Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2024.053379

ARTICLE

Advancing 5G Network Applications Lifecycle Security: An ML-Driven Approach

Ana Hermosilla[1,2,*], Jorge Gallego-Madrid[1], Pedro Martinez-Julia[3], Jordi Ortiz[4], Ved P. Kafle[3] and Antonio Skarmeta[1,2]

[1]Information and Communications Engineering Department, University of Murcia, Murcia, 30100, Spain

[2]Research & Development Department, Odin Solutions S.L., Murcia, 30007, Spain

[3]Network Architecture Laboratory, Network Research Institute, National Institute of Information and Communications, Tokyo, 184-8795, Japan

[4]Engineering and Applied Techniques Department, University Center of Defense at the Spanish Air Force Academy, Murcia, 30729, Spain

*Corresponding Author: Ana Hermosilla. Email: ana.hermosilla@um.es

Received: 30 April 2024 Accepted: 18 July 2024 Published: 27 September 2024

ABSTRACT

As 5th Generation (5G) and Beyond 5G (B5G) networks become increasingly prevalent, ensuring not only network security but also the security and reliability of the applications, the so-called network applications, becomes of paramount importance. This paper introduces a novel integrated model architecture, combining a network application validation framework with an AI-driven reactive system to enhance security in real-time. The proposed model leverages machine learning (ML) and artificial intelligence (AI) to dynamically monitor and respond to security threats, effectively mitigating potential risks before they impact the network infrastructure. This dual approach not only validates the functionality and performance of network applications before their real deployment but also enhances the network's ability to adapt and respond to threats as they arise. The implementation of this model, in the shape of an architecture deployed in two distinct sites, demonstrates its practical viability and effectiveness. Integrating application validation with proactive threat detection and response, the proposed model addresses critical security challenges unique to 5G infrastructures. This paper details the model, architecture's design, implementation, and evaluation of this solution, illustrating its potential to improve network security management in 5G environments significantly. Our findings highlight the architecture's capability to ensure both the operational integrity of network applications and the security of the underlying infrastructure, presenting a significant advancement in network security.

KEYWORDS

Network application; network function virtualization; machine learning; security; 5G

1 Introduction

During the last decades, the widespread adoption of mobile technologies has considerably affected how people are connected, as Internet consumption shifted from wired devices to wireless and

portable terminals. In response, mobile technologies (defined by standardization institutions such as 3rd Generation Partnership Project (3GPP) [1] or European Telecommunications Standards Institute (ETSI) [2]) have advanced significantly to meet the demands and needs of the end users. However, as occurs with any technological advancement, every new development comes with additional risks and challenges, particularly in terms of security. This is especially relevant in 5G and Beyond 5G (B5G) networks since, unlike previous architectures, the network is available to third parties, who develop and deploy their applications over it. Therefore, there is a clear need for novel and more complete solutions to tackle such a challenge.

Furthermore, some enabler technologies for 5G and B5G networks such as Network Function Virtualization (NFV) or Software Defined Networks (SDN) also expose their own vulnerabilities. Decoupling the control plane from the data plane entails a new set of risks [3], as well as the dynamic deployment of resources on demand [4]. As networks grow in complexity, the surface where attacks can happen increases too, hence security actions and countermeasures need to cover the whole horizon of threats. In response to these challenges, Machine Learning and Artificial Intelligence (ML/AI) mechanisms are extensively utilized to detect security threats and devise countermeasures, as evidenced in [5]. However, securing such an infrastructure also requires ensuring that the applications deployed over the infrastructure are working correctly throughout their whole lifecycle. Consequently, methodologies and mechanisms are being designed and proposed to ensure the correctness of the network applications to be deployed over 5G and B5G networks [6,7]. In particular, the term network application refers to an application specifically designed and developed to be executed over a 5G or B5G network, as they aim at taking advantage of all the advances and benefits those networks offer, e.g., the optimization of 5G resources [8]. Moreover, it is expected that those advances will become widespread and globally used, not only for big telco companies but also for small developers. Thus, offering some tools to properly develop small developers' solutions becomes also crucial, as their applications must be secure enough not to result in a threat to the underlying shared infrastructure.

In this way, this work proposes a model that offers a comprehensive solution: it not only automatically validates the operation of network applications upon deployment but also responds in real time to existing threats by effectively deploying countermeasures. In this way, the network application behavior and performance are evaluated, and also how it affects the networks where it is instantiated, as well as how it reacts when certain countermeasures are needed and enforced. To present this model, its components and functions must be clearly defined, accompanied by a High-Level Architecture representing it. Subsequently, for model validation, an implementation of the architecture is performed, involving the selection and integration of existing tools that execute the model's functionality. To the authors' knowledge, this is the first solution that simultaneously addresses both the validation and security of applications, while also possessing the capability to detect threats and compute countermeasures.

The rest of the paper is organized as follows. Section 2 provides an overview of existing solutions that address security in state-of-the-art 5G infrastructures. Section 3 details the model that represents the proposed solution. Section 4 defines a High-Level Architecture representing the aforementioned model, while Section 5 presents how to instantiate that architecture and the mechanisms involved. Section 6 presents the instantiation of the architecture as well as a set of experiments conducted to demonstrate the validity of the solution. Finally, Section 7 draws the conclusions and presents the future lines of work.

2 Background

The introduction of NFV is one of the main enablers of 5G and B5G, as it allows the dynamic deployment of the required components in the required location [9]. In an NFV infrastructure, Virtualized Network Functions (VNFs) are deployed on commodity hardware and connected through a virtualized network, providing greater flexibility and scalability than traditional network architectures. However, it also introduces security challenges, since the number of vulnerable elements of the architecture increases and the attack surface is expanded [10]. This is because a) having multiple instances running dynamically on the same hypervisor entails certain security risks, as they share the same physical resources; and b) having new entities in the architecture, e.g., the NFV-Orchestrator, implies that those entities must also be protected. Furthermore, these security challenges can appear due to the malfunction of a certain instance, e.g., being attacked, or because of an external attack focused on the architecture itself [11]. One potential way to enhance the security related to these issues in an NFV infrastructure is to use reactive AI-powered solutions [12] that can detect and respond to those security threats in real time, as shown in [5]. Concretely, AI-powered solutions increase awareness of the system since they enable the real-time analysis of data from different sources, allowing a substantial enhancement in the decision-making process. Thus, the system can better and faster detect threads and react more efficiently. A clear example of such a solution is shown in [13], where authors present an autonomic resource control architecture to allow fast detection and adaptation to workload changes in an automatized way, supporting multiple administrative domains and with integrated ML and AI mechanisms at its core.

In this line, the integration of ML/AI mechanisms to enhance infrastructures' security has been extensively studied in recent scientific literature, as shown in [14,15]. For example, authors in [16] integrated autonomous security by means of E2E slicing management, continuously monitoring the security requirements. Also, reference [17] presented a model-driven federated learning approach for managing and orchestrating the system in order to detect and prevent cyber-attacks, integrating a B5G security framework by means of a multi-domain and multi-tenant orchestrator. In [18], a comprehensive AI-driven framework is presented, based on a Network Performance Management that automated both fault detection and root-cause analysis. In [19], a cybersecurity reaction methodology based on Artificial Immune Systems was presented, adapting an evolutionary computing paradigm where the countermeasures are tailored to the level of risk faced by the assets of the protected system. In the field of autonomous systems, authors in [20] enhanced the architecture with ML native optimizations, to improve the orchestration, the efficiency of the network as well as the security, although focusing on Network Slices rather than on the infrastructure itself. Also in this line, authors in [21] developed a reactive zero-touch approach for NFV infrastructures, with security services that continuously monitor traffic in order to detect possible network vulnerabilities and apply countermeasures to mitigate the possible threats. In [22], authors presented the INSPIRE-5Gplus architecture, a security architecture designed to enable automated and smart security management of B5G by means of enforcing Security Service Level Agreements and ensuring Quality of Service. In [23], a novel system architecture and simulation model for machine learning orchestration in cloud environments was proposed, aiming at automating the training and deployment of using Distributed Machine Learning (DML) AI model. And finally, authors in [24] presented the Hexa-X project, applying AI/ML methods to networks, with the aim to improve the efficiency of the operation of 6G networks, by means of introducing data-driven approaches into their Management and Orchestration.

However, none of the above solutions consider validating the applications to be deployed, as they only focus on the infrastructure or the network *per se*, and none of them consider the testing nor evaluation of the applications deployed over it.

In that sense, as the network applications' behavior must be reliable, ensuring they meet some basic requirements before their deployment over real-world infrastructure becomes of utmost importance. As previously mentioned, the concept of a network application refers to an application specifically designed and developed to be executed over a 5G or B5G network, with the aim of taking advantage of all the advances and benefits those networks offer [8]. In this way, if network applications are deployed in a controlled environment to be tested, and they pass some tests that prove those basic requirements, the infrastructure can ensure that they meet at least a certain standard of quality and will not be a potential danger. In that sense, the risk is detected before the execution of the network application can cause any damage. To solve such a challenge, authors in [7,25] tried to offer some mechanisms to ensure the integrity of the applications prior to their deployment over the shared infrastructure using Continuous Integration and Continuous Deployment (CI/CD) mechanisms [6]. Those CI/CD mechanisms are used to test the applications in an automatic manner and ease the adoption of these technologies by 5G newcomers. Nevertheless, the platforms proposed fail to address reactive security measures and attack detection mechanisms. Reactive AI and ML systems have been widely employed for network and network security automation, in particular in the area of threat detection. Nonetheless, detection after an infection or an attack occurs might be already late, thus mechanisms to ensure the behavior of the elements to be deployed over the network are envisioned as necessary. In this line, the project 5GEVE [26] had as its primary goal the creation of an architecture to support the definition, execution, and validation of experiments, focusing on the testing and validation of the applications deployed on an end-to-end 5G infrastructure. We can also find the project VITAL-5G [8], with the aim of showcasing the benefits of 5G-based network applications via real-life trials by means of an experimentation service portal to create, deploy, monitor and (re)configure the network applications. But, again, they focused on the experiments, leaving behind a reactive system to improve the security of the infrastructure.

