IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 February 2023, accepted 8 April 2023, date of publication 20 April 2023, date of current version 9 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3268759

RESEARCH ARTICLE

A Network Traffic Mutation Based Ontology, and Its Application to 5G Networks

ZUJANY SALAZAR[1], FATIHA ZAÏDI[2], HUU-NGHIA NGUYEN[1], WISSAM MALLOULI[1], ANA ROSA CAVALLI[1], AND EDGARDO MONTES DE OCA[1]

[1]Montimage EURL, 75013 Paris, France
[2]Université Paris-Saclay, CNRS, ENS Paris-Saclay, Laboratoire Méthodes Formelles, 91190 Gif-sur-Yvette, France

Corresponding author: Fatiha Zaïdi (fatiha.zaidi@lri.fr)

This work was supported in part by H2020 Project SANCUS under Grant 952672, in part by INSPIRE-5Gplus under Grant 871808, and in part by SPATIAL under Grant 101021808.

ABSTRACT This paper presents an ontology based on mutation techniques for the modelling of cybersecurity attacks and its application to 5G networks. Main concepts of network protocols, mutation operators, flow of network packets and network traffic are introduced. An ontology is designed based on different mutation operators that allow to design models that can be assimilated with known and unknown attacks. This approach has been implemented in our open source 5G network traffic fuzzer, 5Greplay, and has been applied to three use cases that are representative of attacks against 5G networks: NAS Replay attack, Denial of Service by Sending Malformed NGAP Packets and 5G encapsulation of IoT traffic.

INDEX TERMS Ontology, mutation technique, network traffic, 5G networks, fuzzing techniques.

I. INTRODUCTION

The 5G networks represent a paradigm of next generation wireless technology. They will be faster and able to handle more connected devices than the existing 4G networks. They are composed of different enabling technologies, such as Software-Defined Networking (SDN), Network Function Virtualization (NFV), Multi-access Edge Computing (MEC), Cloud-Native Core Network (CCN) and Network Resource Slicing (NRS).

Although these technologies have been the subject of various research works, they introduce a new set of cybersecurity challenges that still need to be investigated. Regarding cyberattacks, different techniques have been developed in order to detect and execute them. One of the techniques used is fuzz testing, which is a software testing technique that relies on the injection of random, invalid or unexpected data to cause the malfunctioning or a crash of the system.

Human languages are commonly studied from three main perspectives: syntax, semantics, and behaviour [1]. Syntax studies rules to define grammatically well-formed sentences, semantics considers the meaning of words and phrases, and behaviour is the study of the context-dependent features of a language.

We study network protocols from these perspectives. Then, every protocol has (i) a syntax that defines the sequencing of data elements or bits that are considered to be valid, and determines how to read the data in the form of fields, (ii) a semantic that refers to the interpretation or meaning that computers give to each field, and (iii) a behaviour, that we will call behaviour, dimension that considers the data in its context, this is when the data should be sent, and how fast.

The work described in this paper extends the research and open-source solution 5Greplay, presented by the authors in [2] and [3]. Notably, in this paper, we present an ontology based on mutation techniques for the modelling of cybersecurity attacks and its application to 5G networks, that we demonstrate by means of our 5G network traffic fuzzer, 5Greplay.

The idea is to develop an ontology based on different mutation operators that will allow to design models that can be assimilated with known attacks and with 0-day attacks. Attacks can be simple attacks consisting of simple actions, or more complex attacks based on the composition of

The associate editor coordinating the review of this manuscript and approving it for publication was Tao Zhou.

different actions. This approach will facilitate the description and execution of attacks in order to check the robustness of 5G networks and their components. It will also facilitate the automated detection and design of mitigation techniques to resist to attacks.

The main contribution of this paper is the design of an ontology for mutation techniques by introducing classes of the concepts of network protocol, mutation operators, network packet, flow of network packets, network traffic. To the best of our knowledge there is an absence of a unified formalism for describing the fuzzing process, in particular the mutation-based fuzzing techniques. We believe that a formalism to describe network-enabled fuzzing would improve network security by allowing automation of test case creation, and facilitating the creation of complex test cases scenarios.

The formalism we described in Section IV differs from the state of the art because it was specially designed to mutate network protocols and also allows the design of complex attacks. Network-enabled mutation has the peculiarity that there may be dependency between network packets or between packet flows. Therefore, when mutating a packet, our formalism also considers the packets that depend on it, in addition to other characteristics of its context that we also discuss in this article. Finally, the operators that we defined in our formalism can be combined to create complex protocol-based attacks.

The paper is organized as follows: Section II contains the necessary background on ontologies and 5G networks to understand the rest of the article, and the case studies presented; Section III presents related works; Section IV presents the 5G network fuzzer; Section V-D presents the ontology of the network mutation functions, and Section V the experimentations to illustrate our approach. Finally, Section VI provides the conclusion and perspectives.

II. BACKGROUND

In this section, we present the necessary background for understanding the rest of the article. First, we quote a definition and list the basic steps to build an ontology. Then, we present basic concepts of 5G networks, that will help the reader to understand the case studies presented in Section V.

A. ONTOLOGIES

Research has defined an ontology as the basic terms and relations comprising the vocabulary of a subject, together with the rules for combining terms and relations to define extensions to the vocabulary [4].

Ontologies differentiate from taxonomies, as they contain full specifications of a domain, including relationships, such as ''composition'', ''if-then-else'', ''and'', ''or'', ''not'', ''equal''. Moreover, an ontology contains resources to add new classifications.

Ontologies can be defined by formal means, for instance, by using axioms to specify the intended meaning of domain elements. However, informal methods, such as, UML class diagrams, entity-relationship models, and semantic nets, are also accepted.

The basic steps to build an ontology are:

1) Determine the domain and scope of the ontology
2) Search and envisage reusing ontologies in the literature
3) Enumerate important terms in the ontology
4) State the classes and the class hierarchy
5) State the properties of classes and slots
6) State the facets of the slots
7) Create instances

B. 5G CORE NETWORK

In the 4G network architecture, User Equipment (UE) like smartphones or cellular devices, connects over the LTE Radio Access Network (E-UTRAN) to the Evolved Packet Core (EPC) and then further Data Networks (DN), like the Internet. 5G networks divide the LTE core and implements each function separately, so that they can run independently of each other. Moreover, 5G network functions are virtualized, then they do not need dedicated hardware. This enables the 5G core to decentralized 5G nodes, and to be very flexible. In fact, 5G networks can be adapted through network slicing, to have multiple logical ''slices'' of functionality optimized for specific use-cases, all operating on a single physical core within the 5G network infrastructure. 5G core Service-Based Architecture proposed by the 3GPP[1] is showed in Figure 1. It is composed of the following major components [5]:

1) The Access and Mobility Management Function (AMF) is as a single-entry point for the UE connection. Based on the service demanded by the UE, the AMF selects a Session Management Function (SMF) for handling the user session.
2) The User Plane Function (UPF) transports the IP data traffic (user plane) between the User Equipment (UE) and the external networks.
3) The Authentication Server Function (AUSF) assists the AMF to authenticate the UE and access services of the 5G core.
4) Further functions like the Session Management Function (SMF), the Policy Control Function (PCF), the Application Function (AF) and the Unified Data Management (UDM) function manage the policy control framework, applying policy decisions and accessing subscription information, to control the network behaviour.

III. RELATED WORKS

This Section cites previous works in Mutation-based Fuzzing Testing Formalization, Network-enabled and 5G protocol fuzzers, 5G security test generation, as well as, a survey on 5G Threats.

A. MUTATION-BASED FUZZING TESTING FORMALIZATION

Various formal bases to improve this critical fuzzing approaches has been proposed in the literature. On the software testing field, research [6] has characterized the input

[1] https://www.enisa.europa.eu/publications/sdn-threat-landscape

evaluation and selection components of fuzzing, based on static analysis concepts. In their model, they define the notions of input, concrete and abstract states, and how the inputs are mapped from a concrete state trace to an abstract state by an abstraction function. Moreover, authors have defined some software fuzzing technique using their model concepts in order to demonstrate its generality. Their results proved that the choice and combination of abstraction functions are significant and can modify the effectiveness of fuzzing.

Fuzzing testing has been also represented using mutation trees, in particular [7] applied it to an industrial control system, in order to improve the efficiency of the mutation process and the insufficient amount of tests in the old fuzzing technology for industrial control systems (ICS).

B. NETWORK-ENABLED AND 5G PROTOCOL FUZZERS

Dedicated fuzzers for protocols commonly used in telecommunications have been proposed. T-Fuzz [8] follows a generation-based fuzzing approach, relying on message models with full protocol specifications to test the 3GPP Non-Access Stratum (NAS) protocol, used in LTE and 5G networks. For the Resource Control layer (RCC) protocol that operates between the UE and gNB, a fuzzer based on the ASN.1 description language has been proposed [9]. The fuzzer extracts information about ongoing RRC messages by means of protocol description files of RRC from 3GPP, and uses it to mutate RRC messages. The adaptive fuzzer recognizes individual fields, sub-messages, and custom data types according to specifications when fuzzing the content of existing messages.

The Next Generation Application Protocol (NGAP) that operates between the UE and the AMF, has also been tested with fuzzing techniques. Mutation algorithms based on partition weight have shown to be able to reduce fuzzing time by generating samples that are more likely to produce anomalies in the 5G core [10]. The main limitation of this approach is that the number of samples that are required to test and calculate the weights of each protocol field during the tuning phase of the algorithm, must be adequate to get an accurate calculation of field weights, and this process could take a considerable amount of time.

In conclusion, network traffic mutation is gaining attentions for searching vulnerabilities. Nevertheless, to the best of our knowledge, there is an absence of a unified vocabulary for describing it. Some works have approached this problem, but they are merely taxonomies, that do not contemplate the relationship between the entities interfering on the process of network traffic mutation, as an ontology would do. We believe that such an ontology would improve network security by allowing automation of test case creation.

The ontology we described in Section IV differs from the state of the art because it was specially designed to mutate network protocols. Network-enabled mutation has the peculiarity that there may be dependency between network packets or between packet flows. Therefore, when mutating a packet, our ontology also considers the packets that depend on it, in addition to other characteristics of its context that we will discuss later in this article.

C. 5G SECURITY TEST GENERATION

The evolution of 5G mobile networks towards a service-based architecture (SBA) comes with the emergence of numerous new testing challenges and objectives. First, 5G deployment introduces a brand-new set of technologies, such as, the network function softwarization enabled by the software defined networking (SDN) and network functions virtualization (NFV), Mobile edge computing (MEC), and Network Slicing (NS). These require to be tested from a functional point of view; but, also from a non-functional point of view in order to determine the sanity of the system based on indicators such as data throughput performance, latency, scalability, robustness, etc. Finally, 5G SBA introduces new cybersecurity threats, with some of the previously adopted security and privacy mechanisms becoming ineffective or not applicable in 5G due to the changes in the architecture and the advent of new services [11]. This requires the creation of new sets of security test cases and tools specifically targeting 5G security concerns.

The 3GPP standardization organism has proposed a wide set of test cases to check functional and non-functional requirements in 5G network products. For example, in the TS 33.117 catalogue [12] of general 5G security assurance requirements, they propose a group of test cases to verify if network products providing externally reachable services are robust against unexpected inputs. The target of these tests are the 5G protocol stacks (e.g., Diameter stack). Other specifications, such as the TS 33.512 [13], concerning the security assurance of the Access and Mobility management Function (AMF), provides test cases to verify that this 5G component is properly protected against specific vulnerabilities.

Regarding security testing, 5G issues have been the subject of numerous studies. Standardization organisms list collections of threats and vulnerabilities [14], also investigated by academia [11], [15], and Industrial researchers [16], [17]. Several sources have studied the problematic of replay attacks in 5G networks, that could expose the system to Man-in-the-Middle (MiTM) or Denial-of-Service (DoS) attacks. Technology reports identify the possibility of malicious actors performing a DoS or MiTM attack by replaying Packet Forwarding Control Protocol (PFCP) messages that manage GPRS Tunnelling Protocol (GTP) tunnels. The ENISA organization [18] alerts that an AMF can be vulnerable to replay attacks of Non-Access Stratum (NAS) signalling messages between the UE and AMF on the N1 interface. Moreover, ENISA adds that the security of a network slice could be compromised if an attacker spoofs a genuine network manager by obtaining access to an insecure network management interface, or just by replaying or modifying a valid message.

In order to overcome these issues, numerous research has been done in the matter of detection and prediction of 5G cyber-attacks. Several Intrusion Detection Systems (IDSs) have been proposed [19], [20]. Nevertheless, to the best of our knowledge, there is a lack of open-source solutions that enable to manually create or edit existing 5G network protocol packets and injecting them in a network, allowing to easily test the proposed detection schemes.

