D5.3 FINAL REPORT ON ORCHESTRATION SOLUTIONS AND CYBERSECURITY FRAMEWORK FOR 6G-NTN

Revision: v.1.0

Work package	WP 5
Task	T5.2, T5.3
Due date	31/12/2025
Submission date	12/01/2026
Deliverable lead	TH-SIX
Version	1.0
Authors	Farid Benbadis (TH-SIX), Alice Piemonti (MAR), Vito Cianchini (MAR),
Reviewers	Tomaso De Cola (DLR), Alessandro Guidotti (UNIBO)
Abstract	*This deliverable describes the proposed orchestration and management architectures for 6G networks over ad hoc constellations. The main focus will be drawn on the split of different functions and VNFs across the proposed architecture in WP2 with a particular attention on open interfaces and generic frameworks for integrating AI and ML algorithms. The document will also assess the cybersecurity threats and vulnerabilities induced by the proposed virtualisation framework. A comprehensive threat analysis stemming from extending the 6G network to the satellite payloads will be conducted. Led by Task 5.2 with input from Task 5.3.*
Keywords	6G-NTN, Dynamic Orchestration, Autonomous Monitoring, Kubernetes, Security, Non-Terrestrial Networks, Machine Learning, API, Authentication, Scalability

Document Revision History

Grant Agreement No.: 101096479
Call: HORIZON-JU-SNS-2022
Topic: HORIZON-JU-SNS-2022-STREAM-B-01-03
Type of action: HORIZON-JU-RIA

Version	Date	Description of change	List of contributor(s)
V0.1	10/09/2025	Document creation	Farid Benbadis (TH-SIX)
V0.1	10/12/2025	Document submission for internal review	Farid Benbadis (TH-SIX)
V0.1 Rev	17/12/2025	Document reviewed with comment	Alessandro Guidotti (UNIBO)
V0.2	21/12/2025	Document updated	Vito Cianchini (MAR)
V0.9	25/12/2025	Final version for submission	Farid Benbadis (TH-SIX)
V1.0	12/01/2026	Approved for submission	Alessandro Vanelli-Coralli (UniBo)

DISCLAIMER

Co-funded by the European Union

6GSNS

Project funded by

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Economic Affairs, Education and Research EAER
State Secretariat for Education, Research and Innovation SERI

6G-NTN (*6G Non Terrestrial Network*) project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101096479. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them. This work has received funding from the Swiss State Secretariat for Education, Research and Innovation (SERI).

COPYRIGHT NOTICE

Project co-funded by the European Commission in the Horizon Europe Programme		
Nature of the deliverable:	**R**	
Dissemination Level		
PU	*Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page)*	✓
SEN	*Sensitive, limited under the conditions of the Grant Agreement*	
Classified R-UE/ EU-R	*EU RESTRICTED under the Commission Decision No2015/ 444*	
Classified C-UE/ EU-C	*EU CONFIDENTIAL under the Commission Decision No2015/ 444*	
Classified S-UE/ EU-S	*EU SECRET under the Commission Decision No2015/ 444*	

* R: Document, report (excluding the periodic and final reports)

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

DATA: Data sets, microdata, etc.

DMP: Data management plan

ETHICS: Deliverables related to ethics issues.

SECURITY: Deliverables related to security issues

OTHER: Software, technical diagram, algorithms, models, etc.

EXECUTIVE SUMMARY

This deliverable presents the research and development work conducted within WP5 of the 6G-NTN project, focusing on orchestration, monitoring solutions, and cybersecurity threats for 6G Non-Terrestrial Networks. It is an update of D5.2, entitled Initial report on orchestration and monitoring solutions and cybersecurity threats for 6G-NTN, submitted at M18, and it addresses the critical challenge of integrating satellite-based network infrastructure with terrestrial 6G systems while ensuring optimal performance, efficient resource management, and a robust security framework.

The work is structured around three main pillars. First, we present a Machine Learning (ML) based orchestration platform built on a Kubernetes cluster hosting an Open5GS 5G Core network. This platform enables dynamic Virtual Network Function (VNF) placement through continuous metrics collection via a monitoring system, ML-based resource usage prediction (CPU and memory), and automated orchestration decisions. The ML pipeline, orchestrated using Prefect workflows, generates forecasts that are directly transmitted to the network orchestrator, enabling proactive resource allocation before demand thresholds are reached.

Second, we conducted a comprehensive theoretical study on VNF placement strategies in hybrid terrestrial-LEO satellite architectures. Using discrete-event simulations across four NTN traffic ratios (30%, 50%, 70%, and 85%), we evaluated five placement policies: baseline (terrestrial-only), move-to-LEO, duplicate-on-peak, greedy heuristic, and flash-crowd. Our findings reveal that the flash-crowd policy demonstrates exceptional value for emergency scenarios, reducing packet loss by 81% with a minimal cost increase. The critical insight is that LEO placement should not be viewed as a performance optimization strategy but rather as a coverage extension technology for areas lacking terrestrial infrastructures.

Third, we address cybersecurity concerns through two approaches. We implemented a proactive security mechanism based on graph generation and attack path reconstruction from Kubernetes deployment descriptors, enabling risk identification before actual deployment. Additionally, we conducted a thorough security assessment of the distributed 6G-NTN architecture, analyzing the implications of disaggregating network functions (DU, CU, Core) across satellites connected by inter-satellite links (ISLs). Our analysis confirms that current 3GPP security standards (IPSec ESP, IKEv2, DTLS) provide adequate protection for F1 and Ng interfaces, though we identify potential vulnerabilities related to payload processor attacks and future quantum computing threats that warrant consideration for systems expected to operate until 2045.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

3GPP	3rd Generation Partnership Project
5GC	5G Core Network
6G	Sixth-Generation Wireless
AI	Artificial Intelligence
AMF	Access and Mobility Management Function
AUSF	Authentication Server Function
BS	Base Station
BVLoS	Beyond Visual Line-of-Sight
BWP	Bandwidth Part
C2	Command and Control
C2CSP	C2 Link Communication Service Provider
CoW	Cell on Wheels
CP	Cyclic Prefix
C-SWaP	Cost Size Weight and Power
CT	Core Network and Terminals
E2E	End-to-End
EAB	External Advisory Board
ECC	Electronic Communication Committee
EASA	European Union Aviation Safety Agency
ECC	Electronic Communication Committee
FR	First Responder
GEO	Geostationary Earth Orbit
gNB	Next-Generation Nobe-B
GNSS	Global Navigation Satellite Systems
GSO	Geostationary Orbit
HAP	High Altitude Platform
HD	High Definition
HV	Host Vehicle

IoT	Internet of Things
ISL	Inter-Satellite Link
INL	Inter-Node Link
LEO	Low Earth Orbit
LIDAR	Light Detection and Ranging
LoS	Line of Sight
M2M	Machine-to-Machine
MC	Multi-Connectivity
MC data	Mission Critical data
MCPTT	Mission Critical Push-to-Talk
MC video	Mission Critical Video
MFCN	Mobile/Fixed Communications Networks
MEO	Medium Earth Orbit
ML	Machine Learning
MNO	Mobile Network Operator
NASA	National Aeronautics and Space Administration
NGSO	Non-Geostationary Orbit
NLoS	Non Line-of-Sight
NR	New Radio
NRF	Network Repository Function
NTN	Non-Terrestrial Network
OOBE	Out-of-Band Emission
PAPR	Peak-to-Average Power Ratio
PCF	Policy Control Function
PHEM	Pre-Hospital Emergency Medicine
PPDR	Public Protection and Disaster Relief
PTT	Push-To-Talk
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
Rel	Release

RF	Radio Frequency
RIC	Radio Intelligent Controller
SAR	Search and Rescue
SD	Standard Definition
SI	Study Item
SMF	Session Management Function
SNO	Satellite Network Operator
SON	Self-Organizing Networks
T	Task
TN	Terrestrial Network
TR	Technical Report
TS	Technical Specification
UAM	Urban Air Mobility
UAV	Uncrewed Aerial Vehicle
UAV-C	UAV controller
UC	Use Case
UDM	Unified Data Management
UE	User Equipment
UPF	User Plan Function
USSP	U-Space Service Provider
vLEO	Very Low Earth Orbit
VLoS	Visual Line-of-Sight
VNF	Virtualised Network Function
VR	Virtual reality
VTOL	Vertical Take-Off and Landing
WI	Work Item
WPAN	Wireless Personal Area Network
WP	Work Package

1 INTRODUCTION

In the rapidly evolving landscape of wireless communication technologies, the transition from 5G to 6G Non-Terrestrial Networks (NTNs) represents a pivotal moment in the quest for enhanced connectivity, unprecedented data speeds, and transformative applications. As we stand on the cusp of this new era, characterised by the convergence of advanced technologies and the proliferation of interconnected devices, the need for a robust and adaptable infrastructure becomes increasingly apparent. This document serves as a comprehensive exploration of the key drivers, challenges, and opportunities inherent in the migration towards 6G non-terrestrial networks. By delineating the scope and objectives of 6G-NTN, elucidating the underlying principles of dynamic orchestration and autonomous monitoring, and delving into the intricacies of cybersecurity in a rapidly evolving threat landscape, this document seeks to provide a holistic understanding of the technological advancements shaping the future of wireless communication. Through a combination of theoretical analysis, practical insights, and forward-looking perspectives, this document aims to inform and inspire researchers, industry stakeholders, and policymakers alike, catalysing innovation and driving progress towards a more connected, intelligent, and secure digital future.

1.1 SCOPE AND OBJECTIVES

This deliverable is part of WP5, Tasks 5.2 and 5.3. Its scope and objectives are multifaceted, reflecting the complexity and breadth of the research and development efforts within 6G-NTN. At its core, this document aims to provide a comprehensive exploration of the transition from 5G to 6G non-terrestrial networks, elucidating the key innovations and potentials to be explored within this domain. By delving into various aspects such as dynamic orchestration, autonomous monitoring, and cybersecurity, the document seeks to address the evolving challenges and opportunities associated with the next generation of wireless communication technologies. Moreover, it aims to define the structural framework and technical solutions required to realize the vision of 6G non-terrestrial networks, including new virtualised and cloud-native architectures, ML-based traffic prediction techniques, and proactive security mechanisms. Through this comprehensive analysis, the document aims to lay the groundwork for future research endeavours and industry initiatives aimed at advancing the state-of-the-art in wireless communication technologies and driving the development of more resilient, efficient, and secure network infrastructures.

1.2 RELATION TO OTHER WORK PACKAGES IN 6G-NTN

The relation of Tasks 5.2 and 5.3 to the rest of the 6G-NTN project is illustrated in Figure 1.

WP5 was closely interconnected with all other work packages, enabling a coherent and system-level approach to the project objectives. This interdependency ensured that the orchestration, monitoring, and security solutions developed in the work-package were aligned with the architectural, technological, and use-case-driven requirements defined elsewhere in the project.

WP1 provided overall coordination and project management throughout the execution of WP5 activities. It ensured that Tasks 5.2 and 5.3 were carried out according to the project timeline and that their outputs were consistent with the global objectives and milestones of the 6G-NTN project.

WP2 defined the use cases and system requirements that guided the design choices in WP5. The orchestration mechanisms, VNF placement strategies, and security analyses developed in Tasks 5.2 and 5.3 were derived directly from these requirements, ensuring relevance with respect to targeted scenarios and performance expectations.

WP3 proposed the 3D network architecture for 6G-NTN, including terrestrial, aerial, and satellite components. WP5 built upon this architectural framework to study the integration of NTNs into an end-to-end 6G system. The theoretical VNF placement study conducted in WP5 relied on the architectural assumptions and connectivity models introduced in WP3, enabling a consistent evaluation of hybrid TN and NTN deployments.

WP4 focused on the design of air-interface technologies and data-driven algorithms for NTN. While WP5 did not directly modify radio-layer mechanisms, it complemented WP4 by addressing higher-layer orchestration, monitoring, and management aspects. The orchestration solutions developed in WP5 were designed to operate on top of the network behaviors and constraints identified in WP4.

WP6 was responsible for dissemination, standardisation, and exploitation activities. WP5 contributed to these efforts by providing concrete technical results, experimental platforms, and analytical insights that supported demonstrations, publications, and standardisation-oriented discussions.

At the time of execution, Tasks 5.2 and 5.3 primarily focused on VNFs deployed in virtualized environments. The work completed within WP5 includes both an experimental ML-driven orchestration platform and a theoretical evaluation of VNF placement strategies, laying the ground for assessing end-to-end performance and security implications of 6G-NTN architectures. The results produced in WP5 now serve as consolidated inputs for the overall project conclusions and future research directions.

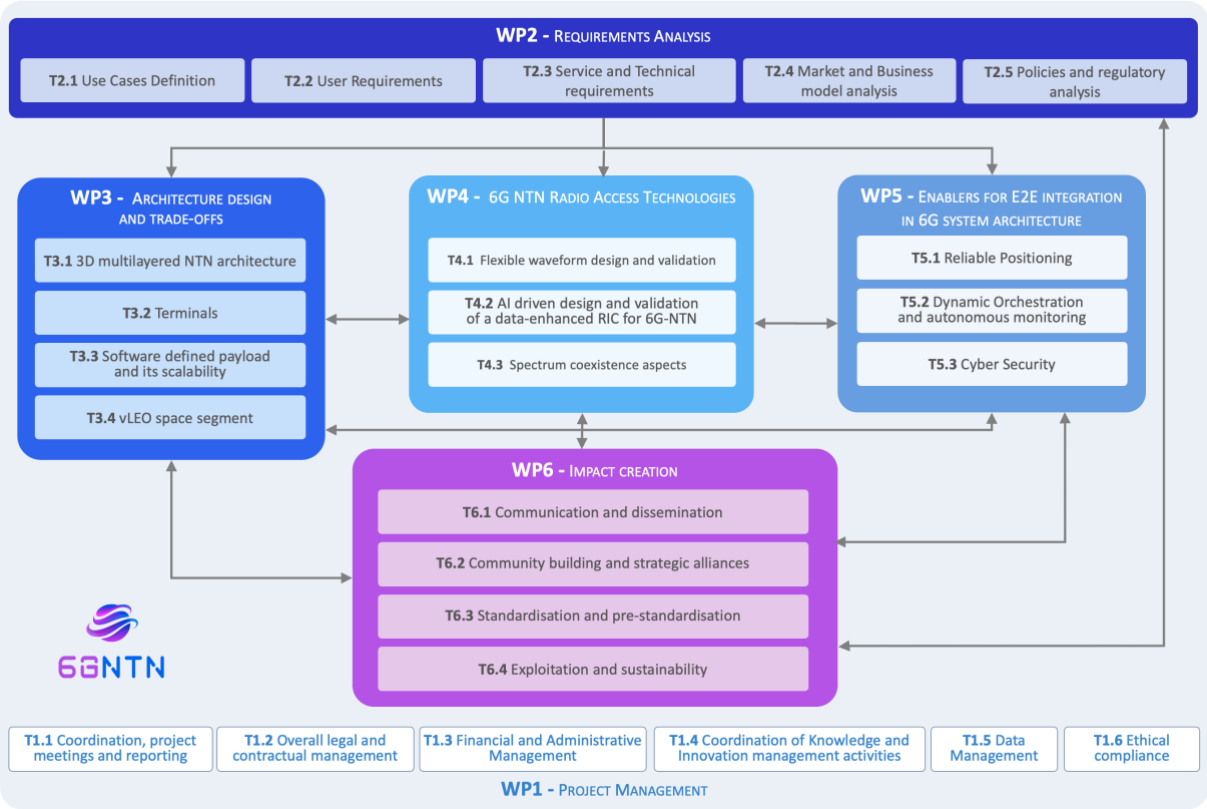

FIGURE 1: 6G-NTN WORK ORGANISATION

1.3 STRUCTURE OF THE DOCUMENT

This document is structured to guide the reader from the general context of 6G Non-Terrestrial Networks to the concrete architectural, analytical, and security contributions developed within tasks 5.2 and 5.3 of WP5.

