Hindawi
Security and Communication Networks
Volume 2018, Article ID 9291506, 21 pages
https://doi.org/10.1155/2018/9291506

Research Article

5G NB-IoT: Efficient Network Traffic Filtering for Multitenant IoT Cellular Networks

Pablo Salva-Garcia,[1] **Jose M. Alcaraz-Calero**,[1] **Qi Wang**,[1] **Jorge Bernal Bernabe**,[2] **and Antonio Skarmeta**[2]

[1] *University of the West of Scotland, UK*
[2] *University of Murcia (UMU), Spain*

Correspondence should be addressed to Pablo Salva-Garcia; pablo.salva-garcia@uws.ac.uk

Received 31 May 2018; Revised 7 November 2018; Accepted 19 November 2018; Published 10 December 2018

Academic Editor: Petros Nicopolitidis

Internet of Things (IoT) is a key business driver for the upcoming fifth-generation (5G) mobile networks, which in turn will enable numerous innovative IoT applications such as smart city, mobile health, and other massive IoT use cases being defined in 5G standards. To truly unlock the hidden value of such mission-critical IoT applications in a large scale in the 5G era, advanced self-protection capabilities are entailed in 5G-based Narrowband IoT (NB-IoT) networks to efficiently fight off cyber-attacks such as widespread Distributed Denial of Service (DDoS) attacks. However, insufficient research has been conducted in this crucial area, in particular, few if any solutions are capable of dealing with the multiple encapsulated 5G traffic for IoT security management. This paper proposes and prototypes a new security framework to achieve the highly desirable self-organizing networking capabilities to secure virtualized, multitenant 5G-based IoT traffic through an autonomic control loop featured with efficient 5G-aware traffic filtering. Empirical results have validated the design and implementation and demonstrated the efficiency of the proposed system, which is capable of processing thousands of 5G-aware traffic filtering rules and thus enables timely protection against large-scale attacks.

1. Introduction

Internet of Things (IoT) applications are widely envisioned as a major use case in the forthcoming fifth-generation (5G) mobile networks and would account for one-quarter of the global 41 million 5G connections in 2024 [1]. Meanwhile, security is a top concern for large-scale IoT deployment, which is subject to new, disparate kind of threats and attacks. The constrained nature of IoT devices in terms of memory, computation, and power, as well as the unattended, pervasive and dynamic network environment, makes them appealing to attackers. Diverse types of evolved cyber-attacks, for instance, Distributed Denial of Service (DDoS) attacks, which rely on infected bots, are starting to appear in IoT [2, 3]. For example, the Mirai attack in 2016 took down major websites via massive DDoS using hundreds of thousands of compromised IoT devices [4]. Low-Power Wide-Area Network (LPWAN) protocols employed in IoT scenarios, such as NB-IoT [5] defined in 3GPP 13 release [6], are not ideal environments to perpetrate DDoS based on high-rate brute force attacks, due to their associated low bit rate (60kpps uplink). Nonetheless, variants of DDoS attacks, based on low-rate methods [7], fit perfectly in these environments, since they exploit techniques such as sending partial HTTP requests, sending small packets, or keeping sessions open from going to idle time-out.

Similarly, some other kinds of attacks, e.g., those based on unauthorized access, or data leakage, are difficult to detect and mitigate. Infected IoT devices might disclose private personal data of their owners, such as localization, owner identity data, or even video from their featured video cameras. Unfortunately, security and privacy mechanisms are difficult to enforce in the final device, and therefore, the network infrastructure should be ready to self-protect the whole network and system, without involving necessarily the potential malicious IoT device.

Thus, in order to counter dynamically and on demand those cyber-threats in a 5G-enabled IoT network, the network operator might need to filter, mirror, divert, and differentiate IoT packets in the edge access network and in the core of the 5G network. Ideally, this traffic control and management should be performed accordingly at any packet encapsulation level required in LTE/5G Networks. This may include multiencapsulation required to support user mobility and carrier-isolation, any field of the inner packet headers, the tenant the IoT device is associated with, or even any field of a particular IoT-specific protocol, e.g., the Constrained Application Protocol (CoAP) [8], used by the affected IoT device, among others.

Network operators should be able to offer advanced Security-as-a-Service solutions, exploiting the flexibility provided by Software-Defined Networking (SDN) and Network Function Virtualization (NFV) to detect dynamically cyber-threats, and react accordingly, enforcing proper and timely countermeasures, either at the core or at the edge of the network, including dynamic enforcing of pertinent filtering rules to drop malicious traffic coming from the myriad of IoT devices.

The increasing number of technologies using network virtualization, where traffic is usually encapsulated to support multicarrier 5G-enabled services, raises the challenge of managing encapsulated traffic efficiently. Like in LTE and 5G, NB-IoT traffic might require to cope with smart objects mobility, which involves dealing with another level of encapsulation, e.g., through the General Packet Radio Service (GPRS) Tunneling Protocol (GTP).

In a basic network environment, and using predefined matching filters like Linux Netfilter, each packet will traverse all the filtering rules until matching a rule. That will cover layer-3 and layer-4 headers and also application layer payload, when l7-filter is used. Indeed, diverse researches on functional enhancements for efficient traffic filtering have already been provided in the state of the art [9–12]. However, there is still a lack of filtering mechanisms able to perform traffic filtering in multicarrier and mobility scenarios for IoT traffic, being able to deal with the encapsulation requirements imposed by both edge and core network segments of the 5G multitenant networks, capable of performing traffic filtering and deep packet inspection in NB-IoT traffic. Moreover, there is a lack of security frameworks that can assist the security management in order to provide self-healing and self-repairing capabilities to the NB-IoT networks, adapting dynamically the network traffic filtering to the current contextual conditions.

Our proposed filtering mechanism in this paper allows inspecting and analyzing traffic without having to create any tunnel interfaces to deencapsulate the traffic. It allows filtering beyond the first encapsulated layer and dealing with any packet and header of any inner encapsulated traffic to cope with mobility and multitenancy requirements of virtualized 5G networks. The filtering predicates allow classifying packets in Linux kernel space based on any packet fields in any header and encapsulated packet. The benefits are manifold, encompassing scalability, performance, and flexibility, since there is no need to create tunnel interface to perform the deencapsulation, and traffic filtering in kernel space provides an efficient approach.

Moreover, the proposed filtering mechanism has been integrated into a security framework to achieve resilience and autonomic reconfiguration of the filtering rules in order to counter cyber-attacks as low-rate DDoS.

The contributions of this paper are manifold:

(i) A new security framework is presented with an autonomic control loop to enable self-organizing networking based self-protection.

(ii) This paper focuses on an encapsulation-aware traffic filtering approach especially devised for virtualized, multicarrier, narrowband, and 5G-aware IoT networks.