Therefore, besides the significant amount of functional and non-functional proposed test cases, as well as the previously mentioned collections of 5G network threats and vulnerabilities and intrusion detection approaches, the testing of 5G network components and IDSs remains a challenge. This is in part due to the lack of publicly available labelled data sets containing realistic user behaviour and up-to-date attack scenarios [21].

Open-source tools, such a Tcpreplay, aim to solving this issue by enabling to replay malicious traffic patterns on IDSs. More recent versions of the tool include the capability to replay traffic on web servers. Within the Tcpreplay suite, Tcpwrite allows editing, creating and replaying PCAP files in a network. However, it targets modifying IP, TCP and UDP attributes and fields. Other packet manipulation solutions such as Scapy are also not 5G-oriented.

Here, we propose 5Greplay, an open-source solution to perform fuzz testing of 5G networks. 5Greplay aims to facilitate the testing process of 5G virtual network functions and IDSs by allowing to forward network packets from one network interface card (NIC) to another with or without modifications. The tool supports the implementation of test cases by letting the users create specific scenarios using PCAP files that contain conventional 5G network traffic and execute them on a target network.

D. 5G THREATS

This section aims to synthesize security threats and in some cases cyber-attacks, that target 5G. For the categorization, we have considered the main technologies that interact in 5G networks: SDN, NFV, VNF, RAN, virtualization technologies, network slicing, Multi-access Edge Computing (MEC), cloud computing and IoT. Additionally, based on 5G Service-Based Architecture (SBA) defined by the 3GPP [22], we grouped the threats into three main different physical domains: Core network, access network or RAN, and UE, as shown in Figure 1. In our physical classification we assume that Core threats are also applicable to the Edge network, as it has been considered as a reduced version of the Core network [23].

Table 1 summarizes the main threats and attacks targeting 5G for enabling technology and physical domain. When Table 1 shows more than one physical domain, this means that the vulnerability could be exploited starting from more than one physical domain. Moreover, in its column *5Greplay*, the table indicates the threats that could be reproduced by means of the ontology described in Section IV, and therefore, implemented with the 5Greplay fuzzer.

It is worth noting that when we assign to a threat or attack a specific enabling technology, or physical domain in Table 1, we are assuming that the vulnerability that was exploited to perform the malicious activity was inherent to the mentioned technology, and the exploitation was made in the particular physical domain. However, the effects of this exploitation, in general, are reflected in more than one technology and physical domain. For example, if an attacker exploits a weakness in an SDN platform, it would be classified, according to our categorization as an SDN threat, located in the Core domain, even though it probably would have consequences in the network slicing management, on the NFVs, and edge and UE domains.

1) CLOUD COMPUTING

In the last few years, the adoption of cloud computing services has become a very attractive alternative for both small companies and big multinationals, due to the possibility of having increased IT resources at reduced capital costs. Nevertheless, cloud computing platforms and their native applications have been demonstrated to have significant privacy and security vulnerabilities [48].

5G networks rely heavily on cloud technologies. First, 5G Core has a Service-Based Architecture (SBA) intended for deploying a cloud-native network, enabling to build and modify VNF faster, with greater scalability and flexibility than in previous mobile networks. Cloud-native 5G also enables network slicing, hence the network performance can be adapted to the client's specific service requirements. Moreover, 5G networks migrated from Distributed Radio Access Network (D-RAN) to Cloud or Centralized RAN (C-RAN). Therefore, cloud computing weaknesses impact heavily on 5G network components (e.g., SDN, MEC, or IoT devices).

Some of the most prominent threats of cloud computing according to [49] are (i) interception of cloud data, (ii) interruption of internet connection, and (iii) breaching of the cloud server. Additionally, merging of cloud computing and IoT exposes IoT platforms to cloud-induced vulnerabilities such as those contained in the Open Web Application Security Project (OWASP) top 10 [50]. Applications that rely on cloud computing services can compromise mobile devices. This possibility can be leveraged by external attackers to remotely breach both private and public corporate networks. Moreover, cybercriminals can use cloud services on their own. By having the capacity to do so, a cybercriminal may provide powerful resources to process power and storage, and appear and disappear easily [51].

2) NETWORK SLICING

Network slicing, which is enabled by SDN and virtualization technologies, allows 5G networks to provide and customize services on-demand depending on the client's necessities.

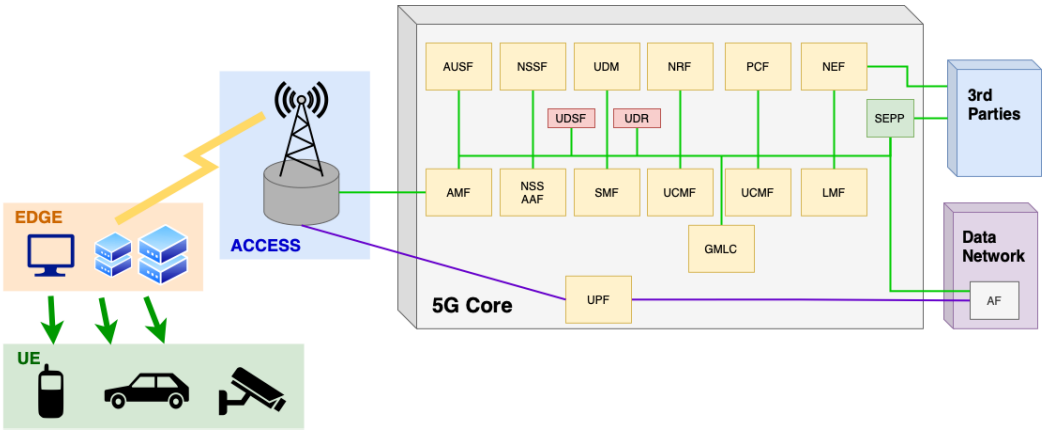


FIGURE 1. 5G service-based architecture (SBA) [22].

Security and isolation between slices are fundamental but complex, because physical isolation is not always possible, and the VNF of network slice instances are implemented on common cloud-based infrastructures. A malicious actor could use the capacity and elasticity of one slice to exhaust the resources of another target slice, a make it out of service. Additionally, certain network functions in the control plane, such as NSSF, are mutual to multiple slices. Therefore, an attacker may eavesdrop on the data of one slice by accessing the shared functions from another slice in an unauthorized manner. Likewise, the UE can simultaneously access more than one network slice. Consequently, a UE could be abused to initiate a security attack from one slice to another [28], [29].

Multi-factored attacks are also possible [28]. In such a scenario, by means of malware an attacker could reduce the resources of one slice, thus causing DoS to the actual subscriber. Side-channel attacks, combined with incorrect isolation between network slices, may lead to data exfiltration.

Slice management also contains critical threat vectors if not secured. Without securing network slice management interfaces, attackers may gain access to them, thus illegally create network slice instances. Consequently, a malicious actor could perform fraudulent activities, such as, DoS attacks, false charging, eavesdropping on data, or extraction of sensitive information to accomplish other attacks.

3) VIRTUAL NETWORK FUNCTIONS

A VNF is a network service that the NFV allows to place in a virtualized form in dedicated hardware technology. VNFs are part of the NFV architecture since normally their threats are the result of the exploitation of code vulnerabilities. This differs from the rest of NFV threats, which are exploits of weaknesses in network protocols. However, in this survey, we consider NFV threats and VNF apart.

The VNFs are pieces of software, then they are exposed to the exploitation of software vulnerabilities in their code. For example, improper input validation, buffer overflows and underflows during read or write operations, dynamic memory deallocation, poorly defined restriction of operations within the boundaries of a memory buffer, integer overflow, path traversal, or vulnerable software components (i.e. libraries, frameworks, and other software modules, etc) are software-related vulnerabilities that can lead to attacks against a VNF, and compromise the rest of the 5G architecture [30]. Moreover, academic research [31], [32] and standardization groups [33] report authentication threats that may lead malicious actors to access data or perform unauthorized actions, and consequently have a range of issues, including information exposures, DoS attacks and arbitrary code execution.

According to ENISA, one of the most important vulnerability sources of the VNFs is the edge devices with open application programming interfaces (APIs) [34]. In specific, their abuse is feasible done through the exploitation of vulnerabilities in MEC applications. Moreover, SDN controllers maintain interfaces with a variety of network components, using VNFs. The latter exposure makes the susceptible to amplification and flooding attacks, which through a small stream of requests from masqueraded malicious parties, can create a flood of responses. Flooding attacks are capable to affect a range of different external interfaces that the network provides, such as the radio interface, interfaces to external networks like the World Wide Web or other mobile networks. Finally, the 3GPP has listed numerous threats that are inherent of the specific network functions (i.e., AMF, UPF, SMF, etc) and their related network protocols (e.g., SCTP, NGAP, NAS-5GS, etc) [35].

4) NETWORK FUNCTION VIRTUALIZATION

NFVs enable placing various network functions in different network components based on their performance requirements and eliminate the necessity for function or service-specific hardware. This technology is vulnerable to authentication threats, due to spoofing of information parameters in different VNFs, unauthorized use of VNF predefined

TABLE 1. 5G security threats and cyber-attacks classified by enabling technology and Physical domain, pointing out the applicability of 5Greplay to reproduce them.

Enabling Technology	Threat	Physical Domain	Ref	5Greplay
Cloud Computing	Interception of cloud data	CORE/EDGE, ACCESS, UE	[24], [11]	
	Interruption of internet connection	CORE/EDGE, ACCESS, UE	[24]	
	Breaching of cloud servers	CORE/EDGE, ACCESS, UE	[24], [25]	X
	Exploitation of applications	CORE/EDGE, ACCESS, UE	[26], [27]	X
	Use of cloud resources to perform criminal activities	CORE/EDGE, ACCESS UE	[26]	
Network Slicing	Exploitation of common physical resources between different slices, to perform DoS attacks	CORE/EDGE, ACCESS	[28], [29], [11]	
	Exploitation of common VNF between different slices, to perform eavesdropping	CORE/EDGE, ACCESS	[28], [29]	X
	Exploitation of a UE with access to several slides	CORE/EDGE, ACCESS	[28], [29], [22]	X
	Multi-factored attacks, using malware to cause DoS attacks	CORE/EDGE, ACCESS	[28]	
	Abuse of management interfaces to illegally create network slice instances, leading to DoS attacks, fraudulent activities, or eavesdropping	CORE/EDGE, ACCESS	[28], [22]	X
VNF	Improper Input Validation, Buffer overflows & underflows during Read or Write operations, Dynamic memory deallocation, Memory buffer bounds, Integer overflow, Path traversal, Exploitation of vulnerable software components: libraries, frameworks, etc	CORE/EDGE, ACCESS	[30]	
	Authentication and Authorization broken access	CORE/EDGE, ACCESS	[31], [32], [33]	
	Open APIs abuse	CORE/EDGE, ACCESS	[34], [11]	X
NFV	Authentication threats, due to spoofing of information parameters, unauthorized use of VNF predefined accounts, weaknesses in password policies, and traffic spikes	CORE/EDGE, ACCESS	[35]	X
	Elevation of privilege via incorrect verification of access tokens	CORE/EDGE, ACCESS	[35], [22]	
	Exploitation and Abuse of a third party hosted network functions, a lawful interception function, or a weakly designed or configured API	CORE/EDGE, ACCESS	[34], [22], [36]	X
	Access the personal data stored in the log files	CORE/EDGE, ACCESS	[34], [22]	
	5G Network protocol (e.g. GTP, Diameter, NGAP, NAS-5GS, JSON) exploits	CORE/EDGE, ACCESS	[37], [22]	X
SDN	Spoofing	CORE/EDGE, ACCESS	[34], [38], [39]	X
	Tampering	CORE/EDGE, ACCESS	[38], [40] [39]	
	Elevation of Privilege	CORE/EDGE, ACCESS	[38]	
	Information Disclosure	CORE/EDGE, ACCESS	[38], [39], [41]	
	DoS attacks	CORE/EDGE, ACCESS	[34], [38], [39], [41], [40], [11]	X
	Malicious Northbound Applications	CORE/EDGE, ACCESS	[34], [41], [11]	
	Configuration issues	CORE/EDGE, ACCESS	[41], [40], [42], [11]	
Virtualization	Hypervisor-based attacks, VM-based attacks	CORE/EDGE, ACCESS	[24]	
	VM image-based attacks	CORE/EDGE, ACCESS	[24], [25]	
	Container image-based attacks, Cross-container attack	CORE/EDGE, ACCESS	[26]	
Multi-access Edge Computing (MEC)	Exploitation of open APIs to perform DoS and man-in-the-middle (MitM) attacks, malicious mode problems, privacy leakages, and VM manipulation	EDGE	[11], [22], [43]	X
	Cross-contamination, CaaS and PaaS based attacks	EDGE	[22]	
RAN	Spectrum resource abuse, Radio frequency jamming, Session hijacking, Signaling fraud	ACCESS	[34]	
	Fake access network node, Flooding attacks	ACCESS	[34], [22]	X
	Signaling storms	ACCESS	[34], [11]	
	False buffer status attacks, Micro-cell attacks, Message insertion attacks	ACCESS	[44]	
	RRC protocol-based attacks	ACCESS	[37], [22]	
IoT Devices	Exploitation of unpatched APIs	UE	[45]	X
	Exploitation of authentication vulnerabilities	UE	[45], [46]	X
	Physical attacks, Reconnaissance attacks	UE	[46]	
	DoS attacks, Malware, Botnets, Ransomware	UE	[46], [47]	X

accounts or attributes (e.g., guest, ctxsys), weaknesses in password policies, and traffic spikes. Furthermore, if authorizations for accounts and applications in the NFV are not reduced to the minimum required for the tasks they have to perform, the elevation of privilege via incorrect verification of access tokens is possible [35].