Section 1 introduces the deliverable by defining its scope and objectives within the 6G-NTN project and by clarifying its relation to the other work packages. It provides the necessary context to understand how Tasks 5.2 and 5.3 contribute to the overall project vision.

Section 2 discusses the transition from 5G to 6G Non-Terrestrial Networks, highlighting the motivations, challenges, and innovation potentials associated with integrating terrestrial and non-terrestrial components. This section positions the work of WP5 within the evolution of 6G systems.

Section 3 presents the ML-based orchestration platform developed in the project. It details the underlying virtualized infrastructure, the Kubernetes-based execution environment, the monitoring and telemetry stack, and the orchestration logic. This section also describes the machine learning pipeline used for resource usage prediction and its integration with the network orchestrator to enable proactive VNF management.

Section 4 is dedicated to a theoretical study on VNF placement in a hybrid TN and NTN architecture. This study is conducted independently from the experimental platform and relies on discrete-event simulations. It evaluates multiple placement strategies under varying NTN

traffic ratios and jointly presents performance results and their discussion to assess the cost/benefit trade-offs of dynamic VNF relocation.

Section 5 addresses cybersecurity aspects related to cloud-native and distributed 6G-NTN architectures. It first presents a proactive security analysis method based on graph generation and attack path reconstruction from Kubernetes deployment descriptors. It then analyzes the security implications of distributing RAN and core network functions across satellites and ground infrastructure, with a focus on 3GPP-compliant security mechanisms and emerging threats.

Finally, Section 6 concludes the document by summarizing the main findings of the orchestration, theoretical evaluation, and security analyses, and by outlining their implications for future research and deployment phases of the 6G-NTN project.

2 FROM 5G TO 6G NON-TERRESTRIAL NETWORKS

The evolution of 5G into beyond 5G (5G-Advanced) and 6G networks aims at responding to the increasing need of our society for ubiquitous and continuous connectivity services in all areas of our life: from education to finance, from politics to health, from entertainment to environment protection. It is generally understood that the terrestrial network alone cannot provide the flexibility, scalability, adaptability, and coverage required to meet the above requirements, and the integration of the NTN component is a key enabler.

In this section, we describe how the work done in Tasks 5.2 and 5.3 can help achieve some of the objectives of the project detailed in the Direction of Work.

The dynamic network orchestrator we are designing is a crucial component in achieving some key objectives and exploring the innovation potentials of the 6G-NTN project. This section details how the ICS will respond to these objectives and innovation potentials over the next 18 months.

- **Objective #6:** *AI-enhanced Radio Intelligent Controller (RIC)*

The system can integrate AI-enhanced RIC capabilities to analyse traffic patterns and predict variations at both large and small scales. This predictive capability will enable the our system to proactively allocate resources, adjust network configurations, and optimize performance across the 3D network infrastructure. The AI algorithms will be continuously refined to improve accuracy and responsiveness.

- **Objective #7:** *VNF orchestration*

The dynamic network orchestrator will leverage advanced VNF orchestration techniques to integrate TN and NTN components within the 6G Edge and Core architectures. By utilizing lightweight micro-service orchestrators, the system will dynamically deploy VNFs and edge services on physical nodes. We believe this approach will ensure low-latency, high-throughput, and reliability services across the network.

Regarding the innovation potentials we planned to explore within the project, we provide here a list of the ones we target.

- Innovation potential #1: *Performance enhancement*

The dynamic orchestrator will significantly enhance network performance compared to 5G-NTN by employing advanced AI algorithms and dynamic orchestration techniques. These capabilities will optimize latency, data rates, and the number of devices managed by the network. It will ensure that new services requiring high performance are achievable, providing seamless connectivity to any 6G-NTN terminal.

- **Innovation potential #2:** *Ubiquitous connectivity and Resiliency*

The system will play a crucial role in reinforcing network resilience and providing ubiquitous coverage. By dynamically adjusting network configurations and resource allocations in response to real-time traffic variations and network conditions, it will ensure global service continuity and meet diverse Quality of Service (QoS) requirements. This capability will be essential for maintaining network operations during outages and other disruptions, with the ability to reconfigure resources in short time (order of a minute).

- Innovation potential #5: *Solutions as-a-Service*

By leveraging virtualised capabilities and cloud computing techniques, our dynamic orchestrator will enable fast (near-instantaneous) deployment of network functions. This approach will support Network as a Service (NaaS) and Infrastructure as a Service (IaaS) models, allowing for flexible and scalable service offerings. The system main goal is to ensure that network resources are efficiently utilised and can be quickly reconfigured to meet changing demands.

- Innovation potential #6: *Space Edge Computing*

Because the system we are designing is supposed to include all the components of our core network, it will incorporate Space Edge Computing to enhance latency-sensitive services by bringing computation closer to data sources. This will be particularly beneficial in remote and hard-to-reach locations. The orchestrator will manage and orchestrate edge computing resources located at various non-terrestrial layers, including satellites, High Altitude Platforms (HAPs), and aerial base stations.

- **Innovation potential #7:** *Fast adaptation to traffic variations*

Finally, the orchestrator will provide high dynamicity and re-configurability to absorb traffic variations at both large and small scales. By continuously monitoring traffic patterns and predicting future demands, it will dynamically adjust network configurations and resource allocations. This will ensure that the network can handle fluctuating traffic loads efficiently, maintaining optimal performance and user experience.

The Dynamic network orchestrator is poised to address several key innovation potentials of the 6G-NTN project, enhancing network performance, connectivity, and service flexibility. By integrating advanced AI, dynamic orchestration, and space edge computing capabilities, it will ensure that the 6G-NTN network is robust, resilient, and capable of meeting future demands.

3 ML-BASED ORCHESTRATION PLATFORM

The experimental platform developed for the 6G-NTN project is designed to provide a controlled environment for evaluating orchestration strategies, virtualized network functions, dynamic resource management, and real-time monitoring. Its architecture is based on a fully virtualized infrastructure composed of seven virtual machines hosted on a VMware ESXi hypervisor. All virtual machines operate under Ubuntu Server 22.04 LTS, ensuring uniformity across the environment, and simplifying automation tasks.

The platform, described in Figure 2, is structured into distinct functional layers. The first layer, the reverse proxy, provides secured external access and unified traffic entry. The second layer hosts the monitoring part and the entity that receives resources forecast and applies the orchestration logic. The third layer is a Kubernetes cluster that acts as the execution substrate for containerized network functions, monitoring agents, and the 5G Core. Together, these layers form an integrated and coherent system supporting the study of dynamic orchestration and network behaviour under various load conditions.

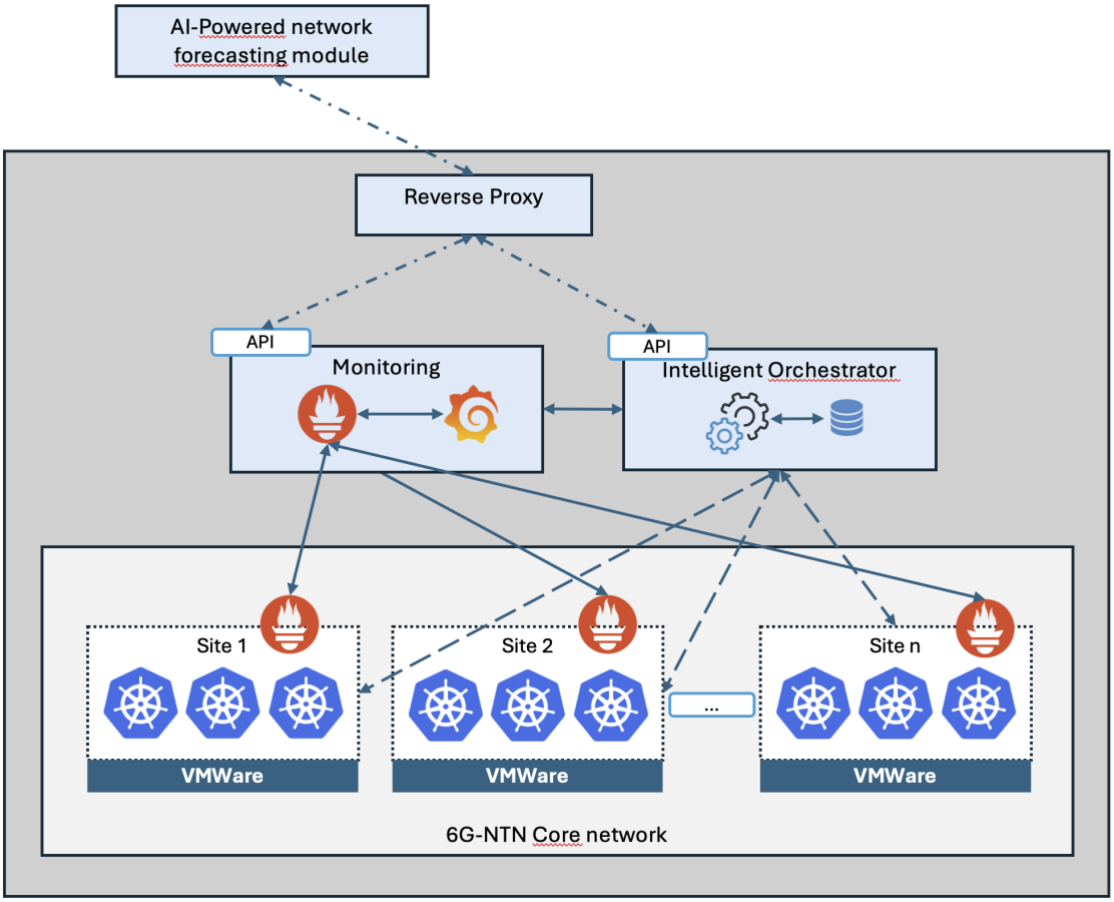

FIGURE 2: HIGH-LEVEL VIEW OF THE SOLUTION ARCHITECTURE DESCRIBING DIFFERENT COMPONENTS AND THE INTERACTIONS BETWEEN THEM

3.1 HARDWARE INFRASTRUCTURE AND VIRTUALIZATION

The entire platform is deployed on top of the VMware ESXi hypervisor, a virtualization layer that allows multiple independent virtual machines (VMs) to run concurrently on the same physical hardware while ensuring strong isolation between them in terms of execution, memory, and networking. In our case, this isolation is a key requirement, as it contributes directly to the security guarantees expected by Thales SIX.

VMware ESXi provides mature and well-established virtualization mechanisms, including advanced virtual networking features such as virtual switches and VLAN-based segmentation. These capabilities allow the separation of management, control, and data-plane traffic, which is essential in complex, security-sensitive environments. From an operational standpoint, the hypervisor also offers fine-grained resource allocation and monitoring, enabling each VM to be provisioned according to its functional role.

All seven virtual machines composing the platform are allocated CPU, memory, and storage resources based on their responsibilities. Nodes hosting the Kubernetes cluster are assigned higher compute and memory capacities to handle containerized workloads, while nodes dedicated to ingress management or orchestration are dimensioned according to these specific needs. In addition, a homogeneous operating system is deployed across all VMs, which reduces integration complexity and facilitates maintenance.

The choice of VMware ESXi was primarily driven by practical and operational considerations rather than by a theoretical comparison of alternative hypervisors. The entire virtualized infrastructure hosting our platform is already managed using VMware ESXi, making it the de facto standard within the existing environment. This choice ensures compatibility with established operational processes, security policies, and tooling, and avoids introducing additional complexity or operational risks. As a result, no dedicated trade-off study between different hypervisors (e.g., KVM or Hyper-V) was conducted, as VMware ESXi already fulfilled all functional and security requirements of the platform.

3.2 ACCESS NODE AND REVERSE PROXY

The first virtual machine assumes a dual role that is essential to the security and accessibility of the entire platform. It acts simultaneously as an external *ssh* access gateway and as the reverse proxy for all http application-level services exposed to users and automation tools.

As the main entry point into the platform, this node ensures that all incoming connections follow well-defined access policies (IP source and public key accesses). It performs all necessary filtering functions, provides a secure interface for remote administration, and ensures that no backend component is directly reachable from outside the controlled environment. All traffic passes through this gateway before being distributed internally.

In addition to access control, the same node hosts the reverse proxy responsible for forwarding HTTPS traffic to backend services. These services include the orchestrator's REST API, the HTTP interface of Prometheus and the Grafana visualization dashboard. This design also enables certificate management, routing rules, authentication layers and monitoring hooks to be centralised in one location, greatly simplifying maintenance.

3.3 ORCHESTRATOR NODE

The second virtual machine hosts the orchestrator, which constitutes the decision-making brain of the platform. The orchestrator receives forecasts of anticipated resource consumption and evaluates strategies for relocating or replicating virtualised network functions. It is through this component that experiments involving smart orchestration, predictive modelling or resource-aware VNF placement are performed.

The orchestrator exposes a REST API that allows external tools to submit forecasts which triggers orchestration cycles. This API is intentionally designed to be modular so that new algorithms can be integrated without modifying the overall platform structure. The orchestrator does not execute VNFs itself, it issues commands to the Kubernetes cluster, manages metadata associated with VNF lifecycles, and maintains a global view of resource utilisation.

The orchestration logic may include different strategies. Its objective is to determine when and where VNFs should be migrated, duplicated or consolidated, based on current and predicted system conditions. The orchestrator is therefore essential for evaluating dynamic management policies envisioned for 6G and beyond.

3.4 5 NODES KUBERNETES CLUSTER

Five virtual machines form a Kubernetes cluster that provides the execution environment for all containerised network functions and monitoring components. Each node runs the required container runtime and system agents, forming a distributed substrate capable of hosting a full 5G/6G Core network. A fundamental requirement for the platform was the use of an open-source solution supporting native multi-node deployments in order to obtain a realistic execution environment aligned with the research objectives.

We selected Open5GS and free5GC because, at the start of the project in January 2023, they were the only open-source 5G core implementations that were both sufficiently mature and actively maintained for realistic multi-node deployments. Other options such as Open6GCore[1] have since emerged as promising candidates, but the first public documentation and publications for this 6G-oriented core appeared only around mid-2024, when the project was already midterm of its timeline and changing plans was no longer realistic.

In practice, Open5GS and free5GC were also the only platforms that the team was able to install and operate on our distributed multi-node testbed after numerous unsuccessful attempts with alternative stacks. Several deployment iterations were carried out, including experiments with OpenAirInterface's 5G core, but these never resulted in a stable configuration. This empirical experience, combined with the open-source and standards compliant nature of Open5GS and free5GC, ultimately justified focusing the experimental evaluation on these two implementations.

Finally, while there are several commercial 5G core solutions from vendors such as Ericsson, Nokia, and Huawei, the project explicitly required an open-source core to ensure full transparency, modifiability of the software stack, and reproducibility of the experiments. Commercial, closed-source products were therefore out of scope, even when technically attractive, because they could not satisfy these openness and research-driven constraints.

Both solutions were evaluated. In practice, each presented significant challenges when deployed on a real multi-node Kubernetes cluster. Ensuring full interoperability between components, stabilising inter-service communication, and achieving a 100% functional end-to-end deployment consumed a substantial amount of engineering effort. Despite these difficulties, Open5GS ultimately demonstrated better stability across multiple nodes and an easier integration with the monitoring and orchestration layers. For these reasons, Open5GS was selected as the core network implementation for our platform.

The Kubernetes cluster therefore hosts the complete Open5GS 5G Core, with all major control-plane functions and User Plane Function (UPF) deployed as containerised network functions. These include the Access and Mobility Management Function (AMF), the Session Management Function (SMF), the Authentication Server Function (AUSF), the Unified Data Management (UDM), the Policy Control Function (PCF), and the Network Repository Function (NRF), in addition to the User Plane Function (UPF).

. The modular architecture of Open5GS enables the cluster to be reorganised or extended depending on the experiment being conducted, including scenarios involving multiple UPFs, failure recovery, controlled overload, or dynamic VNF migration.