Considering the works presented and to the best of the authors' knowledge, none of the explored solutions use ML or AI algorithms to improve infrastructure's security dynamically, in conjunction with validating the applications prior to their deployment. Therefore, we consider highly interesting explore an architecture integrating both solutions, enabling them to mutually enhance each other's capabilities.

3 Reactive Validation Model

The analysis of existing works in Section 2 reveals that currently there are no integrated solutions that combine an ML-AI-enhanced reactive system with pre-deployment testing to ensure security over 5G infrastructures. That is, attempting to address the problem of security in 5G infrastructures from the point of view of both the infrastructure itself (in terms of monitoring it and enhancing security by deploying the required countermeasures, powered by ML/AI) and the liability of the applications that run on it. Therefore, this work aims to fill this gap by offering a model to be implemented by combining all these aspects. In that sense, the proposed model includes a Reactive System, working in conjunction with a network application Validator (NAV). The first entity is able to dynamically deploy countermeasures depending on what is considered most appropriate at the time, making such decisions with the help of ML algorithms and AI. The second entity is capable of evaluating the network applications before their deployment in the production environment. By combining the capabilities of both systems, it is possible to address the complexity of handling security in 5G infrastructures from a wider perspective, on one hand, the monitoring and dynamic deployment of countermeasures when required, and on the other hand, ensuring the correctness of the applications to be deployed. It is noteworthy that the model can perform both tasks automatedly. In this way, it is feasible to

evaluate if the application misbehaves, e.g., it saturates the network, it conflicts with other applications, even to see how it responds to a simulated attack, or any other performance or behavioral evaluation considered and aligned with the specific application at stake. Furthermore, by having a reactive system with monitoring capabilities monitoring the infrastructure, it is possible to observe how the application affects negatively (or not) the platform. Additionally, the system also evaluates the effectiveness of the executed countermeasures on the application.

3.1 Network Application Validator

Concerning NAV, this entity is in charge of a) instantiating the network applications to be tested, b) executing a set of relevant tests over them, and c) generating a report of the adequacy of the applications. The instantiation is delegated to an NFV-Orchestrator (NFV-O), to be compliant with the ETSI NFV approach [27]. During the test execution phase, called *Validation Process*, the application is tested using predefined and app-driven tests. The first ones are generic tests which evaluate general performance indicators and side effects on the architecture, e.g., bandwidth, potential security breaches due to weak credentials, etc., meanwhile the app-driven ones are those tests that evaluate intrinsic performance and behavior indicators of the network application itself, e.g., correct operation of its Application Programming Interfaces (APIs). Using the results of those tests, NAV generates a report indicating the correctness (or not) of the network application. In this way, developers will know how their app performs in a real 5G scenario, while infrastructure owners can verify whether the application behaves as expected, and asses any potential risk or impact on the infrastructure.

The workflow of the NAV is shown in Fig. 1. Specifically, the network application, following the standards defined in [28–30], is packaged in the shape of NFV Descriptors, which contain all the necessary information to instantiate it. Using these descriptors, their instantiation would occur over a testing 5G platform, closely resembling the production 5G environment where it would be eventually located. Concerning the validation phase, a CI/CD engine would be used to test the applications in an automatic and autonomous manner. In that sense, we differentiate two main events during the NAV work, as shown in Fig. 2: on one hand, the instantiation of the network application over the 5G infrastructure, in charge of NFV-Orchestrator, and the execution of the tests, performed by the CI/CD engine.

In that sense, to validate the network application, the first step is to deploy it using the NFV-Orchestrator. Once is running over the 5G infrastructure, the CI/CD engine executes the tests over them, and finally, a report will be generated, including the tests' results. This process will be explained in more detail in Sections 3.3 and 4.1.1.

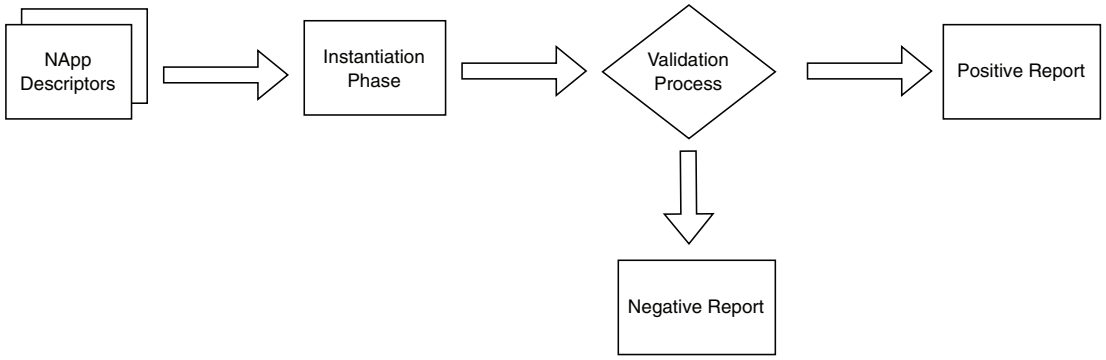

Figure 1: Network application validator flow

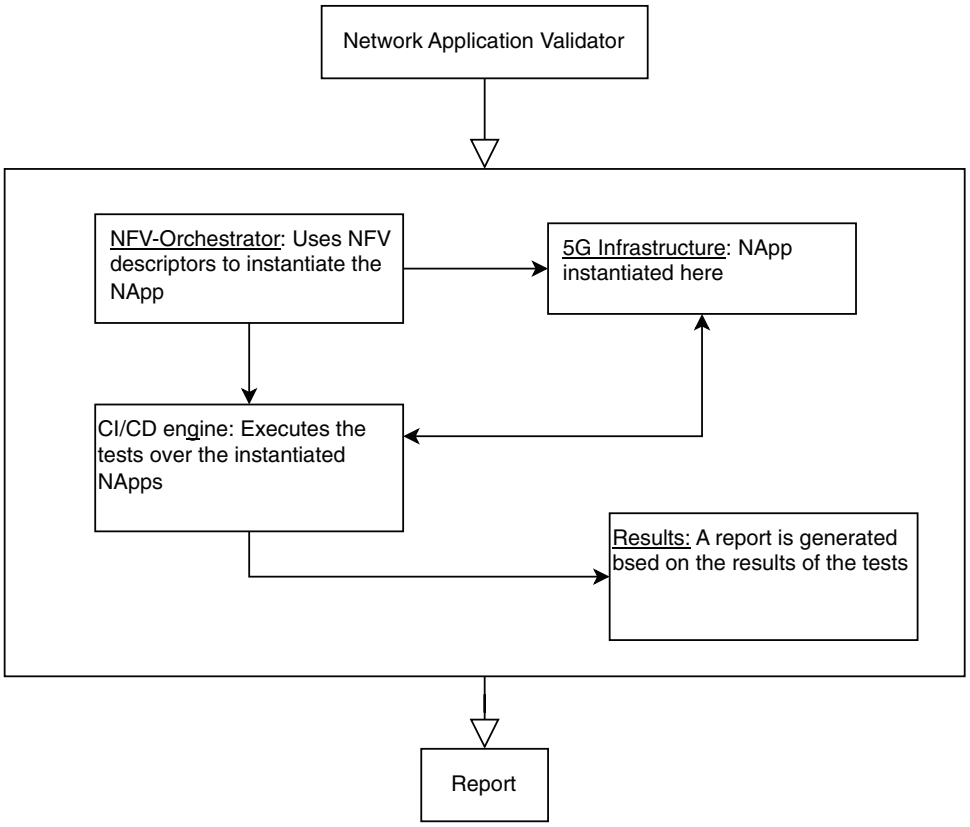

Figure 2: Network application validator components flow

3.1.1 Descriptors

As aforementioned, the network application is packaged in the shape of the so-called NFV Descriptors. However, these packages are limited in scope to the instantiation of the network application. To ease and automatize the testing process, a Test Descriptor Template has been designed, to include the information required to launch the related tests. This descriptor references both the general and app-driven tests and its format and content is presented in Listing 1:

Listing 1: Descriptor for test information and phases

```
- test_info: # Test general info and relations w/NApp and testbed
    {id, NApp_id, testbed, description}
- test_phases: # Test definition in three test_phases
    - setup:
        - testcases:
            {id, type, scope, name, parameters}
    - execution:
        {batch_id, executions[id, name, instances]}
    - validation:
        {id, execution_id, file_report}
```

The introduction of this descriptor is highly interesting, as enables the Network Application Validator to dispose of the tests in a similar format to the NFV Descriptors, thus allowing test automation. A detailed example of a fulfilled Test Descriptor Template with real data can be found in [31].

3.2 Reactive System

The Reactive System is the entity responsible for continuously monitoring the infrastructure and maintaining the security in a dynamic and reactive manner. To do so, some monitoring agents are deployed over the infrastructure, gathering data from the network and from other relevant components, i.e., the nodes where the applications are instantiated. This data is analyzed by Machine Learning (ML) and Artificial Intelligence (IA) mechanisms that enable real-time detection and response to security threats. The system actively computes countermeasures based on a continuously updated knowledge base and real-time monitoring data. Consequently, it can identify potential threats and deploy appropriate countermeasures effectively. To perform this, the proposed model follows a closed-loop architecture, as it works autonomously with no human intervention. The Reactive System flow is shown in Fig. 3, and it involves the following components:

- An Aggregator, which retrieves the telemetry data gathered by the monitoring agents deployed over the infrastructure.
- An Analyzer, which processes the data gathered from the Aggregator to identify security events. It uses a complex event processor, powered by AI/ML, to identify those events in the monitored system by processing data streams., Furthermore, the system enhances its accuracy by continually re-learning the relationship between the monitoring data received and the evolving states of the system, as this information is introduced in its knowledge base.
- A Decision Engine, which is notified when the Analyzer detects an event. It identifies the necessary actions for addressing the event, pairing each detected event with appropriate solutions, and using a Case-Based Reasoner (CBR) to match the events with their corresponding countermeasures. To do so, the CBR has been trained to learn the correlation between different events and their potential countermeasures so that when a new event occurs, it suggests suitable countermeasures. Then, those countermeasures are executed.