Similarly, NFVs are exposed to exploitation and abuse threats, such as the exploitation of third-party hosted network functions, a lawful interception function, a weakly designed or configured API with inaccurate access control rules, or poorly configured systems/networks; as well as, the unauthorized access to a function when hosted outside the operator's network. Moreover, accessing the personal data stored in the log files can lead to remote access exploitation and compromise the system integrity [34]. Finally, NFVs can be attacked by exploiting vulnerabilities of their native protocols, e.g., GTP, Diameter, NGAP, NAS-5GS, JSON [37].

5) SOFTWARE DEFINED NETWORKS

One of the main novelties in the 5G architecture is the comprehensive virtualization of the Core network [34], facilitated by the NFV and the SDN, that allow easy management and innovation through abstraction. However, in the absence of proper security mechanisms, for example, the lack of TLS adoption, malicious actors could perform a man-in-the-middle, and launch various attacks by impersonating or gaining unauthorized access to the controller, or modifying the channels [34], [38], [39], [40].

Similarly, as the policy enforcement process is distributed across physical switches, security threats and new risks of information disclosure are introduced. A malicious actor can identify the action applied to a packet type by performing packet processing timing analysis [38], [39], [41]. Also, as the SDN controller manipulates flow rules in the data forwarding elements, it is vulnerable to DoS attacks [11], [34], [38], [39], [40], [41].

SDN facilitates third party applications to be included in the architecture, then an application could introduce its vulnerabilities to the system [11], [34], [41]. Simple faults in network applications might lead to the breakdown of the control plane and failure of network functionality. Moreover, traffic high-jacking and re-routing is possible, an illegitimate appropriation of routing group addresses by corrupting the routing tables. Finally, network security policies and protocols can have network vulnerabilities that affect the layers and interfaces of the SDN framework [11], [40], [41], [42].

6) VIRTUALIZATION PLATFORM

5G networks are deeply based on virtualization technologies, that allows placing VNF in virtual machines. Virtualization platforms face different threats, depending on the different virtualization approaches followed in the network. In this subsection, we focus mainly on server virtualization software threats and container-based threats.

Research divided the server virtualization security threats into three main categories: hypervisor-based attacks, VM-based attacks, and VM image attacks [24]. A hypervisor-based attack is an exploit in which a malicious actor takes advantage of vulnerabilities in the program used to allow multiple operating systems to share a single hardware processor. If attackers gain command of the hypervisor, all the VMs and the data accessed by them will be under their full control to utilize. Furthermore, it could compromise the control of the underlying physical system and the hosted applications. Some well-known attacks (e.g., Bluepill, Hyperjacking, etc.) insert VM-based rootkits that can install a rogue hypervisor or modify the existing one to take complete control of the environment. Since the hypervisor runs underneath the host OS, it is difficult to detect these sorts of attacks using regular security measures.

VM-based threats include VM escape, where malicious actors break the isolated boundaries of the VM and start communicating with the operating system directly by passing through the virtual machine manager (VMM) layer, such an exploit opens the door to attackers to gain access to the host machine and launch further attacks. VM sprawls, which occurs when numerous VMs exist in the environment without proper management or control, and since they retain the system resources (i.e., memory, disks, network channels etc.) during this period, these resources cannot be assigned to other VMs. Cross VM-side channel attacks, when a malicious VM penetrates the isolation between VMs, and then access the shared hardware and cache locations to extract confidential information from the target VM. VM image threats comprise inside-VM attacks, where a VM image is infected with malware or OS rootkits at run-time; and outdated software packages in VMs, that can pose serious security threats in the virtualized environment, for example, a machine rollback operation may expose a software bug that has already been fixed [25].

Regarding the container management security threats, the two major types of risks that we examined are the compromise of an image or container and the misuse of a container to attack other containers, the host OS and other hosts. A container image that is missing critical security updates, or has an improper configuration, embedded malware or clear text secrets, can be the target of exploitation that compromised the security of the rest of the system. Likewise, images often contain sensitive components like an organization's proprietary software and embedded secrets. If connections to registries are performed over insecure channels, the contents of images are subject to the same confidentiality risks as any other data transmitted in the clear [26]. By default, in most container runtimes, individual containers are able to access each other and the host OS over the network. If a container is compromised and acting maliciously, allowing this network traffic may risk other resources in the environment. Moreover, a container running in privileged mode has access to all the devices on the host, thus allowing it to essentially act as part of the host OS and impact all other containers running on it.

7) MULTI-ACCESS EDGE COMPUTING

The development of Multi-access Edge Computing enables cloud computing capabilities to the edge of mobile networks, allowing service providers to deploy applications to end-users. One of the main security challenges is the need for open APIs, to provide maintenance for federated services that developers use to serve contents to MEC applications and end-users [11], [43]. The adoption of open APIs often create vulnerabilities that an attacker could exploit by impersonating third parties to launch diverse attacks on the MEC ecosystem. For example DoS and man-in-the-middle (MitM) attacks, malicious mode problems, privacy leakages, and VM manipulation. ENISA studies how virtualization technologies' threats (described in the section above) affect MECs. Additionally, they state that common hardware resources can lead to cross-contamination; exploitation of vulnerabilities in the common host platform, Container-as-a-Service (CaaS) and Platform-as-a-Service (PaaS) also impact the security [22].

8) RADIO ACCESS NETWORK CONTROL

ENISA in its Threat Landscape has resumed the 5G security issues related to RAN control [34].

A base station (BS) could be compromised by a malicious actor masquerading a legitimate user, this would facilitate different types of attacks, such as man-in-the-middle or network traffic manipulation. Besides, a compromised access network could enable an attacker to forge configuration data and launch other actions, for example, DoS attacks. The illegal use of the spectrum resources could also allow an attacker to take a specific spectrum band by imitating a legitimately licensed unit.

Flooding of radio interfaces with requests can consume component resources, and consequently reduce the quality or provoke the complete shutdown of the radio frequency provided by the component. Jamming the radio frequency by disrupting the network radio frequency (NRF) could cause the core network and related services to become inaccessible for affected users. Finally, signalling manoeuvres has also been cited, for example signalling the interconnection between networks for fraud. Signalling storms can be launched by malware or apps, and they can overload the bandwidth at the cell, the backbone signalling servers, cloud servers, and may also deplete the battery power of mobile devices.

Additionally, attacks based on false buffer status reports has been reported [44]. In this case, an attacker can exploit the buffer status reports of access network components such as Basic Service Sets (BSS) to obtain information (e.g. packet scheduling, load balancing, and admission control algorithms). Moreover, they could perform message insertion attack to initiate a DoS attacks; as well as, the exploitation of micro BSs that are not as physically secure as macro BSs used in pre-5G networks.

Finally, Radio Resource Control (RRC) protocol-based attacks at the Control Plane have been described [37]. The RRC protocol is a control protocol that manages a radio bearer of the L3 layer, related to radio resource establishment, reconfiguration, and announcement between the UE and RAN. Through the RRC protocol attack, attackers are able to launch various attacks, such as subscriber ID tampering, DoS attacks against BSs, and authentication bypass.

9) IoT DEVICES

Cybercriminals are constantly searching for vulnerabilities in IoT devices for opportunities to steal information, extort businesses, and take control of computer systems remotely.

An optimal security approach should protect the services, hardware resources, information and data, both in transmission and storage. However, this is very challenging since IoT devices are diverse compared to traditional computing devices, making them more vulnerable in different ways [52]. For example, most IoT devices are designed to be deployed at a massive scale, creating a network of nearly identical appliances with similar characteristics. Thus, this similarity amplifies the magnitude of any vulnerability in the security that may significantly affect many of them. Similarly, the expected number of links interconnected between the IoT devices is unprecedented, and many of these devices can establish connections and communicate with other devices automatically. This nature of interconnection on IoT devices implies that if a device is poorly secured and connected, it can affect the security and the resilience of the whole network.

Added to these, most IoT user-interactive applications are web and/or mobile-based, designed mostly with application programming interface (API), using PHP, Java, XML, and HTML. An unpatched API may be susceptible to various attacks exposing the entire system to malicious attacks [45]. IoT-based attacks may come in various forms, and the most common of them are the following [46]: (i) physical attacks, that affect hardware components, (ii) reconnaissance attacks, that comprise unauthorized discovery and mapping of systems, services, (iii) denial-of-service attacks, (iv) access attacks, where unauthorized persons gain unauthorized access of devices, (v) malware, that intent to hijack sensors' functions and spread in the IoT infrastructure, (vi) botnets, (vii) ransomware, targeting data storage, and (ix) privacy attacks, such as data mining on databases, cyberespionage, eavesdropping, tracking a user's movements, or password-based attacks.

IV. NETWORK MUTATION ONTOLOGY

We believe that network protocols can be defined using three main concepts: syntax, semantic, and behaviour. **Syntax** defines the sequencing of data elements or bits that are valid, and determines how to read the data in the form of fields. **Semantic** refers to the interpretation or meaning that computers give to each field. **Behaviour** considers the data in its context, this is when the data should be sent, and how fast for non-functional behavioural aspects.

Let us consider, as a running example, the Transmission Control Protocol (TCP). The TCP syntax is the group of

rules that defines that a TCP header is composed by at least 20 bytes, of which the first 2 bytes are a piece of information, that must be interpreted as a field apart from the following 2 bytes that correspond to another piece of information, etc. The semantics of the protocol gives to the first 2 bytes in the header the meaning of the *source port number*, followed *the destination port number*, etc. The behaviour determines that, for instance, a TCP ACK signal must arrive after a TCP SYN signal.

In a mutation-based fuzzing testing strategy, the idea is to generate new test cases, called also mutants, by making syntactic changes in already existing test cases, to therefore inject them into the system under test. Ideally, the mutants must be syntactically correct in order to discard test cases that the systems under test cannot interpret [53]. Moreover, mutants can also be generated by modifying the inputs from a behaviour' perspective. Continuing with the TCP example, supposing that the system under test is a TCP server, and a test case is a group of TCP packets, that semantically constitute the TCP handshake message exchange. A behavioural mutant of the test case can be the same message exchange, but with the client acknowledgment message before the server synchronize message. The client acknowledgment message would be syntactically and semantically correct, but it would violate the behaviour of the protocol.

In our ontology, mutant operators can only make changes of the syntax, and behaviour, the data in its context, of an object. Semantic changes, the interpretation that computers give to each field, are not possible as the ultimate objective of this ontology is to represent a testing process, and modifying the interpretation or meaning of a message would imply to modify the system under test.

As described in Section II-A, ontologies can be defined by formal and informal methods. Figure 2 is a UML diagram that summarizes the classes of our ontology and the relationship between them. Meanwhile, Section IV-A formally defines them.