A monitoring system is also deployed on top of this cluster. Prometheus acts as the central time-series engine and scrapes metrics from a node exporter running on each virtual machine, providing system-level indicators such as CPU usage, memory consumption, disk activity, and process saturation. Prometheus additionally collects application-level metrics emitted by Open5GS. Grafana runs alongside Prometheus and offers the visualisation environment. Dashboards are available externally through the reverse proxy hosted on the access gateway node.

3.5 METRICS COLLECTION AND EXPLOITATION

The platform integrates Prometheus as the central telemetry engine. Prometheus continuously collects a broad range of metrics, including compute resource usage across all Kubernetes nodes, detailed performance metrics of the core network, as well as experimental data exposed by VNFs deployed on the cluster. This includes values related to throughput, uplink and downlink load, signalling message rates, pod life-cycle events, scheduling latency and error conditions. Prometheus stores these metrics as time series that can be queried directly by the ML platforme.

Grafana provides the visualisation layer for data exploration, enabling the creation of dashboards that visualise real-time or historical data. We rely on these dashboards to demonstrate the number of VNFs executed on each of nodes, which shows that a VNF has been moved following the reception of a forecast message. This will be described in the demonstration section.

3.6 HIGH-LEVEL ARCHITECTURE VIEW

The experimental platform is deployed within the Thales SIX facilities and is organized around a logically isolated 6G-NTN VLAN, ensuring controlled access, network segmentation, and

security. External components interact with the platform exclusively through a secured gateway, while all internal communications occur within the private VLAN.

External access to the platform is provided via the Internet through two public endpoints, api.6g-ntn.eu and metrics.6g-ntn.eu, both mapped to the same public IP address (192.93.161.124). A firewall enforces strict filtering rules, allowing only SSH (port 22) from specifi source IP addresses and HTTPS/HTTP traffic (ports 443 and 80). This firewall acts as the first security barrier between the public network and the internal infrastructure.

Behind the firewall, an access gateway node plays a dual role. First, it acts as an SSH proxy, serving as the single administrative entry point to the platform. Second, it hosts an HTTP(S) reverse proxy that exposes selected internal services to external users and automation tools. This node is part of the 6G-NTN VLAN and ensures that no internal component is directly reachable from the public Internet.

The core of the platform is composed of a Kubernetes cluster deployed on five virtual machines. Each node hosts Open5GS network functions and runs a Prometheus agent responsible for exporting system-level and application-level metrics. These metrics include CPU and memory usage, process activity, and network-related indicators. The Kubernetes cluster provides a realistic, multi-node execution environment for containerized network functions, enabling experiments related to orchestration, scaling, and dynamic VNF placement.

A dedicated monitoring node hosts Prometheus and Grafana. Prometheus collects metrics from all Kubernetes nodes by scraping the Prometheus agents deployed on each of them. Grafana provides visualization dashboards that allow real-time and historical inspection of system behavior. Access to Grafana is exposed externally through the reverse proxy, enabling remote observation without direct access to the internal network.

An orchestrator node is responsible for decision-making and control. It receives resource usage forecasts generated by the machine learning components. Based on these inputs, the orchestrator issues reconfiguration messages to the Kubernetes cluster, triggering actions such as VNF migration or replication. These control messages are sent directly to the cluster over the internal VLAN.

The machine learning node hosts an execution version of the ML platform used for resource usage prediction. It retrieves monitoring data from Prometheus and generates forecasts related to future CPU and memory consumption of network functions. These forecasts are transmitted to the orchestrator via http REST requests, enabling proactive orchestration decisions. In addition, an external ML platform located at Martel facilities, described in details in Section 3.7 can interact with the system via the Internet, using the exposed APIs to submit forecasts or retrieve results.

Following Figure 3**Error! Reference source not found.** details the architecture of the platform.

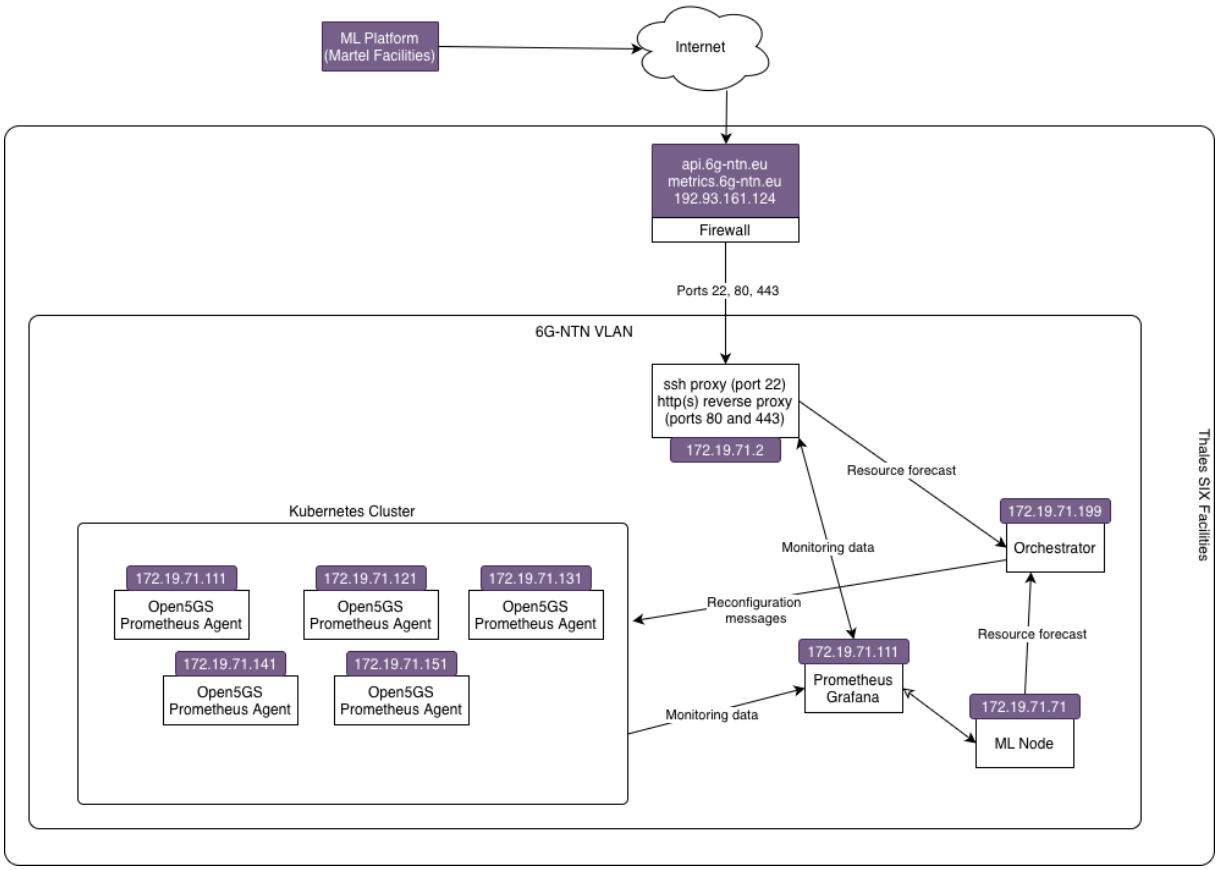

FIGURE 3: NETWORK AND INTERACTION VIEW OF THE ORCHESTRATION PLATFORM.

3.7 ML TECHNIQUES FOR CNF RESOURCE USAGE PREDICTION

This section provides an update to the work presented in Deliverable D5.2, concerning the application of ML solutions for network resource prediction. The initial framework from D5.2 has been significantly enhanced, evolving from a proof-of-concept into a fully orchestrated and automated platform. Key advancements include a robust containerized architecture (see Figure 5) using industry-standard tools for workflow orchestration and a concrete integration with the 6G core network platform environment.

The work reported in D5.2 laid the theoretical groundwork, serving as a proof-of-concept for ML-based resource prediction. This initial phase relied on offline analysis of static datasets, such as historical AMF CPU usage, and employed manual, ad-hoc scripts for training and evaluation. The primary focus was on model exploration, comparing algorithms like ARIMA, LSTM, and Prophet to identify the optimal forecasting approach, but without the infrastructure for automated scheduling or error handling.

Building on this foundation, D5.3 advances the framework into a fully orchestrated MLOps platform, now integrated with the 6G core network prototype. This evolution enables real-time capabilities, transitioning from static analysis to direct data ingestion via Prometheus, which scrapes live metrics from Cloud-Native Network Functions (CNFs). The architecture has been consolidated into a containerized ecosystem, utilizing Docker to deploy the Orchestrator, Workers, Database, and Visualization components as independent units.

The main objective remains the proactive management of network resources by accurately forecasting resource metrics such as CPU and memory usage of Cloud-native Network Functions (CNFs). These forecasts enable the orchestrator (described in section 3.3) to anticipate demand and allocate or deallocate resources efficiently, ensuring service quality while optimizing energy consumption. Compared to the D5.2 prototype (which targeted a single metric – CPU usage of an AMF – using offline data), the current implementation handles multiple metrics (both CPU and memory) on live data from the 6G core prototype. This broadened scope and real-time data integration greatly increase the practical impact of the forecasts.

3.7.1 Architecture

Building upon the architecture described in D5.2(Figure 5), the platform has been consolidated into a container-based ecosystem orchestrated via Docker Compose. This approach ensures portability, scalability, and ease of deployment. The core components of the architecture are detailed below:

- **PostgreSQL Database**: it has been integrated as a durable high-resolution time-series database needed for ML training. A dedicated ingestion service (described later) periodically fetches metrics from Prometheus and stores them in this database, creating an historical dataset for model training and evaluation. A retention policy (2 days of data) is applied to cap the database size and emulate a rolling window of recent data. The database schema is designed to accommodate two distinct data streams:
  - Prometheus metrics: are stored with a sampling frequency of 15 seconds and a retention period of 2 days.
  - Generated predictions: are stored with a frequency of 5 minutes and a retention period of 7 days.
- **Prefect Orchestrator:** The entire ML pipeline – from data ingestion to forecasting – is managed by **Prefect**, a modern workflow orchestration tool. Prefect was chosen over simpler cron jobs or alternative orchestrators due to its powerful features for building, scheduling, and monitoring complex data pipelines. It provides: *Reliability:* automatic retries on failure and detailed logging for each task. *Scalability:* the ability to distribute tasks across multiple worker processes or nodes, enabling parallel execution. The deployment is made through multiple Prefect workers (e.g., separate workers for metrics ingestion, forecasting, and housekeeping) to isolate workloads and scale out when needed. *Parameterization:* easy reconfiguration of flows (e.g., adjusting forecast horizon or model parameters) without modifying code. *Dynamic workflows:* the capability to generate tasks at runtime (not just static DAGs), allowing adaptation of the pipeline based on context (e.g. different metrics or NFs).

  The architecture leverages Prefect's distributed execution model to ensure resource isolation and scalability. This is implemented using Work Pools, which act as dedicated queues for specific types of tasks, and Workers, which are the autonomous processes that poll these pools to execute flow runs. The system employs three distinct Work Pools to categorize workloads:

- *metrics_ingestion*: A pool dedicated to high-frequency, low-latency tasks such as scraping Prometheus metrics.
- *LSTM_forecasting*: A pool allocated for compute-intensive Machine Learning training and inference processes.
- *Postgres_retention*: A pool responsible for background maintenance tasks like database cleanup.

Corresponding Workers (e.g., `*prefect-forecasting-worker-1*`) are deployed to subscribe to these pools. This strategy prevents resource contention, ensuring that heavy model training does not block critical data ingestion, and allows for independent scaling, where additional workers can be added to a specific pool to handle increased load.

- **Minio Object storage:** A MinIO instance is deployed to function as the model registry and storage backend for Prefect flow definitions. This architectural decision decouples code and model artifacts from the execution environment, thereby ensuring version control and facilitating seamless deployment across distributed workers.
- **Grafana:** Used for visualization of both real-time data and forecast results. Grafana dashboards allow operators to monitor live metrics from Prometheus and to compare predicted resource usage against actual consumption (stored in PostgreSQL). This visual feedback helps validate model performance and can highlight when forecasting errors increase (indicating, for example, concept drift or anomalies).
- All these components run as isolated Docker containers on a dedicated virtual machine (VM) within the prototype environment (Figure 4). This containerized architecture ensures reproducibility and easy maintainability of the system across different environments.

```
alice@modele:~/6G-NTN-resource-forecasting/src$ docker ps
CONTAINER ID   IMAGE                                                COMMAND                  CREATED        STATUS       PORTS
                NAMES
ed0c99e836b0   6g-ntn-ml-platform-prefect-forecasting-worker-1      "prefect agent start…"   2 hours ago    Up 2 hours
                LSTM_forecasting_worker_1
d0fd55017376   6g-ntn-ml-platform-prefect-postgres-retention-worker-1   "prefect agent start…"   2 hours ago    Up 2 hours
                postgres_retention_worker_1
06a269328cc4   6g-ntn-ml-platform-prefect-metrics-ingestion-worker-1    "prefect agent start…"   2 hours ago    Up 2 hours
                metrics_ingestion_worker_1
329f60a5c63b   6g-ntn-ml-platform-prefect-forecasting-worker-2      "prefect agent start…"   2 hours ago    Up 2 hours
                LSTM_forecasting_worker_2
87afbf3a4338   grafana/grafana:12.2.1-security-01-ubuntu            "/run.sh"                4 weeks ago    Up 2 days    0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp
                grafana
32df89230b34   postgres                                             "docker-entrypoint.s…"   4 weeks ago    Up 2 days    0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp
                forecasting-postgres-database
3445538c3df7   6g-ntn-ml-platform-prefect-orion                     "/bin/bash -c 'prefe…"   4 weeks ago    Up 2 hours   0.0.0.0:4200->4200/tcp, [::]:4200->4200/tcp
                prefect_orion
c9b91c6dad43   minio/minio:RELEASE.2025-09-07T16-13-09Z-cpuv1       "/usr/bin/docker-ent…"   2 months ago   Up 2 days    0.0.0.0:9000-9001->9000-9001/tcp, [::]:9000
-9001->9000-9001/tcp   minio
b349e605ff85   postgres:15.2-alpine                                 "docker-entrypoint.s…"   2 months ago   Up 2 days    5432/tcp
                prefect_db
```

FIGURE 4: RUNTIME STATUS OF THE CONTAINERIZED INFRASTRUCTURE RUNNING ON THE ENVIRONMENT PROTOTYPE

The interaction between the major components is illustrated in the architectural diagram below (Figure 5).

