify the CPU & Memory resource requirements & limits using suitable admission controller solutions. Setting CPU & memory limits will help OCloud virtualization platform to avoid a rogue CNF container using up all resources thereby starving off other co-located CNF containers. |

**This recommendation can help to mitigate**: T-VM-C-01, T-VM-C-02, T-VM-C-06, T-VL-01, T-ADMIN-01, T-ADMIN-02, T-TS-01

## REC-SDLC Secure Development Lifecycle

**Recommendation**: Software providers should assume that their VNFs/CNFs and VL software contain flaws and have appropriate processes in place to mitigate these cases. To this end, they should adopt a secure software development life-cycle (S-SDLC). A S-SDLC integrates security considerations into the normal software development life-cycle. This ensures that risks, threats, and security mechanisms are formalized alongside the development of VNFs/CNFs and VL.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-SDLC-BP-1 | Define and implement good coding practices. This includes writing documentation, adopting secure programming guidelines, enforcing code quality metrics and performing software tests. In particular, code quality and adherence to coding standards can be checked automatically with tools, such as SonarQube[[13]](#footnote-14). With this approach, the chance of introducing easily exploitable vulnerabilities into the code is reduced. |
| REC-SDLC-BP-2 | Modern continuous integration (CI) and continuous delivery (CD) systems can be used to build new container images from code repositories, such as Git[[14]](#footnote-15) automatically. Thus, every commit can be used to generate a new container image that is immediately deployed for testing in a development environment. Once a new software version passes all tests, it can be deployed automatically to the O-Cloud platform. This step should be safeguarded by an internal approval process. For example, operators can configure their CI/CD pipeline to only start deployment in production after quality assurance, security testing and operations have approved the new version. |
| REC-SDLC-BP-3 | During development, developers may include private keys and configuration files with passwords into their VM/Container images for convenience and testing reasons. However, this information should never be published in a production VM/Container image. Thus, one of important step before actually deploying any VM/Container image to the O-Cloud platform is to verify that no sensitive information is included. In case of Docker, recent versions allow to specify these image differences between development and operations by using multiple FROM commands, i.e., one for development and one production in the respective Dockerfile. |
| REC-SDLC-BP-4 | Define mechanisms and processes that ensure that VM/Container image metadata is signed and verified. Thus, if operators download an image from a party they trust, they can verify the signature and the cryptographic hashes of the image files.  Example: use of Docker Notary and Content Trust[[15]](#footnote-16) |

**This recommendation can help to mitigate**: T-GEN-01, T-IMG-01, T-IMG-03

## REC-SNFLC Security App/VNF/CNF lifecycle

**Recommendation**: Security in the life cycle management of Apps/VNFs/CNFs should be integrated during development, onboarding, instantiation, scaling, migration, and termination. Security lifecycle management must be adapted to work in a more dynamic environment with fast-changing network topology, data flow paths, and network addresses.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-SNFLC-BP-1 | Before an App/VNF/CNF is deployed, the NFO/FOCOM system should verify compliance with its build-configuration standards. A check can include but is not limited to:   * Security of the configuration * Whether the package contains only trusted and expected components * Whether trusted components have been altered (integrity) |
| REC-SNFLC-BP-2 | Apps/VNFs/CNFs can easily be cloned and instantiated. As a result, a security framework must address the following elements:   * Depending on the migration technique, certificates, accounts, media access control addresses, or hardware IDs will look identical after cloning. * Trusted platform modules (e.g., TPM, HSM, Intel’s Trusted Execution Technology, or TXT) may be required to ensure that virtual components can provide attestation or a true status of the security of the underlying hardware and physical location. |

**This recommendation can help to mitigate**: T-GEN-03, T-VM-C-01, T-VM-C-02, T-VM-C-04, T-VM-C-06

## REC-IMGP Image Protection

**Recommendation**: Only images from secure and trusted sources should be used. These images should be scanned to ensure that images do not contain vulnerable software.

The App/VNF/CNF images shall not be packaged with embedded secrets such as passwords or credentials, or any other critical configuration data.

During the onboarding, the authenticity and integrity of the VNF Package should be verified against the signature provided by the VNF provider. Furthermore, Operators should undertake additional security validation of the VNF Package during the onboarding process and operator's signing should be used to certify the VNF as authorized to onboard into the operator's network [i.46], [i.47].

**Best practices to fulfill this recommendation**:

***Image Scanning***

Scanning process needs to identify when Apps/VNFs/CNFs are running with out-of-date packages that need to be updated for security patches. Further, it should identify malware that has been built into an image. The O-Cloud VL should enforce image scans to check and flag the presence of secrets in images.

1. VM/Container images need to be frequently scanned throughout the lifecycle of the App/VNF/CNF

|  |  |  |  |
| --- | --- | --- | --- |
| **During Development** | **Within the registry** | **Before Deployment** | **During Runtime** |
| Perform frequent scanning for known vulnerability or misconfiguration on App/VNF/CNF image during build phase. E.g., Check for malware, secrets stored in image.  Ideal to automate with CI pipeline.  **Responsible**: O-RAN Solution Provider/Vendor | Regular scan for App/VNF/CNF image known vulnerability or misconfiguration stored in the registry. E.g., scan for new vulnerabilities after built.  **Responsible**: O-RAN Solution Provider/Vendor/Operator | Perform frequent scanning for known vulnerability or misconfiguration on App/VNF/CNF image before deployment using admission control techniques. Admission controller could block deployments if the image doesn’t comply with the operator security policies e.g., To check the images used to create CNF pods are secure by checking for malware.  Ideal to automate with CD pipeline.  **Responsible**: O-RAN Solution Provider/Vendor/Operator | Perform continuous scanning/monitoring for known vulnerability or misconfiguration on runtime workloads, check for any open ports, VM/Container escape.  **Responsible**: Operator/O-Cloud Operator |

1. As part of O-RAN App/VNF/CNF development, O-RAN Solution Provider/Vendor is recommended to choose base images and packages used in the O-RAN App/VNF/CNF from a trusted publisher or create in-house base images. Choosing trusted signed images helps to mitigate MITM attacks that can introduce vulnerabilities into the base images available in public registries.
2. As part of O-RAN App/VNF/CNF development, O-RAN Solution Provider/Vendor is recommended to choose minimal base images that bundle only the necessary system tools and libraries required by O-RAN App/VNF/CNF, to limit the attack surface of O-RAN App/VNF/CNF.

***App/VNF/CNF Image Signing***

**Responsible:** App/VNF/CNF Solution Provider/Vendor and/or Operator

1. App/VNF/CNF Solution Provider creates public/private key pair for code signing.
2. CA verifies public key belong to the owner and issue CA certificate with attached public key.
3. Hash function on App/VNF/CNF Image returns image Digest.
4. Image Digest is encrypted with private key.
5. Signed App/VNF/CNF Image contains CA Certificate, Digest, Hash function.

***App/VNF/CNF Image Verification***

**Responsible**: Operator or O-Cloud Operator

1. Verify CA certificate for authenticity.
2. Decrypt digest with public key
3. Compare Decrypted digest with Digest computed as result of Hash function on App/VNF/CNF Image.
4. Verification is successful if both digest matches.

***Proposed Signing/Verification Models***

1. Software signing by Solution Provider/Vendor, Software verification at runtime by O-Cloud Operator
2. Software signing by Solution Provider/Vendor, Software verification by operator before onboarding vendor images to the operator registry
3. Software signing by Operator, Software verification at runtime by O-Cloud Operator
4. Software signing by Solution Provider/Vendor, Software verification at runtime by O-Cloud Operator and/or Operator.
5. Software signing by solution provider/vendor, Software verification by Operator and software re-signing by Operator before onboarding vendor images to the operator registry, software verification at instantiation.

**Other best practices:**

* Least privilege access that limits and controls access to the images’ repository and the repository of source code, secure storage, and automation for verifying the configuration of images before loading.
* Utilize a trust model such as Docker Content Trust with digital signatures to ensure runtime verification of the integrity and publisher of specific image tags.
* Implement only digitally signed host images to validate the integrity of the software used on the O-Cloud platform. Make use of this feature where possible in order to prevent and/or detect attempts by adversaries to compromise the host image.
* The registry is most sensitive part of the system and should be run in an HSM.
* The App/VNF/CNF images shall not be packaged with embedded secrets such as passwords or credentials, or any other critical configuration data. Secrets have to be stored outside the App/VNF/CNF images. For example, K8s secrets or external secrets managers (like Vault) can be used to store secret and sensitive information hence preventing the threat of packaging secrets in CNF container images. Secrets required by containerized applications can be injected into the CNF container from the External Secrets Manager as required.
* Encrypting VNF volume / swap areas: Virtual volume disks associated with VNFs may contain sensitive data. Therefore, they need to be protected. The best practice to secure the VNF volume is by encrypting them and storing the cryptographic keys at safe locations. The TPM module can also be used to securely store these keys. In addition, the hypervisor should be configured to securely wipe out the virtual volume disks in the event a VNF is crashed or intentionally destroyed to prevent it from unauthorized access. VM swapping is a memory management technique used to move memory segments from the main memory to disk, which is used as a secondary memory in order to increase system performance in case the system runs out of memory. These transferred memory segments can contain sensitive information such as passwords and certificates. They can be stored on the disk and remain persistent even after system reboot. This enables an attack scenario whereby a VM swap is copied and investigated to retrieve any useful information. One way to avoid this kind of attack is to encrypt VM swap areas. Linux based tools such as dm-crypt can be used for this purpose.

**This recommendation can help to mitigate**: T-GEN-01, T-GEN-03, T-IMG-01, T-IMG-02, T-IMG-03, T-IMG-04

## REC-LOG Logging, Monitoring and Alerting

**Recommendation**: The continuous security of O-Cloud software depends on the ability to identify attacks and reconstruct them. The earlier operators identify suspicious behavior and possible attacks, the faster they can initiate countermeasures. In the best case, they are able to stop attacks before severe harm is done. In the worst case, they at least are able to reduce the attacks’ severity and collect important details on the attack itself. This knowledge is important to find and fix flaws and vulnerabilities to prevent similar attacks in the future. Therefore, a key activity is to perform logging, monitoring and alerting.

