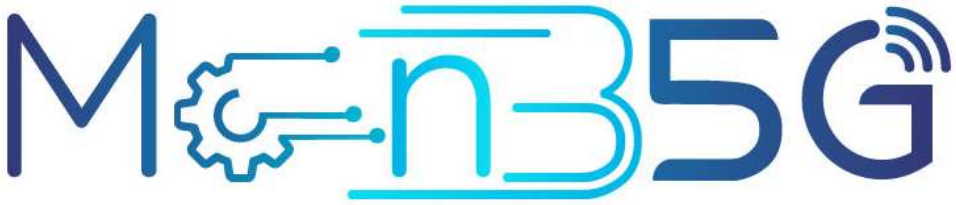
871780 — MonB5G — ICT-20-2019-2020



Deliverable D1.8
Final Publishable Activity Report

Document Summary Information

Grant Agreement No	871780	**Acronym**	MonB5G
Full Title	Distributed Management of Network Slices in beyond 5G		
Start Date	01/11/2019	**Duration**	42 months
Project URL	https://www.monb5g.eu/		
Deliverable	D1.8 – Final Publishable Activity Report		
Work Package	WP1		
Contractual due date	M42	**Actual submission date**	07/06/2023
Nature	Report	**Dissemination Level**	Public
Lead Beneficiary	CTTC		
Responsible Authors	Engin Zeydan (CTTC)		
Contributions from	Zhao Xu (NEC), Eric Gatel (BCOM), Sławomir Kukliński (ORA-PL), Vasiliki Vlahodimitropoulou (OTE), George Tsolis (CTXS), Luis Garrido (IQU), Engin Zeydan (CTTC), Selva Vía (CTTC), Adlen Ksentini (EUR), Anne Marie Bosneag (LMI), Ashima Chawla (LMI), Sihem cherrared (ORA-FR)		

Disclaimer

Copyright message

TABLE OF CONTENTS

List of Figures

List of Tables

List of Acronyms

Acronym	*Description*
3GPP	*Third Generation Partnership Project*
5GC	*5G Core*
5GS	*5G System*
ACT	*Actuator*
AD	*Anomaly Detection*
AE	Analytic Engine
AI	Artificial Intelligence
AMF	Access and Mobility Management Function
API	Application Programming Interface
BSS	Business System Support
CN	Core Network
DB	DataBase
DDOS	Distributed Denial-of-Service
DE	Decision Engine
DMO	Domain Manager and Orchestrator
DNN	Deep Neural Network
DNS	Domain Name System
DRL	Deep Reinforcement Learning
DSM	Domain Slice Manager
EPC	Enhanced Packet Core
EEM	Embedded Element Manager
eMBB	Enhanced Mobile Broadband
ES	Experimental Scenario
ETSI	European Telecommunications Standards Institute
FCAPS	Fault, Configuration, Accounting, Performance, Security
G5IAD	Graph-based Interpretable Anomaly Detection
GCN	Graph Convolutional Network

HA-DRL	Heuristically Assisted DRL
IDM	Infrastructure Domain Manager
IDMO	Inter-Domain Manager and Orchestrator
IDSM	Inter-Domain Slice Manager
InP	Infrastructure Provider
ISM	In-Slice Management
IoT	Internet of Things
ITU	International Telecommunication Union
K8	Kubernetes
KNN	k-nearest neighbours
KPI	Key Performance Indicator
LCM	Lifecycle Management
LDA	Linear Discriminant Analysis
LSTM	Long-Short Term Memory
LXC	Linux Containers
ML	Machine Learning
MANO	Management and Orchestration
MaaS	Management as a Service
ML	Machine Learning
MNO	Mobile Network Operator
mMTC	Massive Machine Type Communications
MS	Monitoring System
NF	Network Function
NFVI	Network Function Virtualization Infrastructure
NFVO	Network Function Virtualization Orchestrator
NSI	Network Slice Instance
NFVI	NFV Infrastructure
OSS	Operation System Support
PCA	Principal Component Analysis
PoC	Proof of Concept

QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
RAN	Radio Access Network
RBAC	Role-Based Access Control
RNN	Recurrent Neural Networks
RTSP	Real Time Streaming Protocol
SDN	Software Defined Networks
SFC	Service Function Chain
SON	Self-Organizing Network
SLA	Service Level Agreement
SFL	Slice Functional Layer
SML	Slice Management Layer
SM	Slice Manager
SO	Slice Orchestrator
TASAC	Time Aware Slice Admission Control
TD	Technology Domain
ToD	Time of the Day
TQFL	Trust deep Q-learning Federated Learning
TSDB	Time Series Database
UC	Use Case
UDM	Unified Data Management
UE	User Equipment
UV	Univariate
uRLLC	Ultra-Reliable Low-Latency Communication
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VNF	Virtual network Function
VR	Virtual Reality
XAI	interpretable Artificial Intelligence

ZSM	Zero-touch network and Service Management

Executive summary

The MonB5G project (https://www.monb5g.eu/) has been co-funded by the EU through the Horizon 2020 framework. **Twelve organizations** from **eight European countries** (Greece, Spain, France, Ireland, Poland, Germany, Cyprus and Finland) have participated, and the total duration of the project was **42 months**. MonB5G project, capitalised on artificial intelligence (AI) for smart, flexible and automated management of 5G and 6G network resources. The project used **AI-based mechanisms for zero-touch management and orchestration of massive-scale network slices**. With the application of AI algorithms, networks acquired **self-configuration, self-monitoring, self-healing and self-optimization capabilities,** without the need for human intervention. MonB5G project presented **hierarchical, fault-tolerant, AI-based, automated network management framework** that includes security and energy efficiency techniques for orchestrating many parallel network slices. The concept of **"network slicing"** referred to the division of a virtual network infrastructure into multiple customized and independent segments and the suitable allocation, per case of use, of network resources to efficiently serve different requirements. Each network slice, assisted by AI, ensured **the seamless operation of demanding applications** that require high capacity and low latency. In addition to the provision of reliable and high-quality services, the use of **AI enhanced infrastructure security and improved energy efficiency of networks**.

The MonB5G system divided the centralized management system into **many management subsystems**, through an optimal adaptive assignment of monitoring, analysis, and decision-making tasks across multiple domains to achieve the **concept of Zero touch Service management (ZSM).** AI-driven zero-touch closed-loop management was based on the three administrative elements of **monitoring system (MS), analytic engine (AE), and decision engine (DE)**, with feedback interfaces used to reconfigure MS, AE, and DE to meet energy efficiency, security and scalability objectives along with network automation and service management goals. MonB5G has chosen **two use Proof of Concepts (PoCs)** to be demonstrated on the partners' (CTTC and EURECOM) 5G testbeds, including **Zero-touch service management and orchestration** across technical and administrative domains, enabling network operators to ensure end-to-end cross-domain SLAs, and **AI-assisted policy-driven security monitoring and enforcement**.

The demonstration of PoC-1: zero-touch multi-domain service management scenario showcased the capabilities of the MonB5G system in managing and orchestrating services across multiple domains without human intervention. Through the implementation of zero-touch techniques, such as automated service management and orchestration, resource optimization, and efficient anomaly detection, PoC-1 demonstrated the potential of MonB5G to deliver end-to-end service management with stringent service level agreements (SLAs). In the **demonstration of PoC-2: AI-assisted security monitoring and enforcement**., the focus shifted to the security aspects of the MonB5G system and the problem of security threats in B5G networks. Specifically, it focused on AI-assisted policy-driven security monitoring and enforcement to detect and mitigate attacks, ensuring the integrity and reliability of the network. By leveraging AI-driven techniques, such as attack detection, mitigation, and enforcement of security policies, PoC-2 showcased the ability of MonB5G to effectively safeguard network slices against various security threats by deploying a secure and resilient security system. These two PoC demonstrations highlighted the importance of **incorporating intelligent, zero touch service management, energy efficient and security mechanisms** into the MonB5G

architecture to ensure the automation, resilience, sustainability, and protection of the network infrastructure.

Major achievements of the MonB5G project can be summarised as follows: The MonB5G project incorporated **AI-driven and decentralized management and orchestration architecture**, aiming to achieve ZSM for network slices. This involved the development of an AI-driven In-Slice Management (ISM) concept that reduced the number of external slice interfaces and separates slices' management plane. Additionally, a multi-domain orchestration framework provided a strong separation between different orchestration domains. **In terms of security, privacy, and resiliency**, the MonB5G project introduced an AI-based closed control loop framework to detect and mitigate attacks on network slices. It also explored the concept of Security orchestration and Security as a Service, emphasizing the importance of a trusted architecture for network slicing deployment. The project demonstrated the capabilities of the closed control loop in detecting and mitigating attacks through three specific use cases: in-slice mMTC DDoS attacks on AMF, aLTEr attacks involving traffic steering and VNF instantiation, and attack detection and mitigation on the Federated Learning (FL) training process. **Regarding technology enablers,** the MonB5G project defined novel end-to-end (E2E) slice Key Performance Indicators (KPIs) for monitoring the performance of network slices. It employs graph-based learning techniques for slice KPI prediction and explored the use of FL to reduce SLA violations in beyond 5G network slicing. AI-based intra and inter slice admission control mechanisms are also developed to optimize resource allocation and ensure efficient slice management. Additionally, MonB5G focuses on energy efficiency by developing decentralized cross-domain Energy Efficient DEs and implementing energy-saving techniques at the RAN and Edge. Furthermore, the MonB5G project contributed to the research community **by publishing 5G datasets collected from its testbeds**, providing valuable resources for future investigations and advancements in the field. MonB5G also contributed to relevant standard bodies (and groups therein) in **ITU-T, 3GPP and ETSI** and several dissemination and communication activities. The MonB5G consortium has published **29 journal and 37 conference papers** in prestigious international journals and conferences throughout the duration of the project (5 more conference papers were accepted). Moreover, the contributions in three white papers by the 5G-PPP WGs and in the “Towards Natively Intelligent Networks” chapter of the 6G architecture book titled "Towards Sustainable and Trustworthy 6G: Challenges, Enablers and Architectural Design" illustrate its impact on the transition from 5G and beyond to a well-developed 6G architecture. The MonB5G consortium participated in a variety of outreach events, including presentations, workshops, tutorials, and demonstrations, which are detailed in Deliverable D7.7.

In summary, the MonB5G project combined AI-driven network management, service orchestration, and security measures to achieve zero-touch service management, ensure security and privacy, and optimize network performance. By leveraging technology enablers and innovative approaches, MonB5G aimed to enhance the capabilities of network slicing and contribute to the development of future B5G networks. Overall, the MonB5G project's potential impacts can also extend beyond technical advancements and have the potential to **generate socio-economic benefits, drive innovation, and positively influence** the wider society by improving network efficiency, enhancing service quality, strengthening security, promoting sustainability, and contributing to economic growth for operators and service providers.

1 Introduction

The MonB5G project has been a 42-month research project that brought together 12 partners from 8 different European countries. MonB5G aimed to establish a decentralized and intelligent framework for zero-touch management and orchestration to support network slicing in 5G LTE and beyond. This deliverable provides an overview of the project's functional blocks, methodology, architecture, artificial intelligence (AI) -driven techniques employed in the MonB5G system. This deliverable also highlights the experimental scenarios and their outcomes, demonstrating the effectiveness of the MonB5G solutions in achieving zero-touch network and service management with end-to-end service level agreements (SLAs) and policy-driven security monitoring.

The MonB5G architecture is described, illustrating the decentralized and intelligent nature of the system. It showcases how the components interact and collaborate to achieve zero-touch management and orchestration of network slices in a scalable and efficient manner. The functional blocks of the MonB5G system are introduced, encompassing the Monitoring System (MS), Analytics Engine (AE), Decision Engine (DE), and Actuators (ACT). These blocks play crucial roles in collecting network data, analysing it using AI techniques, making informed decisions, and implementing actions for efficient network management and security. The methodology employed in the MonB5G project is outlined, providing an overview of the performance validation process and the key performance indicators (KPIs) used for evaluation. The methodology encompasses the comparison of baseline mechanisms, representative use cases, and the KPIs defined in each work packages separately.

Additionally, this deliverable delves into the AI-driven techniques employed in the MonB5G system, focusing on the monitoring system, analytic engine, decision engine, security techniques, and energy efficiency techniques. These AI-driven techniques enable advanced anomaly detection, resource allocation, slice admission control, security monitoring, and energy optimization within the MonB5G framework. The experimental scenarios, namely PoC-1 and PoC-2, are presented with corresponding evaluation results. PoC-1 results focuses on zero-touch network and service management with end-to-end SLAs, demonstrating the capabilities of MonB5G solutions in multi-domain service management, anomaly detection, elastic slice management, resource allocation, and slice admission control. PoC-2 results emphasize AI-assisted policy-driven security monitoring and enforcement, showcasing MonB5G's effectiveness in detecting and mitigating attacks, ensuring the robustness of learning algorithms, and enforcing security policies.

In conclusion, this deliverable provides an overview of the MonB5G project, highlighting the functional blocks, methodology, architecture, and AI-driven techniques employed. The experimental scenarios demonstrate the successful implementation and validation of MonB5G solutions, showcasing their ability to achieve zero-touch network and service management with end-to-end SLAs and policy-driven security monitoring. The outcomes of the project contribute to the advancement of intelligent and decentralized management and orchestration frameworks for future 5G LTE and beyond networks.

2 MonB5G functional blocks

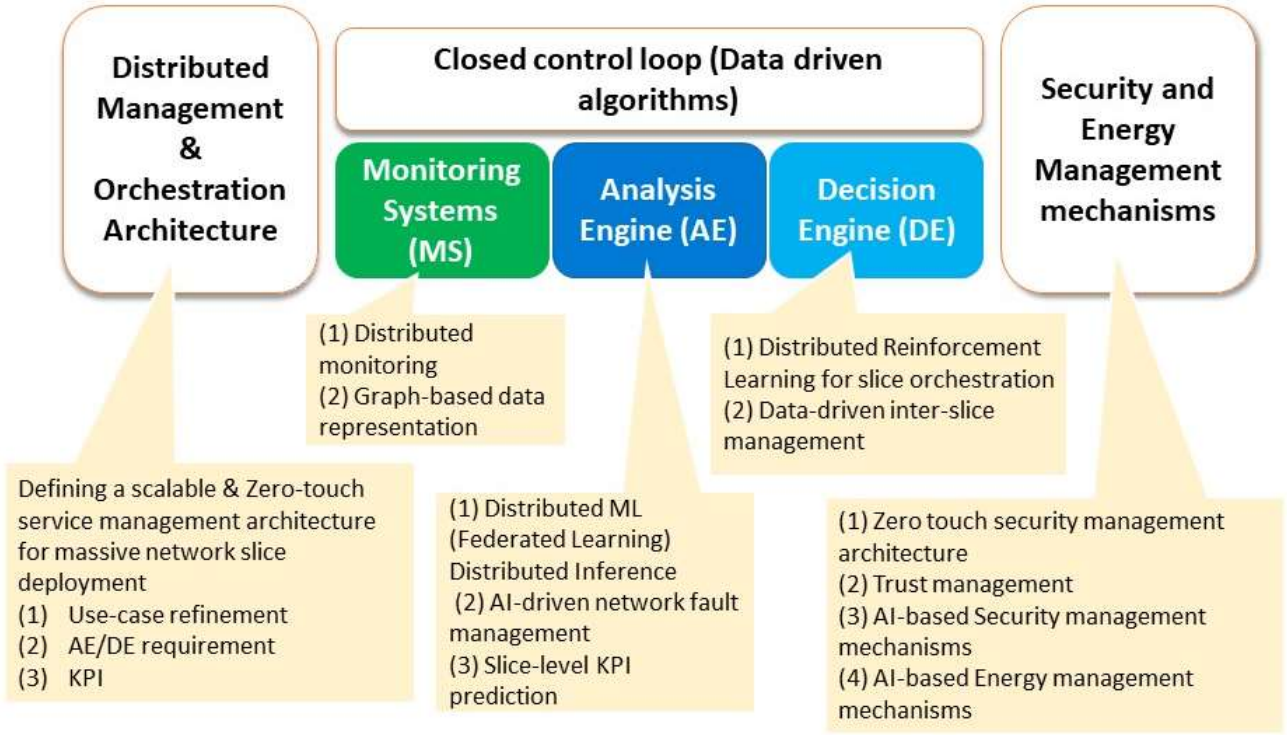


Figure 1 MonB5G Functional Blocks

The MonB5G project is divided into three functional blocks as shown in Figure 1,each treated in one dedicated work package (WP). The distributed Management & Orchestration Architecture functional block covers the activities dedicated to the definition of the MonB5G architecture that should be scalable and uses self-automated solutions to ensure the concept of zero-touch management aiming at supporting a massive deployment of network slices in beyond 5G networks. This functional block corresponds to the activity of WP2, which besides the MonB5G architecture, covered MonB5G use-case refinement, AE/DE requirement, as well as KPI to be achieved by the architecture. The second functional block concerns the activities related to the closed-control loop devised in MonB5G, which is composed of the triplet monitoring system (MS), analytic engine (AE), and decision engine (DE). While MS and AE-related research tasks have been conducted in WP3, tasks related to DE were done in WP4. The research activities and mechanisms dedicated to MS covered, for example, distributed mechanisms and Graph-based data representation. Regarding AE, the research activities covered algorithms and mechanisms that address the following topics: (i) Distributed ML (Federated Learning) and Distributed Inference; (ii) AI-based driven network fault management; (iii) Slice-level KPI prediction. Regarding DE research activities, they covered algorithms and mechanisms devised on distributed Reinforcement learning for slice orchestration and data-driven intra and inter-slice management. The final functional block of the project is securing 5G networks featuring network slicing. The security

research activities were divided between WP2 and WP5. WP2 leveraged the MonB5G architecture to include security and trust management, while WP5 covered the algorithms and mechanisms to reinforce the security of the network slices. Besides security, WP5 has a dedicated task to energy saving, where AI-based mechanisms were devised. Work Package 6 (WP6) focused on the integration of the MonB5G network architecture elements, verification of their interoperability, integration, and testing of the proposed AI-driven MonB5G mechanisms within the deployed architecture, as well as the demonstration of two proof-of-concept scenarios: PoC-1, which involves zero-touch multi-domain service management, and PoC-2, which entails AI-assisted security monitoring and enforcement.

3 Methodology overview

MonB5G technical development strategy focused on four key phases as shown in Figure 2.