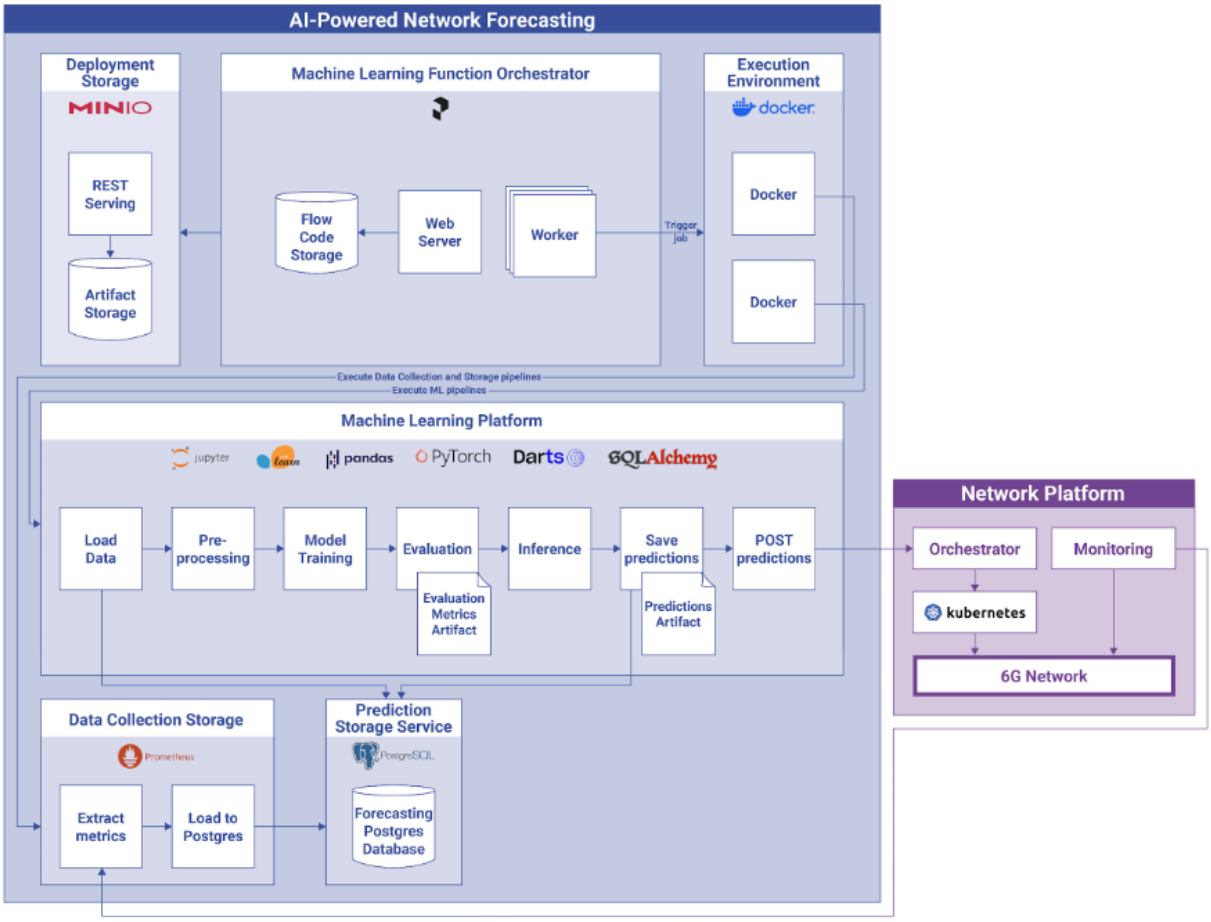

FIGURE 5: AI-POWERED NETWORK FORECASTING ARCHITECTURE

3.7.2 Implementation approaches and improvements

The fundamental forecasting approach continues to leverage **Long Short-Term Memory (LSTM)** neural networks, as these are well-suited for capturing temporal dependencies in resource utilization time series. However, the process around model training and inference has been significantly **refined and automated** compared to D5.2. In particular, the scope of forecasting has expanded (covering multiple metrics on a live system) and the entire pipeline is now orchestrated and resilient, rather than being a manual offline experiment.

The end-to-end forecasting process is implemented as Prefect flows, which consist of a series of dependent tasks executed in a coordinated manner.

The data ingestion process relies on a Prefect workflow (`*prometheus-to-postgres*`) that runs at regular intervals (every minute) to collect fresh operational data.

During each execution, the workflow queries the Prometheus HTTP API to obtain the CPU and memory consumption of the pod running UPF, looking back at 1 minute data and with a frequency of 15s. The retrieved values are added to the time-series dataset stored in PostgreSQL, which forms the historical basis for training. This mechanism ensures that the forecasting models are continuously exposed to up-to-date information and remain sensitive to short-term variations in resource demand.

A second Prefect workflow (`*ml-pipeline*`) is triggered every five minutes and is dedicated to the machine learning pipeline. It starts by retrieving a recent window of historical data from PostgreSQL. The data is then prepared for modelling. This preparation includes reshaping it into a unified time-series structure, enforcing a consistent sampling interval of five minutes by aggregating the finer granularity points using the max function, interpolating missing values when needed, and applying scaling to normalise the range of measurements.

Once the data has been prepared, an LSTM model is trained using the Darts library. The model is trained using the full time series. The framework performs an internal validation check to monitor the loss, and early stopping is applied to end the optimisation when no further improvement is detected. This prevents the model from overfitting and ensures that the version selected for forecasting is the best performing one from the training session. PyTorch Lightning manages the internal optimisation routine, including the validation checks and checkpointing of the best model.

When training is complete, the updated model produces a prediction for one time step ahead, which by default corresponds to the next five-minute interval based on the configured resampling frequency. The framework is capable of producing longer or multi-step forecasts if required for specific orchestration strategies by adjusting the steps parameter. The generated forecast, together with its timestamp and relevant metadata (network function, metric name, node, predicted value), is then pushed to the orchestrator and stored in PostgreSQL's forecasted table for historical archiving and visualization

This automated and orchestrated approach represents a major enhancement over the manual experiments described in D5.2. It transforms the earlier concept into a reliable, production-ready service. The use of Prefect flows ensures that each step is executed in order, monitored, and logged. Failures in any task (e.g., a temporary Prometheus query failure) will trigger automatic retries, and any persistent issues are visible in the Prefect dashboard for troubleshooting. The entire pipeline can run continuously and hands-free, providing a steady stream of predictions for consumption by the network orchestration logic.

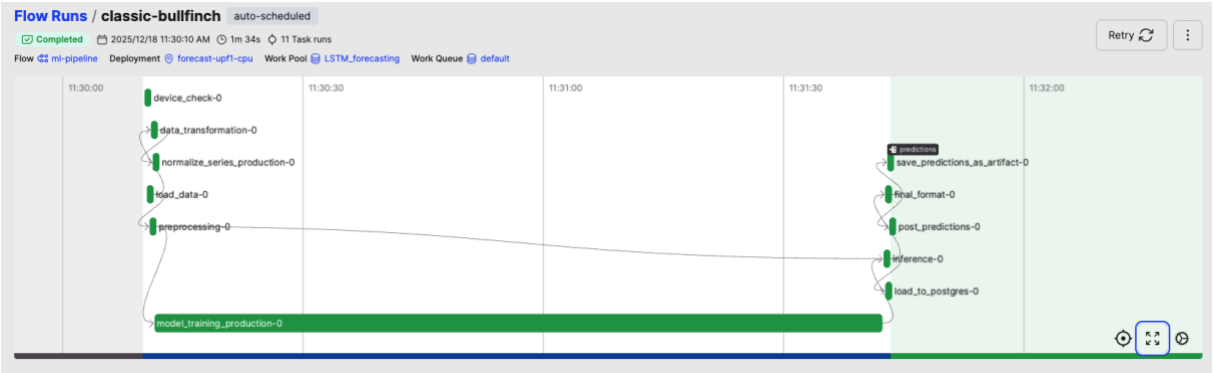

FIGURE 6: ML FORECASTING PIPELINE ORCHESTRATED AS A PREFECT FLOW

Forecasting Horizon and Frequency

The prediction interval was a key area of refinement. The initial goal was a 1-minute forecast executed every minute. However, performance benchmarking on the demonstration VM revealed that reducing the resampling frequency to 1 minute would significantly expand the

dataset size, resulting in training cycles that would exceed the desired prediction interval and make real-time forecasting infeasible. To ensure system stability and prevent execution queue buildup, the interval was adjusted to 5 minutes. The current 5-minute configuration maintains operational value for orchestration while ensuring system reliability. With substantially more powerful hardware (e.g., GPU acceleration or additional CPU cores), the platform could potentially support shorter intervals, though this would require both faster computation and the ability to handle larger datasets efficiently.

3.7.3 Integration with Network Platform

A key achievement in this last phase of the project is the successful integration of the forecasting platform with the 6G-NTN Core Network prototype, demonstrating a practical application of ML-driven network automation in a prototype deployment environment. The integration occurs at two main levels: (1) data acquisition from the 6G core network environment, and (2) ML forecasts dispatch to the network orchestration system.

3.7.3.1 Data Acquisition

The entry point for integration is the collection of real-time metrics from the network monitoring infrastructure. The CNF pods/containers whose resource usage we want to predict (such as the UPF) are deployed and orchestrated by the Network platform on a Kubernetes cluster. A Prometheus instance, already running within this environment, is configured to automatically discover and scrape metrics from these components (for example, via Kubernetes service discovery and cAdvisor metrics for pods). The AI-powered forecasting platform connects to this Prometheus server as its primary data source, treating it as an internal metrics provider. This integration is non-intrusive: it does not require any modification to the CNFs or the Prometheus setup, since it leverages existing monitoring data. The data acquisition is therefore low-latency and secure by virtue of the platform's network placement.

3.7.3.2 Exposing Forecasts for Consumption by Network Platform orchestrator

After the ML platform has trained its models on the collected data and generated resource usage forecasts, the resulting predictions are delivered directly to the network platform's orchestration layer through a dedicated API. In practice, as soon as new forecasts are produced, the predicted metrics for upcoming time intervals are transmitted by the forecasting module to the 6G network prototype's orchestrator via the orchestration API endpoint. This push-based integration ensures that predictions reach the orchestrator in a timely manner, without requiring the orchestrator to query or access any internal storage of the ML platform.

With these predictive insights available, the network orchestrator can immediately incorporate them into its management logic. For example, if a forecast indicates that the CPU load on a particular CNF will exceed a specified threshold within the next hour, the orchestrator can proactively initiate a scaling action or resource reallocation before that threshold is reached. This direct communication between the ML platform and the orchestration component establishes a rapid, event-driven feedback loop for network automation.

3.7.3.3 Testing and monitoring

After deployment, the integration was verified by checking the Prefect Orion UI and Grafana. The Prefect UI showed that the flows were running on schedule and succeeding (Figure 6), and Grafana dashboards were able to display both live metrics and the forecast values (queried respectively from the PostgreSQL *input and forecast* table) (Figure 7).

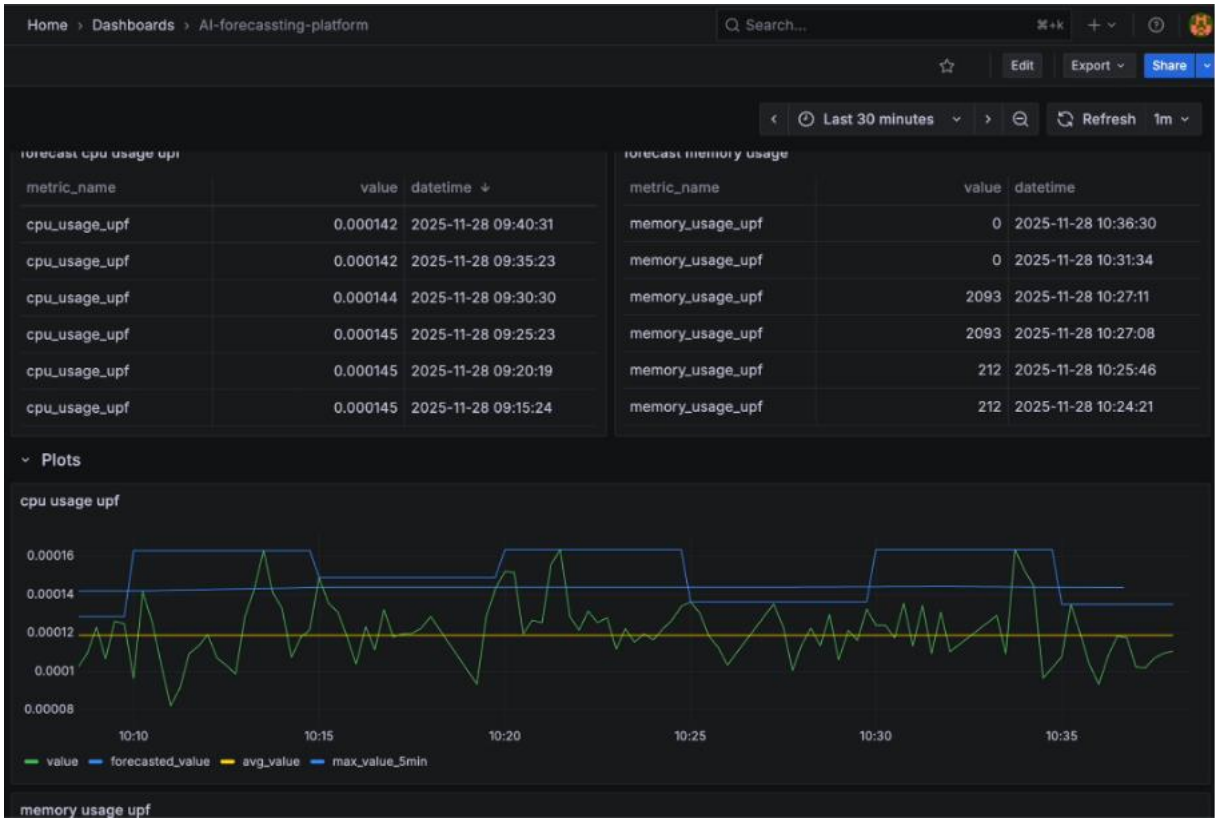

FIGURE 7: GRAFANA DASHBOARD VISUALIZING REAL-TIME CPU/MEMORY USAGE AND THE CORRESPONDING FORECASTS

In summary, the integration with the 6G core network platform demonstrates a complete end-to-end workflow for ML-driven resource management: from collecting data out of a live network environment, to generating predictions using advanced ML models, and finally to making those predictions accessible to the orchestration layer for proactive network management. The entire pipeline runs continuously and autonomously, validating the concepts presented in D5.2 in a prototype environment and providing a solid foundation for future enhancements, ready for evaluation in the context of the 6G-NTN project.

4 VNF PLACEMENT THEORETICAL STUDY

This section introduces a theoretical study conducted independently from the implementation and deployment of the experimental platform described in the previous sections. The objective of this study is to analyze, from a purely conceptual and analytical perspective, whether dynamic VNF placement and relocation mechanisms are beneficial in an hybrid terrestrial and non-terrestrial network architecture. More specifically, it aims to evaluate the trade-off between the potential performance gains brought by VNF relocation, such as reduced latency, improved SLA compliance, or better resource utilization, and the associated costs, including increased operational complexity, signaling overhead, and migration-related expenses. By decoupling this analysis from any specific implementation constraints, this study provides a general understanding of the conditions under which VNF relocation is justified, and identifies scenarios where such mechanisms are either advantageous or unnecessary.

This study presents a comprehensive performance evaluation of VNF placement strategies in hybrid architectures combining terrestrial datacenters and LEO satellite constellations. Through discrete-event simulations, we investigate the impact of NTN traffic ratios (30%, 50%, 70%, and 85%) on network performance metrics including latency, SLA compliance, operational costs, and resource utilization.

We evaluate five placement strategies: baseline (terrestrial-only), move-to-LEO (permanent migration), duplicate-on-peak (temporal replication), greedy heuristic (cost-benefit optimization), and flash-crowd (emergency response).

This study addresses the following research questions:

1. At what NTN traffic threshold does LEO VNF placement become performance-beneficial compared to terrestrial-only deployment?
2. What is the cost-benefit trade-off between different placement strategies across varying NTN traffic ratios?
3. How do different policies (permanent migration, temporal replication, greedy optimization) compare in terms of latency, SLA compliance, and operational costs?
4. What are the resource utilization characteristics (CPU, memory) of LEO nodes under different placement strategies?

4.1 RELATED WORK

Virtualization and softwarization have enabled flexible placement of VNFs across terrestrial and non-terrestrial infrastructures, but most existing work has focused either on purely terrestrial edge computing, or on satellite-only resource allocation, with relatively little attention to integrated, empirically evaluated terrestrial/LEO deployments. The following state of the art organizes prior work into three main strands: VNF placement in terrestrial networks, satellite network virtualization, and hybrid satellite-terrestrial architectures, and highlights how the cited studies motivate a joint treatment of placement and migration decisions in hybrid terrestrial/LEO systems.

4.1.1 VNF placement in terrestrial edge networks

Taleb et al.[2] provide a comprehensive survey of Multi-Access Edge Computing (MEC), emphasizing how placing compute and storage resources at or near base stations can drastically reduce end-to-end latency and backhaul load compared to centralized cloud architectures. Building on such architectural insights, Mach et al.[3] investigate mobile edge computation offloading and placement strategies, often formulating the problem as an (integer) optimization to jointly minimize latency and bandwidth usage for delay-sensitive applications. Taken together, these works show that ILP-based and related placement formulations can yield substantial latency reductions on the order of several tens of percent by moving VNFs from core to edge, but they focus on terrestrial MEC without considering the additional constraints and dynamics introduced by LEO satellites.