(iii) A prototype of a deep packet inspection method is presented, using a kernel space mechanism in order to have full control of encapsulated traffic required in virtualized NB-IoT networks.

(iv) The filtering mechanism has been integrated in the autonomic and security management architecture devised in a joint collaboration between two EU H2020 projects: Anastacia (in IoT security) [13] (H2020 Anastacia: http://anastacia-h2020.eu/) and SELFNET (in 5G management) [14] (H2020 SELFNET: https://selfnet-5g.eu/).

(v) Empirical performance evaluation of the proposed system is presented and analyzed, over a realistic 5G-compliant virtualized NB-IoT network infrastructure.

The rest of the paper is organized as follows. In Section 2 we analyze the background on filtering techniques as well as scientific related work in the research field. Section 3 introduces NB-IoT leveraged in virtualized 5G architectures and laid out filtering requirements for multicarrier and virtualized 5G-enabled NB-IoT networks. Section 4 overviews the management framework. Implementation and testbed are presented in Section 7. Section 8 reports the experimental results in terms of efficiency, suitability, and scalability. Conclusions and future research activities are drawn in Section 9.

2. Background and Related Work

Despite the considerable number of related work in the area of IoT security, there is still no solution from the 5G data path perspective, where the novel technological advancement of this new paradigm enforces to have mechanisms able to deal with nested encapsulation. Furthermore, as explained in the previous Section 1, the 5G PPP working group also highlights the capability of self-adapting the whole network in a dynamic way as one of their main features. Thus, providing a dynamic management and reconfiguration of the system is a feature that very few studies have taken it into account, and even fewer have studied a framework with an automatic control loop for organizing security policies. References [15–17] are the only studies in this state of the art where a 5G nested encapsulation has been achieved by using kernel filtering techniques, programmable hardware interfaces,

Table 1: Related Works.

Work	Dynamic Management and Reconfiguration.	Cognitive Management Framework	5G RAN and Edge-Core Implementation Setup	Classical IP Support	5G Multitenant Support	5G Mobility Support	Application Layer Support	IoT Support	VNF Solution	Scalable Approach
[20]	•	•	•	•	•	•	√	√	√	√
[37]	√	•	√	√	•	√	√	•	•	•
[16]	√	•	√	√	√	√	√	•	•	•
[17]	•	•	√	√	√	√	√	•	•	•
[21]	•	•	•	√	•	•	√	√	•	√
[19]	√	√	•	√	•	•	√	√	•	•
[38]	•	•	√	√	•	•	√	√	•	•
[18]	•	•	√	√	•	√	√	•	√	•
[15]	√	•	√	√	√	√	√	•	•	•
Our contribution	√	√	√	√	√	√	√	√	√	√

and an extended Intrusion Detection System (IDS) version, respectively. However, those studies are far away from the IoT perspective and no cognitive management framework has been presented. Other studies, such as [18], have used decapsulation and reencapsulation techniques for filtering inner layers of the traffic produced by LTE and 5G networks. Nevertheless, that approach removes the device mobility support through the infrastructure, although mobility support is indispensable for 5G mobile networks. The work in [19] is one of those infrequent studies about security in IoT where a framework using Software-Defined Networking for confining traffic flows of devices is presented. However, although this work implements automated techniques for identifying vulnerable devices and isolating them from the rest of the users devices by generating OpenFlow enforcement rules (OF-rule), it does not have the 5G capability for dealing with multitenant and mobile-device traffic. For reducing both capital and operational expenditure from the point of view of the operators, next-generation mobile networks are adopting softwarization and virtualization technologies. In this way, deployment and service creation becomes flexible and agile. This feature becomes extremely important when the aim is to provide a scalable approach. In [20] authors present a 5G platform for IoT applications, and the platform is able for deploying Virtualized Multiaccess Edge Computing (vMEC) when required. In [21], a hierarchical IoT structure is set up by using several cluster heads that can handle several sensor nodes. A cluster head is assumed to have more computational power and energy resources than a node. In such a case, a sensor node transmits traffic data to the corresponding cluster head at first, and then the cluster head forwards the data to the central server. From Table 1, none of the related work presented has managed to accomplish multitenant support, mobility support, IoT support, application layer filtering, and a dynamic management based on an autonomic framework at the same time. None of them has considered nested encapsulation to be able to create security IoT filtering rules in the edge/core of the network. To the best of our knowledge, this contribution is the first one to be able to provide these capabilities simultaneously and to deploy fine-grained actions for dealing with this kind of complex attacks in the edge/core of a 5G network due to the advanced transversal filtering capabilities supported. In conventional network scenarios, IDSs are often used to evaluate the trustworthiness of network nodes and identify malicious IoT nodes by monitoring their traffic or behavior [22, 23]. Once malicious traffic is detected, several filtering techniques can be used for acting as a firewall.

2.1. Filtering Techniques. The complexity of the nature of the 5G networks requires a deep packet inspection (DPI) for going further into the packet structure. DPI allows an application to look into the data payload when packets are passing an inspection point. Therefore, a noncompliant protocol, viruses, spam, or another kind of useful information in the payload can be found and then it is decided whether the packet should pass or not or should be routed to a different destination. In order to acquire a signature that represents a specific network application, tools and techniques have relied on simple mechanisms that basically compare the content of the packet payload with a set of strings [24–26]. Later, DPI techniques replaced string sets with regular expressions for fasting packet inspection processing [27, 28]. A comprehensive literature review and comparison on the tools and techniques necessary to develop modern DPI systems is presented in [29]. Additional research work has studied the efficiency of traffic filtering and proposed new functional enhancements over traditional firewalls. The work in [9] applies statistics collected from policy segments with the aim of setting up Huffman trees that dynamically adapt to the traffic statistics and ultimately improve the average filtering time. Other techniques rely on early packet rejection to enhance the performance such as the one proposed by [10]. It can be deployed on top of any filtering mechanism to prefilter unwanted expensive traffic. Some other work such as [11] perform reordering of rules and rules fields based on the calculation of the histograms of packet matching rules. In [12], a splay tree firewall is