A. CLASSES

In the following subsection we present some definitions that will be used throughout our mutation ontology:

- Let Pr denote a **network protocol** formed by i number of **fields**, which are ordered pairs of **field names** $FD = \{fd1, fd2, \ldots, fdi\}$ with input domain $Dfd = \{Dfd1, Dfd2, \ldots, Dfdi\}$, and **field values** $V = \{value1, value2, \ldots, valuei\}$, such that $Pr = \{(fd1, value1), (fd2, value2), \ldots, (fdi, valuei)\}$, where $valuei \in Dfdi$
- Let P denote a **network packet** formed by n number of network protocols with input domain Dp, such that $P = \{pr1, pr2, \ldots, prn\}$, where $prn \in Dp$
- Let F denote a **flow of network packets** formed by x number of network packets with input domain Df, such that $F = \{P1, P2, \ldots, Px\}$, where $Px \in Df$

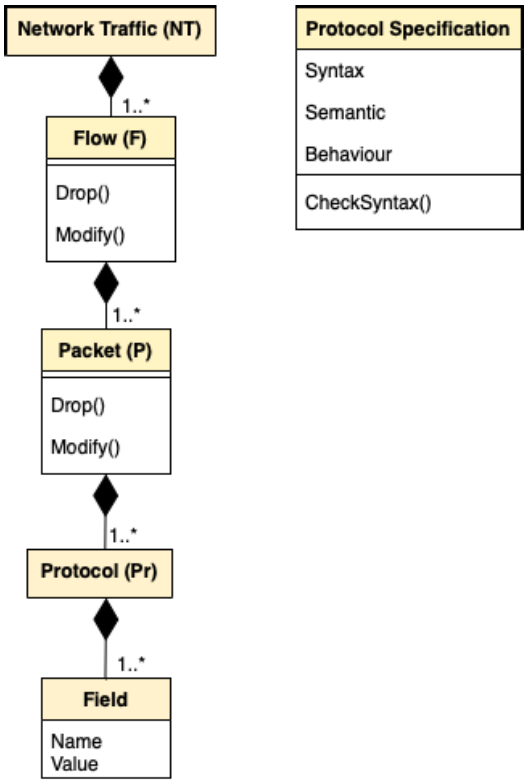


FIGURE 2. Summary of the basic definitions of our ontology (left), and instantiation using 5G protocols (right). Colors differentiate different senders.

- Let NT denote a **network traffic** formed by y number of flow of network packets with input domain Dnt, such that $NT = \{F1, F2, \ldots, Fy\}$, where $Fy \in Dnt$

Figure 3 exemplify the four classes and instantiate them, using the TCP protocol.

B. NOTIONS OF INDEPENDENCE

In the following subsection, we define the independence notions applied to the four basic concepts we defined in Section IV-A.

1) FIELDS

Two fields are independent, if a change in the field value of one does not affect the field value of the other. This notion applies for fields in the same and in different protocols in the same packet, or for fields in different packets. We represent this as: $fdi \perp\!\!\!\perp fdj$. We read the last expression as "fdi is independent of fdj", and $fdi \perp\!\!\!\perp fdj$ do not imply $fdj \perp\!\!\!\perp fdi$.

For example, in a TCP packet, the TCP field *checksum* is used for error-checking of the TCP header, the payload, and an IP pseudo-header. The calculation of this checksum involves the length of the TCP headers and payload in bytes. Therefore, the TCP checksum is dependent on the rest of the TCP fields. On the other hand, if the TCP source port changes, the destination port remains the same and vice versa, therefore, they are independent fields.

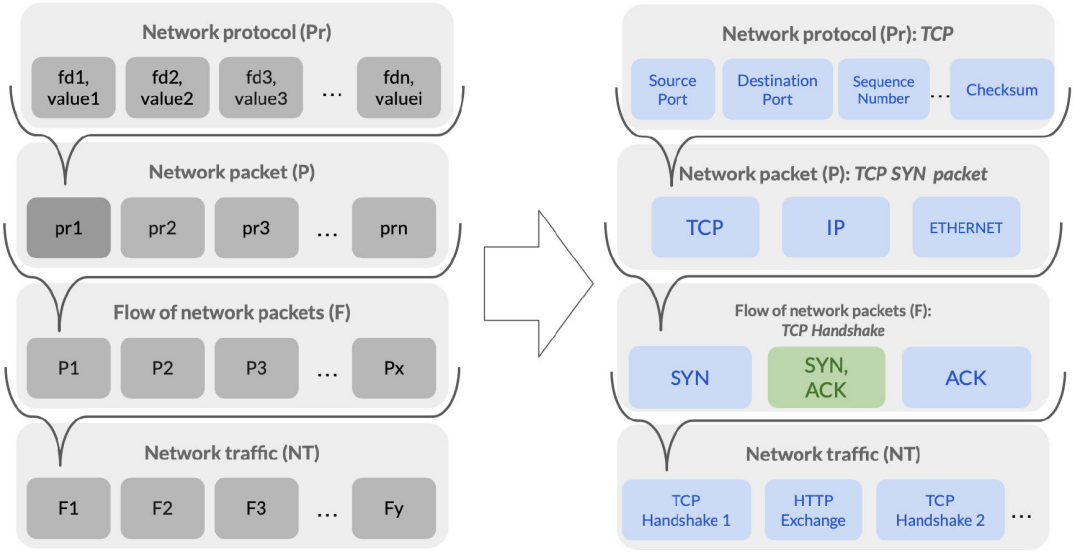


FIGURE 3. Summary of the basic definitions of our ontology (left), and instantiation using TCP protocol (right). Colors differentiate different senders.

2) PROTOCOLS

Two protocols are independent if all their fields are independent. We represent this as: $pdi \perp\!\!\!\perp pdj$. We read the last expression as "pdi is independent of pdj", and $pdi \perp\!\!\!\perp pdj$ do not imply $pdj \perp\!\!\!\perp pdi$.

For example, in a TCP packet, the calculation of the checksum also involves the source IP address, the destination IP address, and the protocol number for the TCP protocol. A TCP field depends on IP fields, therefore, TCP and IP are protocol dependent. On the other hand, if any field of the Ethernet protocol changes, no field of TCP would be affected, therefore, TCP protocol is independent of Ethernet protocol.

3) PACKETS

Two packets are independent if all their protocols are independent. We represent this as: $Pi \perp\!\!\!\perp Pj$. We read the last expression as "Pi is independent of Pj", and $Pi \perp\!\!\!\perp Pj$ do not imply $Pj \perp\!\!\!\perp Pi$.

For example, in a TCP handshake, if the IP source on the TCP synchronization (TCP SYN) packet changes, the IP destination in the correspondent TCP acknowledgment (TCP ACK) packet, must be changed too, to preserve the protocol proper behaviour. Therefore, TCP ACK packet is dependent on TCP SYN packet. However, such a change would not affect a synchronization signal of separate TCP exchange, then this two TCP SYN packets would be independent.

4) FLOW OF PACKETS

Two flow of packets are independent if all their packets are independent. We represent this as: $Fi \perp\!\!\!\perp Fj$. We read the last expression as "Fi is independent of Fj", and $Fi \perp\!\!\!\perp Fj$ do not imply $Fj \perp\!\!\!\perp Fi$.

For example, two separate IP handshakes are independent flows. In the case of HTTP request response exchange flow, by design it is preceded by a TCP handshake flow, and a modification in one of the TCP headers of the packets involved would affect the HTTP exchange. Therefore, HTTP request response exchange flow is dependent on a TCP handshake.

C. MUTANTS

Let P', F', NT' be a syntactically correct network packet, flow of network packets, and network traffic respectively, obtained by making a syntactic, or behavioural changes of P, F, NT. P', F', NT' are known as **mutants** of P, F, NT.

D. MUTANT OPERATORS TYPES

Let R be a rule according to which P, F, or NT are changed. R is known as a **mutant operator**, and it is composed by a context γ and an action σ such that $R = \{\gamma, \sigma\}$.

1) CONTEXT

The **context** determine if the operation will be performed in a single packet, a flow of packets, or the whole network traffic. It is a filter function γ of NT with image equal to

1) a single network packet: $\gamma : NT \rightarrow P$,
2) a flow of network packets: $\gamma : NT \rightarrow F$, or
3) a whole network traffic: $\gamma : NT \rightarrow NT$

For describing the context, we use the following notation:

$$\gamma(NT) = [Level].[protocol].[field].[value\ type].[value] \tag{1}$$

More details of the notation are provided in Table 3.

Moreover, the γ function can be formed of several expressions of the type of equation 1 joined by logical operators, such as **and**, and **or**. For example:

TABLE 2. Mutant operator actions.

Type	Name	Formal name	Description	Atomic action involved	5Greplay
Atomic	Drop	P_DROP	Delete a packet	-	X
	Modify	P_MODIFY	Change a specific attribute on the header of a network packet	-	X
Composed	Duplicate	P_DUPLICATE	Duplicate packet	-	X
	Delay packet	P_DELAY	Delay a single packet	P_MODIFY	X
	Rush packet	P_RUSH	Rush a single packet	P_MODIFY	X
	Permute packet	P_PERMUTE	Exchange the order of two consecutive packets	P_MODIFY	
	Drop flow	F_DROP	Delete all the packets in a flow of packets	P_DROP	X
	Duplicate flow	F_DUPLICATE	Duplicate flow of packets	P_DUPLICATE	X
	Modify flow	F_MODIFY	Change a specific attribute on the header of a flow of packets	P_MODIFY	X
	Permute flow	F_PERMUTE	Exchange the order of two consecutive flow of packets	P_MODIFY	
Complex	Simple packet fuzzing	P_FUZZING	Fuzzing changing randomly field in the protocol header of packets	P_MODIFY	X
	Simple packet fuzzing in time	P_FUZZING_TIME	Fuzzing changing randomly the position packets inside a flow	P_PERMUTE	
	Simple flow fuzzing	F_FUZZING	Fuzzing changing randomly the position of flows inside a network traffic	F_PERMUTE	
	Denial-of-service	DOS	Send messages at the fastest rate the machine can until it breaks the system under test	P_DUPLICATE	X
	Distributed Denial of Service	DDOS	Send messages at the fastest rate the machine can, while changing the IP addresses, until it breaks the system under test	P_DUPLICATE, P_MODIFY	X

TABLE 3. Context notation.

Notation	Shortcut	Description
Level	P	Individual packet
	F	Flow of network packets
Protocol		Network protocol
Field		Network protocol field
Value type	S	String
	R	Range
	RE	Regular Expression
Value		Field Value

- $\gamma(NT) = P.IP.ip\_source.S.8.8.8.8$ is a function that filters all the network packets with an IP source address equal to 8.8.8.8
- $\gamma(NT) = F.TCP.tcp\_port.R.[80 : 88]$ is a function that filters all the network flows with TCP port in range between 80 and 88, including them
- $\gamma(NT) = P.TCP.tcp\_port.RE.. * ([0-9] * 8)\$$ is a function that filters all the network packets with TCP port that end in 8
- $\gamma(NT) = P.IP.ip\_source.S.8.8.8.8$ **and** $P.TCP.tcp\_port.RE.. * ([0-9] * 8)\$$ is a function that filters all the network packets with an IP source address equal to 8.8.8.8 and TCP port that ends in 8 *ip_source*

2) ACTION

The **action** is a function σ of P, F, *or* NT that defines the action that will be applied to the results of the context function. We introduce the three types of actions, that are also summarized in Table 2.

Atomic actions are the most basic ones, and they can not be deduced using other actions, and they are utilized to build composed and complex actions. In our ontology, all the action can be deduced from the DROP and MODIFY actions, applied at the packet level.

Composed actions are combinations of single atomic actions, repeated once or several times. Basic actions performed to a flow or the entire network traffic, can be seen as a composition of atomic actions, therefore, they enter this category. In this category we included the duplication, permutation, and the acceleration or delay in time of packets; as well as, dropping, duplicating, modifying, and permuting a flow of packet. Further composed actions could be proposed using our ontology.

Finally, complex actions are combinations of different atomic or composed actions, and they aim to assist in the generation of complex cyberattacks or test cases. We propose fuzzing a packet field values, the ordering of packets in a flow, and the ordering of flows in the network traffic; as well as, standard and distributed denial of service attacks. Further complex actions could be proposed using our ontology.

In the following subsections, we present the formalization of atomic, composed, and complex actions. For such a purpose, let us consider a network traffic $NT = \{F1, F2, \ldots, Fi, Fj, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, Pj, \ldots, Px\}$, and $Pi = \{pr1, pr2, \ldots, pri, \ldots, prh, \ldots, prn\}$. NT will be muted according to a mutant operator R, with context γ, and action σ. In the following subsections, we present the formalization of the actions we propose to create syntactically correct mutants of NT.

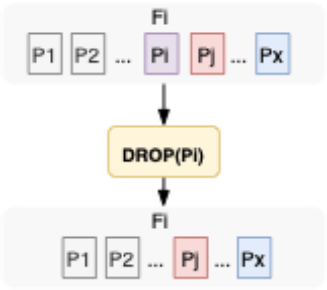


FIGURE 4. Drop packet action.

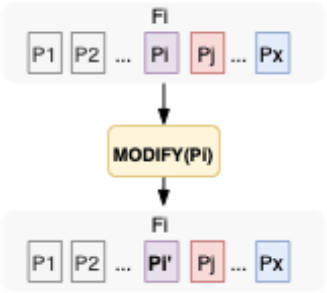


FIGURE 5. Modify packet action.