4.1.2 Satellite network virtualization and NTN

From the satellite side, Giambene et al.[4] survey satellite-5G integration and outline architectures in which SDN and NFV are used to virtualize satellite resources and integrate them with the 5G core. Their analysis highlights that, in non-terrestrial networks, limited on-board processing power and especially inter-satellite link (ISL) capacity can become critical bottlenecks, constraining where VNFs can be placed and how traffic can be routed across a constellation. More recent work by Yuan et al.[5] explicitly formulates joint VNF placement and routing in software-defined satellite-terrestrial integrated networks (SDSTNs) as an integer optimization problem on a time-evolving graph, targeting average latency minimization under tight satellite resource constraints, but still largely at a model-based level without extensive empirical validation on concrete LEO traffic.

4.1.3 Hybrid terrestrial-satellite architectures

Hybrid architectures that integrate terrestrial and satellite segments have been studied mainly from a control and routing perspective, with limited empirical evidence on when migrating VNFs towards the satellite becomes beneficial. Existing work on satellite-5G integration typically proposes high-level architectures and highlights use cases (coverage extension, resilience, multicast), while leaving VNF migration timing and partial migration strategies largely open. Studies such as Yuan et al. already couple placement with routing in dynamic satellite-terrestrial topologies, but they optimize over abstract service demands and do not explicitly characterize “crossover points” at which offloading VNFs to LEO yields net latency or cost gains compared to keeping them on terrestrial edge/central clouds for different NTN traffic ratios.

4.1.4 Positioning of the present contribution

Against this background, the present work extends terrestrial MEC-style VNF placement to a hybrid terrestrial-LEO setting where only a subset of VNFs can be

migrated, and where migration decisions must account for ISL bottlenecks and fast-varying LEO connectivity. In contrast to prior satellite and SDSTN studies that are predominantly analytical, it provides, to the best of current knowledge, a first comprehensive cost/benefit analysis across multiple NTN traffic ratios, explicitly quantifying when LEO-side hosting becomes advantageous and identifying empirical crossover points that were not characterized in earlier work.

4.2 SYSTEM MODEL AND METHODOLOGY

4.2.1 Topology

Our simulation models a hybrid network architecture comprising:

- **Ground Segment:**
  - 3 ground datacenters (ground_0, ground_1, ground_2)
  - Per-node resources: 64 CPU cores, 128 GB RAM
  - Inter-datacenter latency: 10 ms
  - Bandwidth: 10 Gbps
- **Space Segment:**
  - 5 LEO satellite nodes (leo_0 through leo_4)
  - Per-node resources: 64 CPU cores, 128 GB RAM
  - Inter-satellite link (ISL) latency: 3 ms
  - Ground-to-LEO feeder link latency: 15 ms
  - Ground-to-LEO bandwidth: 2 Gbps
  - ISL bandwidth: 1 Gbps

4.2.2 VNF Types

We model seven 3GPP 5G Core VNF types with the following resource requirements:

VNF Type	CPU (cores)	RAM (MB)	Binary (MB)	State (MB)	Processing Delay (ms)
AMF	4.0	8,192	2,048	512	2.0
SMF	2.0	4,096	1,024	256	1.5
UPF	8.0	16,384	1,536	128	0.5
AUSF	2.0	4,096	512	64	3.0
UDM	2.0	8,192	768	1,024	2.5

VNF Type	CPU (cores)	RAM (MB)	Binary (MB)	State (MB)	Processing Delay (ms)
PCF	2.0	4,096	512	128	1.0
CACHE	4.0	32,768	256	4,096	0.2

4.2.3 Traffic Model

4.2.3.1 Traffic generation

Traffic follows a diurnal pattern with:

- Baseline: 1000 requests/second
- Peak multiplier: 3.0 times
- Peak hours: 08:00-11:00, 18:00-21:00
- Simulation duration: 24 hours (288 timesteps of 5 minutes each)

4.2.3.2 NTN Traffic ratios

We evaluate four scenarios with varying NTN traffic ratios:

Scenario	Baseline NTN	Peak NTN	Description
NTN-30	30%	45%	Urban area with supplementary satellite
NTN-50	50%	65%	Semi-rural or suburban with balanced connectivity
NTN-70	70%	85%	Rural or maritime area with satellite dominance
NTN-85	85%	92%	Extreme scenario: polar, or disaster zone

4.2.4 Placement policies

4.2.4.1 Baseline

Description: All VNFs remain on ground datacenters (reference scenario).

Parameters: None (static placement).

This policy represents traditional terrestrial deployment.

4.2.4.2 Move-to-LEO

Description: Permanent migration of selected VNFs (AMF, UPF, CACHE) to LEO when NTN traffic exceeds a threshold.

Parameters:

- NTN traffic threshold: 50-60% (scenario-dependent)
- Latency improvement threshold: 8.0 ms
- Reduced ground-LEO latency: 15 ms

This policy optimizes for NTN users by reducing ground-satellite round trips.

4.2.4.3 Duplicate-on-Peak

Description: Enhanced replication with more aggressive thresholds.

Parameters:

- Load threshold (duplicate): 1.5 times
- Load threshold (remove): 1.1 times
- Node CPU utilization threshold: 35% (reduced from 70%)
- Max replicas per VNF: 3

Ensures duplication triggers more reliably.

4.2.4.4 Greedy Heuristic

Description: Cost-benefit optimization with multi-factor scoring.

Parameters:

- Latency weight: 1.0
- Migration cost weight: 0.5
- Saturation penalty weight: 2.0
- Action threshold: 5.0
- Max actions per timestep: 3

Balances latency improvements against migration overhead.

4.2.4.5 Flash Crowd

Description: Emergency response policy for sudden traffic spikes (5 times multiplier at 14:00).

Parameters:

- Spike hour: 14.0
- Spike multiplier: 5.0 times

- Spike duration: 30 minutes
- Action threshold: 3.0 (more aggressive than greedy heuristic).

Tests resilience to unpredictable surges.

4.2.5 Performance Metrics

We measure the following Key Performance Indicators (KPIs):

- Latency:
  - Average end-to-end latency (ms)
  - 95th percentile (P95) latency (ms)
  - Maximum latency (ms)
- Service Quality:
  - SLA compliance rate (%)
  - Packet loss rate (%)
  - Completed requests
- Operational Costs:
  - Compute cost ($)
  - Migration cost ($)
  - Total cost ($)
- Resource Utilization:
  - Ground CPU average/peak (%)
  - LEO CPU average/peak (%)
  - Memory utilization (%)
- VNF Relocation:
  - Number of migrations
  - Migration volume (MB)
  - Total downtime (ms)
- VNF Distribution:
  - VNFs on ground
  - VNFs on LEO

4.2.6 Simulation Environment

- Simulator: Custom discrete-event simulator implemented with Python language

- Timestep: every 5 minutes
- Duration: 288 timesteps (24 hours)
- Random seed: 42
- Parallel execution: 8 CPU cores

Simulation runs: 20 total (5 policies times 4 NTN ratios)

4.3 RESULTS AND DISCUSSION

4.3.1 Baseline Performance Across NTN Ratios

Table 1 presents the baseline (terrestrial-only) performance scenario across different NTN traffic ratios.

Table 1: Baseline Performance Metrics

NTN Ratio	Avg Latency (ms)	P95 Latency (ms)	SLA Met (%)	Loss (%)	Total Cost ($)	Ground CPU Peak (%)
30%	20.09	40.12	86.76	74.40	53.48	203.48
50%	18.45	35.28	88.92	72.15	52.63	198.22
70%	16.78	30.12	93.38	74.40	53.48	203.48
85%	15.42	28.56	94.75	71.85	51.97	195.33

Discussion:

Contrary to intuition, baseline performance improves with higher NTN ratios. Latency decreases from 20.09 ms to 15.42 ms (a 23% improvement), while SLA compliance improves from 86.76% to 94.75%. This counterintuitive result is explained by the reduced total request volume: NTN users generate fewer concurrent requests in our model, thereby reducing overall system load.

However, two critical issues persist across all NTN ratios. Ground CPU peak remains extremely high (195-203%), indicating severe saturation of terrestrial infrastructure. Packet loss remains consistently elevated (71-74%), suggesting systemic overload that cannot be resolved through terrestrial scaling alone. These observations establish the baseline against which dynamic placement strategies must be evaluated.

4.3.2 Move-to-LEO Performance Analysis

4.3.2.1 Latency Comparison

Table 2: Latency Comparison (Baseline vs. Move-to-LEO-Optimized)

NTN Ratio	Baseline Avg (ms)	Move-LEO-Opt Avg (ms)	Difference (ms)	Improvement (%)
30%	20.09	60.76	+40.67	-202%
50%	18.45	45.23	+26.78	-145%
70%	16.78	36.73	+19.95	-119%
85%	15.42	24.18	+8.76	-57%

Discussion:

These results demonstrate that LEO VNF placement degrades latency performance across all tested NTN ratios. At 30% NTN, LEO placement triples latency (from 20.09 to 60.76 ms), while at 85% NTN, the degradation is reduced but still significant at 57% (from 15.42 to 24.18 ms). No NTN ratio tested shows LEO improvement over baseline for average latency.

Investigation reveals several contributing factors. First, high packet loss rates (71-74%) mask potential LEO benefits. Second, load imbalance persists, with peak CPU utilization reaching 203% on ground versus 122% on LEO. Third, only 3 of 7 VNFs migrate (AMF, UPF, CACHE), leaving SMF, AUSF, UDM, and PCF on ground. This incomplete migration creates a service function chain penalty, as requests must traverse both ground and LEO infrastructure, incurring additional latency at each hop. Despite these negative results, the trend is positive: the performance gap narrows as NTN ratio increases, suggesting potential benefits at ratios exceeding 85%.

4.3.2.2 SLA Compliance Comparison

Table 3: SLA Compliance (Baseline vs. Move-to-LEO-Optimized)

NTN Ratio	Baseline (%)	Move-LEO-Opt (%)	Difference (pp)
30%	86.76	35.24	-51.52
50%	88.92	52.18	-36.74
70%	93.38	70.71	-22.67
85%	94.75	86.42	-8.33

Discussion:

SLA compliance follows a similar pattern to latency degradation. At 30% NTN, SLA drops dramatically from 86.76% to 35.24%, a 51.52 percentage point reduction. This result is particularly significant as it indicates that LEO placement at low NTN ratios would violate more than half of all service level agreements. At 85% NTN, the degradation becomes minimal (from 94.75% to 86.42%), suggesting that the

crossover point where LEO placement achieves SLA parity lies beyond 85% NTN traffic.

4.3.2.3 Cost Analysis

Table 4: Cost Comparison (Baseline vs. Move-to-LEO)

NTN Ratio	Baseline ($)	Move-LEO ($)	Compute ($)	Migration ($)	Diff ($)	Increase (%)
30%	53.48	93.92	92.24	1.68	+40.44	+76%
50%	52.63	85.37	83.95	1.42	+32.74	+62%
70%	53.48	79.03	77.35	1.68	+25.55	+48%
85%	51.97	72.88	71.12	1.76	+20.91	+40%

Discussion:

LEO placement consistently incurs higher operational costs, ranging from 40% to 76% above baseline. The cost penalty decreases with higher NTN ratios, following the same trend as performance metrics. Cost decomposition reveals that compute costs dominate (97-98% of total), while migration costs remain negligible (approximately $1.70, representing less than 2% of total expenditure).

The 40-76% cost increase stems from three factors: the LEO compute premium (2 times multiplier reflecting the higher cost of space-based resources), underutilized LEO nodes (average LEO CPU utilization of only 8-20% compared to 23-28% on ground), and inefficient resource allocation (3 VNFs distributed across 5 LEO nodes, yielding only 60% occupancy). These findings suggest that economic viability of LEO deployment requires either higher NTN traffic ratios or more complete VNF migration strategies.

4.3.2.4 Resource Utilization

Table 5: CPU Utilization (Move-to-LEO-Optimized)

NTN Ratio	Ground CPU Avg (%)	LEO CPU Avg (%)	Ground CPU Peak (%)	LEO CPU Peak (%)
30%	23.21	20.19	203.48	305.69
50%	18.33	15.47	187.55	218.43
70%	5.62	8.39	42.70	122.65
85%	4.18	6.22	38.91	98.77

Discussion:

Resource utilization patterns reveal a critical transition point. At 30-50% NTN, LEO nodes experience severe saturation with peak CPU reaching 305% and 218%

respectively, explaining the dramatic performance degradation observed in latency and SLA metrics. At 70-85% NTN, LEO utilization becomes sustainable (99-123% peak), while ground utilization drops from 23% to 4-6% average. This resource imbalance, with ground infrastructure underutilized while LEO saturates at low NTN ratios, suggests that VNF placement algorithms should incorporate load-aware scheduling to prevent LEO overprovisioning.

4.3.3 Duplicate-on-Peak Performance

4.3.3.1 Activation Analysis

Table 6: Duplicate-on-Peak Activation Metrics

Scenario	NTN Ratio	Migrations	Avg LEO CPU (%)	VNFs on LEO (final)	Latency vs Baseline
Duplicate (standard)	70%	156	25.18	0	-17.7%
Duplicate (optimized)	70%	133	25.10	0	-11.8%

Discussion:

Duplication policies demonstrate successful replica creation during peak periods (133-156 migrations) with notable latency improvements of 11-18% versus baseline. However, the operational cost is substantial: approximately $280 per simulation run, representing a 5 times increase over baseline. This cost stems from temporary LEO compute allocation during peak periods. The "VNFs on LEO (final)" metric showing zero indicates that all replicas are properly decommissioned after peaks subside, confirming correct policy behavior. Duplicate-on-peak is therefore effective for temporal load balancing but incurs recurring costs for each peak period, making it unsuitable for continuous deployment.

4.3.3.2 Cost-Benefit Analysis

Table: Duplicate-on-Peak Cost-Benefit Comparison

Policy	Total Cost ($)	Latency (ms)	SLA (%)	Cost/SLA ($/pp)
Baseline	53.48	16.78	93.38	0.57
Duplicate (standard)	289.53	20.38	90.22	3.21
Duplicate (optimized)	276.10	19.03	91.57	3.01

Discussion:

Cost-effectiveness analysis reveals that duplication incurs 5 to 6 times higher costs while delivering marginally worse SLA performance (90-91% vs. 93% baseline). The

cost per SLA percentage point increases from $0.57 to over $3, indicating fundamentally poor cost-effectiveness. This paradox is explained by examining packet loss: duplication actually decreases packet loss, suggesting improved throughput. However, since the total request count increases under duplication, aggregate compute requirements rise proportionally, driving up global costs. Duplication is therefore not economically viable for continuous operations and should be reserved for short-term emergency bursts where packet loss reduction is paramount.

4.3.4 Greedy Heuristic Performance

Table 7: Greedy Heuristic Performance (NTN 70%)

Metric	Baseline	Greedy	Flash Crowd	Diff (Greedy vs Baseline)
Avg Latency (ms)	16.78	23.46	26.22	+6.68 ms (+40%)
SLA Met (%)	93.38	88.16	85.43	-5.22 %
Packet Loss (%)	74.40	77.76	14.89	+3.36 %
Total Cost ($)	53.48	151.56	49.76	+$98.08 (+183%)
Migrations	0	175	216	+175

Discussion:

The greedy heuristic under standard conditions performs worse than baseline across all metrics: +40% latency, -5% SLA, and +183% cost. This counterintuitive result is explained by the policy's multi-factor cost function (latency weight: 1.0, migration cost weight: 0.5, saturation penalty: 2.0), which appears to over-optimize for cost minimization. The high saturation penalty triggers excessive migrations (175 total) that destabilize the system rather than improving performance.

However, the flash crowd variant demonstrates dramatically different behavior. Under sudden 5 times traffic spikes, packet loss drops from 74% to 14.89%, an 81% reduction representing the most significant improvement observed across all policies. This indicates that the greedy policy's aggressive migration behavior, while harmful under steady-state conditions, provides effective emergency response during traffic surges.