The importance increases because of the dynamic VM/Container deployment. VMs/Containers may be started when demand arises and then quickly be terminated once they are no longer required. Furthermore, the focused nature of VMs/Containers and the application of a microservice architecture means that there are more systems to monitor. As a result, monitoring and logging must handle this dynamic landscape and be able to collect data from many systems at once.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-LOG-BP-1 | Any change to the O-Cloud components 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. |
| REC-LOG-BP-2 | Security logs should be time synchronized and regularly scanned for events of interest. |
| REC-LOG-BP-3 | Consider monitoring for the presence or loading of known vulnerable software that adversaries may drop and exploit to execute code in kernel mode. |
| REC-LOG-BP-4 | Monitor for the deployment of suspicious or unknown container images and pods in your environment, particularly containers running as root. Additionally, monitor for unexpected usage of syscalls such as mount (as well as resulting process activity) that may indicate an attempt to escape from a privileged container to host. In Kubernetes, monitor for cluster-level events associated with changing containers' volume configurations. |
| REC-LOG-BP-5 | Establish centralized logging for the activity of container and Kubernetes cluster components. This can be done by deploying logging agents on Kubernetes nodes and retrieving logs from sidecar proxies for application pods to detect malicious activity at the cluster level. |
| REC-LOG-BP-6 | Monitor logs for actions that could be taken to gather information about container infrastructure, including the use of discovery API calls by new or unexpected users. Monitor account activity logs to see actions performed and activity associated with the Kubernetes dashboard and other web applications. |
| REC-LOG-BP-7 | Consider monitoring process resource usage to determine anomalous activity associated with malicious hijacking of resources such as CPU, memory, and graphics processing resources. Monitor for suspicious use of network resources associated with cryptocurrency mining software. Monitor for common cryptomining software process names and files on local systems that may indicate compromise and resource usage. |
| REC-LOG-BP-8 | Container administration service activities and executed commands can be captured through logging of process execution with command-line arguments on the container and the underlying host. In Docker, the daemon log provides insight into events at the daemon and container service level. Kubernetes system component logs may also detect activities running in and out of containers in the cluster. |
| REC-LOG-BP-9 | Monitor interactions with images and containers by users to identify ones that are added or modified anomalously. In containerized environments, changes may be detectable by monitoring the Docker daemon logs or setting up and monitoring Kubernetes audit logs depending on registry configuration. |
| REC-LOG-BP-10 | Monitor for suspicious or unknown container images and pods in your environment. Deploy logging agents on Kubernetes nodes and retrieve logs from sidecar proxies for application pods to detect malicious activity at the cluster level. In Docker, the daemon log provides insight into remote API calls, including those that deploy containers. Logs for management services or applications used to deploy containers other than the native technologies themselves should also be monitored. |
| REC-LOG-BP-11 | Monitor for unexpected Docker image build requests to the Docker daemon on hosts in the environment. Additionally monitor for subsequent network communication with anomalous IPs that have never been seen before in the environment that indicate the download of malicious code. |
| REC-LOG-BP-12 | Alert the incident responders about a breach / attack on the O-Cloud software via email, SMS or other notification mechanisms. |
| REC-LOG-BP-13 | Alert using a notification to RASP (Runtime application self-protection) modules, or a SIEM about a breach / attack on the O-Cloud software. |

**This recommendation can help to mitigate**: T-GEN-01, T-GEN-02, T-GEN-03, T-GEN-06, T-VM-C-02, T-VM-C-03, T-VM-C-04, T-VM-C-05, T-VM-C-06, T-IMG-01, T-IMG-02, T-IMG-04, T-VL-02, T-TS-01

## REC-SB Secure Boot

**Recommendation**: All servers that are part of O-Cloud Infrastructure should support a root of trust and secure boot to trust that the running host OS, Hypervisor/Container Engine and VNFs/CNFs code were loaded.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-SB-BP-1 | Using trusted platform module (TPM) as a hardware root of trust, the measurement of system sensitive components such as platform firmware, BIOS, bootloader, OS kernel, and other system components can be securely stored and verified. The platform measurement can only be taken when the system is reset or rebooted; there is no way to write the new platform measurement in TPM during the system run-time. The validation of the platform measurements can be performed by TPM’s launch control policy (LCP) or through the remote attestation server. |

For more details see ETSI GR NFV-SEC 007 [i.15].

**This recommendation can help to mitigate**: T-GEN-03, T-IMG-04, T-VL-01, T-VL-02

## REC-ISO Strong Isolation

**Recommendation**: Strong VNFs/CNFs sandboxing, isolation and segmentation should be enforced to make difficult for adversaries to advance their operation through exploitation of undiscovered or unpatched vulnerabilities. O-Cloud virtualization layer shall adopt measures to ensure strong VM/Container isolation among VM/Container workloads to limit the impact of rogue/misbehaving VM/Container on other co-hosted VMs/Containers.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-ISO-BP-1 | In order to prevent privilege escalations, Operators shall ensure that CNFs shall:   * Avoid using binaries with SUID or SGID bit. Instead, specific capabilities required by the files/binaries should be provided using Linux Capabilities set * Avoid root user and sudo privileges within container |
| REC-ISO-BP-2 | Restrict the damage attackers can do once they successfully escaped a container. Again, the Linux kernel provides a couple of security features that can be used to restrict user actions. Among them are Linux security modules providing mandatory access control and user namespaces themselves. |
| REC-ISO-BP-3 | Restrict the large system call interface using seccomp. The seccomp (’Secure Computing’) integrates a BPF-like filter mechanism for system calls into the Linux kernel. This can be used to reduce the number of system calls that a container can effectively use. Moreover, the parameters to system calls can be sanitized. Thus, seccomp reduces one attack surface by reducing the number of exploitable system calls. |
| REC-ISO-BP-4 | Linux kernel allows to restrict users’ capabilities to perform specific operations. In particular, root users have a vast set of permissions. Linux capabilities permit a fine-grained partition of the capabilities bundled into the root user. Thus, instead of being root, different capabilities can be restricted. For example, a user can be root without having the permissions to change network settings. Currently, the Linux kernel has 37 capabilities. Unfortunately, not all capabilities have a specific, well-defined scope. For example, CAP\_SYS\_PTRACE permits the use of ptrace(), while CAP\_SYS\_ADMIN comprises many permissions such as changing the host and domain name, using mount() and umount(), or performing various configurations of storage and memory devices. Restricting capabilities assigned to container processes limits the operations an adversary can use on the host system. |
| REC-ISO-BP-5 | Namespaces and cgroups are the two features that enable container-based virtualization in the Linux kernel. They provide isolation and separation of processes. However, their default configuration is often not as restrictive as possible to support many use cases. Thus, it is important to employ many namespaces and cgroups and tighten their configuration.  For example, the user namespace separates the user IDs on the host system from the user IDs of a container. A user within the container can be root, while on the host system the same user is unprivileged. As another example, the devices control group can restrict what device nodes can be created and be accessed. Other control groups limit the maximum resources a container can use. Thus, they prevent degradation or denial of services in a shared environment. |
| REC-ISO-BP-6 | Linux security modules (LSM) are software components that use a well-defined kernel interface to control and decline certain operations. In particular, LSMs providing mandatory access control (MAC) define rules that further restrict what files a process can access. Two well-known LSM MACs are AppArmor and SELinux. These MACs define policy rules about what a process can and cannot access. The resulting policies are then enforced by the kernel. Thus, adversaries are on the one side further limited on the resources that are exploitable. On the other hand, they have to invest additional time and resources to circumvent and disable the LSMs.  So far, these security mechanisms mostly protect the host from containers. If a container is compromised, an adversary still has to put in the additional effort to compromise the host. However, these protections may not circumvent exploits across containers on the same host. For example, containers on the Docker engine share a common user namespace and storage space for the container images. Thus, an adversary escaping one container may have access to the container images of another container. This can be used to manipulate images and introduce persistent threats or steal information. One safeguard against these cross-container exploitations are extended features of LSM MACs known as multi category security. Using MCS, every container gets a unique identity when it is initialized. Based on this unique identity, the kernel can perform more fine-grained access decisions because two process executing the same binary that would usually have the same policy are now distinguishable.  As a result, access to container specific resources can be restrict for each container. |
| REC-ISO-BP-7 | It is important to secure remote access to the host and containers themselves. While not going into the details of full network security in a cloud environment, each container and host must be protected as part of the security-in-depth approach. On Linux systems, netfilter/iptables and ebtables can be used to create packet firewalls. They limit network access to exposed ports on the host and in the containers. Similar to virtual switches in hypervisors, containers can be connected by a Linux bridge device. Platforms such as Docker use a default bridge device for all containers if no separate network is specified during container initiation. As a result, containers may share a common network segment, which can be exploited. Therefore, ebtables is recommended as an additional firewall configured for the bridge device. |
| REC-ISO-BP-8 | Stronger isolation mechanisms to keep application code constrained within a VM/Container. Such solutions include the private cloud, physical separation.  For containers, many security solutions[[16]](#footnote-17) are used for ensuring strong isolation of resources, such as LXC, LXD, Singularity, Docker (runc), Kata-containers (kata-runtime), and gVisor (runsc). |
| REC-ISO-BP-9 | Shared cache memory partitioning or isolation and assigning of a separate portion of cache memory to each VM can reduce or eliminate successful side channel attacks in a virtualized environment[[17]](#footnote-18). |

**This recommendation can help to mitigate**: T-GEN-01, T-VM-C-01, T-VM-C-02, T-VM-C-05, T-VL-01, T-HW-01

## REC-AUD Security Audit

**Recommendation**: Periodic audit and scan should be conducted of:

* The integrity of images and VM/Containers used in O-Cloud deployments to ensure they have not been modified to include malicious software.
* Accounts and privileges for images repositories.
* All accounts, access lists and the privileges they have been granted to access O-Cloud components and images repositories.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-AUD-BP-1 | Scan code repositories for exposed credentials or other sensitive information. Preemptively search for files containing passwords or other credentials and take actions to reduce the exposure risk when found. |
| REC-AUD-BP-2 | Suspicious accounts/credentials should be investigated and removed. |
| REC-AUD-BP-3 | Routinely check account role permissions to ensure only expected users and roles have permission to modify cloud firewalls, to modify cloud compute infrastructure components, create snapshots and backups, and to create/delete new instances. |