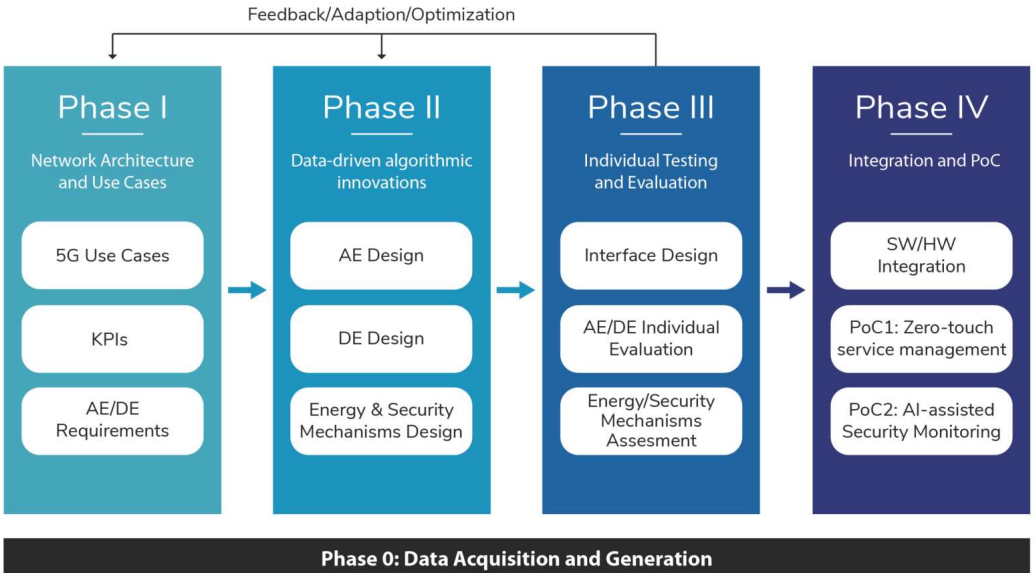


Figure 2: Phases of MonB5G technical development strategy

MonB5G started with data acquisition and generation in phase zero. Later, the first phase involved the exploration of network architecture and use cases. This phase was critical for determining the key performance indicators (KPIs) and the requirements for the access and data exchange mechanisms (AE/DE). Through a thorough understanding of these factors, MonB5G designed a network architecture that is tailored to meet the needs of its users.

The second phase of MonB5G development involved the development of data-driven algorithmic innovations. In this phase, MonB5G focused on designing AE/DE and energy/security mechanisms that leverage the latest advancements in AI and machine learning (ML). By leveraging data-driven insights, MonB5G created algorithms that are highly optimized, efficient, and secure.

The third phase of MonB5G development involved individual testing and evaluation. This phase was critical for ensuring that the network architecture, AE/DE, and energy/security mechanisms functioned as intended. MonB5G conducted interface design tests to assess how the network interacts with user devices. Individual evaluation of AE/DE ensured that the network was functioning optimally, while the assessment of energy/security mechanisms ensured that the network was secure and efficient.

Finally, the fourth phase of MonB5G development is devoted to hardware and software integration for two experimental PoCs. Through these PoCs, MonB5G demonstrates the efficacy of its development strategy and methodology in real-world settings. These PoCs serve as a crucial step in advancing the development of MonB5G and establishing it as a leading technology in the telecommunications industry.

In conclusion, MonB5G's development strategy and methodology represent a significant step forward in the telecommunications industry. By leveraging data-driven insights and algorithmic innovations, MonB5G is creating a network architecture that is tailored to meet the needs of its users. Through a rigorous testing and evaluation process, MonB5G is ensuring that its network is secure, efficient, and functioning as intended. And by conducting experimental PoCs, MonB5G is demonstrating the efficacy of its development strategy and establishing its unique technological advancements in the AI-driven network management field.

4 MonB5G architecture

The MonB5G framework uses the management system decomposition and the MAPE (Monitor-Analyse-Plan-Execute) paradigms as the basis. In our case, the MAPE concept is implemented in a distributed way by means of multiple AI-driven operations. Moreover, the runtime management of slices is distributed and programmable. MonB5G generic management structure is presented in Figure 3 and the architecture with static components (network slices are dynamic) in Figure 4.

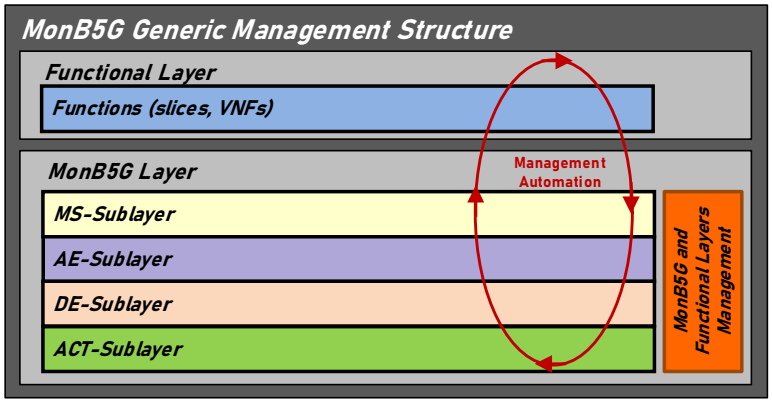


Figure 3: Generic view of MonB5G slice structure

We have also modified the NFV MANO approach slightly by distributing some of the orchestration functions. The key features of the proposed MonB5G framework are the following:

- **Distribution of management operations**. The management operations are AI-driven and pursue different goals. The embedded management concerns nodes, slices, orchestration domains and the E2E slice. Using distributed AI allows for local processing of management information processing, thus reducing the exchange of management information between entities. The AI-driven approach also enables the use of intent-based interfaces.
- **A strong separation of concerns**. In the MonB5G framework, Operation System Support (OSS)/ Business System Support (BSS) of each orchestration domain is focused on the lifecycle management (LCM) of slices and on resource management of this domain, but it is agnostic to slices (i.e., it is not involved in slice runtime management). Each single or cross-domain slice can be seen as a service with its own management platform (called embedded or In-Slice Management, ISM), which is separated from the domain(s) OSS/BSS.
- **ISM capabilities**. ISM is a part of the slice template and is responsible for the fault, configuration, accounting, performance, and security management (FCAPS) of a slice. That approach provides benefits such as isolation of management planes of slices (feature not provided by ETSI NFV MANO nor 3GPP). In the case of multi-domains slices, a special inter-domain component (ISM) is added to the E2E slice template. It interacts with domain-level ISMs to achieve the E2E management of the slice. ISM of each slice may act as a service orchestrator, i.e., it may use the orchestrator to request slice template modifications, and such action is no longer executed by the domain-level OSS/BSS. The request is typically based on the ISM analysis, and the action is related to slice topology update. Since ISM is part of a slice, and it is implemented as a set of VNFs, the resource scaling mechanism can contribute to ISM (i.e., slice

management) performance. Moreover, all the FCAPS functionalities can be dynamically deployed or updated during slice lifetime using the orchestration capabilities of ISM.

- **Hierarchical, E2E slice orchestration.** In MonB5G architecture, there are multiple domain-level orchestrators (they can be domain-specific) and one master orchestrator. This implies the use of domain-specific slice templates. The use of multiple orchestrators contributes to orchestration scalability.
- **Enhanced security of slices.** The use of the ISM concepts provides isolation of the management spaces of different slices, therefore contributing to enhanced slice security. It also limits information exchange between slices and OSS/BSS of each orchestration domain.
- **Support for Management as a Service (MaaS).** MonB5G allows the creation of a "management slice" that can be used for runtime management of multiple slice instances of the same template. In such a case, a new business player, called Slice Management Provider, can be involved in slice management.
- **Programmable, energy-aware infrastructure management.** The infrastructure management system proposed by MonB5G can use the architecture to deploy its services dynamically, in a similar way in which slices are deployed. The framework provides extensions to include energy-aware operations on infrastructure resources by the use of modified, energy consumption aware orchestration. For that purpose, the interface between the orchestrator and the infrastructure is provided.
- **Slice Tenant Portal and slice contract negotiation procedures.** The framework introduces the business portal entity that enables the Slice Tenants to request the deployment of slices based on the selected slice templates and perform lifecycle management operations on their slices. Moreover, the capability for slice contract negotiation procedures between the Slice Tenant and respective stakeholders (e.g., Infrastructure Providers) are facilitated.

The abovementioned features are in line with several ETSI ZSM requirements– the list of the essential requirements of ZSM that are satisfied by the MonB5G management and orchestration framework is provided in Deliverable [MONB5GD24].

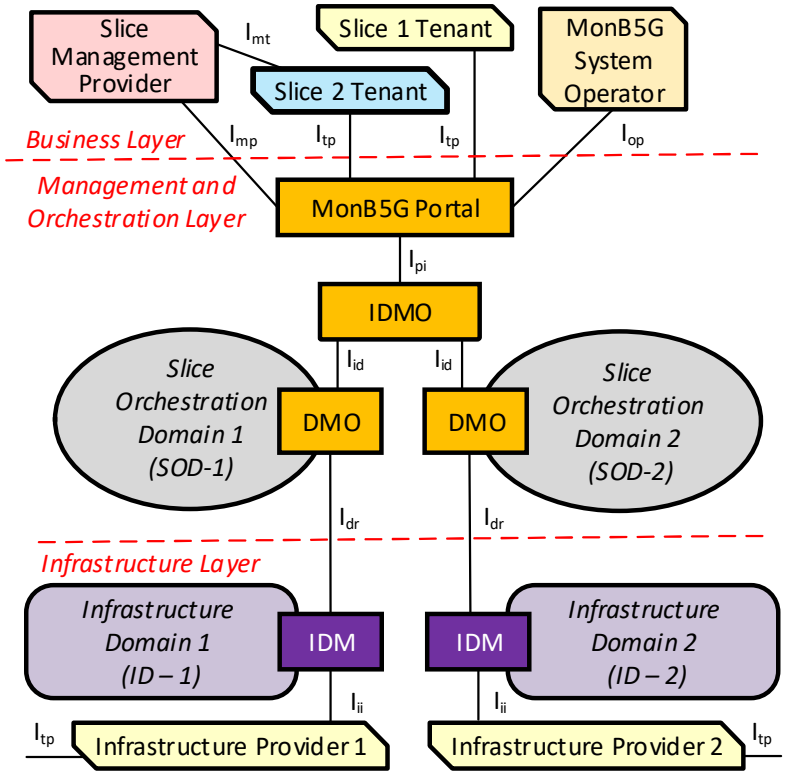


Figure 4. Static components of the MonB5G architecture

MonB5G introduces logical entities for monitoring, analytics, and decision making that are decomposed into **distributed, interacting components** executed at various levels: at the OSS/BBS level, inside the virtualised infrastructure, and embedded in slices. Through local data processing and decision-making, our design aims to achieve two goals: (a) minimize the exchange of (big) data between components to keep management scalable, and (b) significantly reduce the reaction time of data-driven management decisions that can be handled locally. **Reducing the monitoring load** is critical for **carrier-grade performance** in a sliced beyond 5G network, a challenge that has not yet been adequately addressed. This approach requires, however, proper coordination of the "local" management subsystems. The autonomic network management based on feedback loops that is proposed in MonB5G brings significant benefits, but also faces many problems. The most important problems are related to response times (associated with the round-trip time between network elements) and system stability (the managed system is a nonlinear one - the feedback-based control may lead to instabilities and chaotic behaviour). To this end, we propose a hierarchical control scheme with fast local control loops and slow wider-scope ones. Leveraging time-scale decomposition at different levels of the proposed system, we achieve to limit the interference among different feedback-based decisions. We also assume a rich multi-objective environment, where various goals (e.g., energy consumption, statistical multiplexing, slice isolation, etc. vs. performance) may have different weights, and the proposed algorithms should be able to automatically learn to prioritize accordingly. Moreover, we have also introduced the architecture components that are responsible for providing the feedback loop control stability evaluation and restoring.

5 AI-driven MonB5G techniques

5.1 AI-driven techniques for monitoring system (MS)

The MonB5G monitoring system (MS) aims to gather detailed information on network slices deployed in multiple domains. To achieve real-time monitoring for online analysis and reconfiguration of slice KPI in response to unexpected network changes, the system's scalability is critical. To enhance scalability, several strategies were implemented in the design and implementation of the monitoring system. An overview example of the MS is illustrated in Figure 5.

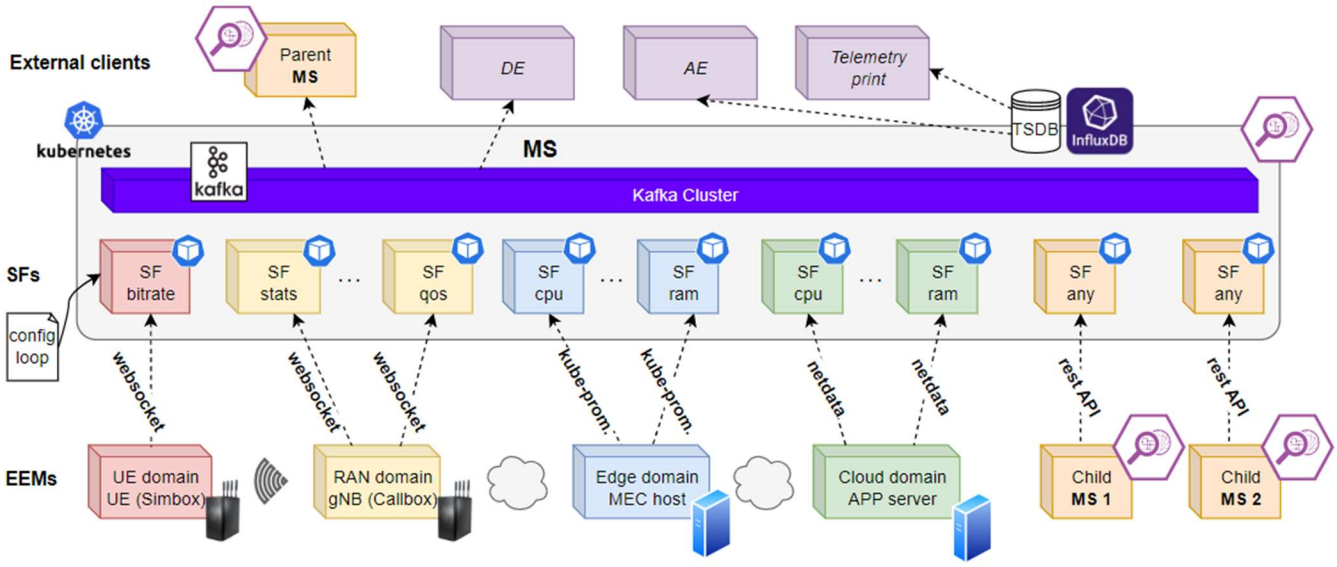


Figure 5. The MonB5G MS integrates various strategies to enhance scalability and facilitate real-time monitoring of network slices in multi-domain networks.

First, the system was conceived from a distributed system, using microservice architecture and implemented as a cloud-native application in a Kubernetes cluster. All system components, including the Sampling Functions (SFs), are implemented as containers, and deployed as pods in a cloud-native architecture. The use of a messaging bus through Kafka, which spans across all nodes, enables communication between components and linear scalability of the system.

Second, the MonB5G monitoring system is designed with a hierarchical architecture that allows multiple sub-monitoring systems to collaborate in a master-slave pattern. With this structure, monitoring tasks are performed at lower levels, limiting data traffic, and reducing reaction time. Monitoring information can be extracted directly from distributed entities and aggregated locally, with the lower-level monitoring systems directly monitoring the VNF and PNF resources, and the higher-level monitoring system ordering lower-level systems to gather data from them.

Third, to reduce the system's resource footprint and improve scalability, MonB5G shares its components with other administrative components. The TSDB and Streaming Bus are accessible to AE and DE, and Sampling Functions can allow more than one target, reducing traffic between MS and Embedded Element Managers

(EEMs) Each sampling loop is implemented by a single Docker image, minimizing the need to frequently create and destroy the sampling function container.

5.2 AI-driven techniques for analytic engine (AE)

MonB5G AE integrates leading-edge AI techniques to provide a variety of slice-specific analytics for inter-domain, cross-domain and network-aware KPI predictions. Due to diversity and complexity of analysis tasks for different technical domains, we explore and develop various AI-driven techniques to enable accurate and cost-efficient KPI analysis.

To enable distributed network slice management, the key contributions focus on AI-driven techniques for distribution of analysis operations from centralized management entities to local ones that associate with functional units near which the management data and tasks are generated. The technical challenges are as follows. On the one side, localized analytics service reduces unexpected communication cost and response delays. On the other side, local analysis lacks a global view of slice/network status, so analysis can be less accurate and even deviate from the actual situations. We develop advanced machine learning methods, such as distributed neural networks and federated learning, to optimally distribute analysis tasks among hierarchical management entities for correct KPI prediction but low overhead and latency.

Figure 6 and Figure 7 illustrate the key innovative AI-driven strategies for decentralized AE towards scalable zero-touch management. The developed AI techniques extract patterns and representations of the local data at the low layers and learn an optimal way to exchange the abstract information between the lower and upper layers of the management hierarchy, by which local predictive performance and time/resource costs are balanced.