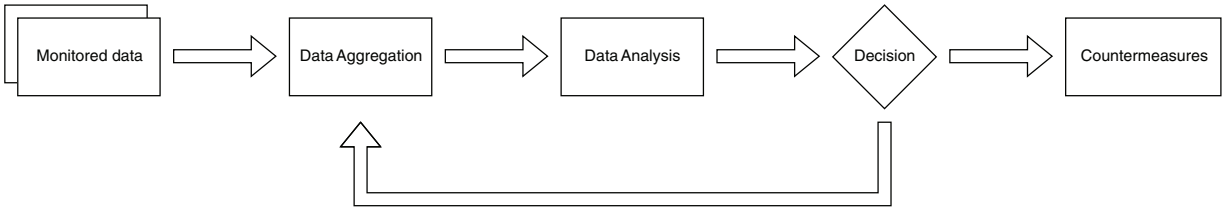

Figure 3: Reactive system flow

After the execution (also called enforcement) of the countermeasures, the whole process starts again, checking whether the problem has been solved or otherwise obtaining new countermeasures and updating its knowledge.

3.3 Joint Process

Once both NAV and Reactive System processes have been presented together with their main advantages, we proceed to explain how they work together, benefiting from their synergies. On one

hand, the validation process is significantly enhanced. This enhancement occurs as it does not only rely on the test executions to validate the application but also incorporates monitoring data, and considers whether the Decision Engine has deemed it necessary to implement any countermeasures. On the other hand, the Reactive System broadens its knowledge base by including test results in its decision-making process. Consequently, it can make more informed decisions, benefiting not only the initial application instantiation but also subsequent ones. This continual learning and feedback enrich the Reactive System's overall effectiveness.

Concerning the joint flow, it starts when the network application is instantiated. Since this model aims to be NFV-aligned, the instantiation is managed via the NFV-Orchestrator. However, it is worth noting that the model is designed to be methodologically agnostic and compatible with different instantiation methods. Once the network application is live, the suite of tests included in the descriptors is performed. It must be noted that during this process, the Reactive System is gathering monitoring data from the testing environment and providing it to the Analyzer that feeds the Decision Engine. Besides, if the Analyzer detects an event, the Decision Engine will be automatically notified to determine if countermeasures are needed. The Decision Engine acts reactively but generates a knowledge base proactively employing all the data extracted from the testing and evaluation phase, selecting the countermeasures that can be potentially applied to the network application based on its behavior (in terms of computational resources) and network patterns. Once the test execution is finished, the results are parsed by the Analyzer and sent to the Decision Engine who, in conjunction with the gathered monitoring data, will assess whether it is necessary to instantiate any countermeasures. If a flaw is detected in a network application, such as a DDoS attack being carried out, the Decision Engine selects and deploys the corresponding countermeasures, the final countermeasure rolling back the network application to a well-known and unaffected version of the network application itself. If no countermeasures are required, the final report with test results is delivered. Otherwise, they will be enforced, and the flow will start again. Fig. 4 illustrates the flow and relationships between the different phases. The logic underpinning this joint flow is detailed in Algorithm 3, meanwhile, Algorithms 1 and 2 describe the internal logic of the Reactive System and the NAV functions, respectively. The names of the data elements are listed in Table 1.

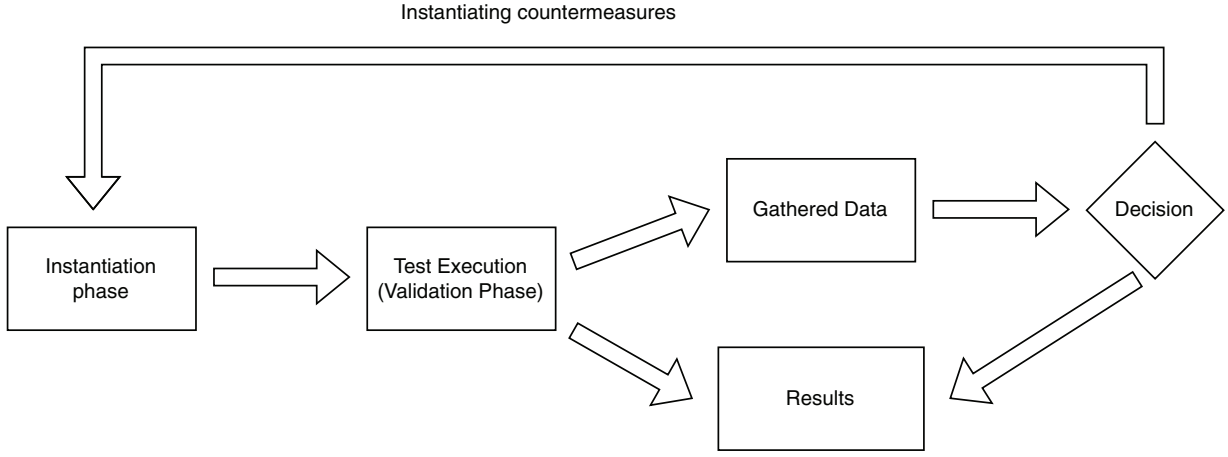

Figure 4: Joint flow including the reactive system and the NAV

Table 1: Data name fields

Field	Definition
A	Aggregator
CBR	Case-base reasoner
CM	Countermeasure
D	Gathered data
DE	Decision engine
DS	Descriptors
I	Instances
M	Match
MA	Monitoring agents
R	Results
R'	Subset of results ∈ DS
T	Tests
tx	Executing tests

After the execution (also called enforcement) of the countermeasures, the whole process starts again, checking whether the problem has been solved or otherwise obtaining new countermeasures and updating its knowledge.

4 Model High-Level Architecture

To be able to validate the proposed model, a high-level architecture including both NAV and Reactive System has been defined. This architecture supports the joint process defined in Section 3.3. Besides, it also includes an NFV-Orchestrator, the 5G infrastructure to be monitored, and the monitoring agents, which together enable the dynamic and on-demand deployment of the network applications to be tested and fulfill the Reactive requirements.

Algorithm 1: Reactive system functions

1: **function** MONITORING_LOOP(MA, N, A)
2: $\sum_{ma \in MA, n \in N}$[data = $ma.monitoring(n)$; $ma.send\_data(A, \text{data})$]
3: **function** REALTIME_ANALYZER(D)
4: $D' =$ EVENT_PROCESSOR(D)
5: **if** $D' \neq \emptyset$ **then**
6: send(D', DE)
7: **function** TESTRESULTS_ANALYZER(R)
8: $R' =$ EVENT_PROCESSOR (R)
9: send (R', DE)
10: **function** EVENT_PROCESSOR(D)
11: $D' = \emptyset$
12: $\sum_{d \in D}$[if ($d \in A \cap T$) then $D' = D' \cup \{d\}$]
13: **return** D'

(Continued)

Algorithm 1 (continued)

function DECISION_ENGINE(D', CBR)
  $CM = \emptyset$
  $\sum_{d' \in D'}[CM = CM \cup \text{MATCH}\ (d', CBR)]$
  if $CM \neq \emptyset$ **then**
    enforce(CM)
function MATCH(d', CBR)
  $CM = \emptyset$
  $match = \text{search}(D', CBR)$
  if $match \neq \emptyset$ **then**
    $CM = CM \cup \{match.countermeasure\}$
  return CM

Algorithm 2: NAV functions

function INSTANTIATE_APP(DS)
  $\sum_{ds \in DS}[a = \text{instantiate}(ds)]$
function EXECUTE_TESTS(T, DS, A)
  $T' = \{t \in T \mid t \in DS \wedge t \in A\}$
  $tx = \text{instantiate}(T')$
  $R = \text{execute}(tx)$
  return R

Algorithm 3: Joint system flow

function JOINT_FLOW(MA, N, A, DS)
  while true **do**
    MONITORING_LOOP(MA, N, A)
    $\sum_{a \in A}[$
      INSTANTIATE_APP(DS)
      $R = \text{Execute\_Tests}(T, DS, a)$
      $R' = \text{Testresults\_Analyzer}(R)$
      $CM = \text{Decision\_Engine}(R', DE)$
      if $(CM \neq \emptyset)$ **then** enforce(CM)
    ]

Regarding the communications between the different modules, we must differentiate between NFV instantiation, Monitoring feedback, and Network Application Validator communications. The first ones are those in which an entity (e.g., the Network Application Validator or the Reactive System) requests the NFV-Orchestrator to deploy either an application or a countermeasure. Monitoring feedback communications are those related to the Reactive System communications with both the monitoring agents deployed on the infrastructure and the countermeasures. Network Application Validator communications are those communications required for the testing process, e.g., execution of application tests or application stress tests, among others. Also, the Inter-Reactive and Validation systems communication (also called east-west communications) between the Network Application Validator and the Reactive System are present to coordinate the complete process while the application is being evaluated. Following the SDN philosophy, the management/control plane is differentiated

from the data plane and, therefore, each component communicates employing each plane depending on its role and the communication peer. All these communications are represented in Fig. 5.