proposed to handle packet rejection and acceptance and can perform splay filters reordering based on a statistical model that utilizes traffic characteristic. Open vSwitch (OVS) (Open vSwitch, http://www.openvswitch.org) is a software switch responsible for providing network connectivity to virtual machines. Since it is programmable, it brings the possibility of applying filtering rules by using standard protocols such as OpenFlow (Specification of Open Networking Foundation cite [30]) and therefore achieving a separation of the data plane from the control plane. As a drawback, OVS only supports a limited number of protocols. Although every new release of this software adds support for new fields or protocols, each version requires changes throughout and consequently a new building, distributing, and installing process is needed. That is the reason why new approaches such as the one presented in [31] and Tu William et al. [32] have a goal to reduce compatibility problems between different kernel versions and OVS versions and provide support for new protocols without recompiling. To this end, a high-level language for programming protocol-independent packet processing such as P4 (P4 Language Consortium, https://p4.org/) is proposed and OVS datapath is implemented entirely using the Enhanced Berkeley Filter (eBPF) for decoupling OVS datapath functionalities from kernel versions. Therefore, by using a compiler that accepts P4 and emits eBPF, generation of new fields/protocols without imposing the necessity to change the OVS version would be feasible. Another filtering approach is to employ byte-matching techniques. In order to allow dynamic inspection of message payloads, a new Netfilter matching extension called u32 was added by Don Cohen [33]. u32 allows jumping between headers and select a specific range of bytes to be inspected. The u32 match feature instructs the module to extract 32 bits (4 bytes) from the packet at any specified location and compares it with a given value. If the field that needs to be extracted is fewer than 32 bits, the extracted data is masked and shifted. Additionally, it also includes a technique to calculate variable header lengths to overcome the problem of dynamic size headers in protocols such as IP or TCP. As a drawback, u32 just allows no more than 100 characters per predicate, which is a high limitation when dealing with encapsulated traffic where a significant number of headers are added to the protocol stacks. Similarly, the BSD Packet Filter (BPF) [34] is a byte-matching filtering mechanism that provides an efficient way of filtering packets in the kernel space. BPF is available on most Unix operating systems and it provides a similar functionality to the u32 module but without its filtering size restrictions. BPF is also available by using the user-space program called iptables which allows configuring Linux kernel firewall implemented within the Netfilter [35]. Iptables uses a set of tables to inspect, modify, forward, redirect, and/or drop packets. The main functions of Netfilter are associated with each one of those tables. Despite the advances in the related work, the existing tools are not able to deal with traffic filtering in virtualized 5G Networks, where traffic from different tenants (multicarriers/operators) needs to be encapsulated to differentiate their users, and mobility scenarios impose another level of encapsulation to be handled in the firewall. The filtering management solution proposed in this paper enables handling dynamically 5G network traffic according to the decisions made by the autonomic security framework, and it is based on BPF as the underlying filtering mechanism to handle efficiently NB-IoT traffic in 5G-enabled networks. It is important to highlight that although BPF is a well-known mechanism, the way how it is used in this publication stresses its capabilities for dealing with the most complex packet structure that it can be seen over the new 5G mobile network infrastructures.

3. NB-IoT Networks in Virtualized and Multitenant 5G Deployments

3.1. NB-IoT Preliminaries. The 3GPP, in release 13 [6, 36], has specified a new cellular radio access interface called Narrowband Internet of Things (NB-IoT), which is optimized for machine type traffic. The specification tries to be as simple as possible to minimize energy consumption, which is crucial for IoT scenarios, considering also difficulties in radio conditions present in these ecosystems. NB-IoT has tight relationships with LTE specification. Indeed it has been integrated into the LTE standard, and therefore, it can be also integrated with virtualized and multitenant 5G-aware architectures as it will be shown in the next section.

The NB-IoT specification minimizes the radio overhead, and it is able to deliver IP and non-IP data. As it can be seen in Figure 1, NB-IoT introduces two new optimizations over the traditional LTE network for the cellular Internet of Things (CIoT), namely, the user plane CIoT (continuous lines in the figure) and the control plane CIoT (dotted lines in the figure). The control plane adds the new IoT-specific Service Capability Exposure Function (SCEF) to deliver non-IP data over the control plane and provides an abstract interface for the network services such as authentication, Access Control or discovery. To allow this, the Mobility Management Entity existing in traditional LTE to deal with user mobility is extended with the new T6a interface to allow the non-IP IoT traffic to be forwarded. The user plane CIoT allows forwarding of data traffic as in traditional LTE, through the Serving Gateway (SGW) and PDN Gateway (PGW).

NB-IoT technology uses licensed band within the frequency band of 180 kHz, adopting one resource block (either in guard-band or in-band) of LTE transmissions. It allows up to 30/60 (DL/UL) Kbps maximum user rate. NB-IoT lacks handover support in the connected state, and only cell reselection in the idle state is supported. It is intended to provide network connectivity to Cat-M1 devices, which send a small amount of data and are not sensitive to delays. Therefore, it does not support QoS directly. The devices are supposed to be active for a while and then go idle using Power Saving Mode (PSM) in order to save battery. It supports Access Stratum (AS) optimization called RRC which allows reducing to the minimum the signalling needed to suspend/resume user plane connection.

3.2. NB-IoT Integration in Virtualized 5G Architectures. Figure 2 depicted the envisaged NB-IoT functional architecture integrated into the upcoming 5G 3GPP releases. Due to

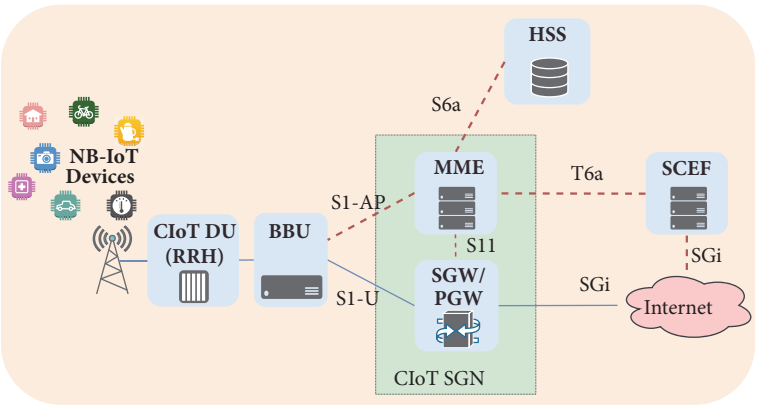

Figure 1: NB-IoT general functional architecture.

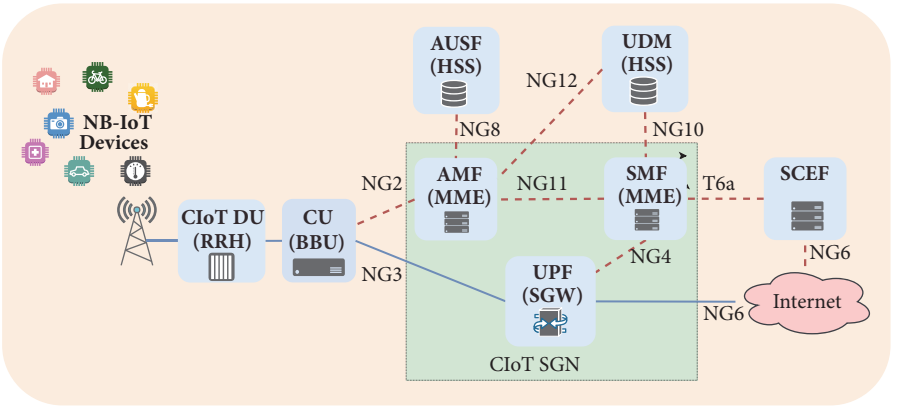

Figure 2: Envisaged NB-IoT functional architecture in upcoming 5G 3GPP releases.

the novel nature of the proposed 5G architecture, the figure shows in parenthesis the relationship between the novel 5G architectural components and the existing LTE components. It has been tailored based on the analysis of all the on-going standardization efforts coming from NB-IoT, 5G RAN, 5G architecture, and a natural way to join them together.