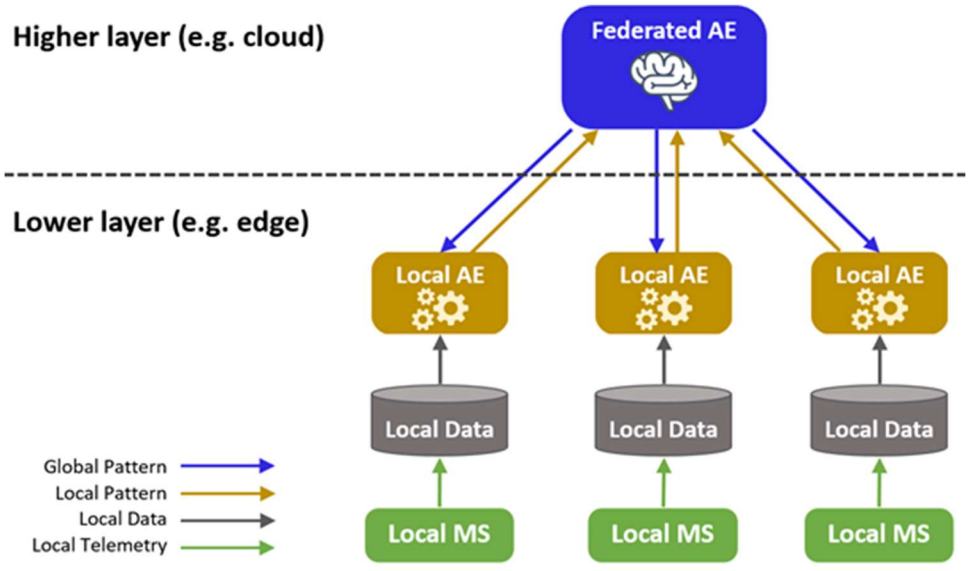


Figure 6: Federated learning for distribution of analysis operations

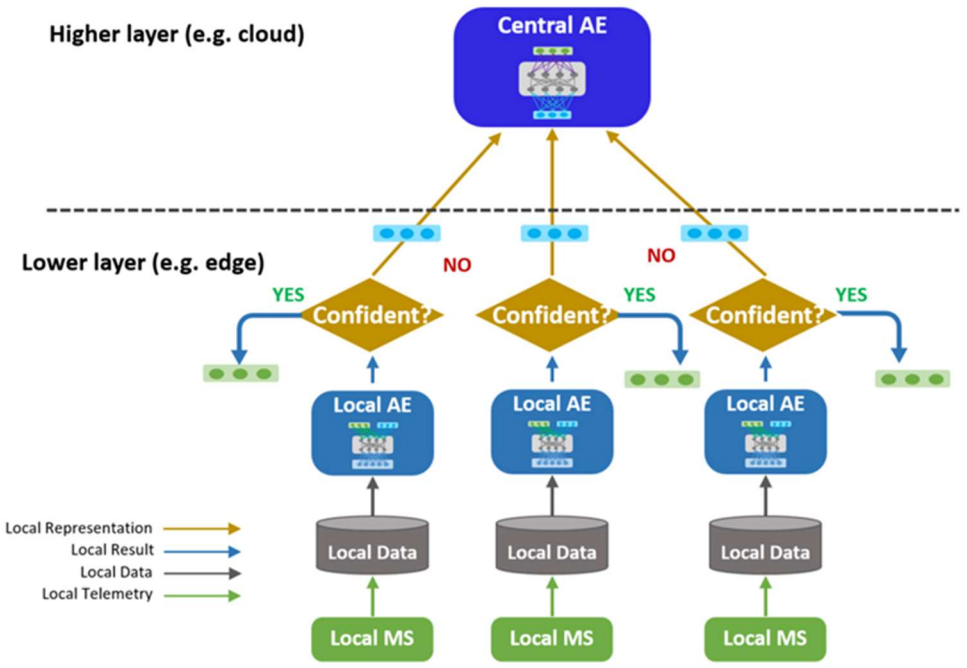


Figure 7: Distributed neural networks for distribution of analysis operations

In addition, we explore other novel ML techniques, such as representation learning, context-aware analytics, and graph-based learning to address diverse challenges in real analysis tasks with superior performance. Putting all together, MonB5G AE has developed the following AI-driven techniques for disparate analysis tasks to facilitate scalable zero-touch slice management.

- Extend federated learning to execute resource forecasting towards SLA violation control A well-designed set of statistical constraints optimizes the federated learning procedure to meet the requirements of distributed network management. The analysis results have been successfully used for decentralized resource allocation in network slices, achieving very low SLA violations.
- Propose distributed deep neural networks (DDNNs) for precise KPI prediction using local and global information. The DDNNs learn an optimal offloading policy that only shifts less confident local predictions to a high-level AE entity for global analysis. As most of analysis tasks are executed locally, it guarantees accurate good predictions while minimizing monitoring overhead and response delay for complex slices involving multiple domains and multiple functional entities in the same domain.
- Introduce graph neural networks (GNNs) to exploit hidden relationships among telemetries for slice KPI forecasting, e.g., slice latency. We integrate GNNs into Recurrent Neural Networks (RNNs) to capture both spatial and temporal patterns for precise prediction of slice latency based on diverse resource KPIs.
- Present task-oriented loss for enhanced local KPI forecasting. Forecasting algorithms are often designed for generic purposes. However, KPI prediction in network management has specific target, e.g., traffic load forecasting for resource allocation. We propose to integrate the target into loss function to generate better predictions for end task. We introduce additional

regularizations in a loss, which penalize over and under allocation of resources as well as resource reallocation settings. By ensuring that the right number of resources are provisioned to a network slice when needed, the novel forecasting technique significantly reduces SLA violation rate.

- Present forecasting-based interpretable anomaly detection for local fault detection. As outperformance of the proposed local forecasting methods, we can precisely capture the local patterns of KPIs. If the true observations deviate from the learned patterns, then faults must happen. We propose to use robust z-score, based on probability theories, to detect faults from predictions without expensive labelling procedure. In addition, explainable AI is explored to interpret the detected faults.
- Enhance decentralized ML towards asynchronism and complete distribution for cross-domain anomaly detection. We propose an optimal information propagation policy, such that all nodes only communicate with their neighbours, without the need of a central coordinator. The fully distributed ML method can largely reduce communication overhead and waiting time but retain a good anomaly detection rate.

5.3 AI- driven techniques for decision engine (DE)

The MonB5G DE employs data-driven algorithms for decision-making in all slice lifecycle management operations (admission control, scaling, migration), which might involve either a single slice (intra-slice), and technological or administrative domain, or even a large number of such slices (inter-slice) and end-to-end operations. Leveraging AI techniques as well as input from the Analytical Engine (AE) (or even raw input from the Monitoring System (MS)), the DE aims at zero-touch management of a massive number of network slices. The objective is to improve the network resource usage and at the same time meet the diverse SLAs. WP4 explores the devise of distributed and scalable AI-driven algorithms for the DE. The MonB5G DE architecture, and hence the proposed DE algorithms, are innovative in the following three dimensions:

- DE can be distributed flexibly across different administrative domains, technological domains, or fine granularities such as one DE per tenant, slice, or even VNF (virtual network function).
- WP4 solutions can manage slices consisting of multiple VNFs across different technological/administrative domains and handle diverse SLAs with sophisticated end-to-end KPIs (e.g., end-to-end delay constraints across an entire, complex VNF graph).
- WP4 solutions go beyond the state-of-the-art both in terms of improving the performance of previously proposed data-driven algorithms for such tasks, and of distributing the components of the data-driven algorithms (e.g., multi-agents) or even the components of the neural network architecture (Distributed DNNs).

There are 3 main tasks (T4.1, T4.2, T4.3) in WP4 each corresponding to a different slice management operation. T4.1 focuses on admission control, T4.2 on intra-slice orchestration, and T4.3 on inter-slice orchestration. D4.1[MONB5GD41] reported the preliminary work on these tasks, while D4.2 [MONB5GD42] mainly extended the algorithms proposed in D4.1 (or in some cases introduced new) to support multiple technological (or even administrative) domains and to improve their scalability, SLA performance, convergence, etc. Also, significant effort was made in D4.2 to clearly validate the scalability of the proposed solutions through simulations. Following, there is a summary of the key contributions in each of the three different tasks (T4.1, T4.2, T4.3).

Admission control is a crucial operation of slice Lifecycle Management (LCM), aiming to maximize the number of co-existing slices while minimizing SLA violations. In D4.1, admission control algorithms based on Deep Q-reinforcement Learning (DQL) and Regret Matching (RM) for RAN and Cloud domains were proposed, while several potential algorithms for slice component placement were identified. In D4.2, two other slice admission approaches were introduced. The first one is a multi-domain data-driven Heuristically Assisted Deep Reinforcement Learning (HA-DRL) scheme based on the A3C DRL algorithm, combined with a heuristic to reduce the learning period of the DRL. The second approach, TASAC (Time-of-day Aware Slice Admission Control), is based on the Time-of-Day traffic curve that reflects the human activity, which has been used to predict resource consumption of eMBB slices and uses a Deep Q-Network (DQN) algorithm for admission control.

Intra-slice orchestration refers to reconfiguration and resource allocation operations for a single slice. Initially, D4.1 focused mainly on domain-specific solutions, while D4.2 included cross-domain operations. To this end, a Service Chain Elastic Management framework called SCHEMA was introduced, based on a Distributed Reinforcement Learning algorithm. SCHEMA has a modular architecture that can support multi-technological domain setups and demonstrates a collaborative agent KPI optimization. Then, an extension of this algorithm, called SafeSCHEMA, was proposed, integrating the important SafeRL framework into the scheme to operate in scenarios where the natural tendency of RL algorithms to explore “any” possible (slice) configuration might be forbidden or prohibitively expensive and therefore the exploration/exploitation components need to be protected.

Inter-slice orchestration lies at a level above its intra-slice counterpart and requires a wider picture of the network and its performance. An algorithm employed for this task must be able to handle multiple slices (possibly spanning across multiple technological domains), which leads to very challenging high complexity problems. In D4.2, a multi-agent DRL algorithm for VNF placement and migration was proposed, utilizing independent DQN agents (one per slice or even per VNF) to radically improve the scalability of the algorithm. This scheme supports multiple VNFs per slice, i.e., an arbitrary, probabilistic VNF graph (which also allows for loops), as well as various realistic end-to-end performance metrics for the average flow served by such a VNF graph. Then, a Federated DRL (FDRL) framework for slice resource allocation at the RAN was introduced, enabling the provisioning of federated learning schemes to further enrich the capabilities of the decision agents. This FDRL framework also incorporated a dynamic agent selection mechanism to enable more efficient collaboration among local decision agents during learning. The benefit of this scheme is that it utilizes local decision agents (one per base station) to account for more timely and accurate information, and it reduces the overhead (due to control information) towards the core network. The last contribution for this task was a probing scheme for VNF bottleneck localization.

Finally, D4.2 introduces a MonB5G method for the coordination of control loops, which is relevant to all three tasks mentioned above. This issue arises when different functions (or sometimes DEs) are trying to optimise a single goal, which can lead to suboptimal results or chaotic system behaviour. To this end, a novel coordination framework was devised, which is compliant with the MonB5G architecture, and it aims at predicting reconfiguration’s impact and hence generating recommendations to minimise conflicts related to network or slice configuration parameters.

5.4 AI-driven security techniques

The environment that has been set up for the testing of AI-driven security techniques is based either on simulation or virtualized infrastructure. When the virtualized infrastructure is used, the hosts are virtualized machines (VM) managed by OpenStack, on which we deploy applications or worker nodes of Kubernetes (K8S) cluster. The network service of the control plane of the 5G system (5GS) and the MonB5G components MS, AE, DE, and ACT are K8s services and pods, while the remaining parts of the 5GS such as the data plane and the simulator of UE and RAN are applications running on the VMs. Moreover, we also keep all host clocks synchronized by the Network Time Protocol (NTP) to have consistent timestamps in logs and to correlate events.

The algorithms implemented for the anomaly detection are tested as a component of the MonB5G AE deployed on the platform.

Description of models

A brief description of the implemented models and the evaluation scores for each model is provided in the following:

1. **Random Forest**

Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

2. **Gradient Boosting Classifier**

Gradient boosting involves three elements:

1) A Loss function

2) A Learning rate for the prediction

3) An additive model to add weak learners to minimize the loss function.

Hyperparameters Tuning for the Gradient Boosting Classifier:

- **Learning Rate**

This determines the impact of each tree on the final outcome. The Gradient Boosting Classifier works by starting with an initial estimate which is updated using the output of each tree. The learning parameter controls the magnitude of this change in the estimates. Lower values are generally preferred as they make the model robust to the specific characteristics of tree and thus allowing it to generalize well. Lower values would require higher number of trees to model all the relations and will be computationally expensive. The model was trained for a number of learning rates to find the optimum learning rate. The rates that were tested were from 0.075 which is considered very small to a very high learning rate of 3. The optimum learning rates were found to be between 0.85 to 1.25, all having the accuracy score of 0.99 which is seen at both the training phase as well as the validation phase of the algorithm. The best learning rate was thus selected for the implementation purposes.

- **n_estimator**

This hyperparameter represents the number of sequential trees to be modelled. The Gradient Boosting Classifier is fairly robust at a higher number of trees, but it can still overfit at a point. Therefore, the value of this hyperparameter was set to 20 for simulation purposes.

3. **XGBoost**

XGBoost is the 'acronyms' for Extreme Gradient Boosting and is an efficient implementation of the stochastic gradient boosting machine learning algorithm. The stochastic gradient boosting algorithm, also called gradient boosting machines or tree boosting, can be used to boost the decision trees in order to calculate accurately predictions for multi-class cases. The main advantage of the XGBoost implementation compared to the gradient boosting algorithm is that the overfitting can be dramatically reduced.

For each supervised model (algorithms), confusion matrices have been implemented, that allows visualization of their performance. In the following, the three main evaluation scores are presented:

- **Precision**

Precision = True Positives / (True Positives + False Positives)

This evaluation score is appropriate when the focus is on minimizing false positives.

- **Recall**

Recall = True Positives / (True Positives + False Negatives)

This evaluation score is appropriate when the focus is on minimizing false negatives. Sometimes, we want excellent predictions of the positive class (i.e., we want high precision and high recall). This can be challenging, as increases in recall often come at the expense of decreases in precision.

- **F1-Score**

Classification accuracy is widely used because it is one single measure used to summarize model performance. F-Measure provides a way to combine both precision and recall into a single measure that captures both properties. Alone, neither precision nor recall tells the whole story. We can have excellent precision with terrible recall, or alternately, terrible precision with excellent recall. F1-score provides a way to express both concepts with a single score.

Once precision and recall have been calculated the two scores can be combined into the calculation of the F1-Score. The traditional F1-Score is calculated as follows: F1-Score = (2 * Precision * Recall) / (Precision + Recall). Like precision and recall, a poor F1- score is 0.0 and a best or perfect F1- Score is 1.0

- **Confusion Matrix**

A confusion matrix is a table to understand how well our model predictions have been performed (which can become especially complex when we have multiple classes and not the classic binary 0/1 problems).

5.5 AI-driven energy efficiency techniques

Energy based Orchestration techniques are studied in two dimensions:

1) Energy-Aware, Multi-Domain Orchestration

The objective of PoC-1 is to optimise the placement of a network slice VNFs locally (intra-domain) for lower energy consumption and latency. We evaluated our proposed solution (SCHE2MA detailed in [MONB5GD54]) through model validation and simulation while demonstrating its ability to jointly reduce average service latency by 103.4% and energy consumption by 17.1% compared to a centralized RL solution. Unlike similar works in the literature, the examined PoC 1 solution is decentralized, eliminating any central point of failure and enabling scalability.

The physical network infrastructure was represented as a graph with nodes and edges, representing the sets of the nodes and links respectively (Figure 8). To accommodate the functions of an SDN-NFV enabled network, the nodes are split into two types. One is dedicated to network switching that is responsible for forwarding the service traffic and the other one has the ability, not only to forward but also to instantiate, terminate or migrate VNFs, up to its physical capacity. Both types of nodes handle the routing of the SFC traffic.

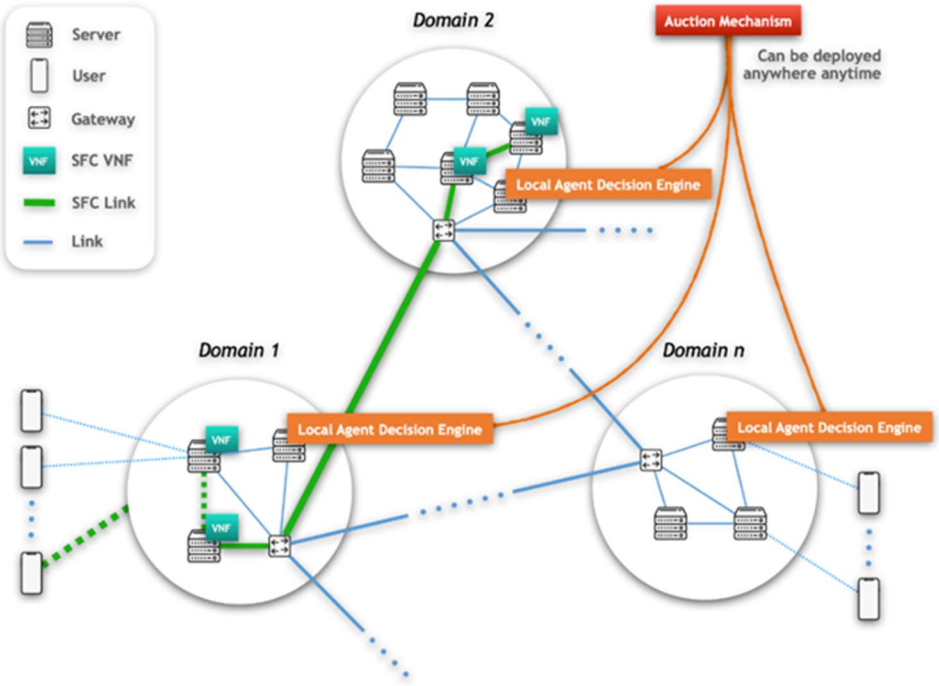


Figure 8: Network graph representation

The network has a finite number of resources that can be measured and tracked through metrics, such as: latency, bandwidth, CPU cores, Random Access Memory (RAM) and disk storage. The instantiation and run-

time of a service VNF in the network servers uses a portion of the aforementioned computational and network resources. These metrics are vital for the decision-making process.

In Figure 9 (a), we depict the average energy consumption of the examined network of 500 simulations for a varying number of users, normalized based on the SCHE2MA performance (%, mJ). The total energy consumption consumed by the network during the operation of a service is composed of 2 different components (see section IV.D of [SCHE2MA]): i) energy consumed by the utilization of the computational resources and ii) the link energy consumption. It should be noticed that average energy consumption shown in Figure 9 (a) is normalized based on the SCHE2MA results and the Y-axis results are in %. We observe that the energy consumption increases almost linearly with the number of users due to the massive number of transmissions. The reason is that introducing more users to the network generates additional requests that consume more energy during each transmission. Therefore, the overall energy consumption of the network is higher. It is evident that the proposed solution can maintain lower energy consumption in all scenarios, reaching almost 17.1% reduction in the case of 100 users. The reason for this behaviour is the ability of SCHE2MA to cluster VNFs into the servers, minimizing the costly communication between servers.

Figure 9 (b) presents the performance of the most critical metric in URLLC services, the average service latency. We observe that the average service latency increases due to insufficient computing resources in servers within the domains as the number of active users grow. However, it must be noted that SCHE2MA outperforms both baselines by offering a 103.4% reduction in latency for the case of 100 users without increasing the energy consumption, which is a considerable performance improvement while also maintaining lower energy consumption than both baselines. That is possible due to VNF clustering in servers, which minimizes the number of transmissions in physical media. SCHE2MA demonstrates a clear indication of its ability to implement better VNF placements that satisfy the latency and energy consumption trade-off.