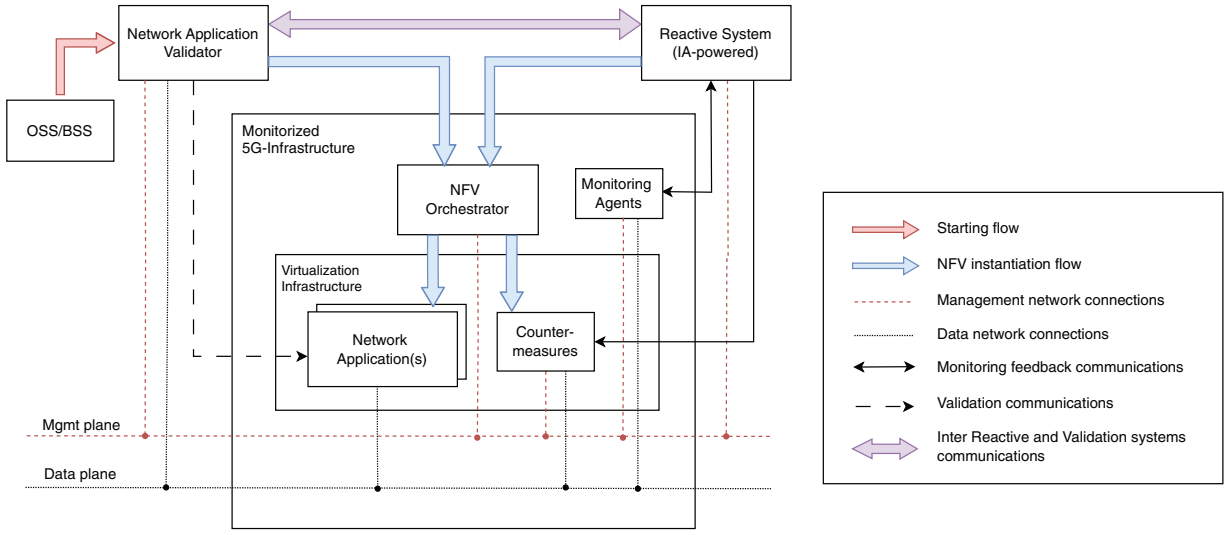

Figure 5: High-level architecture of the proposed model

The communication flow begins when the Operations & Business Support Systems (OSS/BSS) (in a telecom architecture, the entity that initiates the flow; in this scenario, it would be usually the administrator) decides to evaluate an application and requests it to the Network Application Validator. At that point, the Network Application Validator requests the NFV-Orchestrator the deployment of the application and then notifies the Reactive System that the application is going to be evaluated. Depending on the evaluation to be performed and/or the rules for that type of application available to the Reactive System, it will perform different enforcements, for example, evaluating how the instantiation and execution of the application affect the network.

4.1 Components Design of the Proposed Model Instantiation

As previously presented, the proposed model heavily relies on two primary elements, i.e., the Reactive System and the Network Application Validator. In order to validate such a model, this work implements the high-level architecture defined in the previous section using already existing tools or frameworks such as Reactive System and NAV. Among the different frameworks already available and analyzed in Section 2, the Autonomic Resource Control Architecture (ARCA) [32] covers the characteristics to be employed as the Reactive System, as will be further explained in Section 4.1.2. Regarding NAV, the 5GASP platform [33] suits the requirements and offers the characteristics of NAV. The details on the feasibility of 5GASP as a Network Application Validator are detailed in Section 4.1.1.

4.1.1 5GASP Platform as Network Application Validator

Concerning the tool to be used as an implementation of the Network Application Validator, the authors decided to use a reduced version of the 5GASP project platform. The principal motivation for this decision is that it incorporates Network Application testing by design, performed through a CI/CD pipeline, as well as it already considers the inclusion of an NFV-Orchestrator. Additionally, it also includes a component OpenSlice [34] which integrates all the required functionalities effectively.

One of the most important components of this platform is the Network Onboarding and Deployment Services (NODS), as it acts as the main orchestrator of the whole platform. This component receives the information required to deploy the network application, the requests to deploy them, the location where they must be deployed, and the test suites to be executed over them. Besides, it communicates and coordinates the low-level NFV-Orchestrators present in every testbed, as well as the CI/CD manager and agents, and it also coordinates the test pipeline of the network applications, generating a report about the results they obtained after the execution of the tests. This relation is shown in Fig. 6. It should be noted that, in this scenario, only a single site is shown. Nevertheless, the platform is prepared for multi-domain deployments, in which centralized NODS and CI/CD Manager are located, and an NFV-Orchestrator, a CI/CD Agent, and one or multiple Virtualization Infrastructure per site.

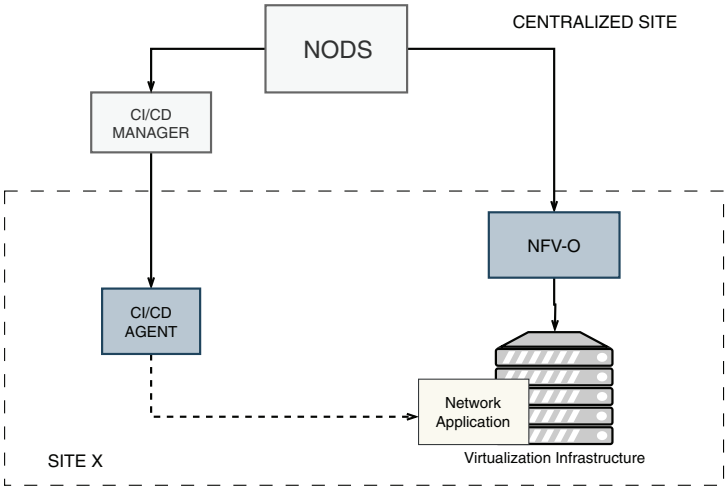

Figure 6: Abstract view of the 5GASP deployment platform

The platform also includes the resources to automatically test and validate the network applications in a controlled environment, to easily monitor them and therefore decide if their behavior is correct or not. To achieve such an ambitious objective, a series of tests are executed, obtaining as a final value some Key Performance Indicators (KPIs) that indicate the compliance of the tested applications. To do so, 5GASP offers a CI/CD pipeline, based on both a CI/CD manager and an agent, in conjunction with a suite of predefined tests.

The CI/CD pipeline is triggered by the NODS, meanwhile the CI/CD Manager is the central entity that coordinates the validation jobs. In this example, only one CI/CD Agent appears, but in multi-domain scenarios all of them will have their own CI/CD Agent. When the execution of the tests has finished, the CI/CD Agent creates a test report including the results that is forwarded to the CI/CD Manager and, finally, to the NODS [31] Besides the validation of the application itself, this platform is also based on the idea of a security methodology to guarantee the correctness of all the steps involved in the development and execution of the applications. In that sense, it covers the full development pipeline, from the design phase of the network application to the creation and execution of the validation tests.

4.1.2 ARCA as Reactive System

Although there are existent solutions that implement the functionality required for the Reactive System previously defined, NICT's ARCA has been selected to implement it. The main reason behind this decision is the fact that ARCA already integrates ML and AI mechanisms in its core,

thus simplifying the instantiation process. In addition, performance evaluations have already been conducted [32], meaning that its maturity level is already one step forward from the alternatives. Moreover, it is NFV compliant, and it has been designed to be distributed and compatible with multiple administrative domains.

ARCA's components are depicted in Fig. 7. Some of them are analogous to the proposed Reactive System in Section 3.2, as most autonomous system solutions are based on the Observe, Orient, Decide and Act/Monitoring, Analysis, Planning and Execution (OODA/MAPE) loop [35]. For example, ARCA's collector is analogous to Reactive System's collector, as well as both analyzers. However, the functionality of Reactive System's Decision Engine is split in ARCA in the shape of a Decider and an Enforcer. This entity is in charge of deploying the selected countermeasures once analysis and decision have taken place, whereas in our proposed Reactive System, this functionality was included inside the Decision Engine itself, being delegated directly to the NFV-Orchestrator.

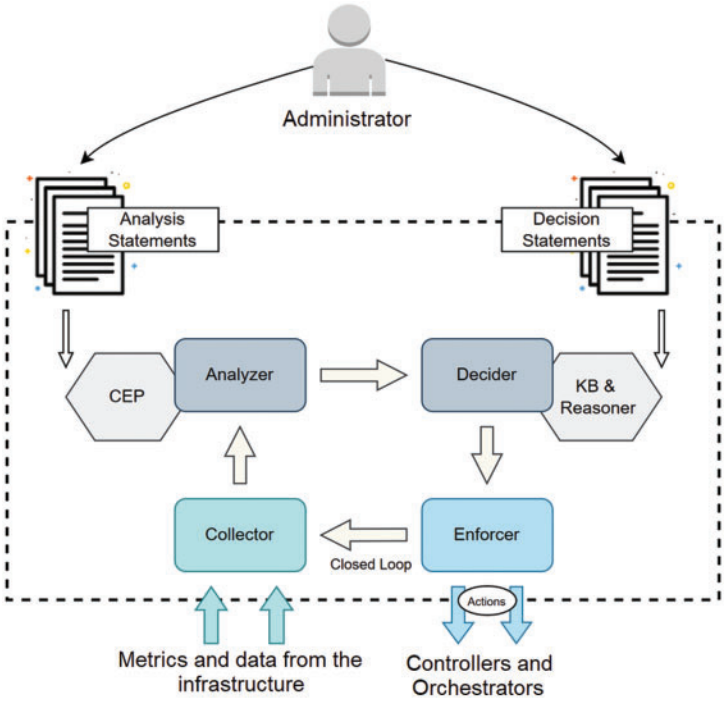

Figure 7: Overview of ARCA components and interactions

ARCA also includes a Complex Event Processing (CEP) mechanism, that correlates all information and determines the situation of the system, as well as a Knowledge Base (KB), including the static knowledge. Finally, administrators specify the analysis and decision statements.

To ensure high fidelity in both event detection and countermeasure decisions, ARCA incorporates several ML algorithms, as well as a CBR. Particularly, the latter component is configured and trained to find out events and computing countermeasures by correlating the current situation with previous similar ones, in particular those provided by the testing and validation phase of the network application. All ML algorithms collaborate to confirm or refute the intermediate decisions. Nevertheless, the final decision will be matched to semantic constraints, which will be provided by the integrators for the overall management system and by the network administrator for particularizing the answer to their domain.

Having explained these two frameworks, we consider that ARCA and the 5GASP platform meet the necessary requirements to be the Reactive System and Network Application Validator of the proposed architecture, respectively, as well as that the elements comprising each of them can be integrated into a single platform to work together.