E. ATOMIC ACTIONS

- **Drop packets:** Drop a packet (see Figure 4).
  $\sigma = P\_DROP(Pi)$
  $\Rightarrow Fx = \{P1, P2, \ldots, Pj, \ldots, Px\}$
- **Modify packets:** Change the value of a field *fdi*, in the protocol *pri*, of the packet Pi, for the value $valuei'$. This operator does not consider dependency relationships between fields, protocols, packets, or flows. As a result, we get a mutation of Pi, that we call Pi' (see Figure 5).
  $\sigma = P\_MODIFY(Pi, pri, fdi, valuei')$
  $\Rightarrow Fx = \{P1, P2, \ldots, Pi', Pj, \ldots, Px\}$ where $Pi' = \{pr1, pr2, \ldots, pri', \ldots, pn\}$, with $pri' = \{(fd1, value1), (fd2, value2), \ldots, (fdi, valuei')\}$

F. COMPOSED ACTIONS

Two actions f and g produces a new action h such that $h(Px) = g \circ f(Px) = g(f(Px))$. In this operation, the action g is applied to the result of applying the function f to Px. Action composition is not necessarily commutative. Successive transformations applying and composing to the right agrees with the left-to-right reading sequence. Composed actions are a special case or a composition of several repeated atomic actions. The following operators can be defined as a composition of the atomic actions written in Table 2. For example:
$F\_DROP(Fx) = P\_DROP(P1) \circ P\_DROP(P2) \circ \ldots \circ P\_DROP(Px) = \{\}$

- **Duplicate packets:** Duplicate a packet or a group of packets.
  $\sigma = P\_DUPLICATE(Pi)$
  $\Rightarrow Fx = \{P1, P2, \ldots, Pi, Pi, Pj, \ldots, Px\}$
- **Permute packets:** Swap the position of Qx with Pi.
  $\sigma = P\_PERMUTE(Qx, Pi)$
  - **Permute two single packets:** Qx is a single packet Px.
    $\Rightarrow Fx = \{P1, P2, \ldots, Px, Pj, \ldots, Pi\}$
  - **Permute a group of packets:** Qx is a group of packets $\{Pi, Pj\}$.
    $\Rightarrow Fx = \{P1, P2, \ldots, Px, \ldots, Pi, Pj\}$
- **Drop flows:** Drop all the packets contained in a flow of packets.
  $\sigma = F\_DROP(Fi)$
  $\Rightarrow NT = \{F1, F2, \ldots, Fj, \ldots, Fy\}$
- **Duplicate flows:** Duplicate all the packets in a flow of packets.
  $\sigma = F\_DUPLICATE(Fi)$
  $\Rightarrow NT = \{F1, F2, \ldots, Fi, Fi, Fj, \ldots, Fy\}$
- **Permute flows:** Swap two flow of packets.
  $\sigma = F\_PERMUTE(Fx, Fi)$
  $\Rightarrow NT = \{F1, F2, \ldots, Fy, Fj, \ldots, Fi\}$
- **Modify flows:** Change the value of a field *fdi*, in the protocol *pri*, in all the packets Pi of the flow Fx, for the value $valuei'$. As a result, we get a mutation of Pi, that we call Pi'.
  $\sigma = F\_MODIFY(Fi, Pi, pri, fdi, valuei')$
  $\Rightarrow NT = \{F1, F2, \ldots, Fi', Fj, \ldots, Fy\}$, where
  $Fi' = \{P1, P2, \ldots, Pi', \ldots, Px\}$

G. COMPLEX ACTIONS

Complex action differentiates from composed action because they aim to implement specific cyber-attacks or tests. They can be seen as a set of predefined scenarios that 5Greplay includes to facilitate its use. Complex actions are made by the composition of composed actions, and users can add more.

- **Simple packet fuzzing:** Fuzzing changing randomly field in the protocol header of packets, to generate several flow mutants to be injected into the system under test.
  $\sigma = P\_FUZZING(Pi)$
  $\Rightarrow Fx' = \{Pi, Pi', Pi'', \ldots, Pi''', \ldots\}$
- **Simple packet fuzzing in time:** Fuzzing changing randomly the position of packets inside a flow, to generate several flow mutants to be injected into the system under test.
  $\sigma = P\_FUZZING\_TIME(Pi)$
  $\Rightarrow Fx' = \{Pi, P1, P2, \ldots, Pj, \ldots, Px\}$,
  $Fx'' = \{P1, Pi, P2, \ldots, Pj, \ldots, Px\}$,
  $Fx''' = \{P1, P2, Pi, \ldots, Pj, \ldots, Px\}, \ldots,$
  $Fx^n = \{P1, P2, \ldots, Pj, \ldots, Px, Pi\}$
- **Simple flow fuzzing:** Fuzzing changing randomly the position of flows inside a network traffic, to generate several network traffic to be injected into the system under test.
  $\sigma = F\_FUZZING(Fi)$

$\Rightarrow NT' = \{Fi, F1, F2, \ldots, Fj, \ldots, Fy\}$,
$NT'' = \{F1, Fi, F2, \ldots, Fj, \ldots, Fy\}$,
$NT''' = \{F1, F2, Fi, \ldots, Fj, \ldots, Fy\}, \ldots,$
$NT^n = \{F1, F2, \ldots, Fj, \ldots, Fy, Fi\}$

- **Denial-of-service:** Send messages at the fastest rate the machine can until it breaks the system under test.
  $\sigma = P\_DOS(Pi) \Rightarrow Fx' = \{Pi, Pi, Pi, \ldots, Pi, \ldots\}$
- **Distributed Denial-of-Service:** Send messages at the fastest rate the machine can, while changing the IP addresses, until it breaks the system under test.
  $\sigma = P\_DDOS(Pi) \Rightarrow Fx' = \{Pi, Pi', Pi'', \ldots, Pi^n\}$

H. THEOREMS

From our formalism definitions, the following theorems can be deduced. These theorems allow optimization of the mutation process by reducing the possible number of mutants, and they are easily provable through mathematical manipulations.

1) PERMUTATION is **commutative** operation: The arguments of the PERMUTATION operation can be exchanged without altering the result
2) DUPLICATE and MODIFY are **commutative**: Both mutant operators can be exchanged without altering the result
3) Composition with DROP operator is always equal to an **empty flow**: Any operator composition including the DROP operator will be equal to an empty flow

I. 5G NETWORK FUZZER

We embody our ontology in our network traffic fuzzer, 5Greplay.[2] It is an open-source solution that generates *mutants* of the network traffic by using *mutant operators*, in order to perform specified security and functional tests on a system, as well as, fuzzing testing. In [2] we presented a first version of the tool, its rule syntax, and original mutant operators, that we extend in this article.

5Greplay main workflow is depicted in Figure 6. The input of 5Greplay is network traffic, in the form of a pcap file or a live network data, a set of mutation rules, and a configuration file. Once a packet is processed by the tool, the context written by the user in the mutation rule will determine if the packet will be mutated or not. Then, if the packet must be mutated, the action, which is also embedded in the mutation rules, will determine which mutation operator must be used to mutate the packet. Finally, the packet is forwarded to the output NIC, together with the non-modified packets, depending on the default action, contained in the configuration file. Moreover, Algorithm 1 formally shows this workflow.

Today, 5Greplay can operate over NAS-5G and NGAP protocols. However, the tool incorporates a plugin architecture for the addition of new protocols. In order to perform different mutations on the incoming 5G traffic, 5GReplay defined a set of mutation operator that can be applied either on the packet or on the flow levels. The list of these operators are provided in the Table 2.

[2]http://5greplay.org

Algorithm 1 5Greplay Main Process

Input: Network traffic (NT) coming from online or offline traffic, mutant operator (R), composed of a context (γ) and an action (σ)
Result: Mutation of input traffic (NT',Fx',or Px')
for $\forall P$ *in* F, $\forall F \in NT$ **do**
  if *5Greplay turned on* **then**
    if $\gamma(NT) == P$ **then**
      $\sigma = P\_ACTION(P)$
    end
    if $\gamma(NT) == F$ **then**
      $\sigma = F\_ACTION(F)$
    end
    if $\gamma(NT) == NT$ **then**
      $\sigma = NT\_ACTION(NT)$
    end
  end
end

J. FUZZING TESTING

As a first approach, we proposed the combination of mutant operators to be performed randomly. Although inefficient in terms of the number of mutations that we could generate, this first approach has allowed us to find vulnerabilities in the 5G core (see Section V). Furthermore, thanks to the theorems proposed in Section IV-H, the number of possible mutants can be reduced.

V. EXPERIMENTAL EVALUATION

To illustrate our ontology, we have formalized the experiments we already performed in our previous work, where we presented our 5Greplay fuzzer [2].

We performed these scenarios against two 5G core open-source solutions, free5GC and open5GS. In both cases, we used the RAN simulator UERANSIM.

A. NAS REPLAY ATTACK

Attackers, with access to the NAS traffic in the 5G interface N1, could intercept a NAS SMC *Security Mode command* clear message sent from the AMF to the UE, copy its NAS sequence number (NAS SQN), and use it to build a NAS SMC *Security Mode complete* message that is replayed to the AMF, or directly intercept a NAS SMC *Security Mode complete* message and replay it to the AMF. If the AMF does not implement a proper integrity protection against this type of attack, the network will not drop the replayed packet. We depict this scenario in the Figure 7.

To perform the NAS-5G SMC Replay attack, a malicious actor must perform the two following actions:

- Duplicate a NAS SMC packet with a specific *Security Mode Complete* field. (NAS_5G.message_type == 93 means that it is an SMC packet)
- Change the value of this *Security Mode Complete* field to a lower level and recompute the checksum of the packet.

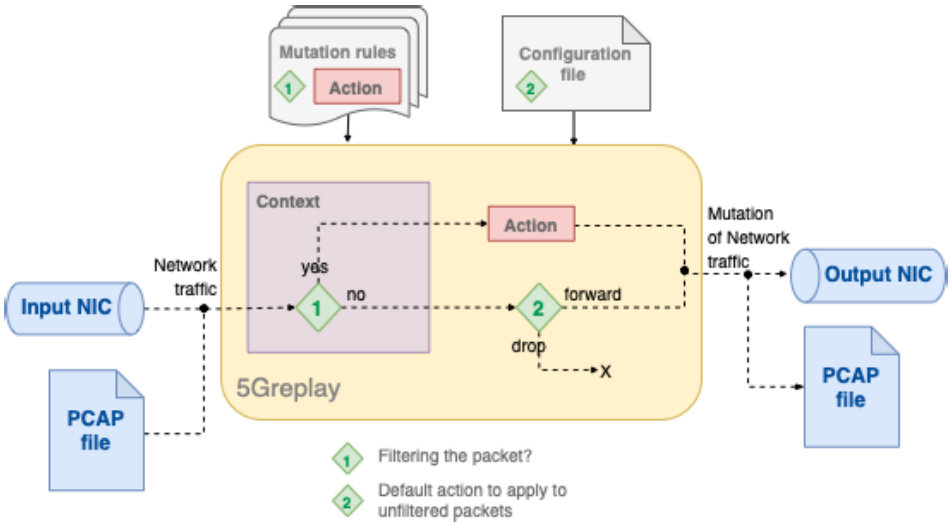


FIGURE 6. 5G replay main architecture.

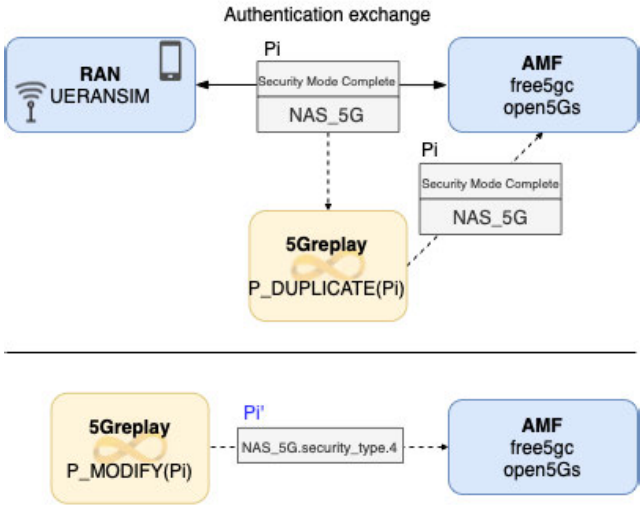


FIGURE 7. NAS-5G SMC Replay Attack. Top: Checking if the AMF is vulnerable to replay attacks. Down: exploiting the vulnerability to change the security level on the network.

(for instance, NAS_5G.security_type == 4 means that there will be no encryption)

The **formalization** of this threat is a follow:

Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{pr1, pr2, \ldots, pri, \ldots, pn\}$, where pri corresponds to the NAS_5G protocol and $pri = \{(fd1, value1), (fd2, value2), \ldots, (message\_type, 93), \ldots, (fdi, valuei)\}$. And $R = \{\gamma, \sigma\}$ a mutant operator, according to which a subset of NT will be filtered, and mutated.

if $\gamma(NT) == P.NAS\_5G.message\_type.S.93 \rightarrow Pi$
$\Rightarrow \sigma = P\_MODIFY(Pi, NAS\_5G, security\_type, 4) \circ P\_DUPLICATE(Pi)$
$\Rightarrow NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$,
$Fi = \{P1, P2, \ldots, Pi', Pi', \ldots, Px\}$,
$Pi = \{pr1, pr2, \ldots, pri', \ldots, pn\}$, and
$pri' = NAS\_5G = \{(fd1, value1), (fd2, value2), \ldots, (message\_type, 4), \ldots, (fdi, valuei)\}$.