**This recommendation can help to mitigate**: T-GEN-01, T-GEN-02, T-VM-C-05, T-IMG-01, T-IMG-03, T-IMG-04, T-ADMIN-01, T-ADMIN-02

## REC-SS Secure Storage

**Recommendation**: The O-Cloud infrastructure should support encrypted storage, for example, block, object, file storage, credentials, and secrets with access to encryption keys restricted based on a need to know.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-SS-BP-1 | Controlled Access Based on the Need to Know should be implemented[[18]](#footnote-19). |
| REC-SS-BP-2 | When possible, store keys on separate cryptographic hardware (e.g. HSM, TPM) instead of on the local system. |
| REC-SS-BP-3 | Use best practices for authentication protocols, such as OAuth 2.0, and ensure traffic that may contain credentials is protected by TLS. |
| REC-SS-BP-4 | Set up monitoring to track access to cache data and identify any unauthorized access attempts. |
| REC-SS-BP-5 | Establish policies for cache data lifecycle, including how long data should be retained in cache and when it should be purged. |

**This recommendation can help to mitigate**: T-GEN-06, T-VM-C-01, T-VM-C-02, T-VM-C-03, T-VM-C-06, T-IMG-03, T-IMG-03, T-VL-01, T-OCAPI-01, T-HW-01

## REC-PHY Physical Security Protection

**Recommendation**: Physical security protection measures should be considered to deny unauthorized access to O-Cloud facilities, equipment and resources.

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-PHY-BP-1 | Reduce the hardware platform (computing platform based for example on the x86 hardware architecture (32bit or 64bit), similar servers) attack surface by removing unneeded interfaces.  Example: Wireless interfaces (e.g., WiFi, 4G), if present, should remain optional for the platform REC-PHY-BP-1 [i.41]. |
| REC-PHY-BP-2 | Enforce protection of the main memory against untrusted peripherals.  Example: The hardware platform exposes an input/output memory management unit, such as VT-x (Intel) or AMD-V (AMD). This feature needs to be enabled by default and can only be disabled by the owner of the hardware platform (not the user) [i.41]. |
| REC-PHY-BP-3 | Ensure the security hardware component quality.  Example: In case the platform embeds a TPM, the latter is certified (e.g. from a Common Criteria perspective according to the Protection Profile TCG Protection Profile PC Client Specific TPM Family 2.0) [i.41].  Only the owner of the hardware platform (not the user) can enable and disable the TPM. |
| REC-PHY-BP-4 | Ensure the hardening of the platform against persistent compromises.  Example: hardware platform firmwares are expected to ensure their integrity by means of state-of-the-art security mechanisms. This notably includes protecting integrity of the code and data stored on the Flash Memory. The modification of firmware code should only be the result of a legitimate, signed update using robust cryptographic protocols. The installation of a signed update older than the version currently installed on the hardware platform shall be considered illegitimate by default. However, for the latter case, the firmware configuration interface shall allow the platform owner to disable this protection. |
| REC-PHY-BP-5 | Ensure the presence of minimal security features within the firmware (Low-level software components shipped with the hardware platform, including UEFI and legacy BIOSes or hardware components embedded software).  Example: The firmware configuration interface provides the following features [i.41]:   * Protecting the access to the firmware configuration interface thanks to a dedicated password * Locking the boot sequence of the platform thanks to a dedicated password * Enabling and disabling input/output interfaces (e.g. USB) * Configuring the boot order (the ordering of the list of devices checked by the BIOS to boot an operating system) * Replacing default Secure Boot keys with custom keys generated by the owner |
| REC-PHY-BP-6 | Allow for inspecting and auditing firmware code.  Example: Mechanisms which prevent firmware code inspection shall be absent or disabled by default. |
| REC-PHY-BP-7 | Data centre physical security protections based on its location and such risk assessments, to determinate the structures, and data centre support systems (sensors, video camera, alarms to a Security Operation Centre…) |
| REC-PHY-BP-8 | Backup system in other area if needed. |
| REC-PHY-BP-9 | Enforcing Multi-Factor Authentication for privileged users could be a mitigation. |
| REC-PHY-BP-10 | Server cabinets fitted with an electronic lock. |
| REC-PHY-BP-11 | Physical Access Control based system with video and entry logs system. |
| REC-PHY-BP-12 | Sensors on the room or the cabinet to generate alarms to the Network / Security Operation Centre. |
| REC-PHY-BP-13 | Limited number of physical interfaces on the on-board NIC of the server and remotely control NIC. |
| REC-PHY-BP-14 | Secure All Endpoints (terminal, physical connectivity…). |
| REC-PHY-BP-15 | Zero trust architecture with security layers including physical access and redundancies, with regular security audit if needed. |

**This recommendation can help to mitigate**: T-VM-C-03, T-VM-C-05, T-VL-02, T-OCAPI-01, T-HW-01, T-HW-02

## REC-RA Remote Attestation

**Recommendation**: A remote attestation (RA) approach should be used to determine the trustworthiness of O-Cloud components. It is a defensive measure that addresses the malicious software execution. In O-Cloud it is obvious the need for applying procedures to verify the integrity of the whole O-Cloud service deployment by the appropriate attestation of the O-Cloud architectural elements, including software and firmware images and associated supporting security sub-systems that will run to instantiate individual VNFs/CNFs and their composition into a O-Cloud service.

The remote attestation requires identifying the root(s) of trust, establishing a chain of trust for the O-Cloud infrastructure, the VL, VNFs/CNFs, and the NFO/FOCOM sub-systems, and verification of the trust chain, so the NFO/FOCOM components can verifiably establish a sufficient level of assurance in the different software elements constituting the VNFs/CNFs and the service(s) that use them.

The remote attestation (RA) capabilities should be used to realize trust establishment in the following use case scenarios:

1. Measurement of VM/Container during launch.
2. Protected VM/Container launch on a trusted O-Cloud platform
3. Measurement of VM/Container during launch and while in use
4. Remote attestation of secret storage
5. Secure VM/Container migration between two trusted O-Cloud platform

**Best practices to fulfill this recommendation**:

|  |  |
| --- | --- |
| **Req ID** | **Description** |
| REC-RA-BP-1 | ETSI GR NFV-SEC 018 [i.13], [i.14], ETSI GR NFV-SEC 007 [i.15] could be used as a security guidance of the remote attestation (RA) approach. |
| REC-RA-BP-2 | All servers part of O-Cloud infrastructure should support measured boot and an attestation server that monitors the measurements of the servers. |

**Example**: Operators might use RA to assess if the overall O-Cloud infrastructure is trustworthy, datacenters might use RA to assess trustworthiness of subsystems they use, and management entities might use RA to assess the trustworthiness of individual infrastructural components. Hence, there are numerous use-cases and scenarios that might be considered where attestation is a fundamental step of creating an overall trustworthy system.

A trustworthy element is the entity which has a component that provides a unique identifier, certification (e.g. through cryptographic signing) and which is able to store measurements and data about the state of that element (including related sub-elements or dependent elements if necessary) in a tamperproof and verifiable form. For example, the TPM2.0 quoting mechanism using the TPMS\_ATTEST data structure is an example of this.

**This recommendation can help to mitigate**: T-GEN-03, T-VM-C-04, T-IMG-04, T-VL-02

## REC-SDD Secure data deletion

**Recommendation**: The O-Cloud platform should delete cryptographic keys and sensitive data using a secure deletion method from both active and backup storage systems.

**Best practices to fulfill this recommendation:**

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-SDD-BP-1 | Review the CSP’s policies and SLAs on data deletion to ensure it has a procedure that effectively deletes data. |
| REC-SDD-BP-2 | Secure overwriting is way to overwrite on existing data to be deleted by new data, replaced with zero or random. |
| REC-SDD-BP-3 | Cryptography method to encrypt the data before storing to the cloud storage. In order to delete the data, the encryption key is being destroyed. This makes the data unrecoverable. |
| REC-SDD-BP-4 | Each data center adheres to a strict disposal policy and uses the techniques described to achieve compliance with NIST SP 800-88 Revision 1 “Guidelines for Media Sanitization” [i.48] and DoD 5220.22-M “National Industrial Security Program Operating Manual.” [i.49]. |
| REC-SDD-BP-5 | Physically destroy media that cannot be sanitized by either crush and deform the drive or shred the drive into small pieces. |
| REC-SDD-BP-6 | Limit access to data backups (through use of roles) to know who has access to the data. |
| REC-SDD-BP-7 | Enable application deployment using a security policy where volumes containing sensitive data mounted by the O-Cloud infrastructure to the application, are unmounted after application initialization guard timer, or based on application state monitoring. |
| REC-SDD-BP-8 | Enable applications to delete application sensitive data from their memory / filesystems based on a trigger from O-Cloud infrastructure. |
| REC-SDD-BP-9 | Enable RASP modules to notify applications to delete application sensitive data from their memory / filesystems based on a trigger from O-Cloud infrastructure. |

**This recommendation can help to mitigate**: T-GEN-05, T-GEN-06, T-VM-C-03

## REC-IDM O-Cloud ID secure management

**Recommendation:** To counter threats associated with ID reuse, mismanagement, and redundancy, the O-Cloud should employ robust strategies for ID generation, validation, and lifecycle management. Properly managed IDs reduce risks of data inconsistencies, unauthorized access, and operational inefficiencies.