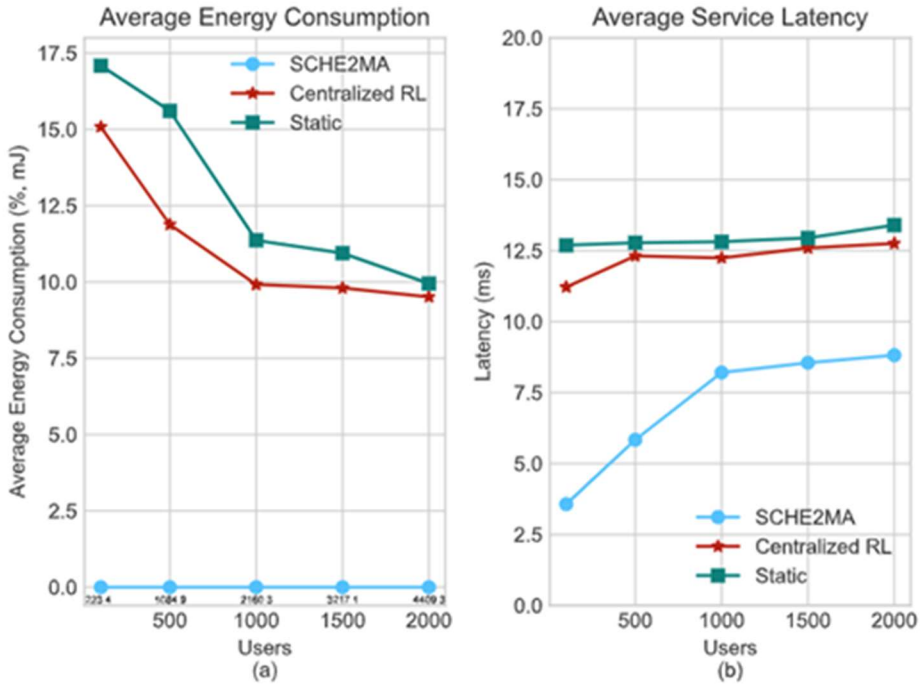


Figure 9: (a) Average energy consumption improvement expressed in percentages, compared to the proposed solution. (b) Average service latency per number of users in multiple traffic scenarios. Lower is better for both figures.

2) Stochastic Federated Learning Scenario

A novel Statistical Federated Learning (StFL) method [Chergui2020] [Chergui2021TWC] [Chergui2021] has been introduced in the Section 4.2 of MonB5G deliverable [MONB5GD53]. The proposed algorithm learns from non-independent and identically distributed (non-IID) datasets in an offline manner, while considering slice-level service level agreement (SLA) long-term statistical constraints. Our approach combines a constrained empirical cumulative distribution function (ECDF) and percentile, both of which are non-convex, to formulate the SLA problem. We introduce a proxy-Lagrangian two-player game strategy to optimize the local FL task while considering both the original SLA constraints and their surrogates. Our StFL method results in a 20% reduction in SLA violations compared to the baseline FedAvg method, and also offers more than x10 overhead reduction and energy efficiency improvement compared to a centralized deep learning approach for SLA-constrained prediction.

Table 1 shows the overhead and the energy consumption obtained by the fully centralized SLA-constrained deep learning algorithm (CCL) and the StFL algorithms (both policy-based and non-policy-based). Let us recall that the main difference between the non-policy and policy-based algorithms is that in the non-policy StFL all the AEs are considered in the Federated Learning process, whereas in the policy-based StFL only a subset of the total number of AEs is considered in each round of the FL process. In particular, in this case, 15 AEs out of a total number of 20 AEs have been considered for the policy-based StFL. Note that starting from the convergence point of StFL algorithms achieved after 21 rounds in the FL algorithms (see Section 6.1.1.4 of

[D3.3] for further details), more than 10 times overhead and energy consumption reduction are obtained in comparison to centralized approach. Comparing the two versions of the StFL, the client subset selection in the policy-based StFL results in a significant reduction in energy consumption (approximately 25 %) relative to the non-policy-based StFL.

Table 1: Overhead and energy comparison between centralized solution and federated learning-based algorithms

Rounds	21	50	60	70	80
Overhead CCL (KB)	1875				
Overhead non-policy StFL (KB)	44.3	105.5	126.6	147.7	168.8
Overhead policy-based StFL (KB)	33.2	79.1	94.9	110.7	126.6
Energy CCL (J)	4.54				
Energy non-policy StFL (J)	0.11	0.26	0.31	0.36	0.41
Energy policy-based StFL (J)	0.08	0.19	0.23	0.27	0.31
Energy Gain w/non-policy FL	**x 41.27**	**x 17.8**	**x 14.8**	**x 12.7**	**x 11.1**
Energy Gain w/policy FL	**x 56. 32**	**x 23. 65**	**x 19.71**	**x 16.90**	**x 14.78**

6 PoC-1: Zero-Touch Network and service management with end-to-end SLAs

Below are the description of the scenario-1 and 2 in that is described in Description of Work and the explanation how MonB5G solutions in the testbed are addressing them.

6.1 Experimental Scenario 1.1: Zero-Touch multi-domain service management

Objectives: The main objective of this scenario is to assess the data-driven management systems in a multi-domain scenario with respect to their ability to guarantee the stringent end-to-end SLA of the Tactile Internet application. Automated zero-touch service management and multiple redundancy mechanisms must ensure practically zero downtime due to the critical, high-availability applications.

Description: In this experimental scenario, multiple Network Function Virtualization Infrastructures (NFVIs), hosted in both project testbeds as well as on AWS infrastructure will be combined to demonstrate Zero-Touch service management in complex multi-domain services. In this scenario the Storage, Compute, and RAN functions of the Tactile Internet application will be hosted in different regions under the control of local Network Function Virtualization Orchestrators (NFVOs) and Decision Engines, but the end-to-end SLA must be honoured. Continuous monitoring and closed-loop autonomic control mechanisms, which will be common across regions and testbeds, will ensure self-healing, self-configuring and self-scaling of services, to address faults and performance issues in any of the service technological domains. In this scenario, the following will be demonstrated:

1. **Continuous monitoring** of Quality of Service (QoS) (e.g., bitrate) and Quality of Experience (QoE) (e.g., video quality) metrics across all technological domains and service functions, by the respective local monitoring engines.
2. Model updates forwarded by the **local monitoring engines** to the centralized Decision Engine where it is assessed **w.r.t. the probability of impacting the SLA**.
3. At the Decision Engine model updates are **federated** and correlated with predicted events (e.g., Peak Planned Occurrences) and service degradation indicators.
4. **Local plans and policies** are forwarded to local decision engines, who remain responsible for implementing them (e.g., scaling the respective VNFs)
5. The "**energy slicing**" subsystem, which is part of the Decision Engines, is responsible for implementing the energy optimization policies, while ensuring that end-to-end SLAs are not affected.

MonB5G solutions described in this deliverable are addressing the above description with the proposed solutions in PoC-1 scenario-1 as follows: In this scenario, **continuous monitoring of QoS metrics and local monitoring engines** are addressed by *Monitoring System and design solution*. **Local plans and policies** with **federated updates** considering **SLA constraints** are addressed using *Stochastic Policy-based Federated Learning solution*. **Prediction of events** for **service degradation** is addressed by *Fault Detector and LSTM and graph-based anomaly techniques*. **Energy efficiency in a slice** is addressed by *Stochastic Policy-based Federated Learning*.

Testbed for PoC-1: Details on providing access for remote users for deployment are as follows. The MonB5G project partners are granted access to a CTTC Testbed Instance (TI), which serves as a virtual experimental environment that comprises an independent and self-contained Network Functions Virtualization ecosystem. Prior to utilizing the resources, the partners have requested their needs and the resources that CTTC can

provide in terms of time and effort. To facilitate this work, a Testbed Instance Requirements template has been provided to each involved partner. Partners were required to fill out the template, which includes the description of the solution or experiment for which the TI will be used, the technical and administrative personnel involved from both sides, the type of Testbed Access needed (remote or physical), the estimated duration of the TI, and most importantly, the resources required.

Hardware details: In this section, we describe hardware details for PoC#1 testbed. The typical resources available in a Testbed Instance include:

- Computing: physical servers, virtual machines, and containers.
- Platform as a Service/Management and Orchestration: Kubernetes.
- Monitoring Systems: MonB5G MS, Prometheus, and Kube-Prometheus.
- 5G (see Figure 10):
    - 5G Core: Amarisoft Callbox 5GC, Free5GC, OAI CN, and Open5GS.
    - 5G RAN: Amarisoft Callbox gNBs and UERANSIM gNB.

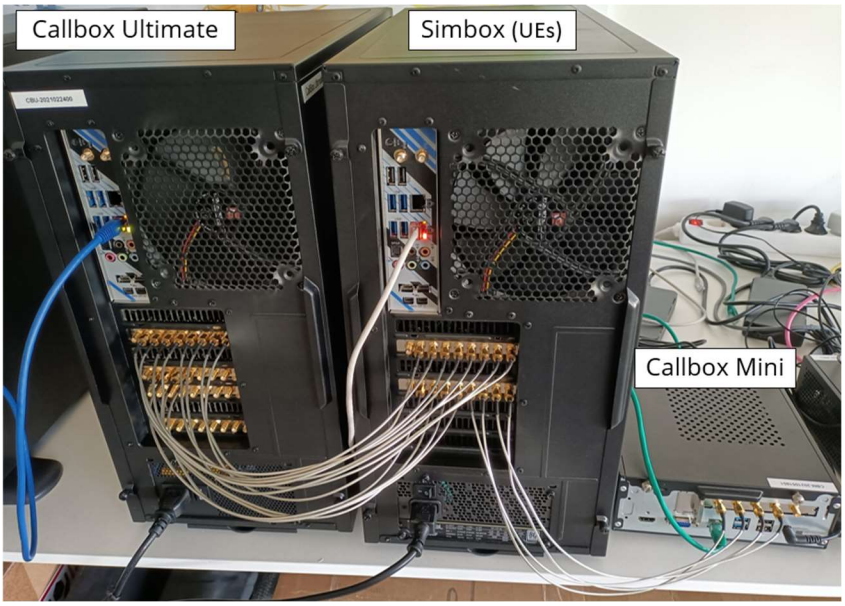


Figure 10: 5G hardware components.

Regarding remote access to the Testbed Instance, CTTC provides partners with a VPN connection. Specifically, a physical server located at CTTC's premises hosts a Linux Container (LXC) with an OpenVPN (OVPN) server service running inside. The OVPN server has been configured to allow access to the specific virtual machines (VMs) involved in the TI, using routing rules.

CTTC's infrastructure exposes a public IP address to the partners, which can be accessed using OVPN client files created and given specifically to them. These client files enable partners to connect to the OVPN server and access the restricted environment, a private network, through Secure Shell (SSH) protocol. Access is granted only after the partner's public IP address or DNS domain has been whitelisted in CTTC's infrastructure firewall. Finally, the traffic between the partner and the OVPN server is established and encrypted using the OVPN client file.

The objective of PoC1 infrastructure setup is to be able to show contributions by the partners and achieve **integration**. PoC#1 setup is given in Figure 10. Both ES 1.1 solution 1 on Federated Learning and ES 1.1 solution 2 on Anomaly detection have three sites, each with a Kubernetes cluster running a monitoring system instance. Site 1 (or A in this figure) and Site 2 (or B in this figure) run the main demo workloads and sampling functions for the monitoring. In site-1 we have univariate and multivariate anomaly detection modules. In Site-1, 2 and 0, we have federated learning modules. Site 0 (or C in this figure) is used for federated learning aggregation server.

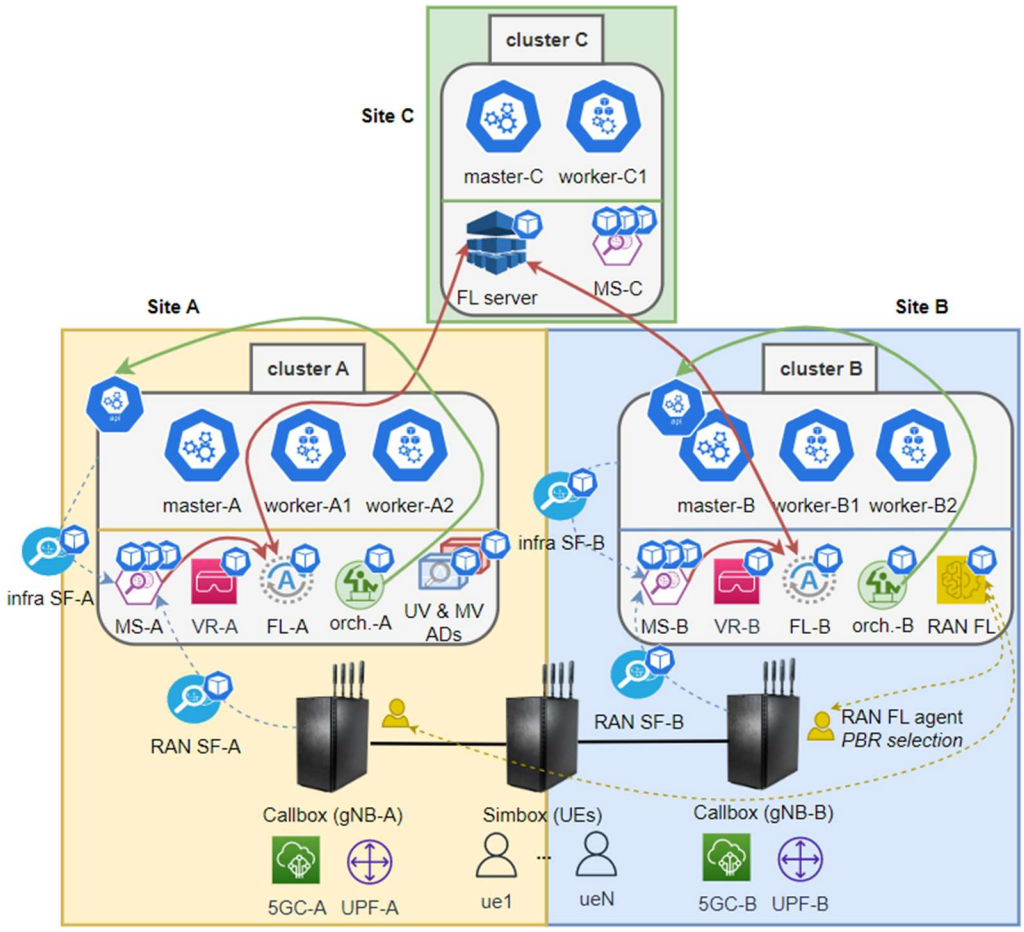


Figure 11: PoC1 Architecture components, infrastructure and setup. FL is from Experimental Scenario 1.1. and ADD ML from Experimental Scenario 1.2

6.1.1 FL experiments

Scaling of the VR video streaming: The experiment carried out consisted of scaling of the VR video streaming server, where FL technique is used to improve the quality of experience of VR video streaming clients connected to BS at each site. To do this, the network below has been deployed with 5G Amarisoft BS, the VR video streaming clients, VR video streaming server and the MonB5G MS, AE, DE and ACT at each FL site. FL facilitates distributed collaborative learning without disclosing original training data where the idea behind

FL is to train the ML model collaboratively among distributed clients without sharing their data and violating the privacy accord. Therefore, FL locates ML services and operations closer to the clients, facilitating leveraging available resources on the network's edge.

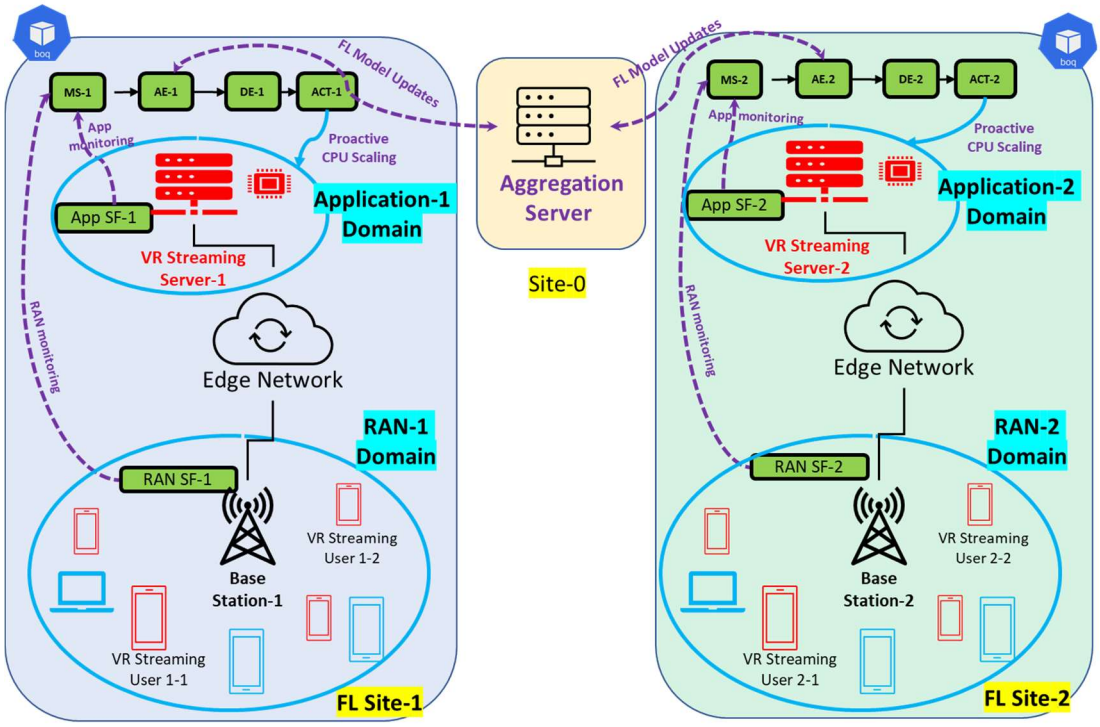


Figure 12: FL deployment and VR video streaming servers, VR video streaming

Figure 12 shows the experimental topology created with FL architecture at CTTC premises. All components of the FL are implemented as pods in K8s environment as separate K8 clusters and each MonB5G component runs as separate container orchestrated by Kubernetes. In our demonstration setup, we have three sites to perform experiments. 5G BS and the VR-streaming servers are in FL site-1 and FL site-2. Finally, the aggregation server is located at site 0. All these sites are connected to each other via service mesh deployment model in K8s. Each VR video streaming users are connected to corresponding BSs at each site for VR video streaming traffic from VR-streaming servers. The number of VR video streaming users are increased in different patterns.