It is also worth mentioning that 5G proposes a fine-grain functional separation of the required functionality of the 5G infrastructure and the usage of Commercial-off-the-Shelf computers, rather than specialized hardware in order to minimize capital and operational costs. The envisioned architecture is composed of the following architectural components:

(i) Distributed Unit (DU) and Centralized Unit (DU) are the architectural components of the Radio Access Network (RAN) and they present an analogy in terms of functionality to the existing LTE RRH (Remote Radio Head) and Base Band Unit (BBU), respectively. 5G proposes a functional segregation of the RAN, fostering the dynamic separation of layers in the RAN stack. It is achieved through the deployment of the protocols of the stack in two architectural components DU and CU, according to the requirements of the deployment and the use case addressed. It is noted that CIoT indicates support for the radio access interface of the NB-IoT specifications.

(ii) Access and Mobility Function (AMF) provides User Equipment (UE) authentication, authorization, and mobility management.

(iii) Session Management Function (SMF) is in charge of managing sessions and allocates IP addresses to UEs. It is also responsible for selecting and manning the User Plane Forwarding Function (UPF) for data transfer.

(iv) Authentication Server Function (AUSF) stores data for the authentication of UE.

(v) User data Management (UDM) stores data about the subscription of UE.

(vi) User Plane Forwarding (UPF) is the mobility anchor for the UE mobility and is in charge of forwarding the UE traffic back and forward to the Internet.

(vii) Service Capability Exposure Function (SCEF) delivers non-IP data over the control plane and provides an abstract interface for the network services such as authentication, Access Control, or discovery.

Another key aspect of 5G architecture is the softwarization and the usage of multitenancy shared resources in

a secure way, fostering the reduction of both capital and operational costs. However, this mobility and multitenancy support for different carriers and telecommunication operators in the network imposes new requirements in the network traffic filtering and it has been the main motivation of this contribution.

It is noted that a comprehensive explanation of all the 5G architectural elements and its reference points is provided in [39].

3.2.1. Network Traffic Filtering Requirements in 5G-Enabled IoT Networks. There are a number of specific requirements for network traffic filtering in 5G IoT networks, listed as follows:

(i) **Multitenant support**: in 5G architectures, the network functional blocks are virtualized as VNFs and different network operators, carriers, and verticals can share the physical infrastructure. The packets need to be encapsulated (e.g., in VXLAN) to differentiate the traffic among them, for management and security reasons. The filtering system needs to deal with this encapsulation.

(ii) **Mobility support**: LTE and 5G networks are subject to the mobility of the UE and, in this case, the mobility of the IoT devices. Although in NB-IoT handover is not supported in a connected stage, cell reselection is supported in the idle state. Mobility in 5G architectures means that packets need to be encapsulated towards the mobility anchor component (UPF in 5G), e.g., using the GTP protocol. The traffic filter will need to be able to handle directly these encapsulation headers.

(iii) **Application layer filtering**: the network traffic filtering system should allow filtering packets for any header/field of any protocol of the OSI stack, including IoT application layer protocols such as CoAP [8].

(iv) **Scalability**: despite that NB-IoT is a Low-Power Wide-Area Network (LPWAN) protocol that requires low bit data rate, the CIoT-RAN and the core of the 5G network will need to cope with the packets of massive IoT devices. Therefore, the network filter(s) will need to efficiently manage the packet filtering process for numerous devices.

(v) **Dynamic management**: IoT networks are volatile and traffic is subject to changing security conditions. Therefore, the management framework needs to automatically adapt the security filtering policies, by enforcing and decommissioning dynamically the rules according to the actual context obtained from real-time monitoring. This dynamic and intelligent management requires relying on softwarized network management and Network Function Virtualization (NFV) technologies for handling such adaptation.

(vi) **Uplink/downlink differentiation**: 5G architectures require having two different Tunnel Endpoint Identifiers (TEIDs) per user, which needs to be handled by the management framework and the filtering agent.

(vii) **Nested encapsulation**: the filtering agent needs to support nested encapsulation for handling simultaneously the traffic encapsulation for both mobility and multitenancy.

4. Cognitive NB-IoT Management Framework

The proposed architecture relies on SDN and NFV technologies, monitoring and reaction tools, cognitive components as well as diverse security enablers and agents to ensure self-protection, self-healing, and self-repairing capabilities in IoT networks and systems. It follows a policy-based security management approach to provide interoperability and higher flexibility to manage security controls over heterogeneous networks, including 5G-compliant IoT networks. The required security actions can be enforced either directly in physical IoT networks or virtual and softwarized appliances. Figure 3 shows the proposed security management architecture.

The **Admin Plane** features pertinent APIs, tools, and graphical interfaces to support administrators on specifying high-level intents about security policies. The policy editor allocated in the plane provides a user-friendly tool to configure security policies using a high-level security policy language, to govern the configuration of the system and network, including not only network traffic filtering but also authentication, authorization, channel protection, and traffic management actions.

The **security orchestration plane** is in charge of deploying and enforcing the security policies on policy-aware security enablers and components, providing a run-time reconfiguration and adaptation of security enablers, whereby the framework is endowed with dynamism and intelligence required for self-healing and self-repairing capabilities. The orchestrator provides autonomic adaptation according to the decisions received from the reaction component.

The *Policy Interpreter* module plays a key role in the refinement of security policies. The high-level policies are first translated into a medium-level security policy language, which allows specifying work-flows related to security procedures in a technology-agnostic way. Then, these policies are refined in specific low-level configurations according to the selected enablers. The policy refinement process is detailed in our previous paper [40].

The *monitoring* component gathers real-time information including security reports, regarding the underlying managed infrastructure, both physical and virtualized. It aims to alert the reaction module when something is malfunctioning. Security probes such as IDS and flow and resource monitoring probes are deployed into the SDN, NFV, and IoT infrastructure domain to give feedback to the monitoring services.