**Best practices to fulfil this recommendation:**

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-IDM-BP-01 | Adopt the UUID generation method in line with RFC 4122 standards within the O-Cloud system. Ensure each newly created object, whether Containers, Pods, Nodes, or Services, is assigned a unique UUID upon its instantiation. Utilize reputable UUID generation libraries or modules to ensure adherence to the standard. |
| REC-IDM-BP-02 | Implement validation mechanisms ensuring each ID's uniqueness during assignment. |
| REC-IDM-BP-03 | Use database-level constraints to ensure ID uniqueness across the O-Cloud. |
| REC-IDM-BP-04 | Monitor the lifecycle of each ID, providing a clear history of its assignment and usage. |
| REC-IDM-BP-05 | Implement automated processes to detect and remove defunct node objects. |
| REC-IDM-BP-06 | Audit node resources before ID assignment and raise alerts on mismatches. |
| REC-IDM-BP-07 | Use configuration management tools to maintain node consistency. |
| REC-IDM-BP-08 | Employ dynamic IP management systems to ensure consistent IP assignments. |
| REC-IDM-BP-09 | Ensure monitoring tools can differentiate between data from old and new IDs. |
| REC-IDM-BP-10 | Notify users/systems when an ID is reused, alerting them of potential ambiguities. |
| REC-IDM-BP-11 | Centralize logging systems with distinct identifiers for each ID. |
| REC-IDM-BP-12 | Regularly update and audit RBAC policies to ensure they reflect current ID assignments. |
| REC-IDM-BP-13 | Before reusing any ID, ensure that the lifecycle of the previous object associated with that ID has been completely terminated. |
| REC-IDM-BP-14 | For any ID that's about to be reused, conduct thorough data cleaning processes to remove all residual data and configurations associated with its predecessor. |

**This recommendation can help to mitigate**: T-O-CLOUD-ID-01, T-O-CLOUD-ID-02, T-O-CLOUD-ID-03

## REC-IDM Identity management for O-Cloud elements

O-RAN Applications hosted on O-Cloud, Services provided by O-Cloud and O-Cloud Nodes use an identity, such as an identifier in an X.509 certificate, for authentication and authorization. To establish consistent and standardized identities throughout O-Cloud, it is essential to implement a uniform identity provider.

The incorporation of an Identity Provider framework ensures uniform identity management that provides strong attested identities, through a combination of diverse attestation mechanisms, for the various elements in O-cloud environment and addresses the threats of Identity spoofing, unauthorized access, inadequate verification of the trustworthiness of entities.

**Recommendation:** SPIRE and SPIFFE are CNCF graduated projects that effectively act as an identity provider for workloads and has been specified in **NIST SP 800-207A** [i.58] which mentions that “service identity should not be subjected to spoofing and should be continuously verifiable. An example of workload identity is a SPIFFE ID which is encoded as a Uniform Resource Identifier (URI) and carried in a cryptographically verifiable document called a SPIFFE Verifiable Identity Document (SVID)”.

SPIFFE provides a structured ID format for applications in [RFC 3986](https://datatracker.ietf.org/doc/html/rfc3986) URI format. The SPIFFE ID adheres to **NIST SP 800-207A** (Zero Trust Architecture Model for Access Control in Cloud-Native Applications) as mentioned above.

**SPIFFE Runtime Environment** (SPIRE), a production-ready implementation of SPIFFE APIs use **Node** and **workload** **attestation** to securely issue **SVID** [X.509 certificate with SPIFFE ID][i.55] to Applications which can be used for service-service communication.

In O-Cloud, services and workloads can be dynamically scaled up, down and scheduled, SPIRE provides a **secure way to verify the identity** of these dynamically changing entities, ensuring that services can **trust** one another in such environments.

**SPIFFE and SPIRE**, as a part of O-Cloud Infrastructure enable services and applications to obtain identity (SPIFFE ID [i.57]) and SVID [X.509 certificate with SPIFFE ID] using attestation policies.

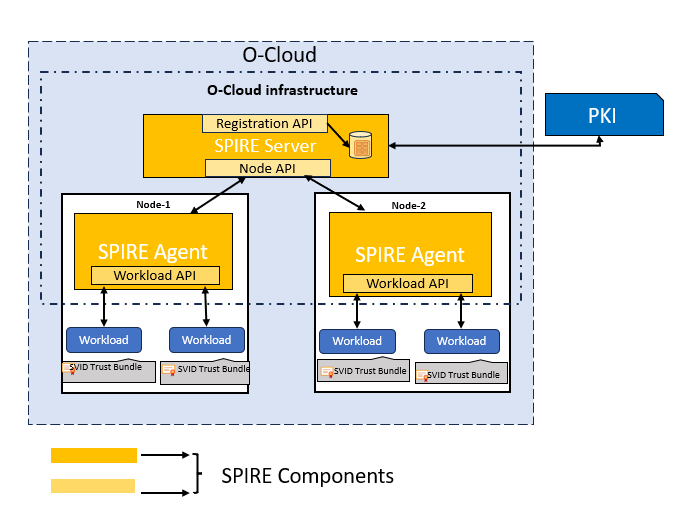


Fig 8.18-1: SPIRE Architecture in O-Cloud

**SPIRE Architecture**: A SPIRE deployment is composed of one or more **SPIRE Servers** and one or more **SPIRE Agents**.

**SPIRE Server**: A SPIRE Server is responsible for **managing and issuing** all **identities** in its configured SPIFFE trust domain. It stores registration entries [i.54] (which specify the selectors that determine the conditions under which a particular SPIFFE ID should be issued) and signing keys, uses node attestation [i.54] to authenticate agents' identities automatically, and creates SVIDs for workloads when requested by an authenticated agent.

**SPIRE Agents**: A SPIRE Agent runs on every O-Cloud node on which an identified workload runs. The agent performs the following tasks:

* Requests SVIDs from SPIRE server and caches them until a workload requests its SVID.
* Exposes the SPIFFE Workload API to workloads on node and attests the identity of workloads that call it.
* Provides SVIDs to the identified workloads.

**Best practices to fulfil this recommendation:**

|  |  |
| --- | --- |
| **BP ID** | **Description** |
| REC-IDM-SPR-01 | Adopt a uniform standardized identity that follow the URI specification as defined by RFC 3986 for all elements within the O-Cloud.  **spiffe://domain/clusterName/namespace/WL-identity** |
| REC-IDM-SPR-02 | The SPIFFE Verifiable Identity Document (SVID) is cryptographically verifiable document can be issued either by SPIRE server locally or through an external Certificate Authority. |
| REC-IDM-SPR-03 | Agent and server architecture model as defined in SPIFFE and SPIRE Concepts [i.54] to be implemented in O-Cloud platform. |
| REC-IDM-SPR-04 | O-Cloud exposes the SPIRE APIs through which identity may be retrieved and/or issued to Applications or services. |
| REC-IDM-SPR-05 | Registration entries maps an identity -- in the form of a SPIFFE ID -- to a set of properties known as selectors that the workload must possess to be issued a particular identity.  For SPIRE to identify an O-Cloud element, it is essential to register the O-Cloud elements with the SPIRE Server, via registration entries.  SMO can use the O2ims interface to push the registration entries for O-Cloud Nodes, Network Functions to the SPIRE server present as part of the O-Cloud platform. |
| REC-IDM-SPR-06 | Attestation Policies are enforced for asserting the identity of O-Cloud elements by gathering attributes and comparing them to a set of selectors defined in the Registration entries.  Node Attestation [i.56] ensures agent authenticate and verify itself when it first connects to a server.  Workload Attestation verifies the identity of the elements by querying information about the process from the node and comparing it with the selectors provided in the Registration entries. |
| REC-IDM-SPR-07 | SPIRE server as part of O-Cloud platform does issuance of SVID and required trust bundle to O-Cloud elements after successful assertion. |

**This recommendation can help to mitigate**: T-O-CLOUD-ID-01, T-O-CLOUD-ID-03

## REC-ADMC Admission Control

Admission Control, within the framework of O-Cloud, operates as a customizable and adaptable security layer. It intercepts API requests after they are authenticated and authorized but before they are executed. It evaluates and decides on the admission of requests to perform specific actions, such as deploying a software image, creating or updating Pods, managing resources, and setting up network policies.

Some of the configurable security policies include:

* Vulnerability scanning: The Admission Control incorporates a robust vulnerability scanning process. This process is integral to identifying and assessing any known security vulnerabilities within a software image. It ensures that only those images that meet the set vulnerability thresholds, indicating a minimized risk of security threats, are approved for deployment. The activation of this scanning feature can be adjusted according to the specific needs and security requirements of different scenarios.
* Image signature verification: In addition to vulnerability scanning, the Admission Control also includes the critical step of verifying the digital signature of software images. This verification process is essential for confirming the authenticity and integrity of the image. It guarantees that the image originates from a trusted source and remains unaltered. The enforcement of this verification can be tailored based on the operational context and the level of security needed.
* Resource quotas and limits: The Admission Control enforces policies on the amount of resources (CPU, memory, storage) that can be consumed by pods or namespaces. This prevents any single application or user from monopolising cluster resources, which can be a security concern in multi-tenant environments.
* Pod security policies: The Admission Control defines a set of conditions that a pod must meet before it is accepted into the system. These policies can enforce the use of non-privileged containers, restrict volume types, and require the use of read-only root file systems, all of which enhance security.
* Immutable containers: The Admission Control enforces policies that prevent changes to running containers, ensuring that the containers remain in their original, secure state as defined at deployment.
* API rate limiting: To prevent abuse and potential denial-of-service attacks, The Admission Control can enforce rate limiting on API requests. This ensures that the API server remains stable and responsive, even under high load or during an attack.

In Kubernetes, the Admission Control is an integral component that significantly enhance the cluster's security and operational integrity. It can be implemented as a plugin that can either modify or reject requests to the Kubernetes API, ensuring compliance with defined security and operational policies. Examples of a such Admission Control include:

* Kubernetes Pod Security Admission: Admission Controller provided by Kubernetes itself. It focuses on enforcing security-related best practices for Pods.
* 3rd party admission plugins: such as OPA. Open Policy Agent (OPA) is a popular third-party admission controller. It extends the functionality of Kubernetes admission control by allowing more complex and customized policy enforcement. OPA uses a declarative language called Rego to define policies that specify the desired state and behaviour of the cluster resources. For example, OPA can be used to require that all container images come from a trusted registry.

In the O-Cloud context, the decision hinges on achieving a delicate balance: on one hand, there's the need for simplicity and seamless integration, which is a strong point of Kubernetes native controllers. On the other hand, there's a demand for advanced, customizable policy management, where OPA excels. Throughout this balancing act, it is crucial to maintain high standards of security and optimal performance, ensuring these implementations are effectively aligned with the broader O-RAN system. Opting for a combination approach might provide the most flexibility, catering to these diverse requirements. This strategy enables the adaptation to varying operational scenarios within O-RAN, each with its own unique set of needs, whether they pertain to security, efficiency, scalability, or compliance.