Monitoring overhead gain calculation for management and orchestration:

Table 2 shows the overhead induced by the baseline *fully centralized MANO deep learning (CCL)* [Chergui2020] and the *Statistical Federated Learning based resource predictor and scaling*. For the computation of the overhead, we have considered that both the datasets and update models are coded in 32 bits. In the uplink between the clients and the aggregation server, the approximate overhead can be calculated as in [MONB5GD3.3]. Starting from the convergence point of *Federated Learning based resource predictor and scaling* at round 8, more than 10 times overhead reduction (approximately 11.11 times) is obtained in comparison with the centralized SLA-constrained algorithm.

Table 2: Monitoring Overhead comparison between centralized solution and federated learning-based algorithms

Rounds	2	5	8	20	30
Monitoring overhead CCL (KB)	52				
Monitoring overhead FL-based resource predictor and scaling (KB)	1.152	2.880	4.608	11.152	17.28
Monitoring Overhead Gain	**x45**	**x18.02**	**x11.2**	**x4.66**	**x3**

Energy Consumption Computation in Management and Orchestration Plane:

Using the same approach as in [MONB5GD5.4] to calculate the energy and using the modeling formulas, energy reduction gain is proportional to the transmitted communication overhead gain. Taking into account that convergence has been achieved at 8 iterations in our experimental case, x11.2 energy gain is achieved in comparison to CCL.

Table 3: Energy consumption comparisons between centralized solution and federated learning-based algorithms

Rounds	2	5	8	20	30
Energy CCL (mJ)	125.6				
Energy FL-based resource predictor and scaling (mJ)	2.7	6.9	11.1	26.9	41.75
Energy Gain	**x45**	**x18.02**	**x11.2**	**x4.66**	**x3.0**

6.1.2 Multivariate Anomaly Detection

We propose a Graph-based Interpretable Anomaly Detection (G5IAD) reference architecture as shown in Figure 13 which comes with the following list of contributions. (1) The importance of slice level KPIs predicted by graph-based representation learning method applied in conjunction with recurrent neural networks (RNN). (2) This solution coupled with an interpretable Artificial Intelligence (XAI) - based solution enables a wider adoption of automated management by telecom operators. The proposed framework G5IAD provides explanations (e.g., what combination of KPI values impacted the slice latency KPI) that can be used by the experts and/or by a flexible DE for maintaining slice SLAs, by using direct information from the interpretable method, compared to methods that compute and compare outcomes for all possible actions.

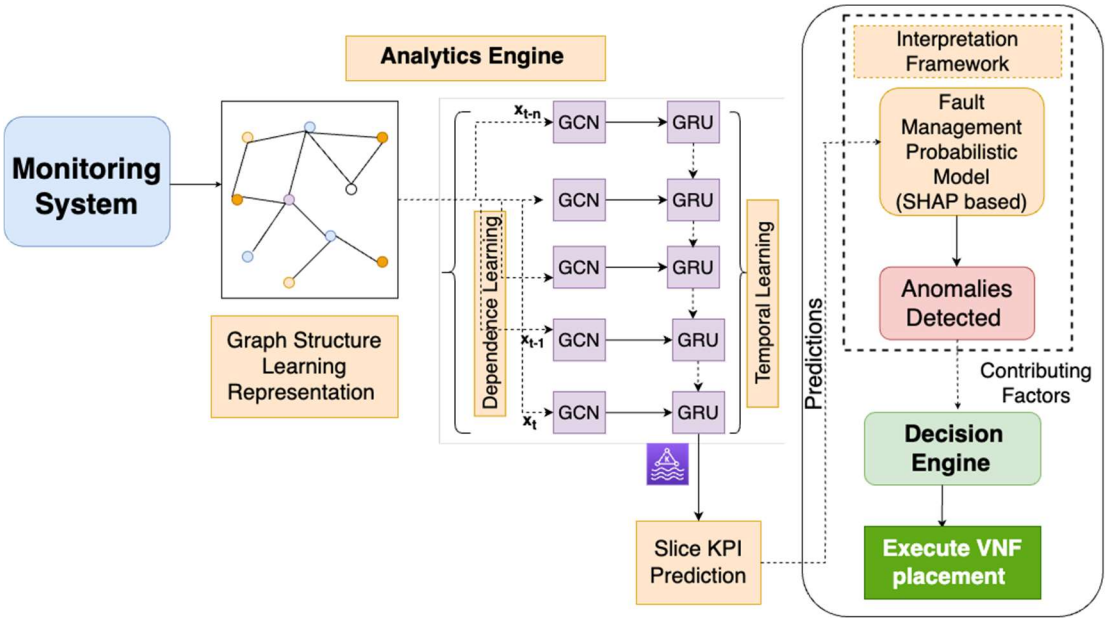


Figure 13: Analytics Engine reference Architecture

In PoC1 scenario 1, we showcase the implementation of multivariate anomaly detection using an intelligent neural network-based algorithm (as shown in Fig 13) powered by MonB5G distributed framework through Docker deployment as shown in

Figure *14*. After training the model with the dataset, using normal instances across multiple variables, Prometheus and Grafana are utilized to monitor and visualize the model's performance. Finally, the trained model *is employed to detect anomalies in unseen test samples.*

```
easchaw@EMB-Y0MPXCM9 Docker  % docker images
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
monb5g_poc1_image   latest    600f3bfb6b55   19 seconds ago   2.45GB
<none>              <none>    091b049232c6   8 minutes ago    2.45GB
<none>              <none>    686148d1f7eb   27 hours ago     2.45GB
<none>              <none>    5bf496be09a6   27 hours ago     2.45GB
<none>              <none>    823fd7f716c8   27 hours ago     2.45GB
<none>              <none>    f22c5ec985d0   27 hours ago     2.39GB
<none>              <none>    00a8e5f6a96b   27 hours ago     2.39GB
<none>              <none>    c6183ec80743   28 hours ago     2.01GB
<none>              <none>    bb18c25ee6f8   28 hours ago     2.01GB
<none>              <none>    9a6bbbd2909d   28 hours ago     2.01GB
<none>              <none>    da4eefe79c3b   29 hours ago     1.77GB
easchaw@EMB-Y0MPXCM9 Docker  % docker run -p 8000:8000 -d monb5g_poc1_image
4970c54a3fb627c2d8ec4aad3e3275ab3e040189b886aa81a1946cb71961fcb5
easchaw@EMB-Y0MPXCM9 Docker  % docker ps
CONTAINER ID   IMAGE               COMMAND                  CREATED          STATUS          PORTS
4970c54a3fb6   monb5g_poc1_image   "python3 MonB5G.py"      3 seconds ago    Up 2 seconds    0.0.0.0:8000->8000/tcp
68ccb087c305   686148d1f7eb        "tail -f /dev/null"      27 hours ago     Up 27 hours
cb0387750ffa   5bf496be09a6        "tail -f /dev/null"      27 hours ago     Up 27 hours
fa40db8beb78   823fd7f716c8        "tail -f /dev/null"      27 hours ago     Up 27 hours
172e3acbcb26   f22c5ec985d0        "tail -f /dev/null"      27 hours ago     Up 27 hours
7ba0dec919ac   00a8e5f6a96b        "tail -f /dev/null"      27 hours ago     Up 27 hours
d76daacac24c   c6183ec80743        "tail -f /dev/null"      28 hours ago     Up 28 hours
de9c045a603d   bb18c25ee6f8        "tail -f /dev/null"      28 hours ago     Up 28 hours
e67cde29361d   9a6bbbd2909d        "tail -f /dev/null"      28 hours ago     Up 28 hours
```

Figure 14: Docker deployed and running.

When Grafana dashboard is investigated, the normalized unseen test sample is depicted, showing CPU, O, and R in Figure 15. It can be observed that the first anomaly was detected at the same time step across all the variables, which was caused by the activation of 8 UEs at a period when only 1 UE was expected in the normal pattern. This resulted in the generation of anomalies in the three variables, with higher R, O, and CPU than expected depicting multivariable anomalies. For the rest of the time, there is no anomaly detected as the data is very similar to what has been trained. This enabler addresses the "reduction of time between NS malfunction and anomaly detection" KPI. Multivariate-based Anomaly Detection detecting anomalies among the multitude of network resources (CPU usage at the server, Outbound traffic at the server, Aggregate downlink bit rate) at the same time.

Figure 15: Multivariate Anomalies been detected by the model (CPU, O, R at same time)

6.1.3 LSTM-Based Anomaly Detection

LSTM-based Anomaly Detection AE (further referred to as AD, originally described in [MONB5GD32]) is deployed in RAN domain and used to detect anomalous changes in traffic originating from a streaming server in the Cloud and consumed by several UEs. As previously described in [MONB5GD61], due to containerization and standardised APIs, the same AD component can be easily configured to operate in other domains e.g., in Cloud to analyse server-side metrics. In this test, it is assumed, that there is a typical time-of-day dependent traffic pattern, which represents the variable user activity (Figure 16). In next generation networks, it is not unlikely to assume, that based on this time-of-day traffic pattern, adequate number of resources will be allocated to the slice for the purpose of efficient resource management. In an event of sudden traffic increase, it is possible that the allocated number of resources will not be enough, and relevant scaling operations will be needed to provide the access to the service for the increased number of users. Analysis of traffic can also help to identify other issues, such as incorrect load balancing, or a complete service failure. Detected anomalous information can be further sent to relevant MonB5G layer components (DEs, AEs) for further processing. One of the possible actions to tackle sudden, unexpected increase of user traffic (i.e., caused by an online event) would be server upscaling, to temporarily increase the service capacity.

Figure 16 depicts emulated time-of-day traffic pattern, by varying the amount of UEs between 1 and 8 (first chart), as well as user traffic (second chart), total user traffic (third chart) and CPU usage (fourth chart).

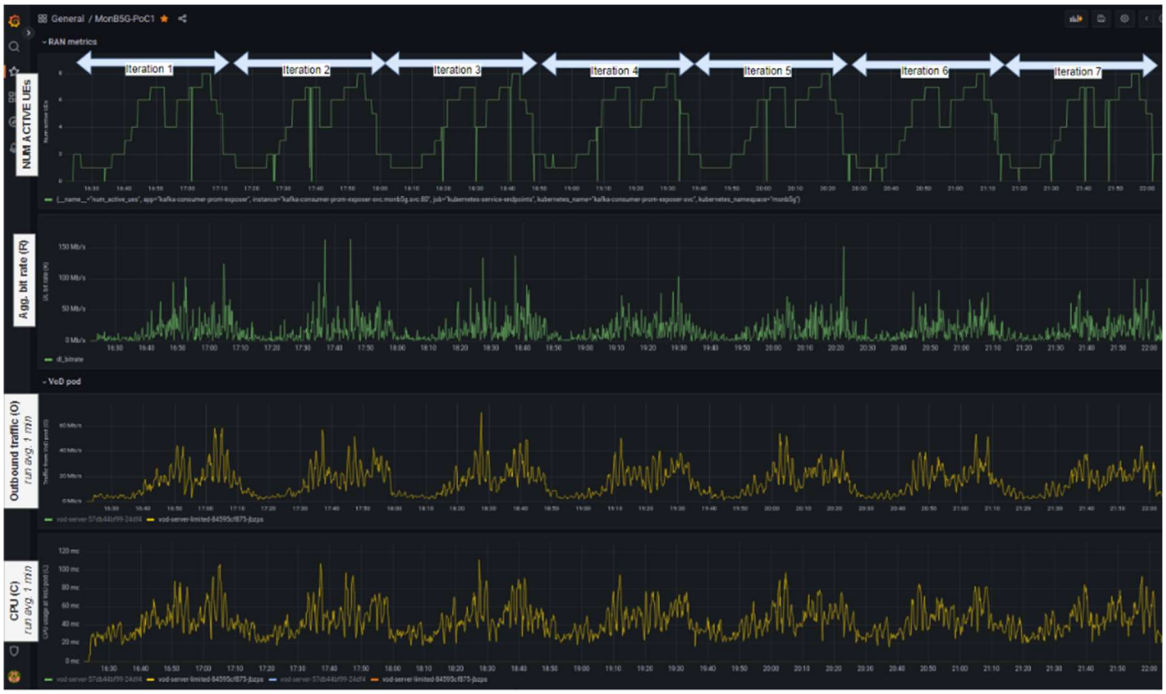


Figure 16: The metrics collected during non-anomalous network operation.

The data was further aggregated and used to train the AD's in-built LSTM model in the offline manner. Data from RAN is collected by the MS via SF and transferred to AD via message bus. The output from AD working on non-anomalous signal is depicted in Figure 17. Orange and yellow lines show real and predicted value respectively. Blue line depicts the error between prediction and actual value. Anomalies are determined based on the error value and a threshold algorithm. In this scenario, a static threshold (red line) has been used. Sign of the error (positive or negative) can also signal what kind of anomaly (traffic increase/traffic decrease) is detected.

Figure 18 depicts data with injected anomalous traffic changes:

- Sudden increase in traffic during night-time (1st, 3rd and 5th green peaks), matched with large, negative error,
- Earlier than usual decrease in traffic (2nd green peak), matched with large positive error,
- Sudden traffic drop (4th green peak), matched with large, positive error.

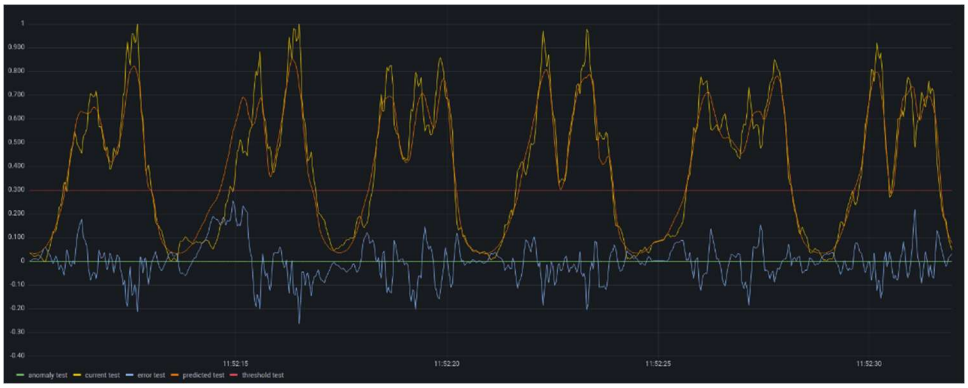


Figure 17: Non-anomalous signal traffic

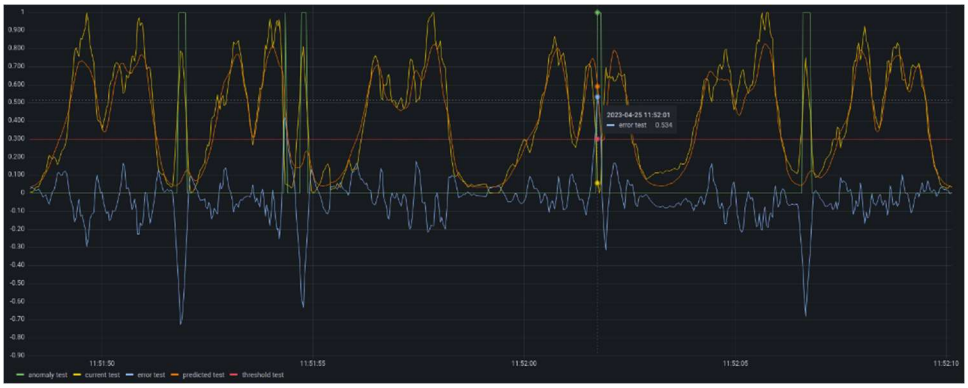


Figure 18: Signal with anomalies

It can be observed that the proposed AD trained on a typical traffic pattern (Figure 17), enables quick detection of traffic deviations (Figure 18). As depicted in Figure 18, in all cases, the anomaly was detected at the beginning of an abnormal traffic change and lasted until after traffic returned to normal. Considering the local deployment, the detection delay is solely dependent on the traffic volume sampling rate and reporting period. This gives the system an early warning and time to take a proper action (i.e., run diagnostics, make a reconfiguration, etc.).

Finally, the link for the video of Experiment Scenario 1.1 can be found here in MonB5G YouTube channel[1].

6.2 Experimental Scenario 1.2: Elastic end-to-end slice management

Objectives: As part of our experiments, we will demonstrate how MonB5G mechanisms react to address local performance issues in multiple technological domains as well as at changes to traffic patterns in various timescales. The ability of these mechanisms to guarantee almost zero latency for Tactile Internet applications

[1] MonB5G PoC1- Scenario 1: FL predictor and anomaly detection: https://www.youtube.com/watch?v=F02zLyhBkpA

by proactively acting (and predicting) spikes in user demand will be assessed. Finally, special emphasis will be on data-driven Radio Resource Management mechanisms to optimize the RAN sub-slice.

Description: As the number of Network Slice Instances (NSIs) increases, the scale and complexity of lifecycle management and slice reconfiguration makes automation a necessity. Each NSI consists of multiple NSSIs, generally one per technological domain (i.e., 5G Core, RAN and transport network), while each technological domain in MonB5G has its own data-driven MS, AE and DE components. In this scenario, in addition to the Tactile Internet application NSI, a massive number of slices will also be emulated in order to demonstrate the following:

1. **Continuous monitoring of each NSSI** by the respective **Monitoring Engine** at appropriate time-scales, to identify performance issues (e.g., deteriorating signal reception).
2. Decision Engines at each domain are able to recover local faults, but also **forward model updates** w.r.t. sub-slice performance to the central Decision Engine.
3. Sub-slice performance data will be **federated** with traffic pattern predictions **at the Decision Engine**, and **proactive actions** will be taken to prevent missing end-to-end service SLAs
4. **Proactive Actions** are implemented by the **respective domain controller** (e.g., forcing UE handovers at the RAN, or traffic steering to avoid bottleneck points)

MonB5G solutions described in this deliverable are addressing the above description with the proposed solutions in PoC#1 scenario #2 as follows: In this scenario, **continuous monitoring of each NSSI metrics by the Monitoring Engine** are addressed by *Monitoring System and design solution*. **Forwarding model updated** with d**ata Federation** for **proactive actions at the Decision Engine and domain controller** is addressed by *DRL-based resource allocation* solution.