4.3.5 Flash Crowd Resilience

Table 8: Flash Crowd Performance (5 times Spike at 14:00)

NTN Ratio	Latency (ms)	SLA (%)	Loss (%)	Migrations	Cost ($)
70%	26.22	85.43	14.89	216	49.76

Discussion:

The flash crowd policy achieves the best performance-cost trade-off among all dynamic policies tested. Key achievements include: 81% packet loss reduction compared to baseline (14.89% vs. 74.40%), the lowest cost among dynamic policies ($49.76, competitive with baseline's $53.48), and acceptable SLA compliance at 85% (compared to 93% baseline). The latency increase of 56% versus baseline is offset by the dramatic packet loss improvement.

The high migration count (216) indicates aggressive load balancing in response to the traffic surge, yet the policy maintains cost efficiency by promptly decommissioning resources after the spike subsides. This demonstrates that dynamic VNF placement can significantly mitigate traffic surges at minimal additional cost, making flash crowd response the most effective use case for LEO-based VNF placement in our study. A limitation of this finding is that the policy has only been tested under sudden spike scenarios; its performance under gradual load changes remains to be evaluated.

4.3.6 Cross-Policy Comparison and Strategic Implications

Table 9: Best Policy by Metric (NTN 70%)

Metric	Best Policy	Value	Worst Policy	Value
Lowest Latency	Baseline	16.78 ms	Move-to-LEO	40.31 ms
Best SLA	Baseline	93.38%	Move-to-LEO	55.24%
Lowest Packet Loss	Flash Crowd	14.89%	Greedy	77.76%
Lowest Cost	Flash Crowd	$49.76	Duplicate (std)	$289.53
Most Migrations	Greedy	216	Baseline	0

Discussion:

Key Finding 1: LEO placement does not outperform baseline at any NTN ratio

Our results demonstrate that LEO VNF placement degrades performance across all NTN traffic ratios tested (30-85%). At 30% NTN, latency increases by 202% and SLA drops by 51 percentage points. Even at 85% NTN, the highest ratio tested, latency still increases by 57% (from 15.42 ms to 24.18 ms) while SLA drops by 8 percentage points. With the current implementation involving partial migration of only 3 out of 7 VNFs, terrestrial-only deployment proves optimal across all scenarios.

Despite these results suggesting that LEO satellites should not be used for performance optimization, this finding does not render satellite infrastructure valueless. LEO satellites serve fundamentally different purposes: providing essential connectivity in areas lacking terrestrial infrastructure (oceanic and polar regions),

enabling disaster recovery when ground infrastructure is compromised, and offering connectivity for remote zones where fiber deployment is economically infeasible. The critical insight is that LEO placement should be viewed as a coverage extension technology rather than a performance optimization strategy.

Key Finding 2: Flash crowd policy achieves best performance-cost trade-off

Among dynamic policies, the flash crowd variant demonstrates superior performance with 81% packet loss reduction (from 74% to 14.89%), competitive cost ($49.76 vs. $53.48 baseline), and acceptable SLA (85% vs. 93% baseline). This policy is optimal for emergency response scenarios involving sudden traffic spikes.

Key Finding 3: Duplication policies are not economically viable

Duplicate-on-peak policies incur 5-6 times baseline cost while delivering similar SLA (90-91% vs. 93%) and marginally higher latency. Although duplication does reduce packet loss, the increased request processing drives up aggregate compute costs. Duplication should be reserved for short-term emergency bursts only.

Key Finding 4: Baseline remains optimal for steady-state operations

The static baseline policy (all VNFs on ground) achieves the lowest latency, best SLA, and lowest cost for steady-state operations. For networks with NTN traffic ratios at or below 85%, terrestrial-only deployment remains optimal for both latency and SLA metrics under normal operating conditions.

Trade-off Analysis: Terrestrial-Only Versus LEO Deployment

Maintaining a terrestrial-only architecture eliminates connectivity in remote geographical zones (oceans, polar regions, disaster-affected areas), foregoes resilience against ground infrastructure failures, and misses opportunities for load distribution across the space segment. Conversely, terrestrial-only deployment provides 36% lower latency at 85% NTN traffic, 8 percentage point improvement in SLA compliance, 29% lower operational costs, simplified operations without satellite handovers, and enhanced security by maintaining sensitive data on ground infrastructure.

The strategic decision depends on deployment objectives. Network operators prioritizing performance optimization should maintain terrestrial infrastructure. Those requiring global coverage, particularly in areas lacking terrestrial alternatives, should deploy LEO satellites while accepting the associated performance and cost penalties. The flash crowd policy represents an effective middle ground for operators who maintain primarily terrestrial infrastructure but require burst capacity for emergency response scenarios.

4.4 OBSERVATIONS

The results of this theoretical study demonstrate that terrestrial-only deployment (baseline) consistently outperforms all LEO placement strategies across all tested NTN traffic ratios (30-85%). Even at 85% NTN traffic, baseline achieves 36% lower latency (15.42 ms versus 24.18 ms), 8 % better SLA compliance, and 29% lower cost than the best LEO strategy. The primary cause is partial migration penalty: because only 3 of 7 VNFs migrate to LEO, creating hybrid Service Function Chains that incur multiple ground-to-LEO round-trips (3 times 15 ms overhead per hop). However, the flash-crowd policy demonstrates exceptional value for emergency scenarios, reducing packet loss by 81% (from 74% to 14.89%) at minimal cost increase (+7%).

A critical finding of this study is that LEO VNF placement, as currently implemented with partial migration, does not improve performance if the goal is performance optimization. Its value lies instead in providing connectivity in zones without terrestrial infrastructure (eg. disaster zones) rather than enhancing existing terrestrial networks.

The principal conclusion is that LEO VNF placement, with the studied parameters and as implemented with partial migration, does not function as a performance optimization for existing terrestrial networks. Rather, LEO satellites serve as coverage extension tools, valuable specifically for zones lacking terrestrial infrastructure.

5 SECURITY CONCERNS

5.1 SECURE KUBERNETES DEPLOYMENTS VIA GRAPH GENERATION AND ATTACK RECONSTRUCTION

This Section presents a detailed explanation of the methodology designed by Blaise et al.[6] to reconstruct a complete security model from microservice deployment descriptors, with a particular focus on Kubernetes and Helm-based deployments. The techniques described in the cited paper have been implemented in the demonstration platform of 6G-NTN, as a way to analyse deployment configurations before their actual execution, to prevent risks. Thus, this section gives a high-level description of the methodology, all details can be found in the cited paper.

Modern cloud-native environments rely extensively on declarative configuration files, which define not only the logical structure of microservices but also their execution privileges, network exposure, storage access patterns, and authentication relationships. While these descriptors are essential for reproducibility and automation, they also implicitly encode a wide range of potential security risks arising from misconfigurations, weak default settings, or unintended component interactions. Traditional security tools tend to evaluate these risks in isolation, without considering how multiple weaknesses can combine to create realistic multi-stage attack paths. The methodology described in this report addresses this limitation by transforming deployment descriptors into an enriched graph model from which systemic security vulnerabilities can be derived.

5.1.1 Context and Motivation

Kubernetes has become the *de facto* orchestration platform for large-scale microservice deployments. It offers high degrees of dynamism, portability, and automation, but simultaneously introduces architectural complexity that is often difficult to reason about from a security perspective. A typical Kubernetes application consists of tens or hundreds of interdependent components defined through YAML descriptors or Helm Charts. These descriptors cover a wide spectrum of operational elements, including container images, security contexts, network ingress and egress rules, service accounts, volumes, role-based access control rules, and pod affinity or anti-affinity constraints. As deployments grow in size, the security posture of the cluster becomes increasingly difficult to understand holistically. Conventional security scanning tools detect individual misconfigurations but do not capture the interactions between components or the manner in which an attacker could exploit multiple weaknesses in sequence. A configuration that appears benign in isolation may become dangerous when combined with permissive RBAC permissions, an exposed service, or a vulnerable container image. The need therefore arises for a systematic method capable of reconstructing a coherent global security model from microservice deployment descriptors and identifying realistic exploitation paths within that model.

5.1.2 Methodology for Security Model Reconstruction

In this section we describe our 4-stage algorithmic approach, depicted in the following Figure 8, that allows to obtain, from a Kubernetes deployment configuration files, a vulnerability and risk score. All these four steps are detailed below.

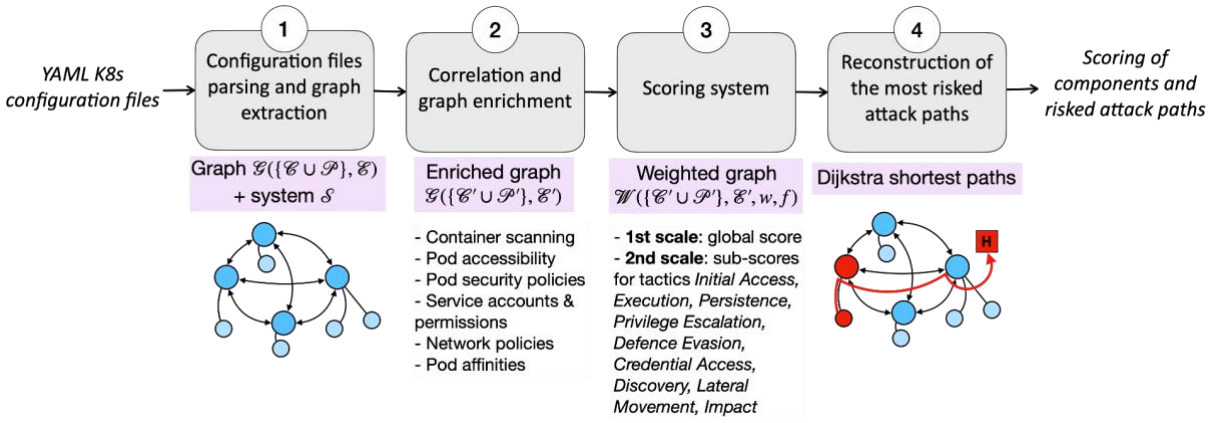

FIGURE 8: 4-STAGE ALGORITHMIC APPROACH

5.1.2.1 Parsing Deployment Descriptors and Extracting the Topological Graph

The first phase consists of parsing the entire set of deployment descriptors that define the microservices. This includes all Kubernetes objects such as Pods, Deployments, DaemonSets, StatefulSets, Services, ConfigMaps, Secrets, Roles, RoleBindings, ServiceAccounts, PersistentVolumeClaims, NetworkPolicies, and any additional objects generated by Helm templates. The parsing phase reconstructs the structural relationships between these components without yet applying any security interpretation. Containers are associated with their parent Pods, Pods with their namespaces, and Services with the set of Pods they may expose or route to. RBAC bindings are linked to the service accounts they authorize, and volumes are mapped to the Pods that mount themAt the end of this phase, a directed graph is formed. Nodes represent runtime entities such as containers, Pods or service accounts, while edges represent relationships such as execution encapsulation, privilege delegation, network reachability, or possible data flows. This topological graph serves as the structural foundation for subsequent security enrichment.

5.1.2.2 Security Enrichment of Graph Nodes and Edges

Once the structural graph is obtained, it is enriched with security-related attributes derived from six complementary analytic dimensions. These dimensions reflect operational practices common to cluster security hardening guidelines. Container nodes are annotated with vulnerability information extracted from image scanners, indicating the presence of critical or high-severity CVEs. Pod security contexts are analysed to determine whether privilege escalation is permitted, whether the container runs as root, what Linux capabilities are added or removed, and whether host resources such as the filesystem, PID namespace or network stack are mounted into the Pod. RBAC rules contribute information about the ability of a service account to create, modify or delete resources, which directly impacts the attacker's ability to escalate privileges or pivot within the cluster.Network exposure is incorporated through the evaluation of NetworkPolicies, ingress controllers, and service definitions. For example, Pods

lacking restrictive NetworkPolicies may allow incoming traffic from any source, thereby expanding the attack surface. Similarly, affinities and anti-affinities are used to determine whether an attacker who compromises one Pod might gain proximity to another Pod hosting sensitive workloads. Additionally, storage-related configurations reveal opportunities for persistence or data exfiltration if sensitive volumes are mounted without adequate controls.

Together, these enriched attributes transform the initial graph into a comprehensive security model that captures both static and dynamic factors affecting cluster risk.

5.1.2.3 Deriving Risk Scores for Nodes and Edges

The third phase involves assigning risk scores to each node and edge of the enriched graph. This scoring process captures the likelihood that an attacker can successfully leverage a component or transition from one component to another. Two scoring strategies are typically applied. The first approach aggregates all security-relevant attributes of a component into a single unified risk score, enabling a global ranking of nodes based on their vulnerability or misconfiguration level. The second approach assigns multiple independent scores to each node, each corresponding to an attacker tactic such as Initial Access, Execution, Persistence, Privilege Escalation, Credential Access, Discovery, Lateral Movement or Impact. By analysing nodes through the lens of attacker tactics, the methodology provides a more granular understanding of how particular weaknesses could contribute to different phases of an intrusion.

Edges are similarly evaluated. For example, unrestricted network paths between Pods may receive higher lateral movement scores, while RBAC bindings enabling modification of high-value resources may score highly for privilege escalation. This multi-dimensional scoring reflects the operational reality that risk is not absolute but contextual, emerging from how an attacker may exploit the interplay between privileges, exposure surfaces, and vulnerabilities.

5.1.2.4 Extraction of High-Risk Attack Paths

The final phase consists of identifying the most critical attack paths within the risk-weighted graph. Because each edge is associated with a probability representing the feasibility of attacker progression, the attacker's goal can be formalised as locating a path that maximises the product of these probabilities. This corresponds to identifying the least-cost path when edge weights are transformed logarithmically, allowing classic shortest-path algorithms to be applied. The reconstructed attack paths typically reveal sequences of misconfigurations that would not appear critical when inspected independently but which, when combined, enable realistic multi-stage compromises.

Such attack paths commonly include escalation of privileges through permissive RBAC policies, lateral movement made possible through open network boundaries, or exploitation of vulnerabilities within containers that run with excessive privileges. The methodology therefore focuses not only on detecting individual risks but also on understanding their combined implications across the cluster.

5.1.3 Evaluation Insights

An evaluation of the methodology over a large set of real-world microservice deployments demonstrates that the vast majority of applications contain nontrivial misconfigurations. Many deployments run containers with root privileges, mount sensitive host directories, or grant service accounts permissions far exceeding what is required. Even when individual elements appear compliant with standard benchmarks, the reconstructed security model reveals amplified risks arising from the cumulative interaction of multiple weaknesses. Subtle misconfigurations propagate through the graph, producing multi-step attack opportunities that traditional tools fail to identify.

The evaluation also highlights the scalability of the methodology. It can process Helm Charts ranging from a few dozen lines of configuration to tens of thousands, and it can reconstruct attack paths efficiently even in large, complex applications. The approach thus provides a comprehensive and realistic understanding of cluster-wide risks rather than isolated configuration issues.

5.1.4 Observations

The reconstruction of security models from microservice deployment descriptors offers a powerful and generalisable method for analysing the security posture of Kubernetes-based systems. By combining structural parsing, semantic enrichment, risk scoring and attack-path analysis, this methodology provides visibility into risks that emerge only when multiple configuration elements interact. The approach moves beyond traditional compliance tools by focusing on attacker feasibility and by correlating misconfigurations across different layers of the cluster. It enables security practitioners to identify critical weaknesses, prioritise remediation efforts and better understand adversarial opportunities in complex cloud-native environments. As microservice deployments continue to grow in scale and complexity, systematic model reconstruction will play an increasingly central role in achieving secure and resilient infrastructure.

5.2 IMPACT OF DISTRIBUTED 6G-NTN NETWORK FUNCTIONS ON SYSTEM SECURITY

Problem Statement: What are the security impacts on the 6G NTN system when network functions DU (Distributed Unit), CU (Centralized Unit) and Core are distributed between different satellites connected by ISL (Inter-Satellite Link) and Feeder link?