**This recommendation can help to mitigate**: T-VM-C-01, T-IMG-01, T-VL-03

# Risk Assessment

The risk assessment of the identified threats against O-Cloud is provided in the O-RAN Security Threat Modeling and Risk Assessment [i.50].

Annex A (informative): Best practices from some of existing main security guidance

Operators, O-Cloud vendors and Cloud providers should ensure that O-Cloud components and interfaces are secure under very strict set of security best practices. Such security guidance include:

* A.1 CISA/NSA Kubernetes security hardening best practices [i.3]
* A.2 CIS Docker security best practices [i.18]
* A.3 ONAP VNFs security best practices [i.24]

1. CISA/NSA Kubernetes security hardening best practices

This section summarizes recommendations and best practices from the CISA/NSA Kubernetes Hardening Guidance. The report details recommendations to harden Kubernetes systems. Primary actions include the scanning of containers and Pods for vulnerabilities or misconfigurations, running containers and Pods with the least privileges possible, and using network separation, firewalls, strong authentication, and log auditing. To ensure the security of applications, system administrators should keep up to date with patches, updates, and upgrades to minimize risk. NSA and CISA also recommend periodic reviews of Kubernetes settings and vulnerability scans to ensure appropriate risks are accounted for and security patches are applied.

Download the full Kubernetes Hardening Guidance: <https://media.defense.gov/2021/Aug/03/2002820425/-1/-1/1/CTR_KUBERNETES%20HARDENING%20GUIDANCE.PDF>

**Kubernetes Pod security**

Pods are the smallest deployable Kubernetes unit and consist of one or more containers. Pods are often a cyber actor’s initial execution environment upon exploiting a container. For this reason, Pods should be hardened to make exploitation more difficult and to limit the impact of a successful compromise.

***“Non-root” containers***: Preventing root execution by using non-root containers limits the impact of a container compromise. container engines allow containers to run applications as a non-root user with non-root group membership. Typically, this non-default setting is configured when the container image is built. Having non-root execution integrated at build time provides better assurance that applications will function correctly without root privileges. (See Appendix A in [i.3]).

***Immutable container file systems***: By default, containers are permitted mostly unrestricted execution within their own context. A cyber actor who has gained execution in a container can create files, download scripts, and modify the application within the container. Kubernetes can lock down a container’s file system, thereby preventing many post-exploitation activities. However, these limitations also affect legitimate container applications and can potentially result in crashes or anomalous behavior. To prevent damaging legitimate applications, Kubernetes administrators can mount secondary read/write file systems for specific directories where applications require write access. (See Appendix B in [i.3]).

***Building secure container images***: Container images are usually created by either building a container from scratch or by building on top of an existing image pulled from a repository. In addition to using trusted repositories to build containers, image scanning is key to ensuring deployed containers are secure. Throughout the container build workflow, images should be scanned to identify outdated libraries, known vulnerabilities, or misconfigurations, such as insecure ports or permissions. One approach to implementing image scanning is by using an admission controller [i.3]. This admission controller could block deployments if the image doesn’t comply with the organization’s security policies defined in the webhook configuration[[19]](#footnote-20).

***Hardening container engines***: Some platforms and container engines provide additional options to harden the containerized environments. A powerful example is the use of hypervisors to provide container isolation. Hypervisors rely on hardware to enforce the virtualization boundary rather than the operating system. Hypervisor isolation is more secure than traditional container isolation. Container engines running on the Windows operating system can be configured to use the built-in Windows hypervisor, Hyper-V, to enhance security. Additionally, some security focused container engines natively deploy each container within a lightweight hypervisor for defense-in-depth. Hypervisor-backed containers mitigate container breakouts.

**Network separation and hardening**

Cluster networking is a central concept of Kubernetes. Communication between containers, Pods, services, and external services must be taken into consideration. Resource separation and encryption can be an effective way to limit a cyber actor’s movement and escalation within a cluster.

***Namespaces***: Kubernetes namespaces are one way to partition cluster resources among multiple individuals, teams, or applications within the same cluster (See [i.3] for more details).

***Network policies*:** Network policies control traffic flow between Pods, namespaces, and external IP addresses. By default, no network policies are applied to Pods or namespaces, resulting in unrestricted ingress and egress trafficwithin the Pod network. Pods become isolatedthrough a network policy that applies to the Pod or the Pod’s namespace. Once a Pod is selected in a network policy, it rejects any connections that are not specifically allowed by any applicable policy object. (See Appendix E in [i.3] for more details).

***Resource policies*:** In addition to network policies, LimitRange and ResourceQuota are two policies that can limit resource usage for namespaces or nodes. A LimitRange policy constrains individual resources per Pod or container within a particular namespace, e.g., by enforcing maximum compute and storage resources. ResourceQuotas are restrictions placed on the aggregate resource usage for an entire namespace, such as limits placed on total CPU and memory usage. If a user tries to create a Pod that violates a LimitRange or ResourceQuota policy, the Pod creation fails. (See Appendixes F and G in [i.3] for more details).

***Control plane hardening***: The control plane is the core of Kubernetes and gives users the ability to view containers, schedule new Pods, read Secrets, and execute commands in the cluster. Because of these sensitive capabilities, the control plane should be highly protected. In addition to secure configurations such as TLS encryption, RBAC, and a strong authentication method, network separation can help prevent unauthorized users from accessing the control plane. The Kubernetes API server should be protected (e.g. by firewall, TLS encryption) and not be exposed to the Internet or an untrusted network.

***Worker node segmentation****:* A worker node can be a virtual or physical machine, depending on the cluster’s implementation. Because nodes run the microservices and host the web applications for the cluster, they are often the target of exploits. If a node becomes compromised, an administrator should proactively limit the attack surface by separating the worker nodes from other network segments that do not need to communicate with the worker nodes or Kubernetes services. A firewall can be used to separate internal network segments from the external facing worker nodes or the entire Kubernetes service depending on the network. Examples of services that may need to be separated from the possible attack surface of the worker nodes are confidential databases or internal services that would not need to be internet accessible.

***Encryption***: Administrators should configure all traffic in the Kubernetes cluster—including between components, nodes, and the control plane—to use TLS 1.2 or 1.3 encryption. Encryption can be set up during installation or afterward using TLS bootstrapping, detailed in the Kubernetes documentation[[20]](#footnote-21), to create and distribute certificates to nodes. For all methods, certificates must be distributed amongst nodes to communicate securely.

***Secrets***: Kubernetes Secrets maintain sensitive information, such as passwords, OAuth tokens, SSH and TLS keys. Secrets can be encrypted by configuring data-at-rest encryption on the API server or by using an external Key Management Service (KMS), which may be available through a cloud provider. (See Appendixes H and I in [i.3] for more details).

***Protecting sensitive cloud infrastructure***: Kubernetes is often deployed on virtual machines in a cloud environment. As such, administrators should carefully consider the attack surface of the virtual machines on which the Kubernetes worker nodes are running. In many cases, Pods running on these virtual machines have access to sensitive cloud metadata services on a non-routable address. These metadata services provide cyber actors with information about the cloud infrastructure and possibly even short-lived credentials for cloud resources. Cyber actors abuse these metadata services for privilege escalation. Kubernetes administrators should prevent Pods from accessing cloud metadata services by using network policies or through the cloud configuration policy. Because these services vary based on the cloud provider, administrators should follow vendor guidance to harden these access vectors.

**Authentication and authorization**

Authentication and authorization are the primary mechanisms to restrict access to cluster resources. Cyber actors can scan for well-known Kubernetes ports and access the cluster’s database or make API calls without being authenticated if the cluster is misconfigured. User authentication is not a built-in feature of Kubernetes. However, several methods exist for administrators to add authentication to a cluster.

***Authentication***: Kubernetes clusters have two types of users: service accounts and normal user accounts. Service accounts handle API requests on behalf of Pods. Authentication is typically managed automatically by Kubernetes through the ServiceAccount Admission Controller using bearer tokens. The bearer tokens are mounted into Pods at well-known locations and can be used from outside the cluster if the tokens are left unsecured. Because of this, access to Pod Secrets should be restricted to those with a need to view them using Kubernetes RBAC. For normal users and admin accounts, there is no automatic authentication method for users. Administrators must add an authentication method to the cluster to implement authentication and authorization mechanisms. The Kubernetes documentation[[21]](#footnote-22) lists several ways to implement user authentication including client certificates, bearer tokens, authentication plugins, and other authentication protocols. At least one user authentication method should be implemented. Administrators should not use weak methods such as static password files. Weak authentication methods could allow cyber actors to authenticate as legitimate users.

***Role-based access control***: RBAC is one method to control access to cluster resources based on the roles of individuals within an organization. RBAC is enabled by default in Kubernetes version 1.6 and newer. Privileges assigned to users, groups, and service accounts should follow the principle of least privilege, giving only required permissions to resources. Users or user groups can be limited to particular namespaces where required resources reside. By default, a service account is created for each namespace for Pods to access the Kubernetes API. RBAC policies can be used to specify allowed actions from the service accounts in each namespace. Access to the Kubernetes API is limited by creating an RBAC Role or ClusterRole with the appropriate API request verb and desired resource on which the action can be applied. Tools exist that can help audit RBAC policies by printing users, groups, and service accounts with their associated assigned Roles and ClusterRoles. (See Appendixes J and K in [i.3] for more details).

**Log auditing**

Logs capture activity in the cluster. Auditing logs is necessary, not only for ensuring that services are operating and configured as intended, but also for ensuring the security of the system. Systematic audit requirements mandate consistent and thorough checks of security settings to help identify compromises. Kubernetes is capable of capturing audit logs for cluster actions and monitoring basic CPU and memory usage information; however, it does not natively provide in-depth monitoring or alerting services.