Then, the *reaction* component is in charge of providing appropriate countermeasures, according to the system model status and monitoring information from the monitoring component. It features a cognitive engine in charge of providing intelligence to the management framework, e.g., by selecting adaptation policies or intents stored in the relevant

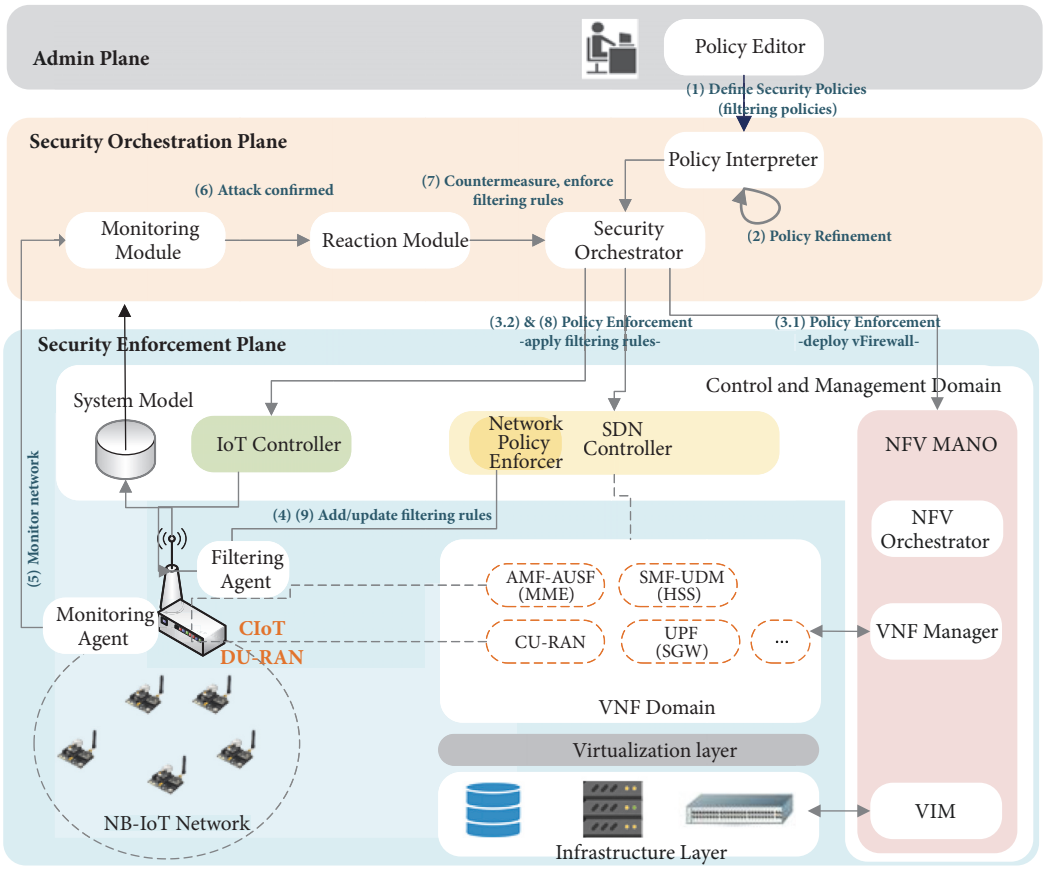

Figure 3: Network management architecture for 5G-enabled IoT networks.

repository and by requiring reconfiguration of the security enablers to cope with the detected attack/threat.

The *Security Orchestrator* supervises the orchestration of the security enablers to be deployed into the Security Enforcement Plane (to be introduced), according to the policy requirements. In addition, at run-time, it analyzes the reaction outcomes and orchestrates the corresponding countermeasures. In this manner, the overall framework aims to achieve self-healing and resilience capabilities, by constantly ensuring the satisfaction of the security requirements defined in the end-user policies.

The **Security Enforcement Plane** is split into three main domains. The *control and management domain* supervises the use of resources and run-time operations of security enablers deployed over software-based and IoT networks. The SDN controllers are in charge of communicating with the SDN-enabled network elements to manage connectivity in the underneath virtual and physical infrastructure. In this sense, the Network Policy Enforcer is in charge of connecting through a southbound API with the Agents deployed in the network, e.g., to enforce filtering rules with a particular filtering agent or a virtual firewall (vFirewall). The Orchestrator is NFV ETSI MANO-compliant to provide support for the secure placement and management of virtual security functions over the virtualized infrastructure. In addition, different IoT controllers are used to managing IoT devices and low-power and lossy networks LoWPANs and LPWANs. These IoT controllers can be deployed at the edge of the network to deploy and enforce Network Security Functions (NSFs) in IoT domains.

The *Infrastructure and Virtualization Infrastructure domain* encompasses both physical machines in charge of holding and supporting storage, computing and networking infrastructure, and the virtualization technologies, to provide Infrastructures as a Service (IaaS). This domain comprises the network elements needed for traffic management (e.g., forwarding, divert, routing etc.), according to the SDN controller rules, as well as the security probes for data gathering needed by the monitoring services.

The *VNF domain* refers to the virtualization infrastructure that holds VNFs deployed to enforce the 5G network functional blocks as well as any Virtual Network Security Function (vNSF) to be deployed by the orchestration plane, such as virtual firewall, vIDS/IPS, vChannelProtection, etc. It is able to provide the defense mechanisms and threat countermeasures requested by security policies.

IoT domain comprises the NB-IoT network (including the CIoT-RAN) as well as the IoT devices to be controlled. This encompasses security enablers, software agents, and actuators required to enforce the security instructions commanded by the orchestration plane. Namely, the filtering agent is deployed in the CIoT-RAN in order to control the