We implement a mutant operator in 5Greplay with context: NAS SMC *Security Mode complete* messages sent by the UE after its authentication; and action: replay it twice to the AMF. Then, we checked the AMF logs, and we monitored the network to verify that the AMF actually received the same packet twice.

After the NAS SMC *Security Mode Command* message, the AMF received a legitimate NAS SMC *Security Mode complete message*, and two NGAP packets with the same UE NGAP ID as the legitimate user. The AMF identified this as not belonging to the same NGAP security context. These two packets corresponded to the replayed packets by 5Greplay and allow us to conclude that the free5Gc AMF is protected again this type of replay attack.

As depicted in the free5Gc AMF log and shown in Figure 8, after the NAS SMC *Security Mode Command* message, the AMF received a legitimate NAS SMC *Security Mode complete message*, and two NGAP packets with the same UE NGAP ID as the legitimate user. The AMF identified this as not belonging to the same NGAP security context. These two packets corresponded to the replayed packets by 5Greplay and allow us to conclude that the free5Gc AMF is protected again this type of replay attack.

B. DENIAL OF SERVICE BY SENDING MALFORMED NGAP PACKETS

This threat intends to check the robustness of the AMF function by sending inconsistent values of NGAP protocol sent over SCTP protocol. In this case, we utilize 5Greplay is his fuzzing mode. For instance, we can change the SCTP protocol identifier from 60 to 0, and put the UE identifier of NGAP to an arbitrary value. We depict this scenario in the Figure 9.

```
2021-05-19T08:40:28-07:00 [INFO][AMF][GMM][AMF_UE_NGAP_ID:7][SUPI:imsi-208930000000003] Send Security Mode Command
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP][192.168.49.4:35118][AMF_UE_NGAP_ID:7] Send Downlink Nas Transport
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP][192.168.49.4:35118] Handle Uplink Nas Transport
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP][192.168.49.4:35118][AMF_UE_NGAP_ID:7] Uplink NAS Transport (RAN UE NGAP ID: 2)
2021-05-19T08:40:28-07:00 [INFO][AMF][GMM][AMF_UE_NGAP_ID:7][SUPI:imsi-208930000000003] Handle Security Mode Complete
2021-05-19T08:40:28-07:00 [INFO][AMF][GMM][AMF_UE_NGAP_ID:7][SUPI:imsi-208930000000003] Handle InitialRegistration
2021-05-19T08:40:28-07:00 [INFO][NRF][DSCV] Handle NFDiscoveryRequest
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP] Create a new NG connection for: 192.168.49.4/172.16.151.12/10.45.0.1:49183
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP][192.168.49.4/172.16.151.12/10.45.0.1:49183] Handle Uplink Nas Transport
2021-05-19T08:40:28-07:00 [ERRO][AMF][NGAP][192.168.49.4/172.16.151.12/10.45.0.1:49183] No UE Context[RanUeNgapID: 2]
2021-05-19T08:40:28-07:00 [INFO][AMF][NGAP][192.168.49.4/172.16.151.12/10.45.0.1:49183] Handle Uplink Nas Transport
2021-05-19T08:40:28-07:00 [ERRO][AMF][NGAP][192.168.49.4/172.16.151.12/10.45.0.1:49183] No UE Context[RanUeNgapID: 2]
```

FIGURE 8. Free5Gc AMF log when replaying Security Mode Complete messages (SMC).

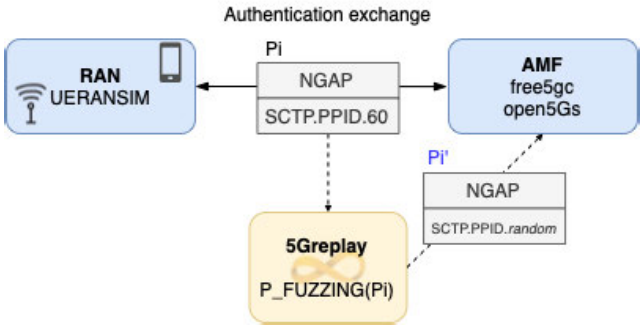


FIGURE 9. Fuzzing NGAP protocol packets.

To perform this attack, we must perform the two following actions:

- Change the value of the SCTP protocol identifier to a random value, for example 0-. This identifier should be SCTP.protoid == 60 (see Figure 9)
- Change the value of the UE identifier field to a random value. (for instance, NAS_5G.amf_UE_id == 1234 which is a random value)

This actions can be done for an authentication response identified by a message type $NAS\_5G.message\_type == 93$. The **formalization** of this threat is a follow:

Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{pr1, pr2, \ldots, pri, prj, \ldots, pn\}$, pri corresponds to the $SCTP$ protocol, and $pri = \{(fd1, value1), (fd2, value2), \ldots, (proto\_id, 60), \ldots, (fdi, valuei)\}$; prj corresponds to the NAS_5G protocol, and $prj = \{(fd1, value1), (fd2, value2), \ldots, (amf\_UE\_id, 93), \ldots, (fdi, valuei)\}$. And $R = \{\gamma, \sigma\}$ a mutant operator, according to which a subset of NT will be filtered, and mutated.

if $\gamma(NT) == P.NAS\_5G.message\_type.S.93 \rightarrow Pi \Rightarrow \sigma = P\_MODIFY(Pi, NAS\_5G, amf\_UE\_id, 1234) \circ P\_MODIFY(Pi, SCTP, proto\_id, 0) \Rightarrow NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, $Fi = \{P1, P2, \ldots, Pi', \ldots, Px\}$, $Pi = \{pr1, pr2, \ldots, pri', prj', \ldots, pn\}$, $pri = SCTP = \{(fd1, value1), (fd2, value2), \ldots, (proto\_id, 0), \ldots, (fdi, valuei)\}$, and $prj = NAS\_5G = \{(fd1, value1), (fd2, value2), \ldots, (amf\_UE\_id, 1234), \ldots, (fdi, valuei)\}$.

We implement a mutant operator in 5Greplay with context: NGAP protocol messages sent by the UE during the authentication exchange; and action: replay them to the AMFs with two modification of the SCTP and NAS_5G fields. Then, we checked the AMF logs, and we monitored the network to verify that the AMF actually received the same packet twice.

When replaying against free5GC, we got an AMF warning, but the simulator keep running and allowed new UE connections, as showed in the Figure 10. Therefore, we conclude that it is protected against this type of malformed NGAP packets. On the other hand, open5GS was not able to handle this packet and the simulator crashed, preventing new connections to the AMF, as depicted in Figure 11.

C. 5G ENCAPSULATING IoT TRAFFIC

This scenario targets applications using 5G infrastructure. We focus on the 5G data plane, and we aim to encapsulate attack payloads in 5G application layer to determine what IP attacks can be realized on 5G, and the impact of attacks on 5G infrastructure. These are some examples of the research questions we can answer by performing the mentioned encapsulation, by means of the 5Greplay tool, implementing the presented mutation operators:

- Can 5G GTP prevent DoS attacks?
- What is the impact of attacks on 5G infrastructure?
- Can DoS attacks on one slice impact another isolated slice?

Figure 12 shows how an attack payload could be inserted into a 5G infrastructure, by means of adding it to the 5G protocol stack.

We depict the workflow of this scenario in Figure 13. Assuming we already have a pcap file containing malicious traffic of a IoT platform, for example, a DoS attack done by means of botnets. After using it as an input for 5Greplay, we use the tool for extracting the payload, modify it, and duplicating. Then, these messages are sent to a 5G core by means of UERANSIM, which is a simulator of the UE and the gnB of a part of the network. We use 5Greplay for manipulating the malicious traffic payload and injecting into UERANSIM which is going to send it to the core network as a UE message. Finally, the core network will send it to its target IoT services, and the attacks will be performed.

```
2021-05-17T07:24:16-07:00 [INFO][AMF][NGAP][192.168.49.4:54593][AMF_UE_NGAP_ID:1] Uplink NAS Transport (RAN UE NGAP ID: 1)
2021-05-17T07:24:16-07:00 [INFO][AMF][GMM][AMF_UE_NGAP_ID:1][SUPI:imsi-208930000000003] Handle Security Mode Complete
2021-05-17T07:24:16-07:00 [INFO][AMF][GMM][AMF_UE_NGAP_ID:1][SUPI:imsi-208930000000003] Handle InitialRegistration
2021-05-17T07:24:16-07:00 [INFO][NRF][DSCV] Handle NFDiscoveryRequest
2021-05-17T07:24:16-07:00 [WARN][AMF][NGAP] Received SCTP PPID != 60, discard this packet
2021-05-17T07:24:16-07:00 [WARN][AMF][NGAP] Received SCTP PPID != 60, discard this packet
```

FIGURE 10. Free5GC AMF log when receiving a malformed NGAP packet.

```
05/12 17:23:47.069: [gmm] INFO: [suci-0-901-70-0000-0-0-0000000001]    SUCI (../src/amf/gmm-handler.c:72)
05/12 17:23:47.069: [amf] WARNING: GUTI has already been allocated (../src/amf/context.c:1045)
05/12 17:23:47.070: [gmm] ERROR: Invalid service name [nudm-sdm] (../src/amf/gmm-sm.c:625)
05/12 17:23:47.070: [gmm] WARNING: gmm_state_authentication: should not be reached. (../src/amf/gmm-sm.c:626)
05/12 17:23:47.070: [core] FATAL: backtrace() returned 9 addresses (../lib/core/ogs-abort.c:37)
/usr/bin/open5gs-amfd(+0x17418) [0x55f750b1d418]
/usr/lib/x86_64-linux-gnu/libogscore.so.2(ogs_fsm_dispatch+0x16) [0x7ff86bb4ec76]
/usr/bin/open5gs-amfd(+0x1bb4e) [0x55f750b21b4e]
/usr/lib/x86_64-linux-gnu/libogscore.so.2(ogs_fsm_dispatch+0x16) [0x7ff86bb4ec76]
/usr/bin/open5gs-amfd(+0x5ec6) [0x55f750b0bec6]
/usr/lib/x86_64-linux-gnu/libogscore.so.2(+0xd718) [0x7ff86bb46718]
/lib/x86_64-linux-gnu/libpthread.so.0(+0x76db) [0x7ff869f416db]
/lib/x86_64-linux-gnu/libc.so.6(clone+0x3f) [0x7ff869c6aa3f]
Open5GS daemon v2.2.7
```

FIGURE 11. Open5GS AMF log when receiving a malformed NGAP packet.

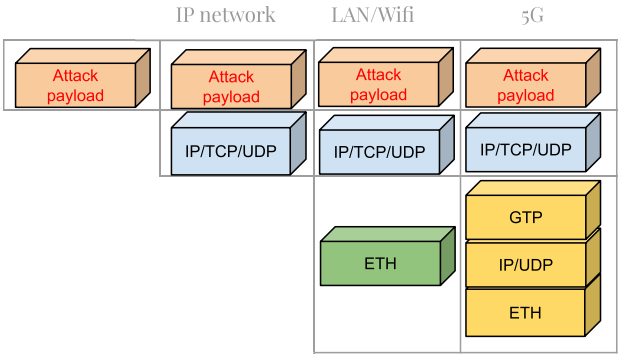


FIGURE 12. Attack payload encapsulation on 5G traffic headers.

Figure 14 depicts the original IoT traffic, and Figure 15 the packets after being encapsulated into the 5G protocol stack, proving that our manipulations worked well.

The **formalization** of the 5G traffic encapsulation, supposing the original traffic is working over TCP, is a follows:

Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{ETH, IP, TCP, \ldots, pn\}$, prn corresponds to the protocol used for the attack. And $R = \{\gamma, \sigma\}$ a mutant operator, according to which a subset of NT will be filtered, and mutated.

If $\gamma(NT) \rightarrow Pi$, where $\gamma(NT)$ is a function that allows to identify a single step of the attack that is desired to encapsulate in the 5G headers

$\Rightarrow \sigma = P\_MODIFY(Pi, ETH, *, \{ETH, IP, UDP, GTP\}) \circ P\_DUPLICATE(Pi)$

$\Rightarrow NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$,

$Fi = \{P1, P2, \ldots, Pi', Pi', \ldots, Px\}$, and

$Pi' = \{ETH, IP, UD, GTP, IP, TCP, \ldots, pn\}$

D. OTHERS

5Greplay has a plugin architecture that enables easily adding new protocols to its stack. Therefore, there is no logical impediment to implement the following threats, but an additional development is required. The following examples involve protocols that are not currently supported by 5Greplay, but that illustrate new use cases of our ontology.