***Logging****:* System administrators running applications within Kubernetes should establish an effective logging, monitoring, and alerting system for their environment. Logging Kubernetes events alone is not enough to provide a full picture of the actions occurring on the system. Logging should also be performed at the host level, application level, and on the cloud if applicable. These logs can then be correlated with any external authentication and system logs as applicable to provide a full view of the actions taken throughout the environment for use by security auditors and incident responders. Within the Kubernetes environment, administrators should monitor/log the following:

* API request history
* Performance metrics
* Deployments
* Resource consumption
* Operating system calls
* Protocols, permission changes
* Network traffic
* Pod scaling

When a Pod is created or updated, administrators should capture detailed logs of the network communications, response times, requests, resource consumption, and any other relevant metrics to establish a baseline.

RBAC policy configurations should be audited periodically and whenever changes occur to the organization’s system administrators.

Audits of internal and external traffic logs should be conducted to ensure all intended security constraints on connections have been configured properly and are working as intended.

Logs can be streamed to an external logging service to ensure availability to security professionals outside of the cluster, identify abnormalities as close to real time as possible, and protect logs from being deleted if a compromise occurs. If using this method, logs should be encrypted during transit with TLS 1.2 or 1.3 to ensure cyber actors cannot access the logs in transit and gain valuable information about the environment. (See Appendixes L and M in [i.3] for more details).

***SIEM platforms***: Security Information and Event Management (SIEM) software collects logs from across an organization’s network. SIEM software brings together firewall logs, application logs, and more; parsing them out to provide a centralized platform from which analysts can monitor system security. SIEM tools have variations in their capabilities. Generally, these platforms provide log collection, threat detection, and alerting capabilities. Some include machine learning capabilities, which can better predict system behavior and help to reduce false alerts. Organizations using these platforms in their environment can integrate them with Kubernetes to better monitor and secure clusters. Open-source platforms for managing logs from a Kubernetes environment exist as an alternative to SIEM platforms.

***Alerting***: Kubernetes does not natively support alerting; however, several monitoring tools with alerting capabilities are compatible with Kubernetes. If Kubernetes administrators choose to configure an alerting tool to work within a Kubernetes environment, there are several metrics for which administrators should monitor and configure alerts. Examples of cases that could trigger alerts include but are not limited to:

* Low disk space on any of the machines in the environment,
* Available storage space on a logging volume running low,
* External logging service going offline,
* A Pod or application running with root permissions,
* An anonymous account being used or gaining privileges,
* Unusual system calls or failed API calls,
* user/admin behavior that is abnormal (i.e. at unusual times or from an unusual location), and
* Significant deviations from the standard operation metrics baseline.

***Service meshes*:** Service meshes are platforms that streamline microservice communications within an application by allowing for the logic of these communications to be coded into the service mesh rather than within each microservice. Coding this communication logic into individual microservices is difficult to scale, difficult to debug as failures occur, and difficult to secure. Using a service mesh can simplify this for developers. The mesh can:

* Redirect traffic when a service is down,
* Gather performance metrics for optimizing communications,
* Allow management of service-to-service communication encryption,
* Collect logs for service-to-service communication,
* Collect logs from each service, and
* Help developers diagnose problems and failures of microservices or communication mechanisms.

***Fault tolerance***: Fault tolerance policies should be put in place to ensure logging service availability. One policy that can be put in place is to allow new logs to overwrite the oldest log files if absolutely necessary in the event of storage capacity being exceeded. If logs are being sent to an external service, a mechanism should be in place for logs to be stored locally if a communication loss or external service failure occurs. Once communication to the external service is restored, a policy should be in place for the locally stored logs to be pushed up to the external server.

***Tools***: Kubernetes does not include extensive auditing capabilities. However, the system is built to be extensible, allowing users the freedom to develop their own custom solution or to choose an existing add-on that suits their needs. One of the most common solutions is to add additional audit backend services, which can use the information logged by Kubernetes and perform additional functions for users, such as extended search parameters, data mapping features, and alerting functionality. Organizations that already use SIEM platforms can integrate Kubernetes with these existing capabilities. Open-source monitoring tools—such as the Cloud Native Computing Foundation’s Prometheus, Grafana Labs’ Grafana, and Elasticsearch’s Elastic Stack (ELK) —are available to conduct event monitoring, run threat analytics, manage alerting, and collect resource isolation parameters, historical usage, and network statistics on running containers. Scanning tools can be useful when auditing the access control and permission configurations by assisting in identifying risky permission configurations in RBAC. NSA and CISA encourage organizations utilizing Intrusion Detection Systems (IDSs) on their existing environment to consider integrating that service into their Kubernetes environment as well. This integration would allow an organization to monitor for—and potentially kill containers showing signs of—unusual behavior so the containers can be restarted from the initial clean image. Many cloud service providers also provide container monitoring services for those wanting more managed and scalable solutions.

**Upgrading and application security practices**

Security of applications running on Kubernetes orchestrated containers is an ongoing process, and it is vital to keep up with patches, updates, and upgrades. The specific software components vary depending on the individual configuration, but each piece of the overall system should be kept as secure as possible. This includes updating: Kubernetes, hypervisors, virtualization software, plugins, operating systems on which the environment is running, applications running on the servers, and any other software hosted in the Kubernetes environment.

The Center for Internet Security (CIS) publishes benchmarks for securing software. Administrators should adhere to the CIS benchmarks for Kubernetes and any other relevant system components. Administrators should check periodically to ensure their system's security is compliant with the current security experts’ consensus on best practices. Periodic vulnerability scans and penetration tests should be performed on the various system components to proactively look for insecure configurations and zero-day vulnerabilities. Any discoveries should be promptly remediated before potential cyber actors can discover and exploit them.

As updates are deployed, administrators should also keep up with removing any old components that are no longer needed from the environment. Using a managed Kubernetes service can help to automate upgrades and patches for Kubernetes, operating systems, and networking protocols. However, administrators must still patch and upgrade their containerized applications.

1. CIS Docker security best practices

CIS Benchmarks are universal security best practices developed by cybersecurity professionals and experts. Each CIS Benchmark provides guidelines for creating a secure system configuration. The following table summarizes recommendations from the CIS Docker Benchmark, specifying how to set up a safe docker configuration.

Download the full CIS Docker Benchmark: <https://www.cisecurity.org/benchmark/docker/>

**Host Configuration**

Create a separate partition for containers

* Harden the container host
* Update your Docker software on a regular basis
* Manage Docker daemon access authorization wisely
* Configure your Docker files directories, and
* Audit all Docker daemon activity.

**Docker Daemon Configuration**

* Restrict network traffic between default bridge containers and access to new privileges from containers.
* Enable user namespace support to provide additional, Docker client commands authorization, live restore, and default cgroup usage
* Disable legacy registry operations and Userland Proxy
* Avoid networking misconfiguration by allowing Docker to make changes to iptables, and avoid experimental features during production.
* Configure TLS authentication for Docker daemon and centralized and remote logging.
* Set the logging level to 'info', and set an appropriate default ulimit
* Don’t use insecure registries and aufs storage drivers
* Apply base device size for containers and a daemon-wide custom SECCOMP profile to limit calls.

**Container Images and Build File**

* Create a user for the container
* Ensure containers use only trusted images
* Ensure unnecessary packages are not installed in the container
* Include security patches during scans and rebuilding processes
* Enable content trust for Docker
* Add HEALTHCHECK instructions to the container image
* Remove setuid and setgid permissions from the images
* Use COPY instead of ADD in Dockerfile
* Install only verified packages
* Don’t use update instructions in a single line or alone in the Dockerfile
* Don’t store secrets in Dockerfiles

**Container Runtime**

* Restrict containers from acquiring additional privileges and restrict Linux Kernel Capabilities.
* Enable AppArmor Profile.
* Avoid use of privileged containers during runtime, running ssh within containers, mapping privileged ports within containers.
* Ensure sensitive host system directories aren’t mounted on containers, the container's root filesystem is mounted as read-only, the Docker socket is not mounted inside any containers.
* Set appropriate CPU priority for the container, set 'on-failure' container restart policy to '5', and open only necessary ports on the container.
* Apply per need SELinux security options, and overwrite the default ulimit at runtime.
* Don’t share the host's network namespace and the host's process namespace, the host's IPC namespace, mount propagation mode, the host's UTS namespace, the host's user namespaces.
* Limit memory usage for container and bind incoming container traffic to a specific host interface.
* Don’t expose host devices directly to containers, don’t disable the default SECCOMP profile, don’t use docker exec commands with privileged and user option, and don’t use Docker's default bridge docker0.
* Confirm cgroup usage and use PIDs cgroup limit, check container health at runtime, and always update docker commands with the latest version of the image.

**Docker Security Operations**

* Avoid image sprawl and container sprawl.

**Docker Swarm Configuration**

* Enable swarm mode only if needed
* Create a minimum number of manager nodes in a swarm
* Bind swarm services are bound to a specific host interface
* Encrypt containers data exchange on different overlay network nodes
* Manage secrets in a Swarm cluster with Docker's secret management commands
* Run swarm manager in auto-lock mode
* Rotate swarm manager auto-lock key periodically
* Rotate node and CA certificates as needed
* Separate management plane traffic from data plane traffic

1. ONAP VNFs security best practices

ONAP provides details on the VNF general security requirements on various security areas such as user access control, network security, ACLs, infrastructure security, and vulnerability management. These requirements cover topics associated with compliance, security patching, logging/accounting, authentication, encryption, role-based access control, least privilege access/authorization. This section summarizes requirements from ONAP, specifying how to integrate and operate VNFs within a robust security environment.