6.2.1 A Multi-Agent Learning for Distributed Resource Allocation in RAN Domain

We propose a distributed architecture for RAN slice resource orchestration based on DRL, consisting of multiple AI-enabled decision agents that independently take local radio allocation decisions without the need for a centralized control entity. We design and implement the overall framework as shown in Figure 19.

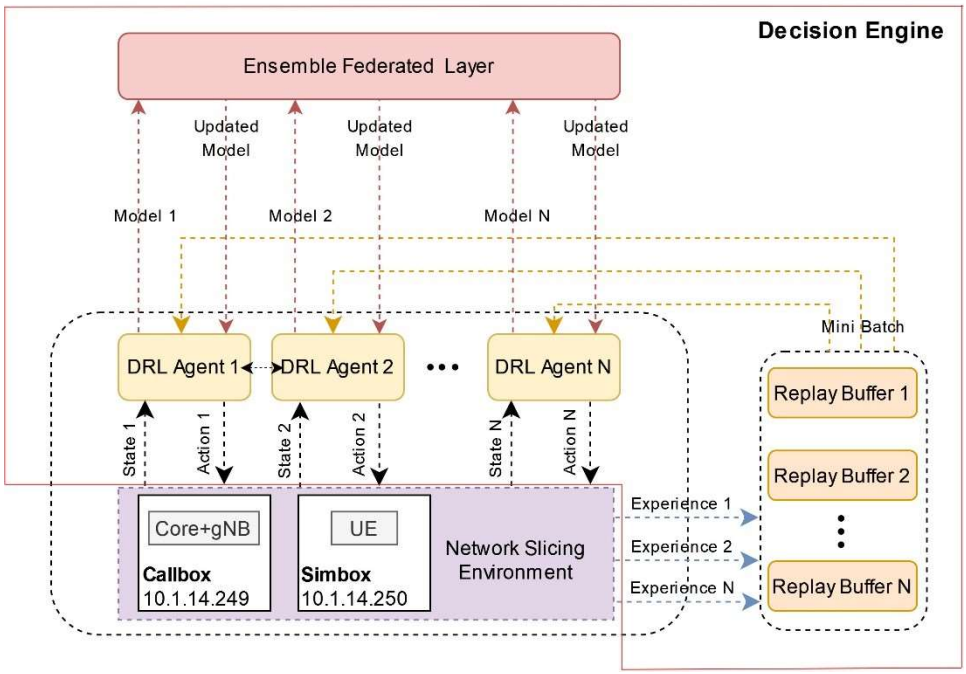


Figure 19: Federated RAN slicing architecture

Figure 19 depicts the evaluation scenario considered in our PoC. We leverage a distributed learning mechanism and multiple decision agents that collaboratively specialize their decision policies onto real-time traffic demands, aided by a coordinated exchange of information to avoid the occurrence of conflicting situations. In particular, we instantiate two gNBs and connect them through a local 5G core. Then, we deploy two eMBB slices, namely Slice A and Slice B, over each gNB, and spawn the setup of a local decision agent for each slice instance. As already detailed in D4.3 [MONB5GD43] and D6.1 [MONB5GD61], exploiting the monitoring system API, the RAN agents can retrieve real-time monitoring information from the Amarisoft RAN platform, which in turn are used to perform training activities pursuing RAN resource allocation. We generate different traffic for each running slice, considering it as the aggregated volume for each UE connected to the specific slice. Moreover, we allow only the agents belonging to Slice A to exchange their local models and perform federation.

For system monitoring and performance analysis reasons, we also design an online dashboard that provides a real-time overview of the solution. Figure 20 illustrates the Grafana-based graphical user interface developed for the testbed. It provides an overview of the key performance metrics of test scenario, as well as the agents-specific metrics such as instantaneous reward, allocation gap, and allocated radio resources. After an initial exploration and training phase, achieves the right trade-off between optimal allocated radio resources and traffic demand accommodation, i.e., exploitation. To this aim, the agents receive partial observations from the running services or slices (e.g., channel quality, consumed resources, etc.) by interacting with each other and with the underlying physical environment. A reward is calculated as an incentive mechanism based on the actions of all agents, indicating how the agents ought to behave. The designed reward function

guarantees adequate performance, impeding over-provisioning and leading to fair resource allocation among the running slices deployed within the same cell.

Figure 20: GUI: Graphical User Interface

6.2.2 Slice Admission Control Based on Traffic Prediction (DE)

Time-of-Day-Aware Slice Admission Control (TASAC) DE allows taking Slice Admission Control (SAC) decisions leveraging the information on the predicted future network traffic intensity to select the most profitable slices for the operator. The use case implemented to evaluate TASAC benefits is presented in Figure 21.

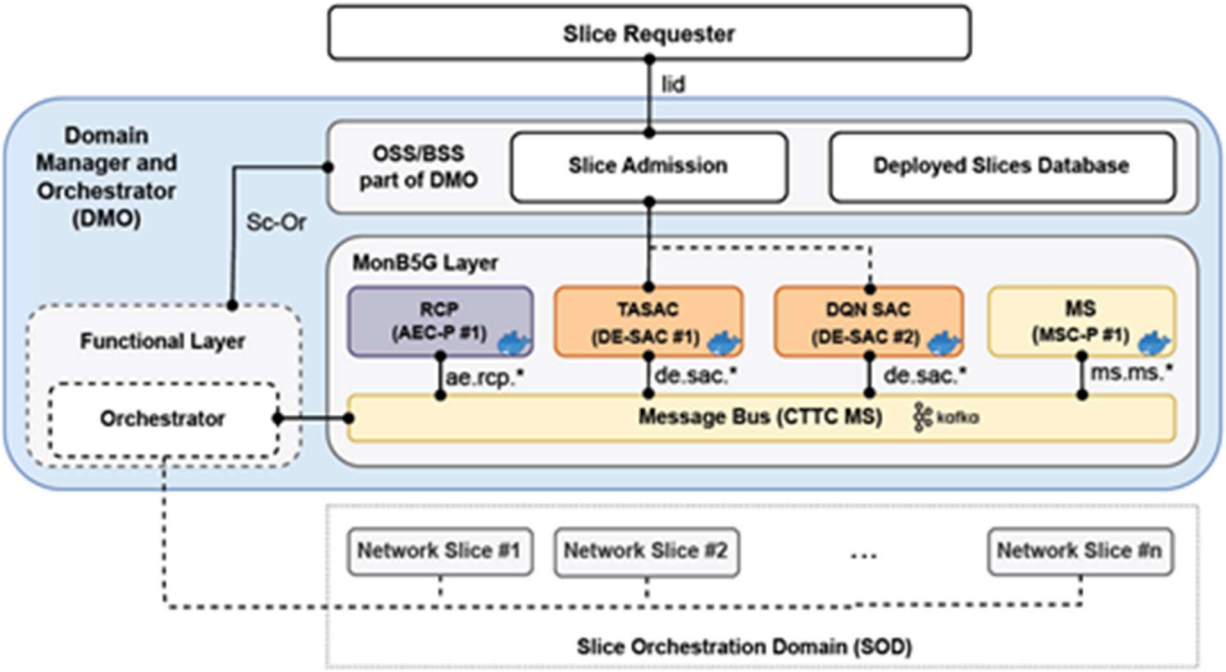


Figure 21: The evaluation approach of TASAC enabler

To emulate the realistic conditions, first, the duration of a day is set to 1440 *tu* (time units) which correspond to the acquisition of monitoring data every 1 min. The influx of SARs is modelled as Poisson process with the rate *tu*. The requested slice types follow a discrete uniform distribution, the maximum resources requested by tenants equal to 10 *u* (generic resource units) for mMTC and 100 *u* for the eMBB slices, and their duration follow the discrete uniform distribution (equivalent to the 30 min to 24 hours deployment). The resources available for the operator to orchestrate slices are *u*. The agent is trained over consecutive episodes which last 1440 *tu* each. During the experiments, the decisions are made based on available bandwidth and ToD activity information derived from the carrier-grade transport network. The results are presented in Figure 22 and in Table 4.

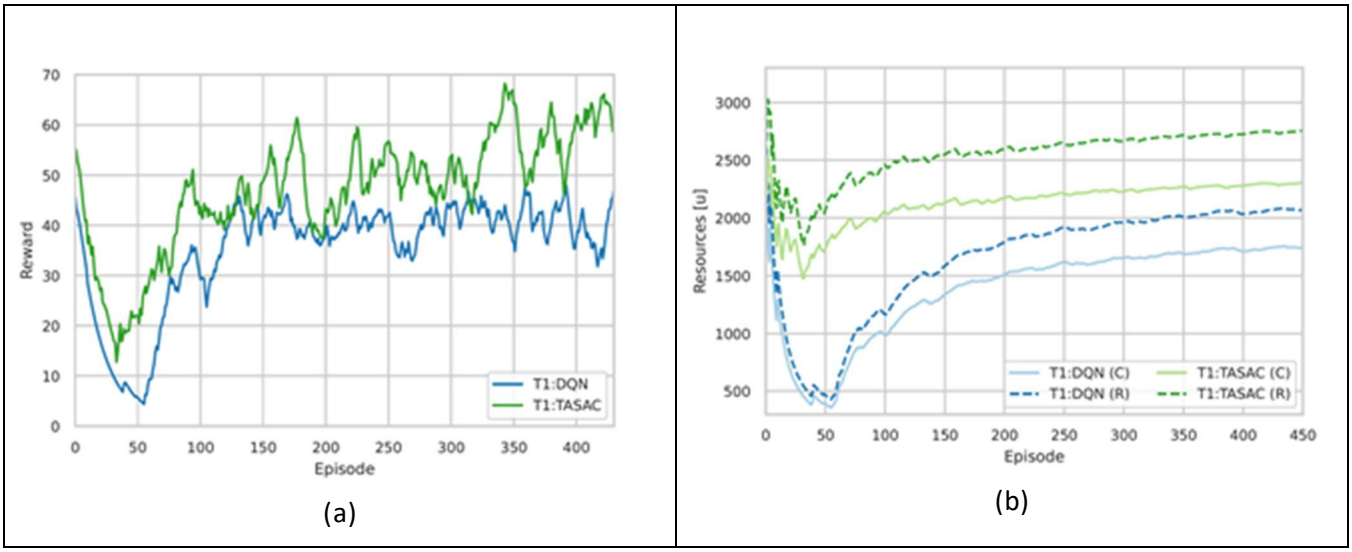


Figure 22: Comparison of TASAC and the baseline DQN-based SAC in terms of: obtained reward (a) and resources (b), requested (R) or consumed (C) by the deployed slices

It can be observed that both the agent's accumulated reward as well as the resource utilization are much better for the TASAC DE than conventional DQN-based DE. In the late stages of operation i.e. when the near-optimal stable policies are reached by the agents the reward gain exceeds over 30%, while the resource utilization gain is around 60%.

Table 4: Comparison of cumulative gains of TASAC and DQN-based DE

Episode	Reward			Requested resources [u]			Consumed Resources [u]		
	DQN	TASAC	Gain [%]	DQN	TASAC	Gain [%]	DQN	TASAC	Gain [%]
50	148.6	755.9	**408.7**	47662.3	107797.1	**126.2**	39718.2	90213.9	**127.1**
150	3793.5	5510.6	**45.3**	167948.6	350174.5	**108.5**	141231.4	292807.7	**107.3**
250	7642.3	10296.2	**34.7**	348340.5	609580.5	**75.0**	293353.0	509734.0	**73.8**
350	11927.0	15664.1	**31.3**	545034.3	877934.7	**61.1**	458988.8	734095.2	**59.9**

Finally, the link for the video of Experiment Scenario 1.2 can be found here in MonB5G YouTube channel[2].

[2] MonB5G PoC1-Scenario 2: A multi-agent learning for distribution resource allocation in the RAN domain: https://www.youtube.com/watch?v=kOrNwMtXjfg

7 PoC-2: AI-assisted policy-driven security monitoring & enforcement

The second Proof-of-Concept "AI-assisted policy-driven security monitoring & enforcement" consists of two experimental scenarios. The main purpose behind this use-case is to show both the efficiency of MonB5G when relying on AI to ensure new security threats detection in addition to their respective mitigation actions, and the proper enforcement of the AI-based techniques through novel trust-based evaluation mechanisms. By leveraging SDN, MANO and NFV technologies and through the usage of the platform built for MonB5G, the effectiveness of the security system has been demonstrated.

The first experimental scenario is about attack identification and mitigation. This scenario applies Machine Learning algorithms (such as gradient boosting,), to identify attacks to slices and automatically apply actions to mitigate them. The objective of this scenario is to demonstrate the robustness of MonB5G for identifying, detecting, and then mitigating the in-slice and cross-slice attacks. Different malicious events can be detected thanks to various MonB5G's components including MS, AE, and DE. This experimental scenario has improved the efficiency of MonB5G for detecting attacks (such as DDOS and alter attacks).

The second experimental scenario is about the robustness of learning algorithms in the face of attacks, such as poisoning attack. First, this scenario has assessed the vulnerability of the distributed learning algorithms to adversarial attacks and how their performance metrics (e.g., precision, accuracy, etc.) are affected by those attacks. Then, the model robustness has been improved by MS, AE, and DE components. The objective of the second scenario is to demonstrate that even under significant numbers of misbehaving entities, distributed learning can be carried out in a robust way. This has been validated using standard metrics used in machine learning.

7.1 Experimental Scenario 2.1: mMTC Attack identification and mitigation

The first experimental scenario aims to provide a solution featuring zero-touch security management of in-slice attack detection and mitigation considering mMTC slices in 5G. The proposed solution relies on machine learning to detect abnormal traffic of MTC devices that could cause DDoS on the control plane of the 5G core network (by flooding the AMF with signalling messages). Hence, it was possible to mitigate the attack by making efficient decisions to prevent flooding of the AMF with traffic and causing DDoS or deteriorating performance for legitimate users. This type of attack can be more effective on mMTC than other 5G services, assuming the very high number of MTC devices supposed to support. In this chapter, we assume that a mMTC slice is composed of a shared sub-slice with other existing network slices, which runs the 5G CN (including the AMF) and gNodeB, and a specific sub-slice to run the application that collects data from the MTC devices.

Figure 23 highlights the interaction among the different actors involved in detecting and mitigating DDoS attacks: the mMTC network slice components (UEs and 5G CN) and the closed-control loop elements (MS, AE, and DE). It is worth noting that the closed-control loop runs in parallel to the mMTC network slice elements and only monitors Attach Requests to detect and mitigate attacks.

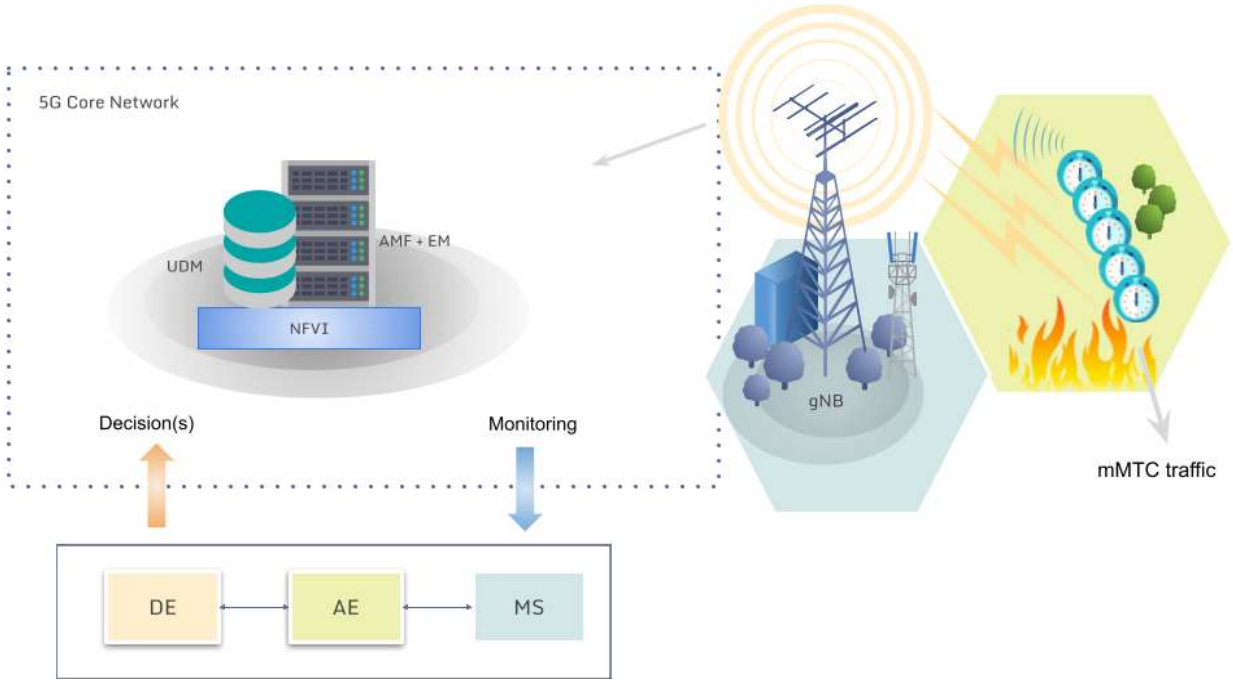


Figure 23: Interaction of the mMTC network slice and the closed-control loop to detect and mitigate DDOS attacks.

We have used a 5G testbed deployed at EURECOM. The testbed has been developed and used in many 5G European projects such as 5GEve[3] and 5GDrones[4]. We have implemented the closed-control loop components (i.e., MS, AE, and DE) and an Element Manager (EM) on top of the AMF. The latter exposes API to 1/ MS to monitor the Attach Request message; 2/ DE to detach and blacklist UEs involved in an attack.

We measure the accuracy of the Gradient Boosting algorithm under both normal and abnormal traffic. On normal traffic, the accuracy denotes how often the system yields a detection rate of UEs that is greater than zero. This does not mean that these devices will get banned, but ideally, a value of 0 should be returned for all devices emitting normal traffic. To evaluate our model on normal traffic (True Positive (FP) = False Negative (FN) = 0), we generate data for 500 normal events and run our detection algorithm on each of them. We then counted the number of UEs for which we obtained a greater-than-zero detection rate versus the total number of UEs in all the events. We generate the event duration randomly (between 30 and 250 seconds). Regarding malicious traffic (True Negative (TN) = False Positive (FP) = 0), we also ran 500 tests, but this time, between 7 and 15 Attach Requests are received every 6 seconds, for a total duration that is random between 30 and 250 seconds.