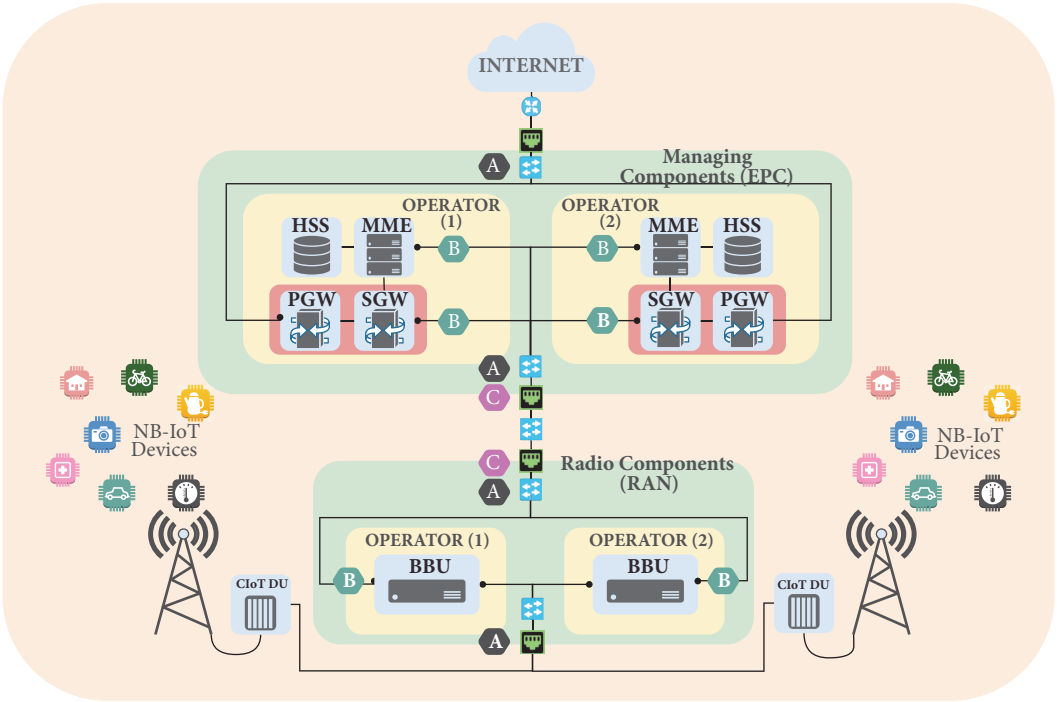

Figure 4: NB-IoT filtering scenario in virtualized and multitenant 5G Infrastructure.

traffic between the particular NB-IoT network according to the filtering rules received dynamically by the Network Policy Enforcer.

5. Virtualized and Multitenant NB-IoT Infrastructure

This section describes an experimental deployment based on a virtualized NB-IoT LTE infrastructure that is deployed in our labs, with several 5G features already supported. For simplicity, Figure 4 provides a simplified view of our deployed infrastructure, where the management plane is omitted. Our infrastructure is composed of 10 computers with an Ubuntu 16.04 and an OpenStack Mitaka release. The deployment employs Neutron and OpenDayLight as SDN controller running the NetVirt Neutron northbound interface provided by OpenDayLight. OpenDayLight utilizes OpenFlow and OVSDB to control the Open vSwitch v2.9 software used to control the data path of the virtual machines. In the figure, only one edge and one core PCs are shown for simplicity, although our lab has two edge nodes and eight core nodes. Every one of the boxes labeled as operator X represents a tenant administrative domain. Each of the tenants has deployed a complete set of VNFs to run the 5G network.

To carry out the deployment of the VNFs, the Mosaic5G (http://mosaic-5g.io/) (evolution of the OpenAirInterface project) infrastructure has been deployed in each of the tenants of the infrastructure. The current version of Mosaic5G allows functional disaggregation of DU and CU although still using the 4G spectrum. Moreover, for the core, the current release still uses MME, HSS, and SGW/PGW terminology; however, it is fully virtualized and running in VNFs. This scenario allows us to have a realistic infrastructure to explore and analyze the NB-IoT traffic along all the network segments.

It is noted that the switches labeled with A in Figure 4 represent the control points used in OpenStack in order to enforce tenant isolation by mean of VLAN, Virtual eXtensible Local Area (VXLAN), or GRE encapsulation. The different points in the data path labeled with B in Figure 4 represent the NB-IoT data plane (using IP connectivity) where GTP encapsulation is present to handle mobility in the devices.

Packets flowing across the infrastructure shown in Figure 4 can be encapsulated into different encapsulation protocols depending on the network segment. The points in the data path labeled with C are a subset of the points labeled with B representing the more complex encapsulation segment for the NB-IoT infrastructure and, at the same time, one of the most efficient ones to apply filtering policies due to the closeness to the hardware (running in physical machines rather than in VNFs). It is especially important when traffic coming from very dense deployments, with potentially hundreds of thousands of NB-IoT devices, need to be handled.

6. Traffic Filtering Process Design

6.1. Filtering Process. The network traffic filtering process is accomplished according to the steps depicted in the architectural Figure 3. The steps are detailed below:

(1) Firstly, in step (1) of Figure 3, the administrator defines proactively his security policy intents, e.g., the filtering policies, employing an interoperable

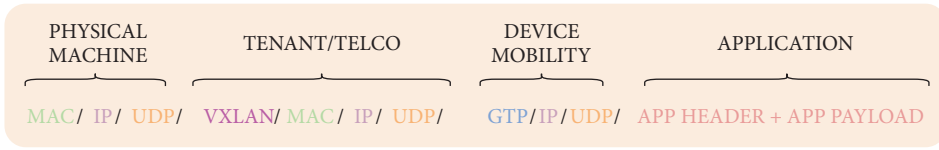

Figure 5: Hierarchical encapsulation.

language, such as MSPL considered in the EU H2020 ANASTACIA project.

(2) Those interoperable policies are refined and translated—step (2)—into particular low-level configurations according to the format required by the specific filtering agent deployed in the network, such as the one shown in our previous work in the context of the Anastasia project [40] or the one in our previous work in the context of the SELFNET project [15].

(3) The Security Orchestrator is notified of the deployment of the rules. Optionally, it might contact the NFV Mano—step (3.1)—to deploy (if not deployed yet) as a VNF the filtering agent acting as a vFirewall, either in the CIoT-RAN or in the core of the network. Then, in step (3.2) it contacts either the SDN Controller or the Network Policy Enforcer to enforce the filtering rules through the northbound API.

(4) The Network Policy Enforcer contacts the filtering agent using a southbound API, e.g., Netconf, in step (4) to enforce the filtering rules. The proposed filtering mechanism is able to deploy the rules in Filtering Agents deployed either in the CIoT-RAN or in vFirewall deployed in the VNF domain of the core of the virtualized 5G Network.

(5) Afterwards, once in run-time, the monitoring agent starts providing monitoring information through probes to the monitoring module, step (5). In this regard, this monitoring traffic is sent through the Pub/Sub Broker.

(6) In case the monitoring module detects an attack based on configured signatures, step (6), it warns the reaction module to make a decision accordingly, and this notification is done using IODEF or IDEMEF standards.

(7) The reaction module component, based on its rule engine, makes a decision to take a proper countermeasure to mitigate the attack, step (7). It might imply adding, for instance, new filtering rules to drop, or divert the traffic coming from a particular infected bot IoT device that is performing a low-rate DDoS attack. This reaction outcome can be done either using a standard such as OpenC2 or by means of tailored mechanisms, as proposed in our proposal in the next section.

(8) The Security Orchestrator contacts again the Network Policy Enforcer to self-reconfigure dynamically the network by deploying the pertinent reaction filtering rules using the northbound protocol, as shown in step (8).

(9) Finally, the filtering rules are configured in the filtering agent through the southbound API, step (9). These rules will consider the network traffic filtering requirements in Section 3.2.1 to cope with the NB-IoT traffic, including nested encapsulation for mobility and multitenant traffic filtering support.