For more details see the full documentation: <https://docs.onap.org/projects/onap-vnfrqts-requirements/en/latest/index.html>

**VNF General Security Requirements**

This section provides details on the VNF general security requirements on various security areas such as user access control, network security, ACLs, infrastructure security, and vulnerability management. These requirements cover topics associated with compliance, security patching, logging/accounting, authentication, encryption, role-based access control, least privilege access/authorization. The following security requirements need to be met by the O-RAN NFs in a virtual environment:

1. The VNF must implement and enforce the principle of least privilege on all protected interfaces.
2. The VNF must provide a mechanism (e.g., access control list) to permit and/or restrict access to services on the VNF by source, destination, protocol, and/or port.
3. The VNF should provide a mechanism that enables the operators to perform automated system configuration auditing at configurable time intervals.
4. The VNF provider must follow GSMA vendor practices and SEI CERT Coding Standards when developing the VNF in order to minimize the risk of vulnerabilities. See GSMA NESAS Network Equipment Security Assurance Scheme – Development and Lifecycle Security Requirements Version 1.0 (https://www.gsma.com/ security/wp-content/uploads/2019/11/FS.16-NESAS-Development-and-Lifecycle-Security- Requirements-v1.0.pdf) and SEI CERT Coding Standards (https://wiki.sei.cmu.edu/ confluence/display/seccode/SEI+CERT+Coding+Standards).
5. The VNF must have all code (e.g., QCOW2) and configuration files (e.g., HEAT template, Ansible playbook, script) hardened, or with documented recommended configurations for hardening and interfaces that allow the Operator to harden the VNF. Actions taken to harden a system include disabling all unnecessary services, and changing default values such as default credentials and community strings.
6. The VNF should support the separation of (1) signaling and payload traffic (i.e., customer facing traffic), (2) operations, administration and management traffic, and (3) internal VNF traffic (i.e., east-west traffic such as storage access) using technologies such as VPN and VLAN.
7. The VNF Provider must have patches available for vulnerabilities in the VNF as soon as possible. Patching shall be controlled via change control process with vulnerabilities disclosed along with mitigation recommendations.
8. The VNF must support only encrypted access protocols, e.g., TLS 1.2/1.3.
9. The VNF must store authentication credentials used to authenticate to other systems encrypted except where there is a technical need to store the password unencrypted in which case it must be protected using other security techniques that include the use of file and directory permissions. Ideally, credentials should rely on a HW Root of Trust, such as a TPM or HSM.
10. VNFs that are subject to regulatory requirements must provide functionality that enables the Operator to comply with ETSI TC LI requirements, and, optionally, other relevant national equivalents.
11. The VNF must be able to authenticate and authorize all remote access.
12. The VNF must log any security event required by the VNF Requirements to Syslog using LOG\_AUTHPRIV for any event that would contain sensitive information and LOG\_AUTH for all other relevant events.
13. If SNMP is utilized, the VNF must support at least SNMPv3 with message authentication.
14. The VNF application processes should not run as root. If a VNF application process must run as root, the technical reason must be documented.
15. Login access (e.g., shell access) to the virtualization layer, whether interactive or as part of an automated process, must be through an encrypted protocol such as TLS 1.2/1.3.
16. The VNF must include a configuration (e.g. a template) that specifies the targeted parameters (e.g. a limited set of ports) over which the VNF will communicate; including internal, external and management communication.
17. Containerized components of VNFs should follow the recommendations for Container Base Images and Build File Configuration in the latest available version of the CIS Docker Community Edition Benchmarks to ensure that containerized VNFs are secure. All non-compliances with the benchmarks must be documented.
18. Containerized components of VNFs should execute in a Docker run-time environment that follows the Container Runtime Configuration in the latest available version of the CIS Docker Community Edition Benchmarks to ensure that containerized VNFs are secure. All non-compliances with the benchmarks must be documented.

**VNF Identity and Access Management Requirements**

The following security requirements for logging, identity, and access management need to be met by the O-RAN NFs in a virtual environment:

1. The VNF must, if not integrated with the Operator’s Identity and Access Management system, support the creation of multiple IDs so that individual accountability can be supported.
2. The VNF must, if not integrated with the operator’s IAM system, provide a mechanism for assigning roles and/or permissions to an identity.
3. The VNF must, if not integrated with the Operator’s Identity and Access Management system, support multifactor authentication on all protected interfaces exposed by the VNF for use by human users.
4. The VNF must support account names that contain at least A-Z, a-z, and 0–9-character sets and be at least 6 characters in length.
5. The VNF must, if not integrated with the Operator’s Identity and Access Management system, comply with “password complexity” policy and support configurable password expiration. When passwords are used, they shall be complex and shall at least meet the following password construction requirements: (1) be a minimum configurable number of characters in length, (2) include 3 of the 4 following types of characters: upper-case alphabetic, lower-case alphabetic, numeric, and special, (3) not be the same as the UserID with which they are associated or other common strings as specified by the environment, (4) not contain repeating or sequential characters or numbers, (5) not to use special characters that may have command functions, and (6) new passwords must not contain sequences of three or more characters from the previous password.
6. The VNF must allow the Operator to restrict access to protected resources based on the assigned permissions associated with an ID in order to support Least Privilege (no more privilege than required to perform job functions).
7. The VNF must set the default settings for user access to deny authorization, except for a super user type of account.
8. The VNF must not store authentication credentials to itself in clear text or any reversible form and must use salting.
9. The VNF must, if not integrated with the Operator’s Identity and Access Management system, support the ability to lock out the userID after a configurable number of consecutive unsuccessful authentication attempts using the same userID. The locking mechanism must be reversible by an administrator and should be reversible after a configurable time period.
10. The VNF must, if not integrated with the Operator’s identity and access management system, authenticate all access to protected resources.
11. The VNF must support LDAP in order to integrate with an external identity and access manage system. It MAY support other identity and access management protocols.
12. The VNF must not identify the reason for a failed authentication, only that the authentication failed.
13. The VNF must provide a means to explicitly logout, thus ending that session.
14. The VNF must provide explicit confirmation of a session termination such as a message, new page, or rerouting to a login page.
15. The VNF must, if not integrated with the Operator’s Identity and Access Management system, enforce a configurable “terminate idle sessions” policy by terminating the session after a configurable period of inactivity.

**VNF API Security Requirements**

This section covers API security requirements when these are used by the VNFs. Key security areas covered in API security are Access Control, Authentication, Passwords, PKI Authentication Alarming, Anomaly Detection, Lawful Intercept, Monitoring and Logging, Input Validation, Cryptography, Business continuity, Biometric Authentication, Identification, Confidentiality and Integrity, and Denial of Service.

The O-RAN NFs in a virtual environment needs to meet the following API security requirements:

1. The VNF should integrate with the Operator’s authentication and authorization services (e.g., IDAM).
2. The VNF must implement the following input validation control: Check the size (length) of all input. Do not permit an amount of input so great that it would cause the VNF to fail. Where the input may be a file, the VNF API must enforce a size limit.
3. The VNF must implement the following input validation controls: Do not permit input that contains content or characters inappropriate to the input expected by the design. Inappropriate input, such as SQL expressions, may cause the system to execute undesirable and unauthorized transactions against the database or allow other inappropriate access to the internal network (injection attacks).
4. The VNF must implement the following input validation control on APIs: Validate that any input file has a correct and valid Multipurpose Internet Mail Extensions (MIME) type. Input files should be tested for spoofed MIME types.

**VNF Security Analytics Requirements**

This section covers VNF security analytics requirements that are mostly applicable to security monitoring. The VNF Security Analytics cover the collection and analysis of data following key areas of security monitoring:

* Anti-virus software
* Logging
* Data capture
* Tasking
* DPI
* API based monitoring
* Detection and notification
* Resource exhaustion detection
* Proactive and scalable monitoring
* Mobility and guest VNF monitoring
* Closed loop monitoring
* Interfaces to management and orchestration
* Malformed packet detections
* Service chaining
* Dynamic security control
* Dynamic load balancing
* Connection attempts to inactive ports (malicious port scanning)

The following requirements of security monitoring need to be met by the O-RAN NFs in a virtual environment.

1. The VNF must support Real-time detection and notification of security events.
2. The VNF must support API-based monitoring to take care of the scenarios where the control interfaces are not exposed or are optimized and proprietary in nature.
3. The VNF must support detection of malformed packets due to software misconfiguration or software vulnerability and generate an error to the syslog console facility.
4. The VNF must support proactive monitoring to detect and report the attacks on resources so that the VNFs and associated VMs can be isolated, such as detection techniques for resource exhaustion, namely OS resource attacks, CPU attacks, consumption of kernel memory, local storage attacks.
5. The VNF should operate with anti-virus software which produces alarms every time a virus is detected.
6. The VNF must protect all security audit logs (including API, OS and application-generated logs), security audit software, data, and associated documentation from modification, or unauthorized viewing, by standard OS access control mechanisms, by sending to a remote system, or by encryption.
7. The VNF must log successful and unsuccessful authentication attempts, e.g., authentication associated with a transaction, authentication to create a session, authentication to assume elevated privilege.
8. The VNF must log logoffs.
9. The VNF must log starting and stopping of security logging.
10. The VNF must log success and unsuccessful creation, removal, or change to the inherent privilege level of users.
11. The VNF must log connections to the network listeners of the resource.
12. he VNF must log the field “event type” in the security audit logs.
13. The VNF must log the field “date/time” in the security audit logs.
14. The VNF must log the field “protocol” in the security audit logs.
15. The VNF must log the field “service or program used for access” in the security audit logs.
16. The VNF must log the field “success/failure” in the security audit logs.
17. The VNF must log the field “Login ID” in the security audit logs.
18. The VNF must not include an authentication credential, e.g., password, in the security audit logs, even if encrypted.
19. The VNF must detect when its security audit log storage medium is approaching capacity (configurable) and issue an alarm.
20. The VNF must support the capability of online storage of security audit logs.
21. The VNF must activate security alarms automatically when a configurable number of consecutive unsuccessful login attempts is reached.
22. The VNF must activate security alarms automatically when it detects the successful modification of a critical system or application file.
23. The VNF must activate security alarms automatically when it detects an unsuccessful attempt to gain permissions or assume the identity of another user.
24. The VNF must include the field “date” in the Security alarms (where applicable and technically feasible).
25. The VNF must include the field “time” in the Security alarms (where applicable and technically feasible).
26. The VNF must include the field “service or program used for access” in the Security alarms (where applicable and technically feasible).
27. The VNF must include the field “success/failure” in the Security alarms (where applicable and technically feasible).
28. The VNF must include the field “Login ID” in the Security alarms (where applicable and technically feasible).
29. The VNF must restrict changing the criticality level of a system security alarm to users with administrative privileges.
30. The VNF must monitor API invocation patterns to detect anomalous access patterns that may represent fraudulent access or other types of attacks or integrate with tools that implement anomaly and abuse detection.
31. The VNF must generate security audit logs that can be sent to Security Analytics Tools for analysis.
32. The VNF must log successful and unsuccessful access to VNF resources, including data.
33. The VNF must support the storage of security audit logs for a configurable period of time.
34. The VNF must have security logging for VNFs and their OSs be active from initialization. Audit logging includes automatic routines to maintain activity records and cleanup programs to ensure the integrity of the audit/logging systems.
35. The VNF must be implemented so that it is not vulnerable to OWASP Top 10 web application security risks.
36. The VNF must protect against all denial-of-service attacks, both volumetric and non-volumetric, or integrate with external denial of service protection tools.
37. The VNF must be capable of automatically synchronizing the system clock daily with the Operator’s trusted time source, to assure accurate time reporting in log files. It is recommended that Coordinated Universal Time (UTC) be used where possible, so as to eliminate ambiguity owing to daylight savings time.
38. The VNF must log the Source IP address in the security audit logs.
39. The VNF must have the capability to securely transmit the security logs and security events to a remote system before they are purged from the system.
40. The VNF should provide the capability of maintaining the integrity of its static files using a cryptographic method.
41. The VNF must log automated remote activities performed with elevated privileges.