1) SMF IMPERSONATION

Let us consider an attack scenario against a 5G user equipment (UE), reported by Positive Technologies [54]. In this scenario, an attacker impersonating the SMF sends a Session Deletion Request packet to the UPF, with the subscriber session identifier (SEID) of the victim. As a result, packet data transmission to the victim's device will stop, but the connection to the network will remain. By means of our ontology, we can describe this attack scenarios as follows:

Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{pr1, pr2, \ldots, pri, prj, \ldots, pn\}$, pri corresponds to the $PFCP$ protocol, and $pri = \{(fd1, value1), (fd2, value2), \ldots, (message\_type, valuex), \ldots, (fdi, valuei)\}$.

And, $R = \{\gamma, \sigma\}$ the mutant operator.

If $\gamma(NT) == P.PFCP.message\_type.RE.!54 \rightarrow Pi$

$\Rightarrow \sigma = P\_MODIFY(Pi, PFCP, message\_type, 54)$

$\Rightarrow NT = \{F1, F2, \ldots, Fi', \ldots, Fy\}$,

$Fi' = \{P1, P2, \ldots, Pi', \ldots, Px\}$,

$Pi' = \{pr1, pr2, \ldots, pri', prj, \ldots, pn\}$, and

$pri' = \{(fd1, value1), (fd2, value2), \ldots, (message\_type, 54), \ldots, (fdi, valuei)\}$.

2) DoS TO A UE VIA HTTP2

Let us consider an attack scenario against a 5G user equipment (UE), reported by Positive Technologies [54]. If a

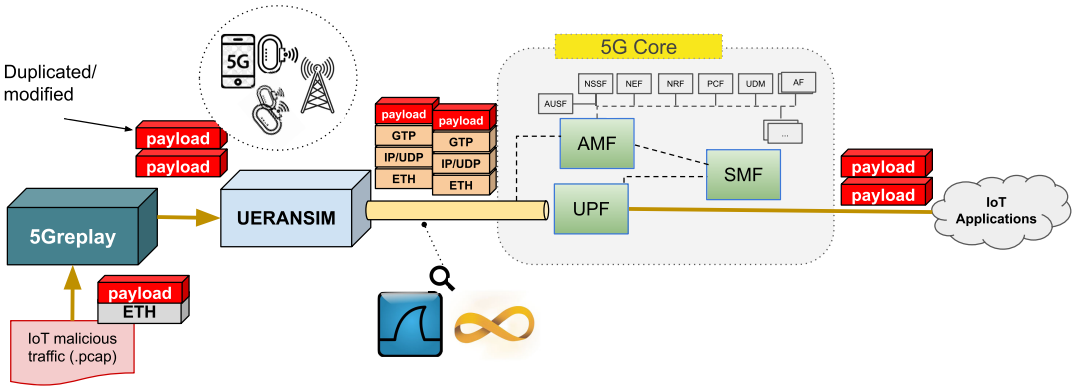


FIGURE 13. Encapsulation workflow.

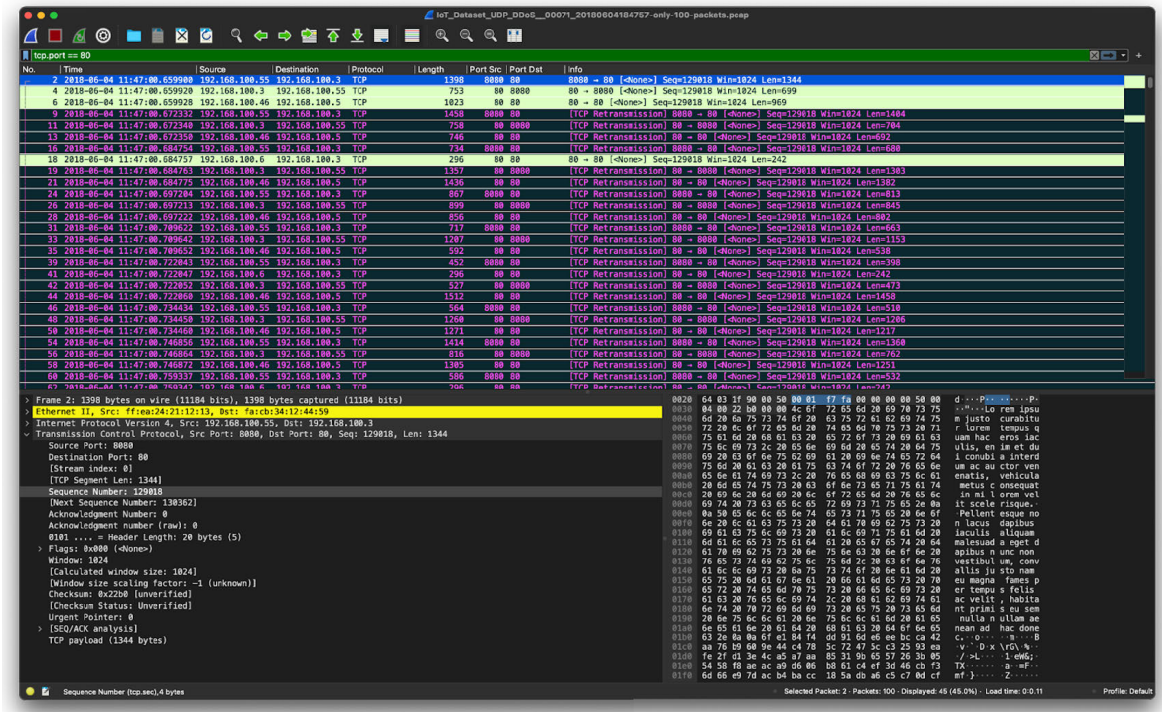


FIGURE 14. Encapsulation workflow.

NRF does not restrict the operations allowed on NF profiles, an attacker with NF profiles information could delete such profiles. To obtain the NF information, attackers can impersonate any network service for other NFs and obtain profile data, such as authentication status, current location, and subscriber settings for network access.

This threat has a CVSS base score of 7.4, which indicates high severity. By means of our ontology, we can describe this attack scenarios as follows:

- Obtaining the NF profile (nflnstanceID)
  Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{pr1, pr2, \ldots, pri, prj, \ldots, pn\}$, pri corresponds to the $HTTP2$ protocol, and $pri = \{(fd1, value1), (fd2, value2), \ldots, (method, valuex), \ldots, (fdi, valuei)\}$.
  And, $R = \{\gamma, \sigma\}$ the mutant operator.
  If $\gamma(NT) == P.HTTP2.method.RE.!GET \rightarrow Pi \Rightarrow \sigma = P\_MODIFY(Pi, HTTP2, method, GET) \Rightarrow NT = \{F1, F2, \ldots, Fi', \ldots, Fy\}$, $Fi' = \{P1, P2, \ldots, Pi', \ldots, Px\}$, $Pi' = \{pr1, pr2, \ldots, pri', prj, \ldots, pn\}$, and $pri' = \{(fd1, value1), (fd2, value2), \ldots, (method, GET), \ldots, (fdi, valuei)\}$.
- Using the obtained information (nflnstanceID) to delete a NF profile
  Let $NT = \{F1, F2, \ldots, Fi, \ldots, Fy\}$, where $Fi = \{P1, P2, \ldots, Pi, \ldots, Px\}$, where $Pi = \{pr1, pr2, \ldots, pri, prj, \ldots, pn\}$, pri corresponds to the $HTTP2$

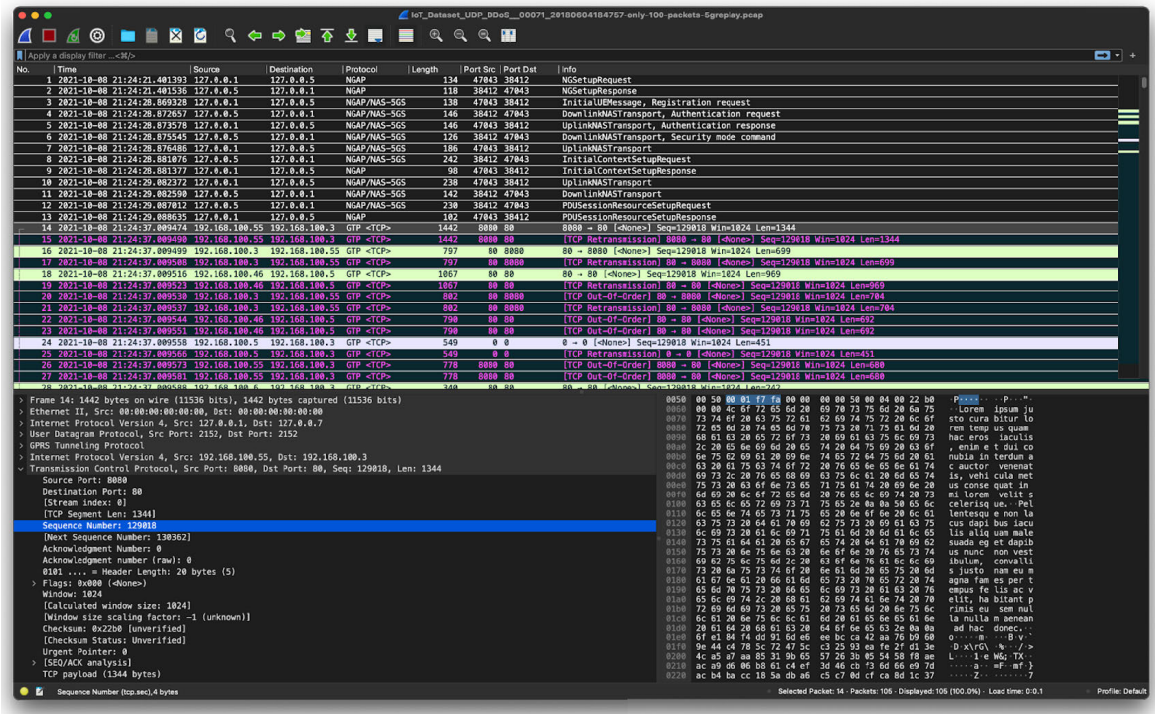


FIGURE 15. Encapsulation workflow.

protocol, and $pri = \{(fd1, value1), (fd2, value2), \ldots, (method, valuex), \ldots, (fdi, valuei)\}$.
And, $R = \{\gamma, \sigma\}$ the mutant operator.
If $\gamma(NT) == P.HTTP2.method.RE.!DELETE \rightarrow Pi$
$\Rightarrow \sigma = P\_MODIFY(Pi, HTTP2, method, DELETE)$
$\Rightarrow NT = \{F1, F2, \ldots, Fi', \ldots, Fy\}$,
$Fi' = \{P1, P2, \ldots, Pi', \ldots, Px\}$,
$Pi' = \{pr1, pr2, \ldots, pri', prj, \ldots, pn\}$, and
$pri' = \{(fd1, value1), (fd2, value2), \ldots, (method, DELETE\ nflnstanceID), \ldots, (fdi, valuei)\}$.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have defined an ontology for network mutation that provides a scientific basis for research work and application of these techniques. The main contribution of this paper is the design of an ontology for mutation techniques by introducing classes of the concepts of network protocol, mutation operators, network packet, flow of network packets and network traffic. To the best of our knowledge there is an absence of a unified formalism for describing the fuzzing process, in particular the mutation-based fuzzing techniques. Some works have approached this problem, but mainly in the software engineering domain. We believe that a formalism to describe network-enabled fuzzing would improve network security by allowing automation of test case creation, and facilitating the creation of complex test cases scenarios.

Based on this ontology, we have designed models of cyber-attacks that we have applied to 5G networks. The proposed approach has been applied to three use cases that represent different attacks against a 5G network. In future work, we plan to introduce ML/AI techniques in order to improve the performance of the fuzzing.