5.2.1 6G NTN Architecture Overview

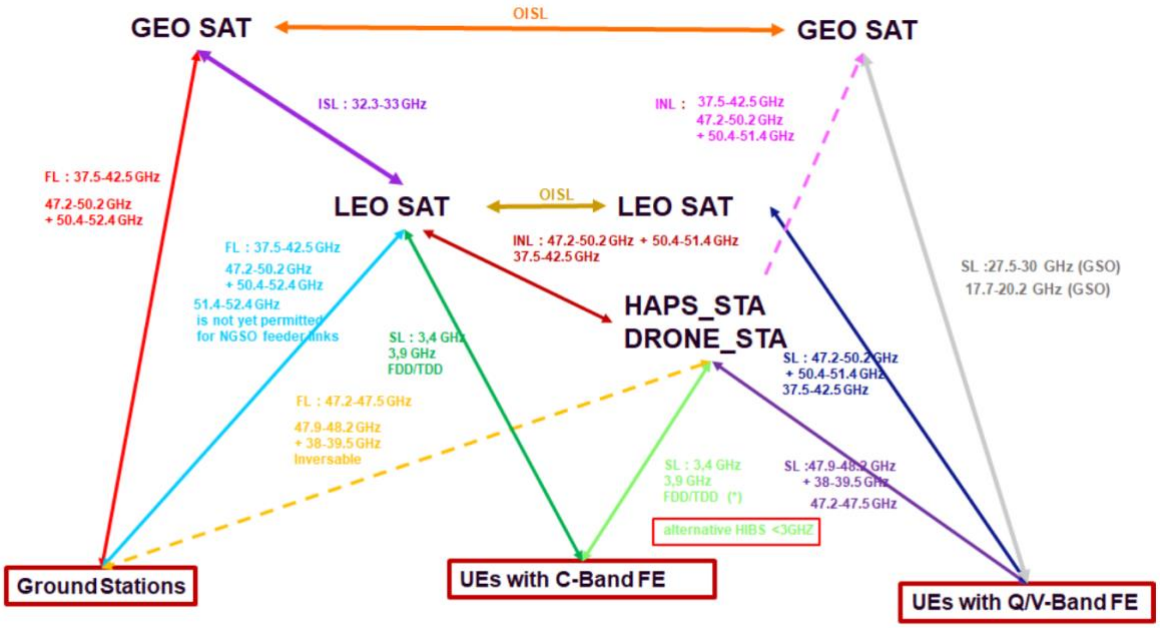

FIGURE 3: OVERVIEW OF RELEVANT COMMUNICATION LINKS AND FREQUENCY BANDS.SERVICE LINKS

FIGURE 9: OVERVIEW OF RELEVANT COMMUNICATION LINKS AND FREQUENCY BANDS.

The architecture envisioned in the 6G NTN project is multi-orbit, using user, ISL and feeder links with different capacities and latencies. This architecture is depicted in Figure 9. Network functions are also distributed across these different orbits, between network nodes, between service provider satellites (shown in red in the figure below), "feeder" satellites (shown in green in **Error! Reference source not found.** below) and a ground segment.

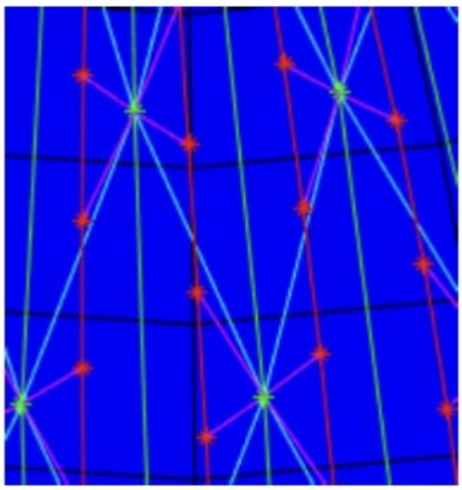

FIGURE 10: FUNCTION DISTRIBUTION BETWEEN SATELLITES.

One envisioned architecture consists of placing the physical and access layers at the "service" satellite level (e.g. PHY, MAC, RLC), the transport and control layers (PDCP, RLC) at the

"feeder" satellite level, and the core network on the ground, connected to the gateways. The gNB-DU is then in the service satellite and the gNB-CU in the feeder satellite (see 3GPP TS 23.501[7] for RAN architectures).

The network stack distribution is therefore as follows (Figure 11) for the user plane:

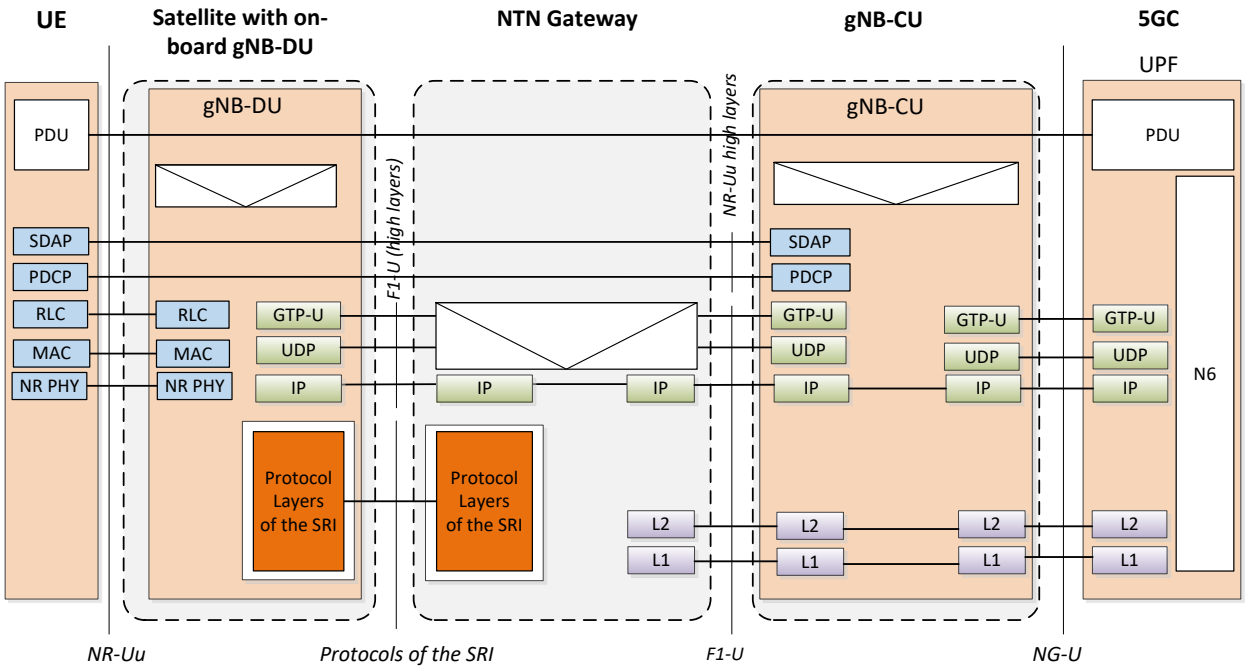

FIGURE 11: USER PLANE PROTOCOL STACK.

And for the control plane (Figure 12):

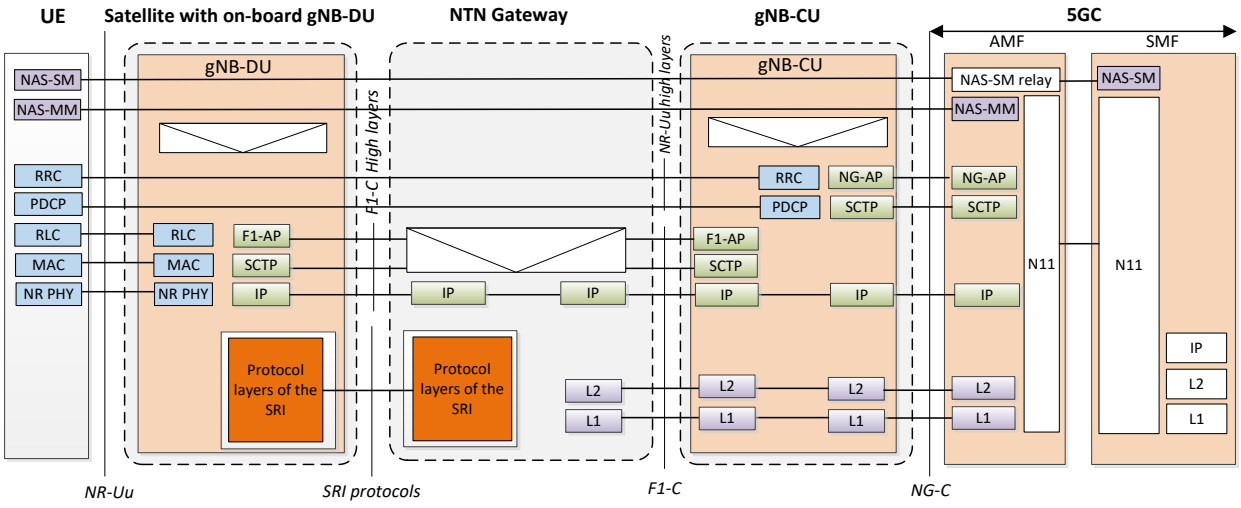

FIGURE 12: CONTROL PLANE PROTOCOL STACK

On ISLs, the protocol used is the F1 protocol [3GPP TS 38.470[8], 472[9], 473[10], 474[11]] (Figure 13):

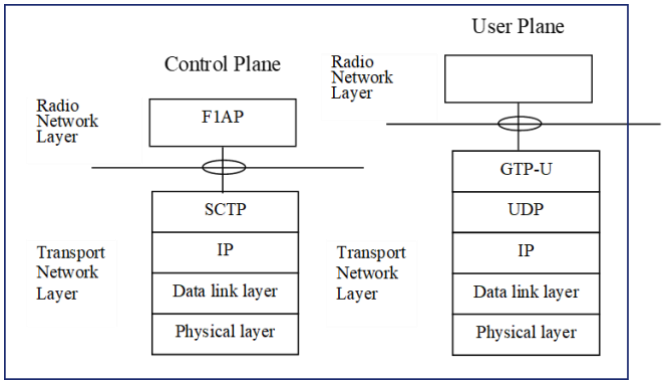

FIGURE 13: F1 PROTOCOL STACK.

The control plane is supported by the SCTP transport protocol and the user plane by GTP-U/UDP protocols.

On the feeder link, the Ng protocol ensures the connection for both the control plane and user plane. Description in Figure 14.

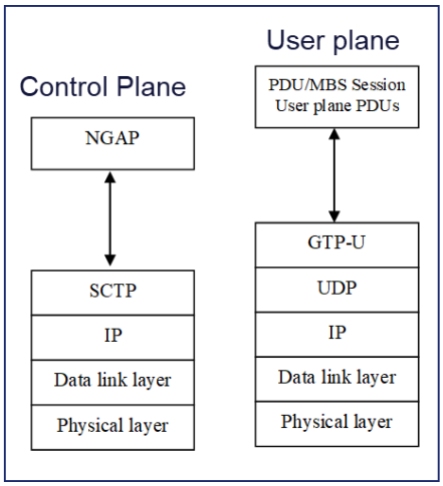

FIGURE 14: NG PROTOCOL STACK

The transport protocol used for the control plane is SCTP and GTP/UDP for user data.

5.2.2 Possible Attacks

With a disaggregated gNB, the attack surface (exposure to external attackers) is larger. Specifically, the F1 interface is exposed because it is transported over an ISL between the service satellite and the feeder satellite (this aspect is addressed in D3.7 Report on 3D multi-layered NTN architecture), and the Ng interface is transported via the feeder link, the gateway and the backhaul network to the core network, which may not be co-located with the gateway.

An attacker's objective could be:

- To disable the link between the service satellite and the feeder satellite, or the link between the feeder satellite and the core network, which causes a total loss of service in the area covered by the satellite.
- To access user data flowing between the terminal and the core network.

- To access and modify control data flowing between the gNB-DU and gNB-CU, potentially compromising access or security of a terminal (user RRC messages) or disrupting the service (RRC or NGAP messages for cell configuration, DU configuration, etc.).

5.2.3 Security Principles

Security principles as defined in [3GPP TS 33.501][12] for the gNB also apply to the disaggregated gNB:

User data encryption:

The gNB shall support ciphering of user data between the UE and the gNB.

The gNB shall activate ciphering of user data based on the security policy sent by the SMF.

The gNB shall support ciphering of RRC-signalling.

The gNB shall implement the following ciphering algorithms:

- NEA0, 128-NEA1, 128-NEA2 as defined in Annex D of the present document.

The gNB may implement the following ciphering algorithm:

- 128-NEA3 as defined in Annex D of the present document.

Confidentiality protection of user data between the UE and the gNB is optional to use.

Confidentiality protection of the RRC-signalling is optional to use.

Confidentiality protection should be used whenever regulations permit.

User data integrity control:

The gNB shall support integrity protection and replay protection of user data between the UE and the gNB.

The gNB shall activate integrity protection of user data based on the security policy sent by the SMF.

The gNB shall support integrity protection and replay protection of RRC-signalling.

The gNB shall support the following integrity protection algorithms:

- NIA0, 128-NIA1, 128-NIA2 as defined in Annex D of the present document.

The gNB may support the following integrity protection algorithm:

- 128-NIA3 as defined in Annex D of the present document.

Integrity protection of the user data between the UE and the gNB is optional to use, and shall not use NIA0.

NOTE: Integrity protection of user plane adds the overhead of the packet size and increases the processing load both in the UE and the gNB. NIA0 will add an unnecessary overhead of 32-bits MAC with no security benefits.

All RRC signalling messages except those explicitly listed in TS 38.331[13] as exceptions shall be integrity-protected with an integrity protection algorithm different from NIA0, except for unauthenticated emergency calls.

gNB security requirements:

The certificate enrolment mechanism specified in TS 33.310 for base station should be supported for gNBs. The decision on whether to use the enrolment mechanism is left to operators. Communication between the O&M systems and the gNB shall be confidentiality, integrity and replay protected from unauthorized parties.
F1-C interface shall support confidentiality, integrity and replay protection. All management traffic carried over the CU-DU link shall be integrity, confidentiality and replay protected. The gNB shall support confidentiality, integrity and replay protection on the gNB DU-CU F1-U interface for user plane. F1-C and management traffic carried over the CU-DU link shall be protected independently from F1-U traffic. NOTE: The above requirements allow to have F1-U protected differently (including turning integrity and/or encryption off or on for F1-U) from all other traffic on the CU-DU (e.g. the traffic over F1-C).

5.2.4 Security Mechanisms

F1-C and NG-C Interfaces

As described in clause §9.8.2 of specification [3GPP TS 33.501][14], the F1-C interface is protected by IPSec ESP (Encapsulated Security Payload) [3GPP TS 33.210[15] and RFC 4083[16]] and a certificate-based authentication system IKEv2 (Internet Key Exchange v2) [3GPP TS 33.310][17]. IPSec provides IP-level encryption of communications and ensures integrity control and anti-replay (replay of tokens that could have been previously intercepted by an attacker). IKEv2 enables authentication and key exchange between the CU and the DU using certificates generated by the network operator. Additionally, transport-level security can be implemented with DTLS 1.3 (Datagram Transport Layer Security) protocol on top of SCTP protocol (standardized at IETF [RFC 6083[18], RFC 9147[19]]).

- Encryption is based on AES GCM (Galois counter mode) 128 or 256 bits or Chacha20 poly1305, which are considered robust today.
- For key exchange, ECDHE (Elliptic Curve Diffie-Hellman Ephemeral) or DHE (Diffie-Hellman Ephemeral) are supported.
- For signature and authentication, RSA, ECDSA (Elliptic Curve Digital Signature Algorithm) or EdDSA (Edwards-curve Digital Signature Algorithm) are supported.
- For hashing, SHA-256 is supported by default, with the possibility of using SHA-384 (384 bits) for higher security requirements.