5 Architecture Instantiation and Component Integration

This section details the instantiation of the proposed model architecture, relying on ARCA and the 5GASP platform as Reactive System and Network Application Validators correspondingly, suggesting already available frameworks to cover the entities conforming to the architecture. By doing so, a later viability validation and quantitative preliminary evaluation is performed. The instantiation is performed using the following components. In order to implement the Network Application Validator functionality, the open-source project OpenSlice [34] is used as NODS, in conjunction with a testing CI/CD pipeline, instantiated as a Jenkins Master and a Jenkins Agent. These tools are the ones selected by the 5GASP platform to implement its functionality, with OpenSlice serving as the *de facto* core of the platform. OpenSlice also includes the functionality of the OSS/BSS, as it acts as the receiver of the verification requests made to the platform, meanwhile Jenkins Master and Agent are responsible for executing the tests over the applications. Concerning the Reactive System, ARCA components conduct its functionality, in conjunction with OpenSource Distributed Mano (OSDM) as an enhanced NFV-Orchestrator acting as the enforcer. Concretely, OSDM is a specialized fork of ETSI's OpenSource Mano [36,37] developed by NICT to better cope with the heavily distributed nature of ARCA. Fig. 8 shows the integration and connections between all these entities, whereas Fig. 9 depicts the communication flow among them.

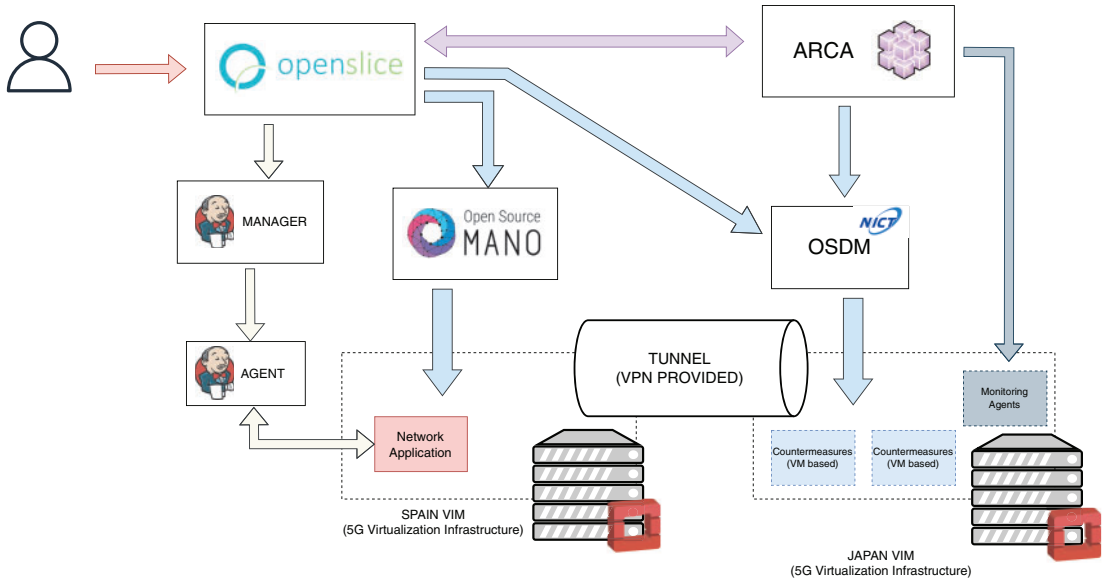

Figure 8: Architecture of the proposed platform after integrating both solutions

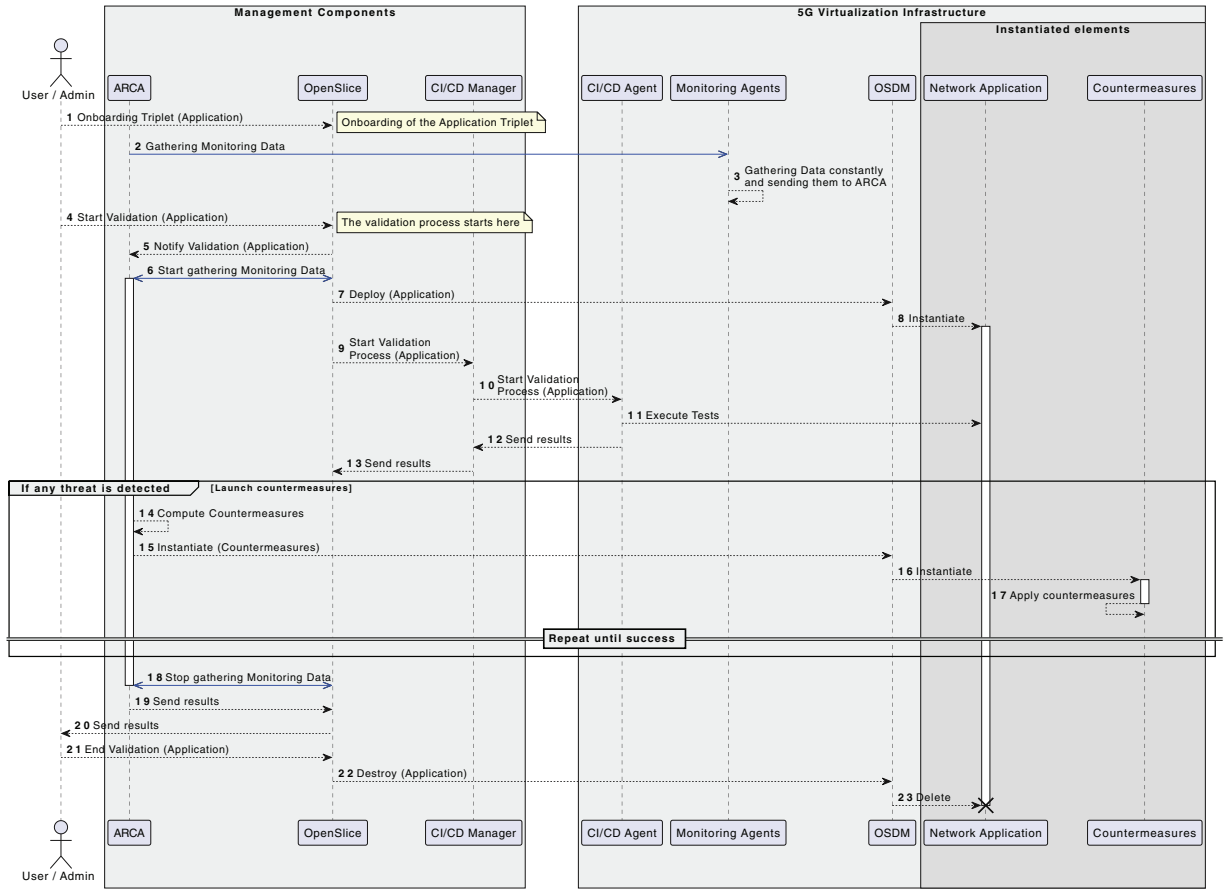

Figure 9: Comprehensive workflow of the communications among all the elements of the proposed instantiation

In the proposed instantiation, the workflow begins with a network administrator or a user onboarding an application to OpenSlice in the form of a triplet (i.e., NFV Descriptors, Test Descriptor, and Network Slice Type (NEST) [38] file). Concurrently, the monitoring agents deployed across the infrastructure are continuously gathering data and sending it to ARCA (Steps 1 and 2). Then, the administrator (or the system automatically, if the deployment had been scheduled) starts the validation process (Step 4). It should be noted here that this process can be initiated in multiple ways, as it may have been programmed previously, or triggered by a user who wants to validate the network application, or even that the instantiation of the application is triggered as a consequence of the execution of another application. When OpenSlice receives the request, it informs ARCA to start a new monitoring process, in case the deployment of the application requires the instantiation of new virtual networks where monitoring was not covered. Once done, OpenSlice asks OSDM to instantiate the application on the corresponding virtualization infrastructure (Step 7). Once the application has been instantiated, OpenSlice asks the CI/CD Manager to start the application validation process with the available test suite, which in turn sends it to the CI/CD Agent deployed in the infrastructure. This Manager/Agent differentiation is due to the fact that the pipeline is designed to be multisite, so agents are deployed in each site and controlled by a single manager. Going back to the verification flow, the agent runs the tests on the application (Step 11) and upon completion, these are sent

back to OpenSlice (Step 13). These results are also gathered by ARCA aggregator to be used as information to infer through its closed-loop if there is any problematic situation occurring (Step 14). If with this information, together with the data obtained from the monitoring, ARCA detects any thread, it computes the necessary countermeasures and asks OSDM to instantiate them (Steps 15 and 16). This process is repeated until the countermeasures are successful or until ARCA runs out of countermeasures to apply. Once this process finishes, ARCA returns the results to OpenSlice, whether it has detected threats or not, and if so, which countermeasures have been effective, and which ones have not. With all this information, OpenSlice prepares a report that is sent to the network administrator or user who initiated the validation process. Finally, once this process is finished, the application is removed from the virtualization platform.

It is noteworthy that, for simplicity, in this case, it is the user or the administrator who has triggered the evaluation of the network application. Still, as aforementioned, this process can occur periodically or automatically when the deployment of a new network application has been scheduled. This diagram showcases the case where the network application is only evaluated and thus deleted when the evaluation phase is finished. However, if the evaluation occurs in the production environment as a scheduled check, the network application will not be erased and will continue its execution till it is terminated, e.g., its functionality is no longer required.

In that sense, using this instantiation, we dispose of a system with the functionality designed in Section 3. Therefore, it becomes feasible to generate a comprehensive test report for the network application behavior, which considers not only the test results but also the monitoring data and countermeasures computed by ARCA.