Figure 24 allows visualizing the results for normal traffic. The points correspond to the event data, while the green line is the anomaly interval. If a point is outside the limit (in green), it was assumed as an attack. For

[3] Online: https://www.5g-eve.eu/, Available: May 2023.

[4] Online: https://5gdrones.eu/, Available: May 2023.

normal traffic, the accuracy is computed as 1 − FP/(TN+FP). Hence, the results show an Accuracy on normal traffic of 96.76%. We expected this result as the interval used in our training data includes around 95% of the data in the training dataset, as depicted in Figure 24.

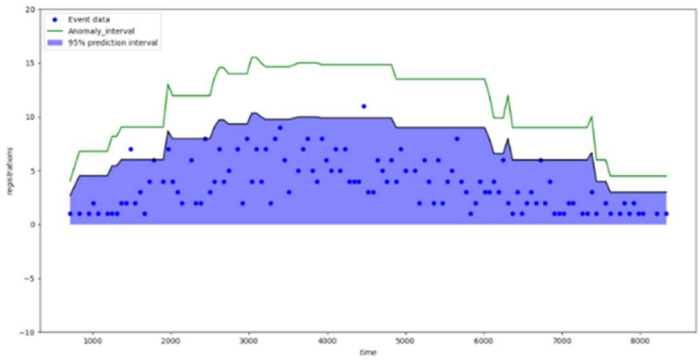


Figure 24: Results of the detection algorithm over normal traffic.

Figure 25 illustrates the results for malicious attacks. For this FN case, the accuracy is computed as 1− FN/(FN+TP). Hence, the results show an accuracy of 83.63%. This represents an excellent result as banning a relevant part of the devices taking part in a DDoS attack is enough to mitigate it. We recall that this is just the detection rate calculated by the AE component, the final decision regarding the devices that should be banned is taken by the DE component.

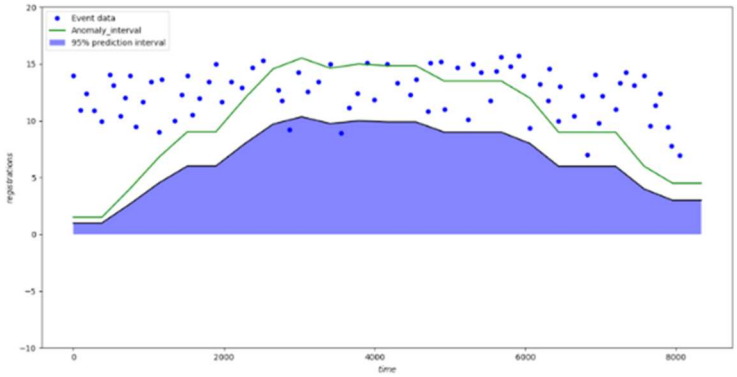


Figure 25: Results of the detection algorithm over abnormal traffic.

For the sake of comparison, we used the same scenarios as for Gradient Boosting to generate normal and abnormal traffic. Then, we applied the statistical method and verified its accuracy in detecting attacks. Figure 26 illustrates the usage of the statistical method in case of an abnormal event. The discontinue green line shows the β(3, 4) curve obtained according to the event duration. The β(3, 4) curve allows us to have a limited path from which all the outside points are considered anomalies, hence potential attacks. The statistical method's results show that 36.0% of the Attach requests are not following the β(3,4) distribution (they are out of the limit path). Therefore, they can be considered potential attacks.

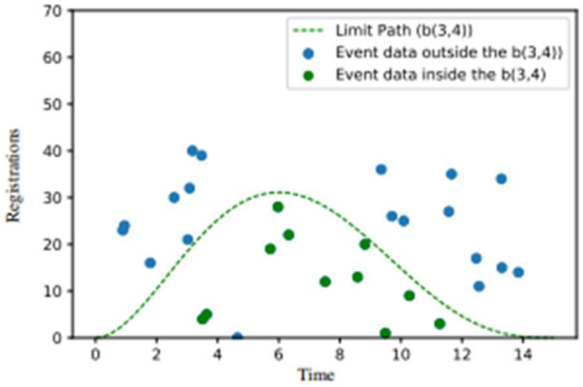


Figure 26: The result of the statics method over abnormal traffic

On the other hand, Figure 27 presents a test of a normal event. The results show that 90.0% of the Attach Requests follow the β(3,4) distribution. The accuracy of the statistical method to detect anomaly are : ((1 − FP/(TN+FP) = 84.21 % (normal traffic), 1 – FN/(FN+TP) = 57.14 % (abnormal traffic))). We remark that these values are weaker than the ones obtained with the Gradient Boosting algorithm. We argue these differences by the fact that the duration estimation has a strong impact on the statistical solution. The shape of the β(3, 4) curve changes drastically according to the duration (noted D), which seriously impacts the accuracy. For instance, we change the duration by +/- ϵ = 2sec, and the obtained results are summarized in Table 5. We see clearly from this table the impact of the duration on the accuracy as a small error on the duration drastically yields a drop in the accuracy. Particularly, if the duration is less than the real one, many points will be out of the curve. In the gradient Boosting algorithm, we do not have this concern, as the latter normalizes the sample period duration and uses the trained model to detect the interval.

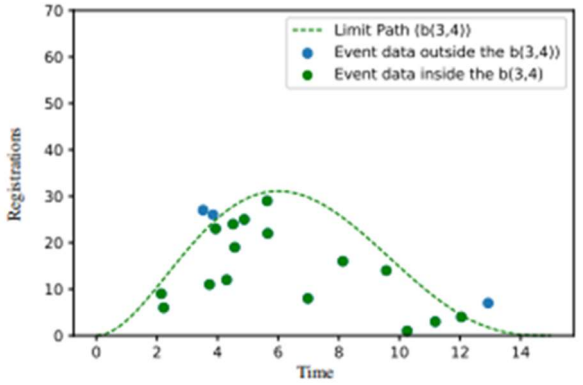


Figure 27: The result of the statics method over normal traffic

Table 5: The accuracy of the statistical model considering different Duration values

Method	GB	Static		
		D - Δt	D	D + Δt
Normal traffic	96.76859	45.18924	84.21052	80.24568
Abnormal traffic	83.633191	30.4156	57.142857	51.86854

Table 6 shows the performance of the Gradient Boosting-based solution when modifying the AE_DETECTION_THRESHOLD value. It is worth recalling that this value is used to derive the detection rate and corresponds to a protecting gap to reduce the impact of the ML prediction error and hence reduce the FP value. We remark that the value allowing to reduce both FP and FN is 2.0. Also, when the AE_DETECTION_THRESHOLD value increases, FP is reduced as the FN increases, whereas when it is reduced, both FP and FN increase.

Table 6: Impact of the AE Detection threshold.

AE_DETECTION_THR.	1.2	2.0	3.0
Normal event	82.98654	96.76859	97.64853
Abnormal event	92.92134	97.64643	75.7584

Finally, the link for the video of Experiment Scenario 2.1 can be found here in MonB5G YouTube channel[5].

7.2 Experimental Scenario 2.2: Robustness of learning algorithms in FL Poisoning Attack

This scenario first assesses the vulnerability of the distributed learning algorithms to adversarial attacks and how their performance metrics (e.g., precision, accuracy, etc.) could be affected by those attacks. Then, the model robustness is improved accordingly, and the new resulting weights and configurations is propagated to the decentralized elements (DE) of the proposed hierarchical architecture of MonB5G. We assume that each decentralized element is operating a local learning module as part of a federated learning algorithm while the centralized element, through its component AE, is serving as a sink for data submission (model updates). In this scenario, MonB5G framework has been evaluated against adversarial attacks, such as poisoning attacks that target training phase. In this use case, poisoning adversarial attack is considered where malicious agent can poison some fraction of the data in order to ensure that the learned model satisfies some adversarial goal. In addition, MonB5G has adopted another variant of poisoning attack where an agent aims to poison the parameter updates (of the federated learning algorithm) to be sent to the centralized element.

[5] MonB5G PoC2- Scenario 1&2: mMTC attack & Federated Learning attacks: https://www.youtube.com/watch?v=rjG6VXFPsEQ

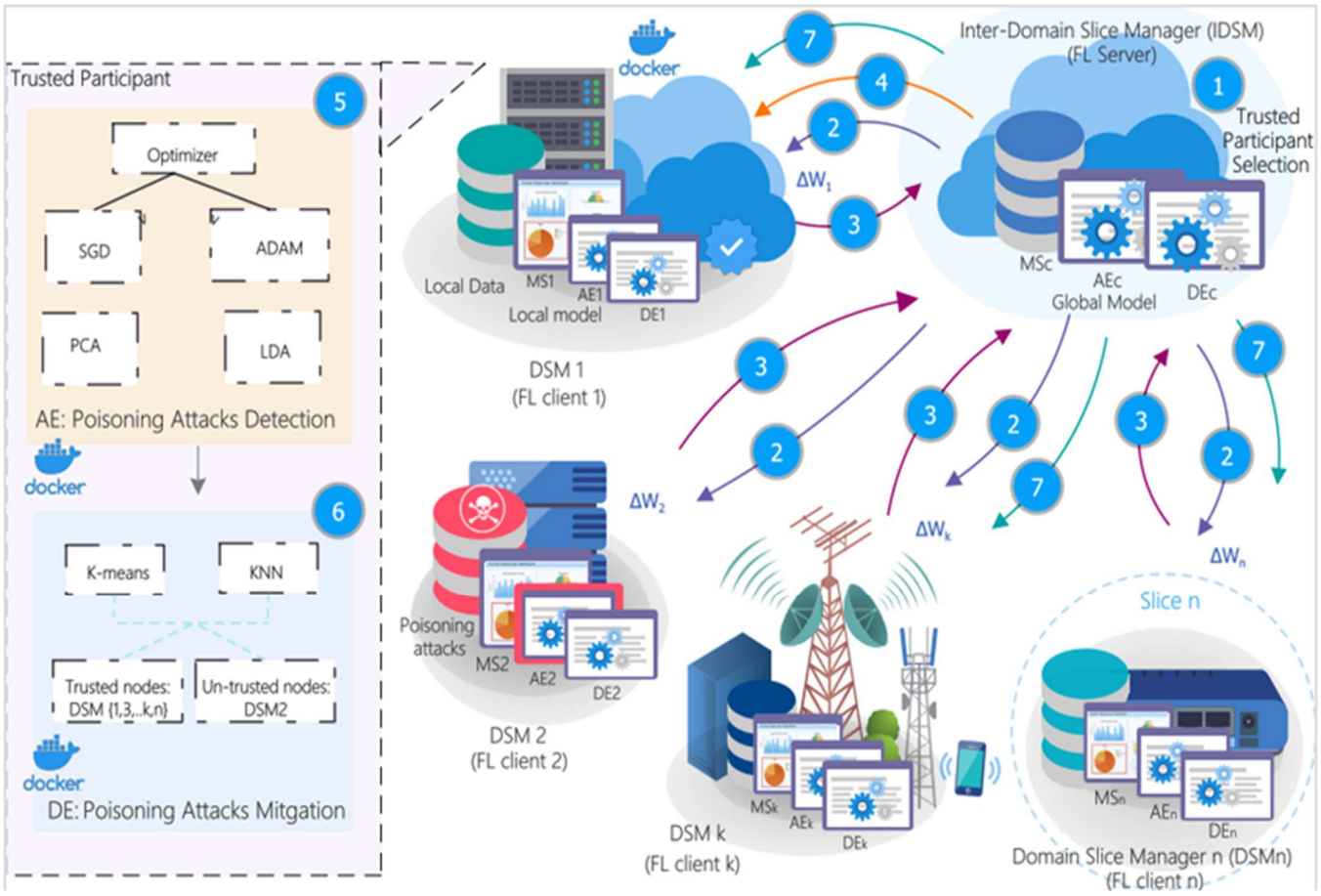
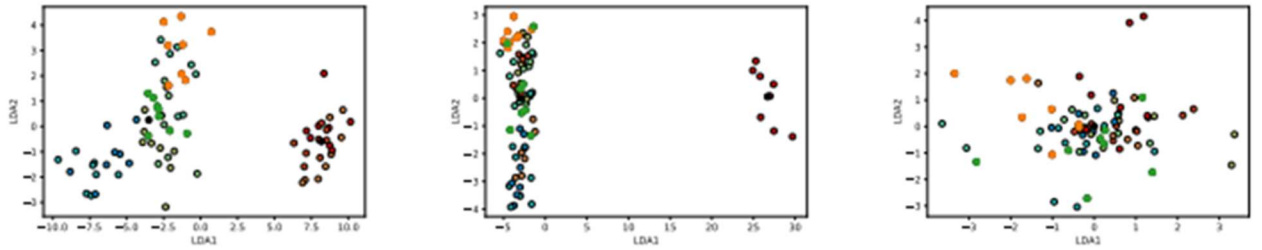


Figure 28: Overview of TQFL Framework.

As depicted in Figure 28, we consider n running network slices that may be initiated by different vertical industries, such as intelligent transportation, Industrial IoT, and eHealth verticals. The running network slices are interconnected to an Inter-Domain Slice Manager (IDSM), which is in charge for the management and orchestration of network slices. To enable ZSM, the IDSM side includes an AE for building learning models and a DE, to make suitable decisions based on AE's outputs. On the other side, each running network slice is managed locally by a Domain Slice Manager (DSM), which also includes a MS for monitoring data and in-slice traffic, and an AE for building learning models.

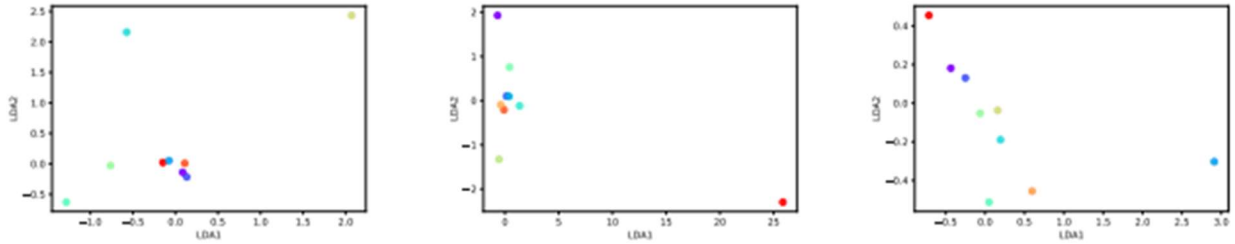
The proposed framework enables to secure federated learning in B5G networks, against poisoning attacks, named TQFL for "Trust deep Q-learning Federated Learning". The design of our framework comprises four main steps, starting from generating a dataset to designing a detection scheme of poisoning attacks: (i) The generation of a realistic dataset about the AMF function's latency of running network slices and its (latency) related parameters. (ii) Building a deep learning model to predict the AMF function's latency of each running network slice in a federated way in order to prevent any latency-related SLA violation. (iii) Building an online DRL model that dynamically selects a network slice as a trusted participant (see step 1). (iv) After the first FL rounds (see steps 2, 3), the trusted participant applies a dimensionality reduction scheme and unsupervised machine/deep learning to detect the malicious participant (s) (see steps 4, 5, 6, 7).

Figure 29 and Figure 32 depict the detection results when combining LDA and K-means on top of ADAM and SGD optimizers, respectively. We also vary the percentage of malicious network slices' DSM and show the trusted nodes that are selected at each FL round (nodes in green color). As we see, our scheme can clearly detect the malicious nodes, even with only one malicious node. Specifically, the trusted node applies both LDA and Kmeans, and then all nodes that are in the same cluster with it are considered correct models, while the nodes (models) that are in the other (s) cluster (s) will be considered as malicious. Therefore, our trust participant selection algorithm helps us not only to select a trusted node but also to determine malicious nodes when performing dimensionality reduction and unsupervised clustering. Moreover, determining the trusted cluster of nodes has also helped the FL server (IDSM) to select a trusted participant for the next FL round.



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

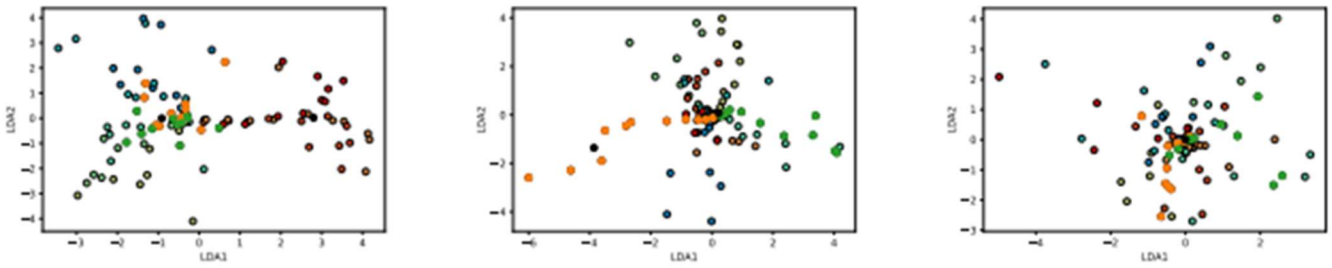
Figure 29: LDA + K-means for different number of malicious nodes (ADAM optimizer).



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

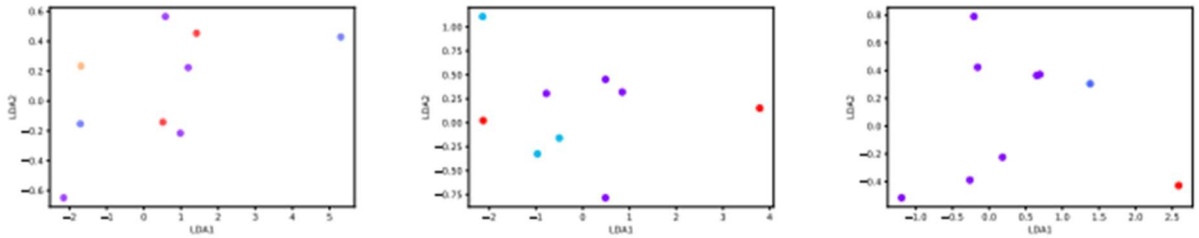
Figure 30: LDA + KNN for different number of malicious nodes (ADAM optimizer)

Figure 30 and Figure 31 also show the clustering of local models when applying both LDA and KNN on top of ADAM and SGD optimizers, respectively. Whatever the number of malicious nodes, we also observe that there are always some isolated points that represent the infected models sent by the malicious DSMs. However, for the LDA technique, we see that the isolated models (infected) are identified better with the ADAM optimizer than with the SGD optimizer (Figs. 6 and 7). Hence, the LDA (with KNN or k-means) technique gives better detection on top of the ADAM optimizer. In fact, these last combinations show the clearest clustering (two separate groups) compared to other algorithms.