6.2. Pattern Matching Filtering Mechanism. This subsection describes in detail the steps indicated in the previous subsection related to the enforcing on the filtering rules into the managed elements.

Our network filtering mechanism is based on BPF [34] to enforce the filtering rules. This filtering mechanism is an efficient way of filtering packets in the kernel space and it is being used today in hardware networking equipment and even in virtual networking software such as OVS, employed mainly to overcome the limitations of OpenFlow regarding packet classification.

Indeed, BPF is used in many management utilities such as tcpdump, libpcap, iptables, ebtables, etc. BPF is in fact not only a language to express filtering policies using a user-friendly high-level descriptive language, but also a built-in compiler (and optimizer) that translates from the high-level BPF program into compiled BPF x86 bytecode.

Therefore, BPF allows system administrators to select and control packets using high-level packet filtering expressions. The following illustrates an example of filter expression using a tcpdump style syntax which will be compiled in a BPF program later on.

```
$ iptables -m bpf --bytecode $ (nbpf_compile
'proto[Start:End] & Mask = Value') $
```

The mechanisms used in this paper is based on this approach but using a match with more complex matching rules where the complexity of the frames crossing the infrastructure is taken into account, as described in the next subsection.

6.3. Hierarchical Encapsulation. Figure 5 illustrates an example of the most complex hierarchical encapsulation available in the NB-IoT multitenant 5G infrastructure, corresponding to the capture of packets in the control points labeled as C in Figure 4.

The first group of headers is related to the communication between physical machines including Medium Access Control (MAC), IP and UDP headers. The second group of headers includes a VXLAN, MAC, IP, and UDP, inserted to isolate tenant traffic, especially for a telecommunication operator sharing the same physical 5G infrastructure as a

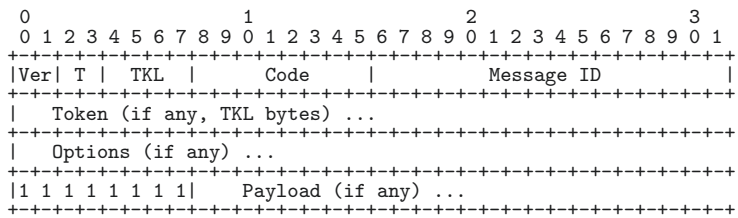

Figure 6: CoAP message format [8].

tenant. This paper proposes to employ the VXLAN protocol to achieve that tenant isolation as an example but other alternative protocols can be used for the same purpose. The next group of headers including GTP, IP, UDP, Application (APP) HEADER, and APP PAYLOAD is used to allow NB-IoT device mobility. GTP is the tunneling protocol employed in NB-IoT and 5G infrastructures to establish the data path for IoT devices with features such as mobility, admission control, etc. Finally, the application header and payload represent the data being sent/received by the devices.

It is noted that a normal IP network uses a very limited subset of these headers, for instance, MAC/IP/UDP /APP-HEADER/APP-PAYLOAD. Compared with that simple case, several additional headers have been added to achieve both multitenancy and mobility of NB-IoT devices.

6.4. Filtering Rules for IoT Traffic in 5G-Aware NB-IoT Networks. In addition to the network protocols specified in the previous section, an application protocol is also considered for filtering, since different network attacks are not identifiable unless the traffic filter matches particular fields at the application layer. In this sense, nowadays, the prominent IoT application protocol is CoAP [8]. It is a lightweight protocol that follows a REST model especially devised for constrained IoT devices (cat-M1) required in NB-IoT. Figure 6 shows the packet structure of the CoAP protocol [8].

7. Implementation and Validation

7.1. Implementing Filtering Rules in Kernel Space. The proposed implementation has been carried out by a filtering agent prototyped in Python using Pika as a library to expose a northbound interfaces receiving intents using the AMPQ protocol [41]. An intent defines what type of traffic should be controlled and the action that needs to be enforced over such traffic. This approach fits perfectly into the novel cognitive NB-IoT Management framework presented in Section 4, and in this way the security orchestration plane is able to deploy and enforce the network security policies, providing a run-time reconfiguration and adaptation. It is important to highlight that old common techniques have been using static sets of filtering policies on a specific rule-based system (e.g., OVS). This work adds dynamicity in terms of creating new filtering rules on demand and using a rule-based filtering system that can be included as a plug-in in the filtering agent. The same intent message will be used as input in the northbound interface of the filtering agent regardless of the plug-in required, thereby using a common way of deploying filtering rules in different filtering systems.

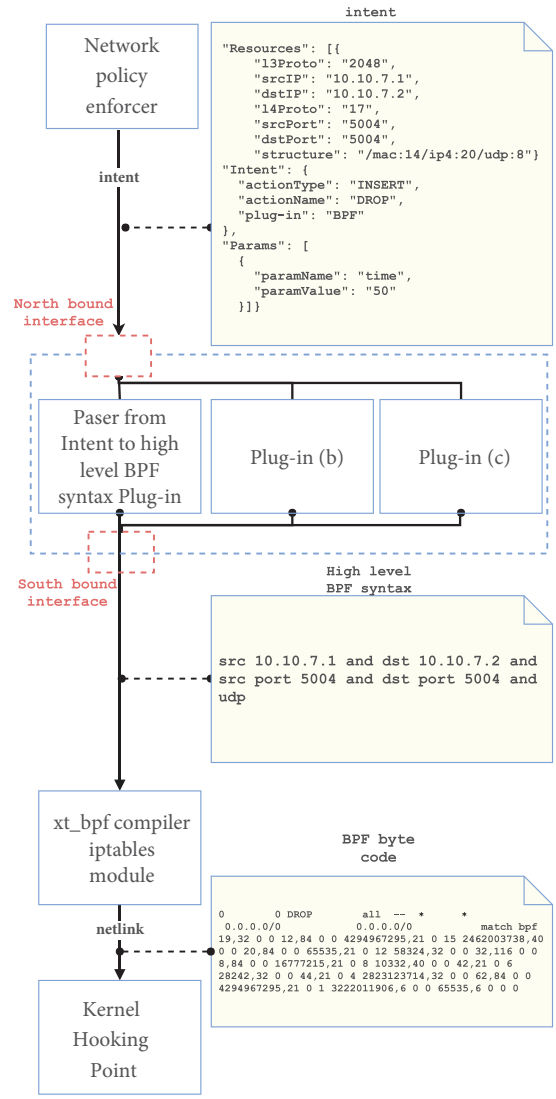

Figure 7: Filtering agent architecture.

As an example, let us assume that the filtering agent is deployed in both points labeled with C in Figure 4 and that the traffic needs to be dropped in order to mitigate low-rate DDoS attacks. The intent for this example under such assumptions is depicted in Figure 7.