6 Proposed Architecture Technical Validation and Quantitative Evaluation

To technically validate our claims, the architecture described in Section 4 has been implemented using the frameworks proposed in Section 5. This instantiation implies the connection of two different sites: the University of Murcia's premises (UMU; Murcia, Spain), where the components conforming to the NAV are located, and the National Institute of Information and Communications' premises (NICT; Tokyo, Japan), where the Reactive System components are. This arrangement demonstrates the feasibility of the model being multisite. Consequently, the experimentation platform facilitates the integration of both remote sites, enabling the construction of end-to-end virtual networks that incorporate functions deployed across these geographical locations. The systems are connected using a wireguard-based VPN that transports both data and controls segregated planes on top of a single connection. On NICT side, we have deployed OSDM and ARCA, created a virtual network with functions deployed by OSDM, and configured ARCA to automatically manage the virtual network, particularly focusing on the detection and mitigation of security threats. On the other hand, at UMU we have deployed OpenSlice as well as a Jenkins Manager and a Jenkins Agent to execute the tests and retrieve the results. Both facilities have a virtualization platform upon which the required components can be deployed, either in the form of an application or as a countermeasure. In addition, to evaluate the integrated platform, we will describe a use case running on top of it, concretely an adaptation of the MIGRATE use case [39]. MIGRATE relies on the idea of the dynamic and transparent migration of a network application from one virtualization infrastructure to another, transparently to the end users and without any data loss. In this case, we use MIGRATE as an example of an application used by a person in Spain that takes a plane to Japan and, when it lands and this movement is detected, the application the person was using is migrated from the Spanish datacenter to the Japanese one. In that sense, efficiency as well as security are improved, since the application would be deployed in the same

country in which the end user is found instead of the country where the journey started, apart from a reduction in communication delays it also implies a lower exposure, since the number of network elements employed for the communication are reduced to the bare minimum.

6.1 Use Case

To give a proper storyboard and ease the comprehension of the performed evaluation to the reader, in the following, we detail one possible context for such a scenario. An end user in Spain is using one of the services allocated in the same country, as it is accessing through his smartphone to some data located in a Spanish data center. To improve the user experience, the application has a cache deployed near the user, to reduce latency and avoid the necessity of retrieving the data from the data center. Then, the user takes a plane to Tokyo. When the user moves and appears in Japan, this movement is detected, and the migration process starts. Besides from the latency improvement when accessing the data, other more elaborated scenarios could be envisioned such as the one proposed in [25], in which law enforcement depending on the visiting country is expected to happen within the national borders. This use case links the virtual world with geopolitics and is also applicable to this scenario. However, in both use cases, prior to deploying an application in the Japanese production testbed, it must be tested to ensure its functioning and proper behavior. Therefore, the application is deployed first over a controlled environment, and a suite of tests (e.g., its resilience against vulnerabilities and how it affects the network) is executed. Meanwhile, ARCA is gathering information on its execution through the monitoring agents. If the test execution reports a success and ARCA does not detect any threats, the Application is deployed again but, this time, over the final infrastructure. This deployment does not mean that the security analysis is over, as there are more monitoring agents surveying that infrastructure, and deploying more countermeasures if anything suspicious is later detected.

Fig. 10 shows how the proposed implementation (previously detailed in Fig. 8) has been adapted to this scenario. In order to enable the communication between both facilities, the dedicated VLANs at each site (one for management traffic and another for data flows) have been interconnected through a WireGuard tunnel, enabling transparent communication between private networks. It is important to note that no dedicated network has been used for this connection. Instead, the WireGuard tunnel, which is UDP-based, operates over the internet and thus is subject to the associated packet loss. As the tunnel interconnects both facilities, it is not required to duplicate the monitoring agents or the countermeasures. However, every component can be obviously deployed indistinctly of the site. Also, there is a difference in the NFV-O deployed on each testbed, being OSM in Murcia and OSDM in Tokyo, on purpose, to show that they are exchangeable with no extra actions or adaptations required. Both sites use OpenStack as *Network Function Virtualization Infrastructure* (NFV-I), so it eases the interoperability between them.

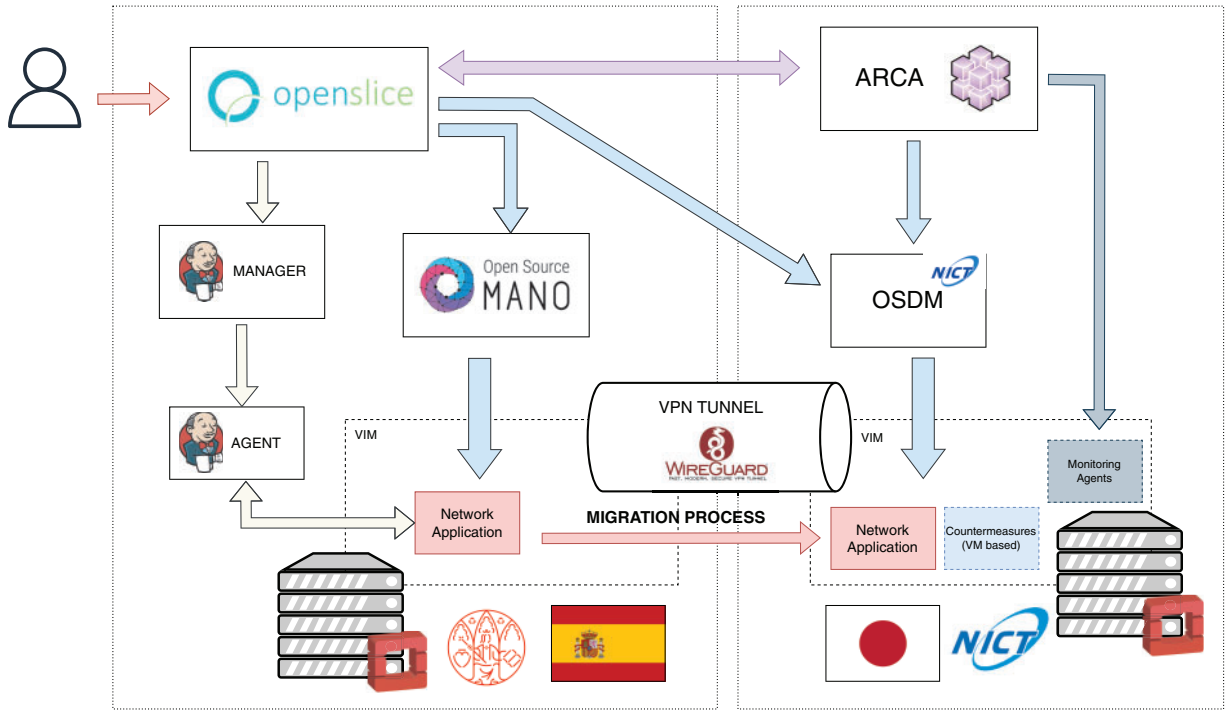

Figure 10: Adaptation of the proposed architecture for the validation scenario

6.2 Evaluation

To evaluate the proposed platform in the context of the discussed use case, two experiments have been conducted. The first one evaluates the performance of the Reactive System, in terms of ARCA's required time to detect a threatening situation and compute the required countermeasures, as well as the time required by the deployment platform to instantiate those countermeasures. And finally, the second one validates the whole flow, where the complete scenario is demonstrated in terms of an application that will be automatically deployed and tested meanwhile ARCA processes the monitored data and launches the required countermeasure. Regarding the testbed used for this experiment, the physical experimentation platform at NICT has five computers with Dell PowerEdge R610 with 2 × Xeon 5670 2.96 GHz (6 core/12 thread) CPU, 48 GiB RAM, 6 × 146 GiB HD at 10k RPM, and 4 × 1 GE NIC. One computer is assigned to the ARCA controller, another to the OpenStack controller and networking, the other two computers are assigned to the OpenStack compute nodes, and the remaining computer is assigned to implement the VPN/Gateway between NICT and UMU. The L2 network that connects all machines is built with an HP ProCurve 1810G-24. In the Murcia site, the platform is composed of three servers, two Dell PowerEdge R640 dedicated to two twins computes nodes for OpenStack Rocky, running Intel Xeon Gold 6138 with 40 cores, 80 threads, and 256 GB of RAM; and the other server being part of a four-nodes PVE, each of them with 256 GB of RAM and 32 threads with an AMD EPYC 730P 16-Core processor, used for running OpenSlice, the OpenStack controller, and the VPN/Gateway between both infrastructures. Both sites are shared environments, and they were in use by other projects during the execution of these tests, so the results are affected by the ongoing usage, and statistical results through multiple executions at different timeslots have been gathered to provide intellectual rigor.

6.2.1 Security Threat Detection Evaluation

The detection of security threats and safety situations, such as external attacks directly on the components of the network or the malfunctioning of those components, as well as the computation of the needed updates to adapt the virtual network to the new situation (threat or safety) is evaluated. From this, we also measured the response time from ARCA to two different scenarios: one with a huge amount of information elements to analyze (380,000), and the other with a more reasonable number of elements to analyze (65,000). These experiments consisted of generating traffic between clients and servers whose communication path crosses the elements managed by ARCA. The traffic generated followed the pattern of a dataset obtained from real measurements of network traffic observed in an Internet exchange point—which is both realistic and anonymous by nature. This dataset recorded, among others, the bandwidth used every 5 s, which is the data we used to generate the traffic in our experiment. We also used the dataset named "Google cluster data 2011" to generate events related to the server side, which is running in a cloud environment consisting of a cluster of machines running OpenStack. To measure the performance, we have used the Cumulative Distribution Function (CDF). The CDF of a real-valued random variable X, also known as the distribution function of X, represents the probability that X will take on a value less than or equal to x [40].

During the execution of the experiment, ARCA received telemetry data obtained from the intermediate network elements located between the clients and servers. The telemetry data is analyzed to find threats or safety situations and determine the actions that must be taken accordingly. We measure the time ARCA requires to complete each operation, shown in Figs. 11–13. Or experiment covers two different scenarios that vary the number of data entries ARCA must analyze to complete each operation. As mentioned before, one scenario had 380,000 entries and the other had 65,000 entries. These numbers depend on the number of managed elements and the number of measurements per second retrieved from them. The envisioned exponential relation between the number of elements to analyze and the analysis time (they must be correlated) is used to validate the good properties of ARCA, mainly the stream processing of monitoring and state information. After executing the experiments, we obtained the following results. In Fig. 11, it is possible to see that most threats are detected in less than 11 ms for 99% of the total cases. In fact, incorporating more information allows ARCA to reduce the detection time to less than 7.2 ms for more than 90% of threats. Similarly, as seen in Fig. 12, ARCA can detect safety situations in less than 15 ms for more than 99% and 11 ms for more than 90% of changes from threat to safety situations. Note that, in this case, ARCA needs more time because more information must be collected to claim that a situation is safe. This can be particularly reflected in the fact that the more information is provided, the more detection time ARCA requires, which is opposite to the results seen for threat detection.