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

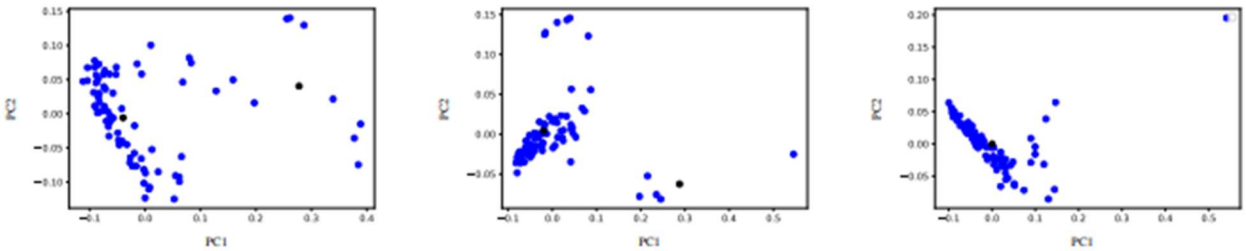
Figure 31: LDA + K-means for different number of malicious nodes (SGD optimizer).



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

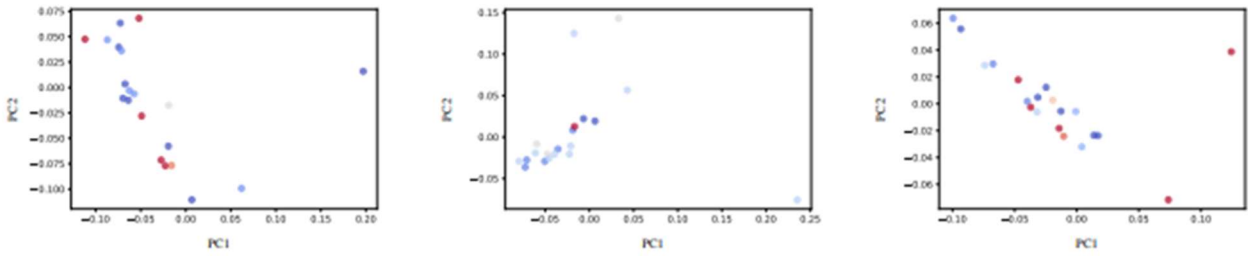
Figure 32: LDA + KNN for different number of malicious nodes (SGD optimizer).

As we did for LDA, we also evaluate the performance of Principal Component Analysis (PCA) technique when combined with clustering unsupervised algorithms. Figure 33 and Figure 35 shows the clustering detection when combining PCA with K-means, on top of the ADAM and SGD optimizers, respectively. We remark that both optimizers succeed in separating and identifying infected models by incorrect data. However, infected models are better identified on top of the SGD optimizer as compared to the ADAM optimizer. Thus, PCA with K-means gives better performance in detecting infected models on top of the SGD optimizer. Indeed, this last combination exhibits the clearest clustering (two distinct groups) compared to other algorithms.



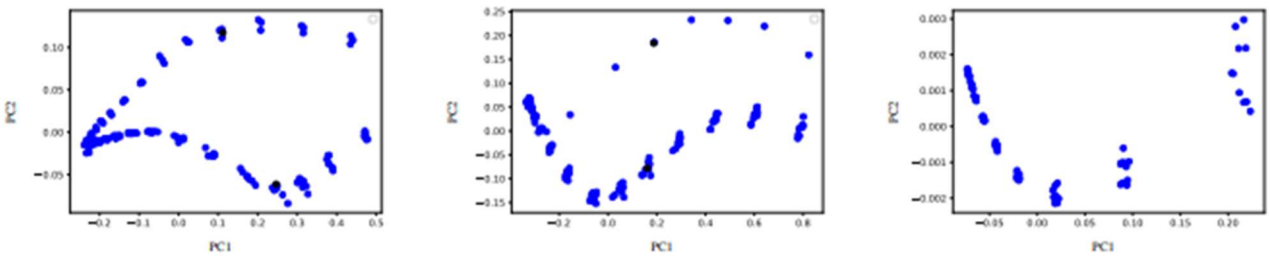
(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

Figure 33: PCA + K-means for different number of malicious nodes (ADAM optimizer).



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

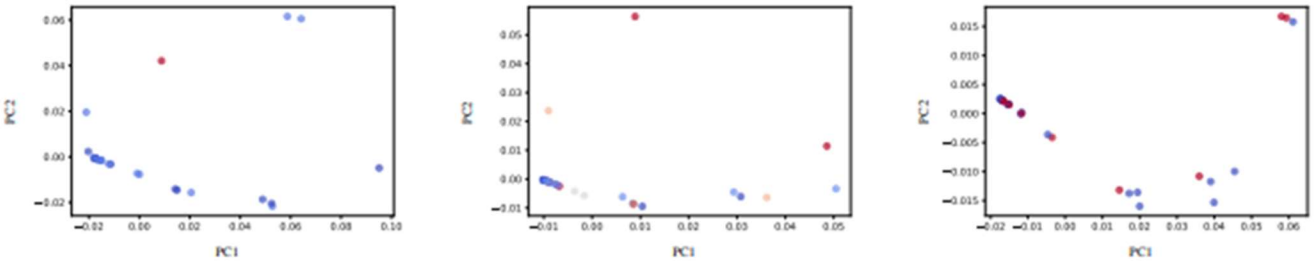
Figure 34: PCA + KNN for different number of malicious nodes (ADAM optimizer)



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

Figure 35: PCA + K-means for different number of malicious nodes (SGD optimizer).

Similarly, Figure 34 and Figure 36 depict the detection when combining PCA with KNN on top of the ADAM and SGD optimizers, respectively. PCA with KNN on top of both optimizers clearly separates correct local models from infected ones and thus enables detection/identification of malicious DSMs. We also see that infected models are better identified when leveraging the SGD optimizer than the ADAM optimizer. Therefore, the PCA technique with either K-means or KNN gives better detection of malicious models on top of the SGD optimizer, which is confirmed in Figure 33, Figure 34, Figure 35 and Figure 36. In fact, when compared to other techniques, these last combinations show the clearest clustering (two separate groups).



(a) Nb of malicious nodes = 3 (b) Nb of malicious nodes = 1 (c) Nb of malicious nodes = 0

Figure 36: PCA + KNN for different number of malicious nodes (SGD optimizer).

Finally, the link for the video of Experiment Scenario 2.2 can be found here in MonB5G YouTube channel[6].

7.3 Experimental Scenario 2.3: aLTEr Attack

The third experimental scenario is demonstrated through an attack called aLTEr attack. aLTEr attack is a MITM attack type and is carried out between the user equipment (UE) and the gNodeB (gNB). It consists of breaking layer two of the user radio bearer, exploiting the user data integrity protection can be missing as a vulnerability to carry out the attack. To conduct the detection and remediation of this attack, a 5G system in SA mode has been deployed on virtualized infrastructures, based on cloud native technology of Kubernetes (K8s) and virtual machines (VM), and different IA tools were used to detect attacks as shown in Figure 37. The mitigation step executed by DE consists of a security policy update on the firewall to deny the private DNS address from the UE. Metrics such as mean time to detect and mean time to react were considered to evaluate the performances of the solution.

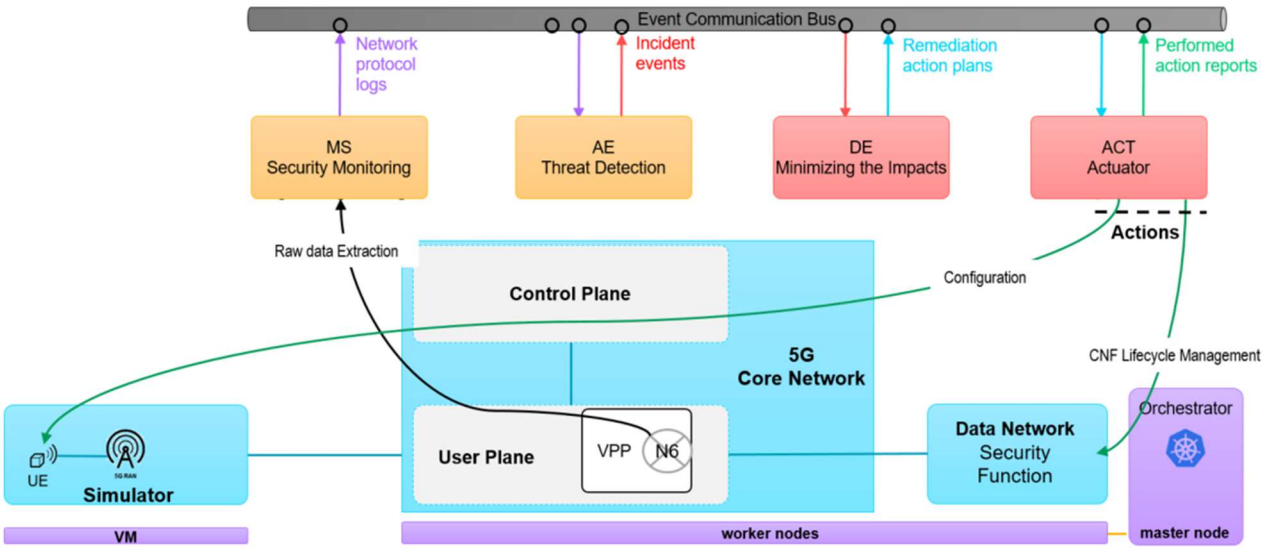


Figure 37: Interaction of the mMTC network slice and the closed-control loop to detect and mitigate DDOS attacks.

In this purpose, a dedicated platform has been built, including the incident response module as a security orchestrator, to detect and respond quickly to minimize the impact on the 5G network. Then a subset of the functions defined in the incident handling guidelines from (NIST.SP.800-61-r2) have been implemented to detect the security incident of an aLTEr attack and minimize its impacts. The implementation leverages the MonB5G components MS, AE, DE, ACT and its architecture to orchestrate security services.

[6] MonB5G PoC2- Scenario 1&2: mMTC attack & Federated Learning attacks: https://www.youtube.com/watch?v=rjG6VXFPsEQ

Such platform allows to run automatically and repeatedly the attack scenario 100 times, and times and logs are collected at the end to calculate metrics such as the Mean Time To Detect (MTTD) and the Mean Time to Respond (MTTR), as depicted in Figure 38.

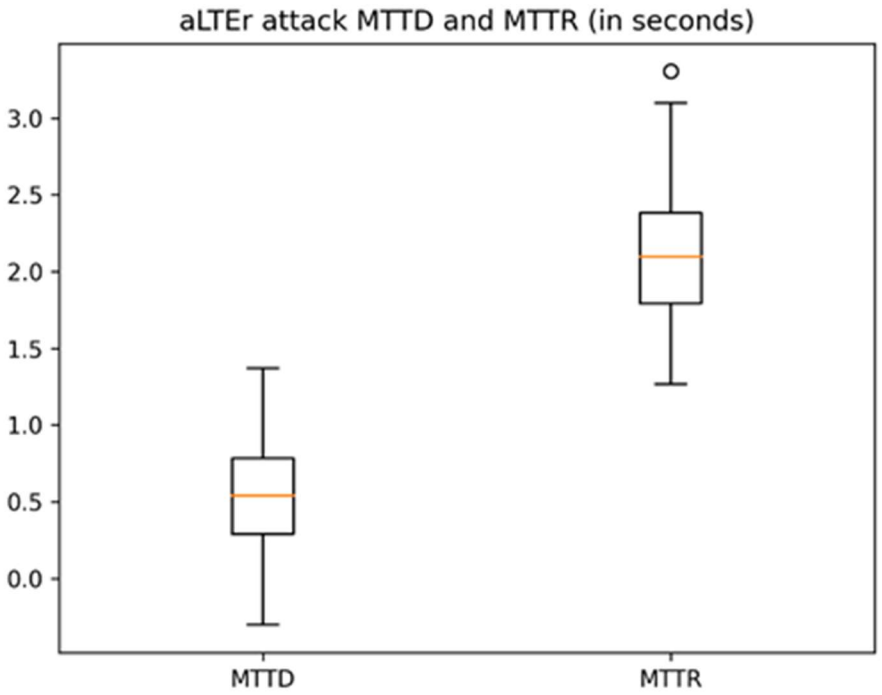


Figure 38: MTTD and MTTR measured over 100 attacks and mitigations.

The results in Figure 38 show the performance of the security orchestrator to detect and remediate aLTEr threat in order of seconds. Although the incident handling playbook suggested by the cybersecurity standards is partly implemented here, this use case demonstrates deploying automation with AI and security tools helps reduce the time to identify and eradicate threats. Moreover, the availability of exposed API to configure security and network functions, and to manage their lifecycle via the domain orchestrator such as Kubernetes master API widens the scope of possibilities to react to a threat, as a result, it enhances the dynamics of defense. All these time-saving advantages in responding to an attack come at a cost. Actually, automation involves the preparation of procedures and their integration into the protected systems and impact assessment on the services.

Finally, the link for the video of Experiment Scenario 2.3 can be found here in MonB5G YouTube channel[7].

[7] MonB5G PoC2- Scenario 3: aLTEr attack: https://www.youtube.com/watch?v=zvte425HeM4

8 Conclusions

In this deliverable we provided an overview of the MonB5G project has successfully capitalized on the power of AI to revolutionize the management and orchestration of B5G network resources. Over a duration of 42 months, the project brought together twelve organizations from eight European countries, with the support of EU funding through the Horizon 2020 framework. The project's focus was on developing AI-driven mechanisms for zero-touch management and orchestration of network slices, enabling smart, flexible, and automated resource allocation. By leveraging AI algorithms, the networks gained self-configuration, self-monitoring, self-healing, and self-optimization capabilities without the need for human intervention. This resulted in the seamless operation of demanding applications with high capacity and low latency requirements. The MonB5G system introduced a hierarchical, fault-tolerant, AI-based, and automated network management framework that incorporated security and energy efficiency techniques. The concept of network slicing, which involved dividing the virtual network infrastructure into customized and independent segments, played a pivotal role in efficiently serving diverse requirements. Each network slice, empowered by AI, ensured the delivery of reliable and high-quality services while enhancing infrastructure security and improving energy efficiency. The project successfully divided the centralized management system into multiple subsystems, employing optimal adaptive assignment of monitoring, analysis, and decision-making tasks across different domains. This approach enabled the realization of ZSM goals and facilitated network automation and service management goals.

The project's efforts culminated in the demonstration of two PoCs (PoC-1 and PoC-2) on the partners' 5G testbeds. PoC-1 showcased the MonB5G system's capabilities in zero-touch multi-domain service management and orchestration, exemplifying its ability to achieve end-to-end cross-domain SLAs without human intervention. Through automated service management, resource optimization, and anomaly detection, MonB5G demonstrated its potential to deliver stringent SLAs and ensure efficient service operation. In PoC-2, the project shifted its focus to the security aspects of the MonB5G system. The AI-assisted policy-driven security monitoring and enforcement mechanisms were developed to detect and mitigate attacks, ensuring network integrity and reliability. By effectively safeguarding network slices against security threats, MonB5G showcased its ability to deploy a secure and resilient security system. Throughout the project, several notable achievements were made. The MonB5G project developed AI-driven and decentralized management and orchestration architecture, paving the way for zero-touch service management. It introduced an AI-based closed control loop framework to detect and mitigate attacks on network slices, while exploring the concept of Security orchestration and Security as a Service. The project also defined novel end-to-end slice KPIs, employed graph-based learning and Federated Learning techniques, and implemented AI-based intra and inter-slice admission control mechanisms. Additionally, MonB5G focused on energy efficiency by developing decentralized cross-domain Energy Efficient Decision Engines and implementing energy-saving techniques at the RAN and Edge. By publishing 5G datasets collected from its testbeds, the MonB5G project contributed valuable resources to the research community, fostering future investigations and advancements in the field. MonB5G also contributed to relevant standard bodies (and groups therein) in ITU-T, 3GPP and ETSI and several dissemination and communication activities (journals, conferences, exhibitions, demos) which are detailed in [MONB5GD77].

9 REFERENCES

[Chergui2020] H. Chergui and C. Verikoukis, “Offline SLA-Constrained Deep Learning for 5G Networks Reliable and Dynamic End-to-End Slicing,” IEEE Journal on Selected Areas in Communications, vol. 38, no. 2, pp. 350-360, Feb 2020.

[Chergui2021] H. Chergui, L. Blanco and C. Verikoukis, “CDF-Aware Federated Learning for Low SLA Violations in Beyond 5G Network Slicing,” in IEEE ICC, 2021.

[Chergui2021TWC] Chergui, Hatim, Luis Blanco, and Christos Verikoukis. “Statistical federated learning for beyond 5G SLA-constrained RAN slicing.” IEEE Transactions on Wireless Communications 21.3 (2021): 2066-2076.

[MONB5GD24] Deliverable D2.4 Final release of the MonB5G architecture (including security), 2022.

[MONB5GD32] Deliverable D3.2 - Final Report on AI-driven Techniques for the MonB5G AE/MS, 2022.

[MONB5GD33] Deliverable D3.3 - Report on Integration and testing of the MonB5G AE and MS, 2023.

[MONB5GD41] Deliverable D4.1 - Initial report on AI driven techniques for the MonB5G DE, 2021.

[MONB5GD42] Deliverable D4.2 - Final report on AI driven techniques for the MonB5G DE, 2022.

[MONB5GD43] Deliverable D4.3 - Report on Integration and testing of the MonB5G DE, 2022.

[MONB5GD53] Deliverable D5.3 - Final report on AI driven MonB5G energy efficiency techniques, 2022.

[MonB5GD54] MonB5G Deliverable D5.4 “Report on implementation and testing of security and energy management techniques”, 2023.

[MONB5GD61] MonB5G Deliverable D6.1 - Technical Report on System Integration and Operation, 2023.

[MONB5GD77] MonB5G Deliverable D7.7 - Final Report on dissemination; standardization & exploitation plans, 2023.

[SCHE2MA] Dalgkitsis, Anestis, et al. "SCHE2MA: Scalable, Energy-Aware, Multidomain Orchestration for Beyond-5G URLLC Services." IEEE Transactions on Intelligent Transportation Systems (2022).