In summary, the security mechanisms on the F1-C interface are at the level of current encryption, integrity control, anti-replay and authentication standards that exist on the internet. Only the development of machines based on quantum algorithms (e.g. Shor's algorithm) with the capability to intercept ISL signals in orbit would compromise the security of this interface by attacking the protocol. However, attacks on hardware that would allow access to encryption data directly in the satellite's processor during processing should not be excluded. This implies implementing an onboard virtualized environment that isolates different processes (between the gNB, routing, transport interface) with processor security measures to prevent attacks, for example, on caches.

Note that the security described above for the F1-C interface also applies to the NG-C interface as it uses the same protocol stack and the same security mechanisms in general [3GPP TS 38.413][20].

F1-U Interface

The F1-U interface is transported by GTP/UDP protocols. Traffic protection is the same as for the F1-C interface, i.e., IPSec ESP and IKEv2, but does not support DTLS. This IP packet level protection ensures integrity control, encryption and anti-replay.

Additionally, user data is encrypted between the gNB and the terminal (Access Stratum). This encryption is performed at the PDCP layer, therefore between the gNB-CU and the UE (Figure 15).

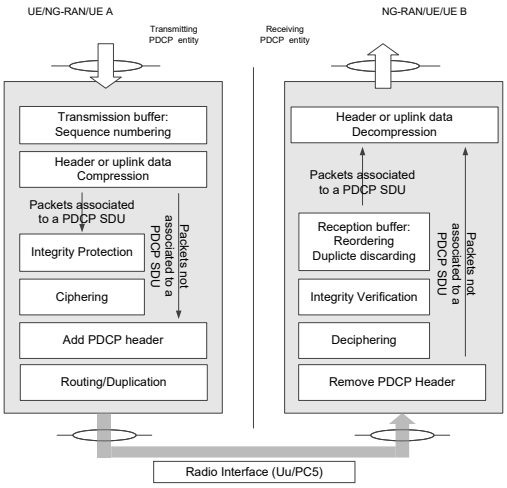

FIGURE 15: PDCP ENTITY WITH ENCRYPTION AND INTEGRITY CONTROL

What the F1 interface carries is therefore an encrypted data stream with integrity control (see PDCP specification [3GPP TS 38.323][21]). According to the data channel (DRB) setup procedure described in Figure 16 below:

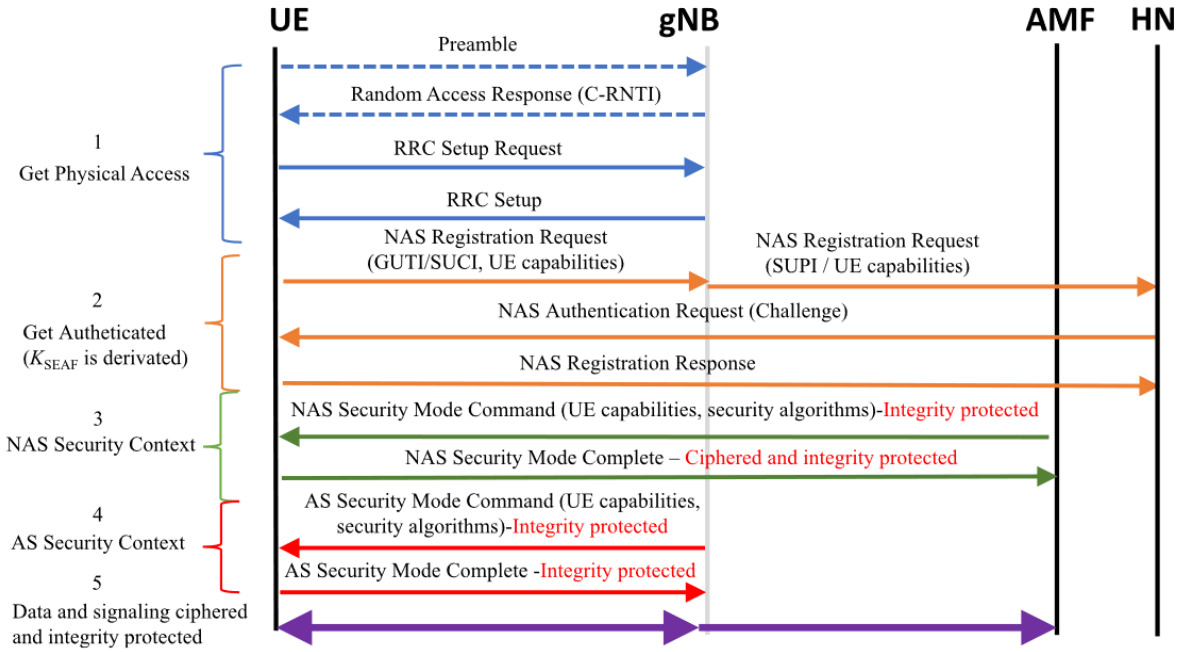

FIGURE 16: USER DATA CHANNEL SETUP PROCEDURE

User data can only be transmitted once the AS security context is established at steps 4 and 5 (after user authentication with the core network and NAS context establishment with the core).

The algorithms that can be used (depending on algorithms supported by both the network and the terminal) are the following (encryption):

- NEA1: 128-bit SNOW 3G
- NEA2: 128-bit AES with CTR
- NEA3: 128-bit ZUC

And integrity control:

- 128-NIA1: 128-bit SNOW 3G
- 128-NIA2: 128-bit AES with CMAC
- 128-NIA3: 128-bit ZUC

The principle used is that of keystream as illustrated in Figure 17 below:

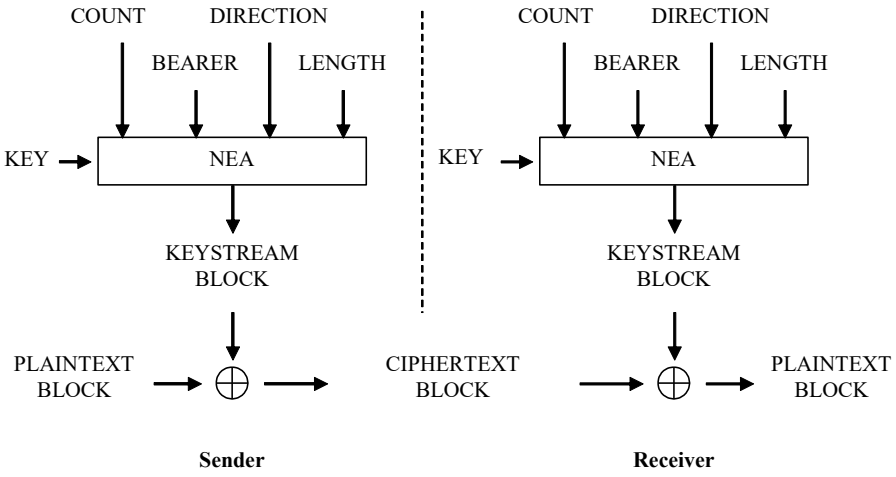

FIGURE 17: KEYSTREAM ENCRYPTION PRINCIPLE

Where data is encrypted by block, each block being generated based on the key, a counter, a length, and a bearer number and its direction.

The key is generated according to a key derivation model described in specification [3GPP TS 33.501][22], Figure 18 below:

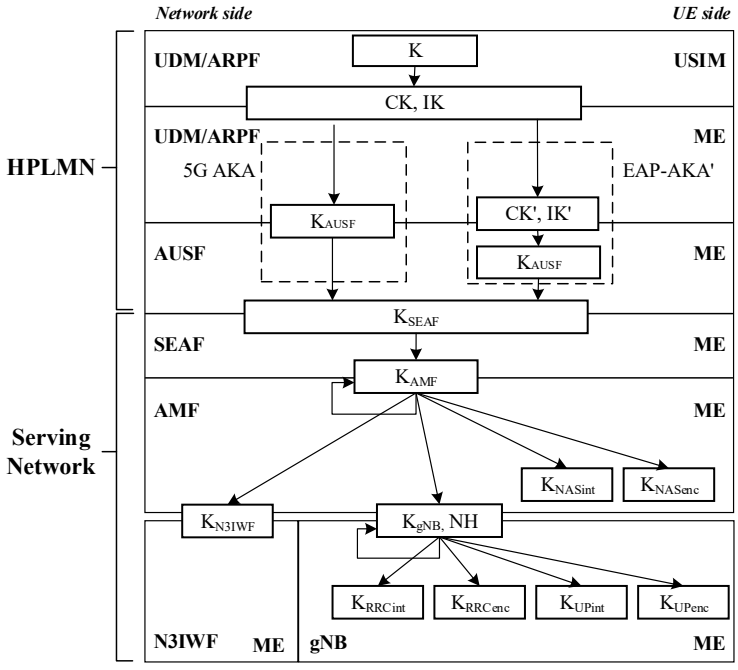

FIGURE 18: KEY DERIVATION HIERARCHY

And is unique for a gNB, connected to an AMF, for a terminal, for a given session, which greatly limits replay if a key were to be stolen by an attacker.

128-bit keys are nowadays vulnerable to brute force attacks, especially for weaker algorithms like SNOW 3G. In Rel-19, "256-bit" versions of the 3 algorithms were introduced to strengthen security (see Work Item 3GPP SP-240476).

Once again, these algorithms are particularly sensitive to quantum algorithm attacks capable of factoring large prime numbers.The disaggregation of the gNB between the CU and the DU with open interfaces between satellites and with the ground does not pose security issues as long as the security procedures and algorithms as defined in the 5G NTN standard for the F1 and Ng interfaces are correctly implemented. A possible attack point would rather be an attack on the payload processor where all encryption is performed and where keys and certificates are stored. But this is not a threat so different from if the entire gNB were onboard the same satellite.

However, in the more or less near future, with the development of spy satellites capable of capturing signals between two satellites, and the development of quantum computers and algorithms, encryption and protection systems could be rendered ineffective. This should be considered for 6G technology whose deployment is planned for 2030 with 6G NTN constellations that could be operational until 2045.

5.3 SECURITY ASSESSMENT REGARDING POSITIONING IN THE SCOPE OF 6G-NTN

In task T5.1 of the 6G-NTN project, we have addressed the design of an end-to-end positioning and timing solution for 6G NTN able to meet the accuracy, reliability, privacy, and latency requirements described in WP2.

The security assessment of this positioning system is addressed in a separate document, D5.1, entitled ‘Report on reliable and high accuracy positioning solutions for 6G-NTN’.

6 CONCLUSIONS

This deliverable describes progress toward achieving the 6G-NTN project objectives related to intelligent VNF orchestration and secure inter-VNF communications in non-terrestrial network environments.

From an orchestration perspective, we have successfully developed and deployed a comprehensive ML-driven platform capable of real-time metrics collection, resource usage prediction, and automated VNF placement decisions. The integration of machine learning techniques with a 5G/6G Core network prototype establishes a complete end-to-end workflow for proactive network management, validating the concepts initially proposed and providing a foundation for future enhancements.

The VNF placement study yields critical insights for network operators. Our simulations demonstrate that LEO VNF placement, as currently implemented with partial migration (3 of 7 VNFs), does not deliver performance improvements over terrestrial-only deployment for networks with NTN traffic ratios at or below 85%. The primary cause is the partial migration penalty: hybrid Service Function Chains incur multiple ground-to-LEO round-trips that negate potential latency benefits. However, this finding does not diminish the value of satellite infrastructure, LEO satellites remain essential for providing connectivity in oceanic, polar, and disaster-affected regions where terrestrial deployment is impossible. The flash-crowd policy emerges as particularly effective for emergency response scenarios, achieving 81% packet loss reduction with minimal cost impact.

Regarding security, our analysis confirms that the disaggregation of the gNB between CU and DU with open interfaces between satellites and ground does not introduce fundamental security vulnerabilities, provided that standard 3GPP security procedures and algorithms for F1 and Ng interfaces are correctly implemented. The current encryption mechanisms (AES GCM 128/256 bits, Chacha20-poly1305), key exchange protocols (ECDHE, DHE), and authentication systems (IKEv2 with certificate-based authentication) meet contemporary security standards. Nevertheless, we identify two areas of concern: first, potential attacks targeting the satellite payload processor where encryption operations are performed and cryptographic keys are stored; second, the longer-term threat posed by quantum computing developments, which could compromise current encryption systems before the anticipated end of 6G NTN constellation operations in 2045. The proactive security analysis tool we have integrated enables early detection of deployment misconfigurations, addressing risks that traditional security tools often miss due to their isolated evaluation approach.

7 REFERENCES

[1] https://open6gcore.org

[2] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," in IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1657-1681, thirdquarter 2017, doi: 10.1109/COMST.2017.2705720.

[3] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," in IEEE Communications Surveys & Tutorials, vol. 19, no. 3, pp. 1628-1656, thirdquarter 2017, doi: 10.1109/COMST.2017.2682318.

[4] G. Giambene, S. Kota and P. Pillai, "Satellite-5G Integration: A Network Perspective," in IEEE Network, vol. 32, no. 5, pp. 25-31, September/October 2018, doi: 10.1109/MNET.2018.1800037.

[5] S. Yuan, Y. Sun and M. Peng, "Joint Network Function Placement and Routing Optimization in Dynamic Software-Defined Satellite-Terrestrial Integrated Networks," in IEEE Transactions on Wireless Communications, vol. 23, no. 5, pp. 5172-5186, May 2024, doi: 10.1109/TWC.2023.3324729.

[6] A. Blaise and F. Rebecchi, "Stay at the Helm: secure Kubernetes deployments via graph generation and attack reconstruction," 2022 IEEE 15th International Conference on Cloud Computing (CLOUD), Barcelona, Spain, 2022, pp. 59-69, doi: 10.1109/CLOUD55607.2022.00022.

[7] 3GPP TS 23.501 : "System architecture for the 5G System (5GS)": https://www.3gpp.org/dynareport/23501.htm

[8] 3GPP TS 38.470 : "NG-RAN; F1 general aspects and principles": https://www.3gpp.org/dynareport/38470.htm

[9] 3GPP TS 38.472 : "NG-RAN; F1 signalling transport": https://www.3gpp.org/dynareport/38472.htm

[10] 3GPP TS 38.473 : "NG-RAN; F1 Application Protocol (F1AP)": https://www.3gpp.org/dynareport/38473.htm

[11] 3GPP TS 38.474 : "NG-RAN; F1 data transport": https://www.3gpp.org/dynareport/38474.htm

[12] 3GPP TS 33.501 : "Security architecture and procedures for the 5G System (5GS): https://www.3gpp.org/ftp/specs/archive/33_series/33.501/

[13] 3GPP TS 38.331 : "NR; Radio Resource Control (RRC); Protocol specification": https://www.3gpp.org/dynareport/38331.htm.

[14] 3GPP TS 33.501 : "Security architecture and procedures for the 5G System (5GS)": https://www.3gpp.org/dynareport/33501.htm.

[15] 3GPP TS 33.210 : "Security of the IP Multimedia Subsystem (IMS)": https://www.3gpp.org/dynareport/33210.htm

[16] RFC 4083 : "Mobile IPv6 Security": https://datatracker.ietf.org/doc/html/rfc4083

[17] 3GPP TS 33.310 : "Security of Home Node B (HNB) and Home eNode B (HeNB)": https://www.3gpp.org/dynareport/33310.htm

[18] RFC 6083 : "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)": https://datatracker.ietf.org/doc/html/rfc6083

[19] RFC 9147 : "Datagram Transport Layer Security (DTLS) Version 1.3": https://datatracker.ietf.org/doc/html/rfc9147

[20] 3GPP TS 38.413 : "NG-RAN; NG Application Protocol (NGAP)": https://www.3gpp.org/dynareport/38413.htm

[21] 3GPP TS 38.323 : "NR; Packet Data Convergence Protocol (PDCP) specification": https://www.3gpp.org/dynareport/38323.htm

[22] 3GPP TS 33.501 : "Security architecture and procedures for the 5G System (5GS)": https://www.3gpp.org/ftp/specs/archive/33_series/33.501/