Once a threat or safety situation is determined, it must be translated into a particular change in the virtual network. As seen in Fig. 13, ARCA needs around 25 ms to perform the computation for more than 99% of the cases studied. In this case, there is a much bigger difference between the amount of information that ARCA manages, and the time required to give an answer. This is because, since most of the available information must be correlated to define the services needed and where they must be placed, the more information available, the more computation time will be needed.

These results validate the feasibility and acceptable reaction time of ARCA to be used in real systems.

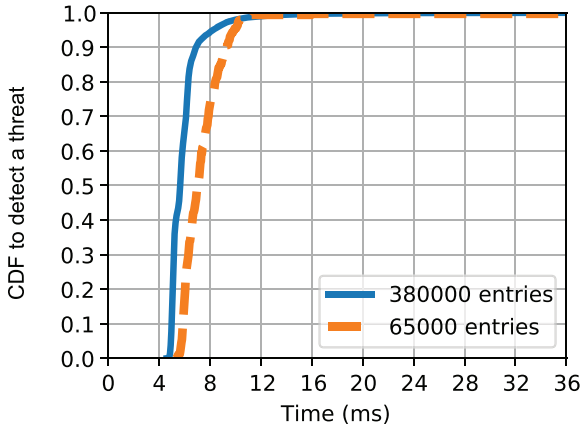

Figure 11: Cumulative Distribution Function of the time required to detect a threat

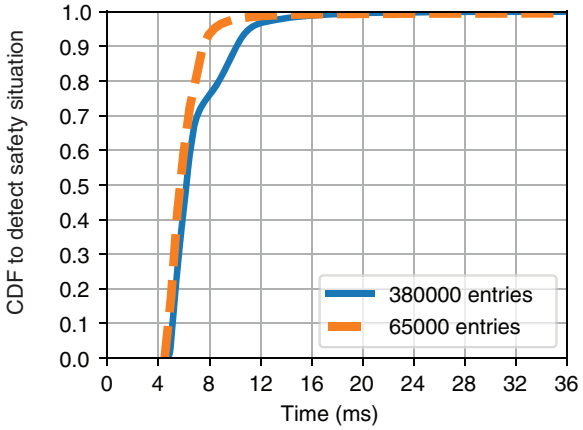

Figure 12: Cumulative Distribution Function of the time required to detect a safety situation

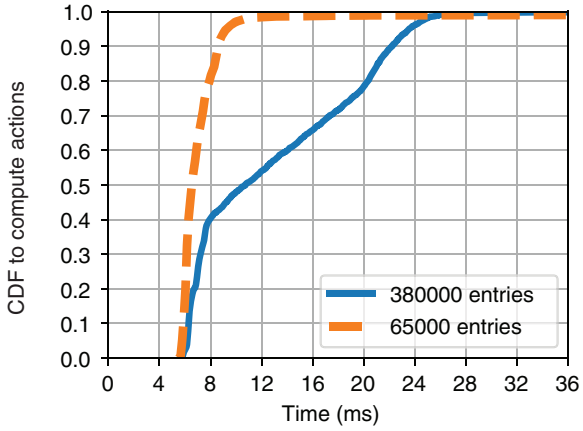

Figure 13: Cumulative Distribution Function of the time required to compute the actions to be enforced

6.2.2 Countermeasure Deployment Validation

Once the required time for ARCA to detect a situation and to compute the required countermeasures has been analyzed, the following step is to evaluate this process in conjunction with the instantiation of the countermeasures themselves on the multi-domain experimentation platform introduced in Section 4. In this case, we evaluate the time elapsed since a threat event is detected (a DDoS attack, malfunctioning of any application, etc.) until the countermeasures deployed are effective, in other words, the time the system is exposed to the security flaw. To do so, we have divided the experiments depending on the number of countermeasures required, assuming that each countermeasure is a network application relying on Linux iptables on top of a Debian image, meaning a small footprint in terms of disk size and memory requirements, and using 65000 simulated elements for ARCA to analyze. Every test has been executed 25 times to ensure statistical wealth, differentiating between three phases: (i) the time required by ARCA to detect the situation and compute the countermeasures, (ii) the time elapsed since ARCA asks for the instantiation till this instantiation starts, and (iii) the elapsed time during the instantiation itself. These results are shown in Fig. 14, together with the confidence intervals (alpha = 0.05).

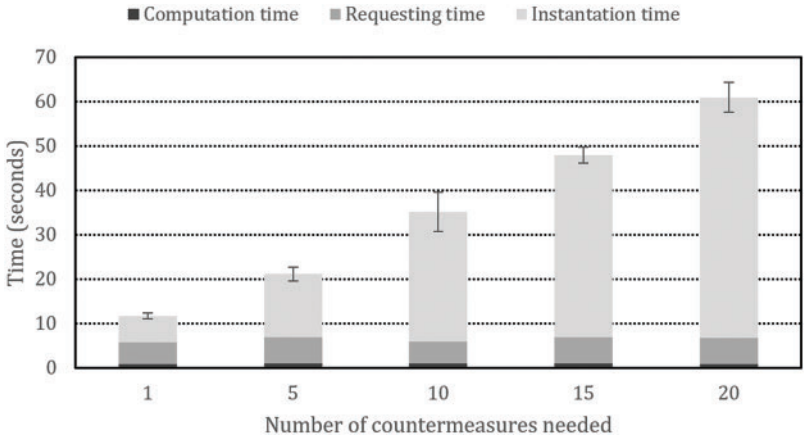

Figure 14: Time required to instantiate the countermeasures depending on its number

As can be seen, the ARCA's processing time is negligible in comparison with the instantiation time. Concerning the request time, it is independent of the countermeasures required. This is due to the fact that a) ARCA processing is quicker, in a different order of magnitude (milliseconds *vs.* seconds) than the request or instantiation processes, even in its worst-case scenario; b) the time required for the petition from ARCA to be processed by OSDM is the same regardless of the number of countermeasures solicited, as they are sent as API calls that are processed in parallel; and c) the virtualization platform cannot parallelize the instantiation process completely, as it inherently demands a substantial allocation of time and resources to deploy the countermeasures, so the more instances required, the more time elapsed.

7 Conclusions

In this paper, we addressed the complex security challenges of 5G Networks by developing a dual-system model that integrates a proactive Network Application Validator with a Reactive, AI/ML-driven security System. This innovative approach not only identifies potential security threats in real-time but also ensures that network applications will have rigorous validation before their

deployment over the production environment. This synergy significantly enhances both the security and functionality of the network applications and their underlying infrastructure, mitigating risks before they can impact the infrastructure or other network applications.

To achieve this, we have defined a model of our solution, detailing its components, the relationships between them and their communication flows. Later, in order to validate the model, we selected and deployed two existing solutions, ARCA and 5GASP platform, which both implement the functionality required by the model. These solutions have been deployed and evaluated in conjunction, thus demonstrating the feasibility and effectiveness of our approach. The relevance of this work relies on demonstrating how two complementary security mechanisms interact (a topic not extensively explored within the realm of 5G networks), thereby providing a higher level of protection and reliability for next-generation networks.

However, it is important to note that this work is not intended as a final or definitive solution but rather as an illustrative example of how different security approaches can be integrated to address a common challenge. The proposed model shows significant potential and is open to further enhancements. For instance, transitioning from conventional VM-based technologies to lightweight virtualization could significantly reduce deployment times for the computed countermeasures. This approach, however, requires further study and evaluation, particularly because some of the developed countermeasures, such as those based on Linux iptables, present challenges in lightweight virtualization environments. Additionally, a comprehensive evaluation of the solution, including the consolidation of components within a unified testbed to minimize latency and maximize efficiency and efficacy is planned in future iterations of this model. Another suggested enhancement involves empowering the NAV and the Reactive System with trust mechanisms, thereby increasing the reliability and accountability not only of the network application but of the whole infrastructure.

Finally, concerning the hierarchical/administrative options the depicted use case offers, in this work it has been implemented as multi-site, with different technological domains but only one administrative domain. A broader scenario with multiple administrative domains would imply multiple OpenSlice and ARCA instances, and an agreement mechanism to avoid the requirement of the user to be part of both administrative domains and having one of the administrative domains as origin for the user, like in telephony roaming scenarios. In that line, smart contracts might be employed to simplify the charging for the services at the destination. Another possibility would be the agreement between administrative domains on access provisioning of local NFV-O to remote OpenSlice and ARCA. The discussed use case could also be considered as a federated multi-domain scenario like in big companies with branches in multiple countries.

Acknowledgement: The authors express their sincere appreciation to all contributors of this paper, as well as to the reviewers and editors for their invaluable feedback and assistance in its enhancement. Additionally, we thank the funding entities whose support made this research possible.

Funding Statement: This work has been supported by Fundacion Seneca-Agencia de Ciencia y Tecnologia de la Region de Murcia- and by the Fulbright Commission in Spain under the Fulbright Grant 00003/FLB/21; by the Spanish Ministry of Science and Innovation under the DIN2019-010827 Industrial PhD Grant, and co-funded by Odin Solutions S.L.; by the European Commission under the NANCY (Grant No. 101096456) and 6G-PATH (Grant No. 101139172) Projects; and by the Spanish Ministry of Economy and Digital Transformation, under the Projects CERBERUS-HADES (Grant No. TSI-063000-2021-62), PROMETEO-6G (TSI-064200-2022-00), MAS4CARE-5G (TSI-065100-2023-006) and Agro6GSense (TSI-064200-2023-8).