REFERENCES

[1] A. Martinich, *The Philosophy of Language*. Oxford, U.K.: Oxford Univ. Press, 2013.
[2] Z. Salazar, H. N. Nguyen, W. Mallouli, A. R. Cavalli, and E. M. de Oca, "5Greplay: A 5G network traffic fuzzer—Application to attack injection," in *Proc. 16th Int. Conf. Availability, Rel. Secur. (ARES)*. New York, NY, USA: Association for Computing Machinery, Aug. 2021, pp. 1–8.
[3] Z. Salazar, F. Zaïdi, W. Mallouli, A. R. Cavalli, H. N. Nguyen, and E. M. D. Oca, "A formal approach for complex attacks generation based on mutation of 5G network traffic," in *Proc. 17th Int. Conf. Softw. Technol. (ICSOFT)*, H.-G. Fill, M. van Sinderen, and L. A. Maciaszek, Eds. Lisbon, Portugal: SCITEPRESS, Jul. 2022, pp. 234–241.
[4] C. Feilmayr and W. Wöß, "An analysis of ontologies and their success factors for application to business," *Data Knowl. Eng.*, vol. 101, pp. 1–23, Jan. 2016.
[5] G. Brown, "Service-based architecture for 5G core networks," Huawei White Paper, 2017, vol. 1.
[6] C. Salls, A. Machiry, A. Doupe, Y. Shoshitaishvili, C. Kruegel, and G. Vigna, "Exploring abstraction functions in fuzzing," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Jun. 2020, pp. 1–9.
[7] G. Dong, P. Sun, W. Shi, and C. Choi, "A novel valuation pruning optimization fuzzing test model based on mutation tree for industrial control systems," *Appl. Soft Comput.*, vol. 70, pp. 896–902, Sep. 2018.
[8] W. Johansson, M. Svensson, U. E. Larson, M. Almgren, and V. Gulisano, "T-fuzz: Model-based fuzzing for robustness testing of telecommunication protocols," in *Proc. IEEE 7th Int. Conf. Softw. Test., Verification Validation*, Mar. 2014, pp. 323–332.

[9] S. Potnuru and P. K. Nakarmi, "Berserker: ASN.1-based fuzzing of radio resource control protocol for 4G and 5G," in *Proc. 17th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2021, pp. 295–300.

[10] Y. Hu, W. Yang, B. Cui, X. Zhou, Z. Mao, and Y. Wang, "Fuzzing method based on selection mutation of partition weight table for 5G core network NGAP protocol," in *Innovative Mobile and Internet Services in Ubiquitous Computing*, L. Barolli, K. Yim, and H.-C. Chen, Eds. Cham, Switzerland: Springer, 2022, pp. 144–155.

[11] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "Overview of 5G security challenges and solutions," *IEEE Commun. Standards Mag.*, vol. 2, no. 1, pp. 36–43, Mar. 2018.

[12] *Catalogue of General Security Assurance Requirements*, document TS 33.117B, 3GPP, 2020.

[13] *5G Security Assurance Specification (SCAS); Access and Mobility Management Function (AMF)*, document TS 33.512B, 3GPP, 2021.

[14] *ETSI GS NFV-Sec 013*, ETSI, Sophia Antipolis, France, Feb. 2017.

[15] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, and V. Stettler, "A formal analysis of 5G authentication," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1383–1396.

[16] *A Guide to 5G Network Security. Conceptualizing Security in Mobile Communication Networks B how does 5G Fit in?* Ericsson, Stockholm, Sweden, 2018.

[17] *5G Standalone Core Security Research*, Positive Technol., 2021. [Online]. Available: https://positive-tech.com/storage/articles/5g-sa-core-security-research/5g-sa-core-security-research.pdf

[18] *ENISA Threat Landscape for 5G Networks*, ENISA, Athens, Greece, Feb. 2021.

[19] M. A. S. Monge, A. H. González, B. L. Fernández, D. M. Vidal, G. R. García, and J. M. Vidal, "Traffic-flow analysis for source-side DDoS recognition on 5G environments," *J. Netw. Comput. Appl.*, vol. 136, pp. 114–131, Jun. 2019.

[20] H. Moudoud, L. Khoukhi, and S. Cherkaoui, "Prediction and detection of FDIA and DDoS attacks in 5G enabled IoT," *IEEE Netw.*, vol. 35, no. 2, pp. 194–201, Mar. 2021.

[21] M. Ring, D. Schlör, D. Landes, and A. Hotho, "Flow-based network traffic generation using generative adversarial networks," *Comput. Secur.*, vol. 82, pp. 156–172, May 2019.

[22] *ENISA Threat Landscape for 5G Networks*, Aug. 2021.

[23] *System Architecture for the 5G System*, document TS 23.501, Version 15.3.0, Release 15, 3GPP, Sep. 2018.

[24] D. Tank, A. Aggarwal, and N. Chaubey, "Virtualization vulnerabilities, security issues, and solutions: A critical study and comparison," *Int. J. Inf. Technol.*, vol. 14, no. 2, pp. 847–862, Mar. 2022.

[25] D. Sgandurra and E. Lupu, "Evolution of attacks, threat models, and solutions for virtualized systems," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–38, Feb. 2016.

[26] M. P. Souppaya, J. Morello, and K. Scarfone, "Application container security guide," May 2021. [Online]. Available: https://www.nist.gov/publications/application-container-security-guide

[27] J. C. D'Orazio and K.-K. R. Choo, "Circumventing iOS security mechanisms for APT forensic investigations: A security taxonomy for cloud apps," *Future Gener. Comput. Syst.*, vol. 79, no. P1, pp. 247–261, Feb. 2018.

[28] "The evolution of security in 5G—A 'slice' of mobile threats," Jul. 2019. [Online]. Available: https://www.5gamericas.org/wp-content/uploads/2019/08/5G- Security-White-Paper8 .15.pdf

[29] S. Zhang, "An overview of network slicing for 5G," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 111–117, Jun. 2019.

[30] *2020 CWE top 25 Most Dangerous Software Weaknesses*. [Online]. Available: https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html

[31] I. Nadareishvili, R. Mitra, M. McLarty, and M. Amundsen, *Microservice Architecture: Aligning Principles, Practices, and Culture*. Sebastopol, CA, USA: O'Reilly, 2016.

[32] F. Van den Abeele, J. Hoebeke, G. K. Teklemariam, I. Moerman, and P. Demeester, "Sensor function virtualization to support distributed intelligence in the Internet of Things," *Wireless Pers. Commun.*, vol. 81, no. 4, pp. 1415–1436, Apr. 2015.

[33] *ETSI GS NFV-Sec 013*, EIS Group, San Francisco, CA, USA, Feb. 2017.

[34] *Threat Landscape and Good Practice Guide for Software Defined Networks/5G*, Aug. 2021.

[35] *Security Assurance Specification (SCAS) Threats and Critical Assets in 3GPP Network Product Classes*, document TR 33.926, Version 16.3.0, Release 16, 3GPP, Oct. 2020.

[36] T. Madi, H. A. Alameddine, M. Pourzandi, and A. Boukhtouta, "NFV security survey in 5G networks: A three-dimensional threat taxonomy," *Comput. Netw.*, vol. 197, Oct. 2021, Art. no. 108288.

[37] H. Kim, "5G core network security issues and attack classification from network protocol perspective," *J. Internet Services Inf. Secur.*, vol. 10, no. 2, pp. 1–15, 2020.

[38] Y. Sung, P. Sharma, E. Lopez, and J. Park, "FS-OpenSecurity: A taxonomic modeling of security threats in SDN for future sustainable computing," *Sustainability*, vol. 8, no. 9, p. 919, Sep. 2016.

[39] P. Krishnan and J. S. Najeem, "A review of security, threats and mitigation approaches for SDN architecture," *Int. J. Innov. Technol. Exploring Eng.*, vol. 8, pp. 389–393, Jan. 2019.

[40] D. Kreutz, F. M. V. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw. (HotSDN)*. New York, NY, USA: Association for Computing Machinery, 2013, pp. 55–60.

[41] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 623–654, 1st Quart., 2016.

[42] A. Akhunzada, E. Ahmed, A. Gani, M. K. Khan, M. Imran, and S. Guizani, "Securing software defined networks: Taxonomy, requirements, and open issues," *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 36–44, Apr. 2015.

[43] L. Zanzi, F. Cirillo, V. Sciancalepore, F. Giust, X. Costa-Perez, S. Mangiante, and G. Klas, "Evolving multi-access edge computing to support enhanced IoT deployments," *IEEE Commun. Standards Mag.*, vol. 3, no. 2, pp. 26–34, Jun. 2019.

[44] R. Khan, P. Kumar, D. Nalin, K. Jayakody, and M. Liyanage, "A survey on security and privacy of 5G technologies: Potential solutions, recent advancements, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 196–248, 1st Quart., 2020.

[45] S. Tweneboah-Koduah, K. E. Skouby, and R. Tadayoni, "Cyber security threats to IoT applications and service domains," *Wireless Pers. Commun.*, vol. 95, no. 1, pp. 169–185, Jul. 2017.

[46] T. Borgohain, U. Kumar, and S. Sanyal, "Survey of security and privacy issues of Internet of Things," *Int. J. Adv. Netw. Appl.*, vol. 6, pp. 2372–2378, Jan. 2015.

[47] C. J. D'Orazio, K.-K. R. Choo, and L. T. Yang, "Data exfiltration from Internet of Things devices: iOS devices as case studies," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 524–535, Apr. 2017.

[48] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," *J. Comput. Syst. Sci.*, vol. 80, no. 5, pp. 973–993, Aug. 2014.

[49] P. Ryznar, "Council post: Cybersecurity and the explosion of augmented reality," Tech. Rep., Sep. 2019.

[50] N. J. D. Patel and D. Pandya, "OWASP top 10 vulnerability analyses in government websites," Tech. Rep., Jun. 2016.

[51] 2016.

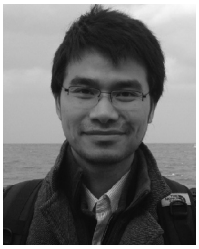
[52] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, *Internet of Things Security*, vol. 88. New York, NY, USA: Academic, Jun. 2017, pp. 10–28.

[53] W. E. Wong and A. P. Mathur, "Reducing the cost of mutation testing: An empirical study," *J. Syst. Softw.*, vol. 31, no. 3, pp. 185–196, Dec. 1995.

[54] *Positive Technologies*, Jan. 2021.

ZUJANY SALAZAR received the M.Sc. degree in computer sciences for communication networks from Telecom SudParis, in 2020. She was a Research and Development Engineer with Montimage, France. Her research interests include simulation and emulation of network traffic patterns and cyberattacks, risk assessment, and monitoring techniques for 5G networks.

FATIHA ZAÏDI received the Ph.D. degree in computer science from the University of Evry, France, in 2001, and the Habilitation degree in computer science from Paris Sud University, France, in 2010. Since 2003, she has been an Associate Professor with Paris-Sud University (newly renamed Paris Saclay University). She has been a Ph.D. advisor to more than ten Ph.D. students. She has developed a passive monitoring technique to test centralized and distributed systems. She is still involved in research projects funded by the National French Agency and also in European projects. Her research interests include addressing formal methods in the software development cycle and particular model-based testing, runtime verification, and parameterized model checking for concurrent and distributed systems. She served as a program committee member for high-ranked conferences. She has also organized several conferences.



HUU-NGHIA NGUYEN received the M.Sc. degree in networks and communicating systems from the IFI, University Claude Bernard of Lyon I, in 2009, and the Ph.D. degree in software engineering from University Paris-Sud XI, in 2013. He is currently a Research and Development Engineer with Montimage EURL company. His research interests include monitoring and testing the composition conformance of distributed systems. His current work focuses on the security and monitoring of high-precision networks.

WISSAM MALLOULI received the degree in telecommunication engineering from the National Institute of Telecommunication (INT), in 2005, and the Ph.D. degree in cybersecurity from Telecom and Management SudParis, France, in 2008. He is currently the CTO of Montimage EURL company, Paris, France. He is also a Professional Trainer. He is working on several collaborative European research projects and has more than 50 scientific publications in popular conferences and journals. His research interests include continuous risk management and cyber defense for critical systems and networks, including cloud-based systems, the IoT, and 4G/5G networks.

ANA ROSA CAVALLI received the Ph.D. degree in mathematics science and informatics from the University of Paris VII, in 1984. She was the Director of the Software-Networks Department, from 2005 to 2015. She has been a Professor with Institut Polytechnique de Paris/Telecom SudParis, since 1990. She is a member of the Research Laboratory CNRS SAMOVAR. She is currently an Emeritus Professor with Telecom SudParis. She has published more than 200 papers in journals and international conferences of high quality. Her research interests include monitoring and testing methodologies for conformance and interoperability testing, security and resilience techniques for the cloud, the IoT, and cyber-physical systems.

EDGARDO MONTES DE OCA received the engineering degree in computing and electronics from Paris-Sud Paris XI, Orsay, in 1985, and the D.E.A. degree in computer science from Pierre and Marie Curie University—Paris VI, Jussieu, in 1986. He was with the Alcatel Research Centre, Marcoussis, and the Ericsson Research Centre, Massy. In 2004, he founded Montimage EURL company, where he is currently the CEO. He is the Leader of the dissemination and exploitation activity of the H2020 INSPIRE-5Gplus Project. His main research interests include building tools to monitor network security and performance, cyber threat intelligence, building portable and secure 5G solutions, and defining the 6G mobile networks of the future. He is a member of the ENISA Business Security Working Group and the 6G-IA Association.