**VNF Data Protection Requirements**

This section covers VNF data protection requirements that are mostly applicable to security monitoring.

1. The VNF MUST provide the capability to restrict read and write access to data handled by the VNF.
2. The VNF MUST Provide the capability to encrypt data in transit on a physical or virtual network.
3. The VNF MUST provide the capability to encrypt data on non-volatile memory. Non-volatile memory is storage that is capable of retaining data without electrical power, e.g. Complementary metal-oxide-semiconductor (CMOS) or hard drives.
4. The VNF SHOULD disable the paging of the data requiring encryption, if possible, where the encryption of non-transient data is required on a device for which the operating system performs paging to virtual memory. If not possible to disable the paging of the data requiring encryption, the virtual memory should be encrypted.
5. The VNF MUST use NIST and industry standard cryptographic algorithms and standard modes of operations when implementing cryptography.
6. The VNF MUST NOT use compromised encryption algorithms. For example, SHA, DSS, MD5, SHA-1 and Skipjack algorithms. Acceptable algorithms can be found in the NIST FIPS publications (https://csrc.nist.gov/publications/fips) and in the NIST Special Publications (<https://csrc.nist.gov/publications/sp>).
7. The VNF MUST use, whenever possible, standard implementations of security applications, protocols, and formats, e.g., S/MIME, TLS, SSH, IPSec, X.509 digital certificates for cryptographic implementations. These implementations must be purchased from reputable vendors or obtained from reputable open source communities and must not be developed in-house.
8. The VNF MUST provide the ability to migrate to newer versions of cryptographic algorithms and protocols with minimal impact.
9. The VNF MUST support digital certificates that comply with X.509 standards.
10. The VNF MUST NOT use keys generated or derived from predictable functions or values, e.g., values considered predictable include user identity information, time of day, stored/transmitted data.
11. The VNF MUST provide the capability of using X.509 certificates issued by an external Certificate Authority.
12. The VNF MUST be capable of protecting the confidentiality and integrity of data at rest and in transit from unauthorized access and modification.

**VNF Cryptography Requirements**

This section covers VNF cryptography requirements that are mostly applicable to encryption or protocol methods.

1. The VNF SHOULD support an automated certificate management protocol such as CMPv2, Simple Certificate Enrollment Protocol (SCEP) or Automated Certificate Management Environment (ACME).
2. The VNF SHOULD provide the capability to integrate with an external encryption service.
3. The VNF MUST use symmetric keys of at least 112 bits in length.
4. The VNF MUST use asymmetric keys of at least 2048 bits in length.
5. The VNF MUST provide the capability to configure encryption algorithms or devices so that they comply with the laws of the jurisdiction in which there are plans to use data encryption.
6. The VNF MUST provide the capability of allowing certificate renewal and revocation.
7. The VNF MUST provide the capability of testing the validity of a digital certificate by validating the CA signature on the certificate.
8. The VNF MUST provide the capability of testing the validity of a digital certificate by validating the date the certificate is being used is within the validity period for the certificate.
9. The VNF MUST provide the capability of testing the validity of a digital certificate by checking the Certificate Revocation List (CRL) for the certificates of that type to ensure that the certificate has not been revoked.
10. The VNF MUST provide the capability of testing the validity of a digital certificate by recognizing the identity represented by the certificate - the “distinguished name”.
11. The VNF MUST support HTTPS using TLS v1.2 or higher with strong cryptographic ciphers.
12. The VNF MUST support the use of X.509 certificates issued from any Certificate Authority (CA) that is compliant with RFC5280, e.g., a public CA such as DigiCert or Let’s Encrypt, or an RFC5280 compliant Operator CA.

NOTE: The VNF provider cannot require the use of self-signed certificates in an Operator’s run time environment.

# Annex: Change history/Change request (history)

|  |  |  |
| --- | --- | --- |
| Date | Revision | Description |
| 2022.01.11 | V01.00.01 | Initial draft version for review |
| 2022.03.03 | V01.00.02 | Update according to notes from the WI core teams. This version addresses ORA-CR009, ORA-CR010, MTR-CR001, ORA.AO-CR011 |
| 2022.03.10 | V01.00.03 | Update according to notes from the WI core teams. This version addresses ALT-CR001 |
| 2022.03.24 | V01.00.04 | Update after spelling check |
| 2022.10.21 | V02.00.00 | Addition of:   * Threats: T-GEN-05, T-VL-03, T-AAL-01, T-AAL-02 * Recommendation: REC-SDD |
| 2022.11.14 | V02.00.01 | Update according to the new O-RAN TR format |
| 2023.02.20 | V03.00.01 | Addition of:   * AAL interfaces * Assets: A-OC-13 to A-OC-20 * Threats: T-AAL-03 to T-AAL-06 |
| 2023.03.20 | V03.00.02 | Update after spelling check |
| 2023.07.05 | V04.00.01 | Integration of SYM CR 0003 |
| 2023.07.05 | V04.00.02 | References are updated to remove version numbers for O-RAN documents |
| 2023.10.16 | V05.00.01 | Integration of WG11 CR-0011 |
| 2023.10.19 | V05.00.02 | Editorial corrections in scope and references |
| 2023.10.26 | V05.00.03 | Editorial corrections in Clause 4.1 |
| 2024.02.19 | V06.00.01 | Additional of:   * Threat: T-GEN-06 * REC-IDM Identity management for O-Cloud elements * Critical service: O-Cloud Admission Control Services |
| 2024.03.05 | V06.00.02 | Addressed editorial comments |
| 2024.03.25 | V06.00.03 | Addressed review comments |
| 2024.06.11 | V06.00.04 | Changes:   * Definition of ‘Sensitive’ * O-Cloud Admission Control in ‘Recommendations and best practices’ * New definitions for certain terms * Alignment of O-Cloud assets with the Threat model |
| 2024.06.20 | V06.00.05 | Changes:   * Addition of references to definitions * Addition of O-Cloud inventory service details * Addition of time synchronization related threat |
| 2024.07.17 | V06.00.06 | Changes:   * Corrections to references * Correction to remove T-IMG-02 from Clause 8.19 to keep document consistency * Regeneration of table of contents |
| 2024.07.22 | V06.00.07 | Changes:   * Editorial (format) changes to the below   + Title of clause 4.3.9   + Renamed title of “revision history” (this table) as per template   + Corrected alignment of title on first page * Regeneration of table of contents |
| 2024.07.25 | V06.00.08 | Changes:   * Editorial (format/typo) corrections   Published as version 07.00 |

1. https://attack.mitre.org/techniques/T1552/003/ [↑](#footnote-ref-2)
2. https://attack.mitre.org/techniques/T1552/002/ [↑](#footnote-ref-3)
3. https://attack.mitre.org/techniques/T1552/004/ [↑](#footnote-ref-4)
4. Graz University of Technology (2018), Meltdown and Specter. [Online] Available at: https://meltdownattack.com [↑](#footnote-ref-5)
5. https://docs.openstack.org/keystone/latest/admin/tokens.html [↑](#footnote-ref-6)
6. https://openid.net/specs/openid-connect-core-1\_0.html [↑](#footnote-ref-7)
7. TSI TS 133 501: "5G; Security architecture and procedures for 5G System (3GPP TS 33.501)" [↑](#footnote-ref-8)
8. draft-ietf-oauth-token-binding-08: "OAuth 2.0 Token Binding", Work in progress [↑](#footnote-ref-9)
9. IETF RFC 8705: "OAuth 2.0 Mutual-TLS Client Authentication and Certificate-Bound Access Tokens" [↑](#footnote-ref-10)
10. https://www.cyberark.com/threat-research-blog/how-i-hacked-play-with-docker-and-remotely-ran-code-on-the-host/ [↑](#footnote-ref-11)
11. https://pages.nist.gov/800-63-3/sp800-63b.html [↑](#footnote-ref-12)
12. https://www.cisecurity.org/white-papers/cis-password-policy-guide [↑](#footnote-ref-13)
13. http://sonarqube.org [↑](#footnote-ref-14)
14. https://git-scm.com [↑](#footnote-ref-15)
15. https://docs.docker.com/engine/security/trust/content\_trust/ [↑](#footnote-ref-16)
16. Olivier Flauzac, Fabien Mauhourat, Florent Nolot, A review of native container security for running applications, Procedia Computer Science, Volume 175, 2020, Pages 157-164, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.07.025. “A review of native container security for running applications “ [↑](#footnote-ref-17)
17. https://arxiv.org/pdf/1606.01356.pdf [↑](#footnote-ref-18)
18. https://www.cisecurity.org/controls/cis-controls-list/ [↑](#footnote-ref-19)
19. The Linux Foundation, "11 Ways (Not) to Get Hacked," 18 07 2018 https://kubernetes.io/blog/2018/07/18/11-ways-not-to-get-hacked/#10- scan-images-and-run-ids. [↑](#footnote-ref-20)
20. https://kubernetes.io/docs/tasks/administer-cluster/securing-a-cluster/ [↑](#footnote-ref-21)
21. https://kubernetes.io/docs/reference/access-authn-authz/authentication [↑](#footnote-ref-22)