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Ana Hermosilla, Pedro Martinez-Julia, Jordi Ortiz, Ved Prasad Kafle, Antonio Skarmeta; software: Ana Hermosilla, Jorge Gallego-Madrid, Pedro Martinez-Julia, Jordi Ortiz; data collection: Ana Hermosilla, Jorge Gallego-Madrid, Pedro Martinez-Julia; analysis and interpretation of results: Ana Hermosilla, Jorge Gallego-Madrid, Pedro Martinez-Julia, Jordi Ortiz; draft manuscript preparation: Ana Hermosilla, Jorge Gallego-Madrid, Pedro Martinez-Julia, Jordi Ortiz; final manuscript preparation: Ana Hermosilla, Jorge Gallego-Madrid, Pedro Martinez-Julia, Jordi Ortiz, Ved Prasad Kafle, Antonio Skarmeta; funding acquisition: Ved Prasad Kafle, Antonio Skarmeta. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used to perform the evaluation are a combination of server usage from Google's ClusterData 2011 [41] and traffic reports from Japan's NIX (National Internet Exchange Point). They would be available from authors upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. 3GPP. The 3rd Generation Partnership Project. Available from: https://www.3gpp.org/about-us/introducing-3gpp. [Accessed 2024].
2. ETSI. European Telecommunications Standards Institute. Available from: https://www.etsi.org/about. [Accessed 2024].
3. Rahouti M, Xiong K, Xin Y, Jagatheesaperumal SK, Ayyash M, Shaheed M. SDN security review: threat taxonomy, implications, and open challenges. IEEE Access. 2022;10:45820–54. doi:10.1109/ACCESS.2022.3168972.
4. Madi T, Alameddine HA, Pourzandi M, Boukhtouta A. NFV security survey in 5G networks: a three-dimensional threat taxonomy. Comput Netw. 2021;197:108288. doi:10.1016/j.comnet.2021.108288.
5. Zehra S, Faseeha U, Syed HJ, Samad F, Ibrahim AO, Abulfaraj AW, et al. Machine learning-based anomaly detection in NFV: a comprehensive survey. Sensors. 2023;23(11):5340. doi:10.3390/s23115340.
6. Direito R, Gomes D, Aguiar RL. Towards a fully automated system for testing and validating netapps. In: 2022 IEEE 8th International Conference on Network Softwarization (NetSoft), 2022; Milan, Italy.
7. Peuster M, Schneider S, Zhao M, Xilouris G, Trakadas P, Vicens F, et al. Introducing automated verification and validation for virtualized network functions and services. IEEE Commun Mag. 2019;57(5):96–102. doi:10.1109/MCOM.35.
8. Trichias K, Landi G, Seder E, Marquez-Barja J, Frizzell R, Iordache M, et al. VITAL-5G: innovative network applications (NetApps) support over 5G connectivity for the transport and logistics vertical. In: 2021 Joint European Conference on Networks and Communications and 6G Summit (EuCNC/6G Summit), 2021; Porto, Portugal.
9. Wang C-X, You X, Gao X, Zhu X, Li Z, Zhang C, et al. On the road to 6G: visions, requirements, key technologies, and testbeds. IEEE Commun Surv and Tut. 2023;25(2):905–74. doi:10.1109/COMST.2023.3249835.
10. Pattaranantakul M. Moving towards software-defined security in the era of NFV and SDN (Ph.D. Thesis). Université Paris Saclay; 2019.
11. He G, Liao X, Liu C. A security survey of NFV: from causes to practices. In: 2023 3rd International Conference on Consumer Electronics and Computer Engineering (ICCECE), 2023; Guangzhou, China.

12. Nespoli P, Papamartzivanos D, Gomez Marmol F, Kambourakis G. Optimal countermeasures selection against cyber attacks: a comprehensive survey on reaction frameworks. IEEE Commun Surv and Tut. 2018;20(2):1361–96. doi:10.1109/COMST.9739.
13. Martinez-Julia P, Kafle VP, Harai H. Exploiting external events for resource adaptation in virtual computer and network systems. IEEE Trans Netw Serv Manag. 2018;15(2):555–66. doi:10.1109/TNSM.2018.2794530.
14. Donatti A, Correa SL, Martins JSB, Abelem AJG, Both CB, Silva FO, et al. Survey on machine learning-enabled network slicing: covering the entire life cycle. IEEE Trans Netw Serv Manag. 2024;21(1):994–1011. doi:10.1109/TNSM.2023.3287651.
15. Coronado E, Behravesh R, Subramanya T, Fernandez-Fernandez A, Siddiqui MS, Costa-Prez X, et al. Zero touch management: a survey of network automation solutions for 5G and 6G networks. IEEE Commun Surv and Tut. 2022;24(4):2535–78. doi:10.1109/COMST.2022.3212586.
16. Asensio-Garriga R, Alemany P, Zarca AM, Sedar R, Kalalas C, Ortiz J, et al. ZSM-based E2E security slice management for DDoS attack protection in MEC-enabled V2X environments. IEEE Open J Vehicular Technol. 2024;5:485–95. doi:10.1109/OJVT.2024.3375448.
17. Saura PF, Bernabe Murcia JM, Zarca AM, Bernabe JB, Skarmeta Gomez AF. Federated network intelligence orchestration for scalable and automated FL-based anomaly detection in B5G Networks. In: 2023 IEEE Future Networks World Forum (FNWF), 2023; Baltimore, MD, USA.
18. Rizwan A, Jaber M, Filali F, Imran A, Abu-Dayya A. A zero-touch network service management approach using AI-enabled CDR analysis. IEEE Access. 2021;9:157699–714. doi:10.1109/ACCESS.2021.3129281.
19. Nespoli P, Marmol FG, Vidal JM. A bio-inspired reaction against cyberattacks: AIS-powered optimal countermeasures selection. IEEE Access. 2021;9:60971–96. doi:10.1109/ACCESS.2021.3074021.
20. Martins JSB, Carvalho TC, Moreira R, Both CB, Donatti A, Corrêa JH, et al. Enhancing network slicing architectures with machine learning, security, sustainability and experimental networks integration. IEEE Access. 2023;11:69144–63. doi:10.1109/ACCESS.2023.3292788.
21. Valantasis A, Psaromanolakis N, Theodorou V. Zero-touch security automation mechanisms for edge NFV: the n-Edge approach. In: 2022 18th International Conference on Network and Service Management (CNSM), 2022; Thessaloniki, Greece.
22. Alemany P, Molina A, Dangerville C, Asensio R, Ayed D, Muñoz R, et al. Management and enforcement of secured E2E network slices across transport domains. Opt Fiber Technol. 2022;73:103010. doi:10.1016/j.yofte.2022.103010.
23. Sakthidevi I, Rajkumar GV, Sunitha R, Sangeetha A, Krishnan RS, Sundararajan S. Machine learning orchestration in cloud environments: automating the training and deployment of distributed machine learning AI model. In: 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2023; Kirtipur, Nepal.
24. Perez-Valero J, Virdis A, Sanchez AG, Ntogkas C, Serrano P, Landi G, et al. AI-driven orchestration for 6G networking: the Hexa-X vision. In: 2022 IEEE Globecom Workshops (GC Wkshps), 2022; Rio de Janeiro, Brazil.
25. Zhao M, Le Gall F, Cousin P, Vilalta R, Muñoz R, Castro S, et al. Verification and validation framework for 5G network services and apps. In: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017; Berlin, Germany.
26. Garcia-Reinoso J, Rosello MM, Kosmatos E, Landi G, Bernini G, Legouable R, et al. The 5G EVE multi-site experimental architecture and experimentation workflow. In: 2019 IEEE 2nd 5G World Forum (5GWF), 2019; Dresden, Germany.
27. ETSI. ETSI GS NFV-SOL 002. Network functions virtualisation (NFV); architectural framework. Technical Report, 2014.
28. ETSI. ETSI GS NFV-SOL 001. Network functions virtualisation (NFV) release 4; protocols and data models; NFV descriptors based on TOSCA specification. Technical Report, 2022.

29. ETSI. ETSI GS NFV-SOL 006. Network functions virtualisation (NFV) release 4; protocols and data models; NFV descriptors based on YANG specification. Technical Report, 2022.
30. ETSI. Network functions virtualisation (NFV) release 3; protocols and data models; network service descriptor file structure specification. Technical Report. Available from: https://www.etsi.org/deliver/etsi_gs/NFV-SOL/001_099/007/03.05.01_60/gs_nfv-sol007v030501p.pdf. [Accessed 2024].
31. Trantzas K, Tranoris C, Denazis S, Direito R, Gomes D, Gallego-Madrid G, et al. Implementing a holistic approach to facilitate the onboarding, deployment and validation of NetApps. In: 2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom), 2022; Athens, Greece.
32. Martinez-Julia P, Kafie VP, Asaeda H. Explained intelligent management decisions in virtual networks and network slices. In: 2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), 2020; Paris, France.
33. Gomes D, Tranoris C. 5GASP Project. Available from: https://5gasp.eu. [Accessed 2024].
34. Tranoris C. OpenSlice: an openSource OSS for delivering network slice as a service. arXiv preprint arXiv:2102.03290. 2021.
35. Jahan S, Riley I, Walter C, Gamble RF, Pasco M, McKinley PK, et al. MAPE-K/MAPE-SAC: an interaction framework for adaptive systems with security assurance cases. Future Gener Comput Syst. 2020;109:197–209. doi:10.1016/j.future.2020.03.031.
36. ETSI. OSM Release Five Technical Overview. Technical Report, ETSI. 2019.
37. Managing B5G Services with OSDM. IPOP 2022. Available from: https://www.pilab.jp/ipop2022/exhibition/panel/iPOP2022_NICT_poster.pdf. [Accessed 2024].
38. GSM Association. Generic Network Slice Template, version 5.0. Technical Report, GSMA. 2024. Available from: https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v5.0-7.pdf. [Accessed 2024].
39. Santa J, Ortiz J, Fernandez PJ, Luis M, Gomes C, Oliveira J, et al. MIGRATE: mobile device virtualisation through state transfer. IEEE Access. 2020;8:25848–62. doi:10.1109/Access.6287639.
40. Deisenroth MP, Faisal AA, Ong CS. Mathematics for Machine Learning. Cambridge, UK: Cambridge University Press; 2020.
41. Google's ClusterData. Google Cluster Data Repository. 2011. Available from: https://github.com/google/cluster-data/tree/master. [Accessed 2024].