Table 2: Encapsulation groups, purpose, and interested fields to be matched.

GROUP	PURPOSE	PROTOCOLS	FIELDS TO BE MATCHED
GROUP 1	PHYSICAL COMMUNICATION	IP	SOURCE IP ADDRESS
			DESTINATION IP ADDRESS
			PROTOCOL
		UDP	SOURCE PORT
			DESTINATION PORT
GROUP 2	TENANT ISOLATION	VXLAN	VNID
		MAC	MAC SOURCE ADDRESS
			MAC DESTINATION ADDRESS
			TYPE
		IP	SOURCE IP ADDRESS
			DESTINATION IP ADDRESS
			PROTOCOL
		UDP	SOURCE PORT
			DESTINATION PORT
GROUP 3	DEVICE MOBILITY	GTP	TUNNEL IDENTIFIER
		IP	SOURCE IP ADDRESS
			DESTINATION IP ADDRESS
			PROTOCOL
		UDP	SOURCE PORT
			DESTINATION PORT
GROUP 4	NB-IoT APPLICATION	COAP	CODE
			VERSION
			MESSAGE ID

The Filter Agent is able to add/update/delete intents coming from the *Network Policy Enforcer* component of the architecture presented in Section 4. The filtering agent will receive intents and then select among all the possible plug-ins registered as interface providers, in order to transform the intent in an implementable and an executable rule. Several plug-ins are supported; for prototyping purposes, this research work has employed a filtering plug-in based on the Linux kernel space using BPF rules. This plug-in converts intents into high-level BPF syntax. An example of a high-level BPF rule is also shown in Figure 7. Then, the high-level BPF syntax is sent to the iptables module named *xt_bpf compiler*. This module compiles the high-level BFP syntax into executable bytecode in the kernel space. This executable bytecode is associated with a hooking point into the Linux networking subsystem (NetFilter) by using the netlink API, so that when packets transverse such hooking point, the bytecode is executed. An example of the compiled BPF bytecode is shown in Figure 7.

7.2. Distributed and Scalable Virtual Firewalls. NB-IoT networks are expected to deal with up to 52,500 devices per cell [6], meaning that, even with low-rate packets per second, the filtering system will need to scale up properly to handle a huge amount of packets in the mobile backhaul. In the worst case there could be a filtering rule per device; however, with current software-based filtering implementations it is not feasible to handle such large quantities of rules and massive traffic in just one firewall. Moreover, this is further complicated in light of the complex rules defined herein that require inspecting the packets according to multiencapsulation imposed in 5G-based NB-IoT networks.

Our solution benefits from leveraging NFV and cloud-computing technologies to deploy dynamically in the RAN backhaul, on demand, virtual Network Security Functions (vNSFs), in the format of distributed vFirewalls. Each vFirewall is in charge of dealing with a subset of the rules according to a network segmentation addressed in a particular RAN. A first filtering agent acts as a load balancer redirecting the traffic quickly to the pertinent vFirewall responsible for handling a subset of rules. The network segmentation and forwarding in the load balancer can be achieved with just one rule per deployed subsequent vFirewall, inspecting the inner IP packet of the encapsulated traffic. Alternatively, this can be done per tenant, by looking into the VXLAN header. This scalable approach enables the deployment of additional vFirewalls according to the network conditions, while our cognitive management framework allows the autonomous configuration of the rules for those vFirewalls.

7.3. Experiment Design. Table 2 provides an example of the different headers explained in Section 6.3 together with all fields that will be matched in our experiments per each of the headers. It is noted that CoAP has been used as the layer 7 protocol as the prominent protocol in NB-IoT deployments.

Table 3: Selected fields of the tests carried out using different Infrastructures.

PROTOCOL	FIELDS	BITS	TEST 1	TEST 2	TEST 3	
IP	src_ip	32	√	√	√	GROUP 1
	dst_ip	32		√	√	
	protocol	8			√	
UDP	src_port	16	√	√	√	
	dst_port	16		√	√	
TOTAL SIZE:			48	96	104	
VXLAN	vni	24	√	√	√	GROUP 2
MAC	src_mac	48	√	√	√	
	dst_mac	48		√	√	
	type	16			√	
IP	src_ip	32	√	√	√	
	dst_ip	32		√	√	
	protocol	8			√	
UDP	src_port	16	√	√	√	
	dst_port	16		√	√	
TOTAL SIZE:			168	312	344	
GTP	teid	32	√	√	√	GROUP 3
IP	src_ip	16	√	√	√	
	dst_ip	16		√	√	
	protocol	8			√	
UDP	src_port	16	√	√	√	
	dst_port	16		√	√	
TOTAL SIZE:			232	408	448	
CoAP	code	8	√	√	√	GROUP 4
	version	2		√	√	
	message_id	16			√	
TOTAL SIZE:			256	418	474	

Three different tests have been designed in order to stress the complexity of the filtering rules and analyze how this complexity affects the scalability in NB-IoT deployments. Each of the tests is related to the number of fields that are matched on each of the protocols available in the payload being filtered. The tests are defined as follows:

(1) Test 1 evaluates rules with predicates for matching up to one field per protocol.

(2) Test 2 evaluates rules with predicates for matching up to two fields per protocol where possible.

(3) Test 3 evaluates rules with predicates for matching up to three fields per protocol where possible.

These three different tests can be applied to different infrastructures. Firstly, these tests can be applied over a classical IP-based infrastructure in order to have a reference point in terms of performance and to be able to evaluable the overhead and complexity imposed by the infrastructure. It will require the use of only the Group 1 of headers (associated with physical communication) indicated in Table 2. Secondly, the tests can also be applied over a Multitenant Infrastructure to evaluate the overhead related to tenant isolation and user-filtering in such an environment. It will require the usage of the Group 1 and Group 2 of headers (associated with physical communication and tenant isolation) indicated in Table 2. Thirdly, the tests can be further applied over an NB-IoT Multitenant Infrastructure where both tenant isolation and NB-IoT mobility need to be handled. It will require the usage of the Group 1, Group 2, and Group 3 of headers (associated with physical communication, tenant isolation, and device mobility) indicated in Table 2. Finally, the tests can also be applied over a Service-aware NB-IoT Multitenant Infrastructure, where not only tenant isolation and NB-IoT mobility need to be handled, but also application-specific filtering for NB-IoT protocols are required. It imposes headers from Group 1, Group 2, Group 3, and Group 4 (associated with physical communication, tenant isolation, device mobility, and NB-IoT Application), as indicated in Table 2.

Table 3 illustrates the combination of the selected fields of the matching rules, grouped according to the protocols available in the three infrastructures analyzed in the three tests carried out in each of these infrastructures. The table contains the size in bytes that need to be matched by the rules for each of the groups. It allows the reader to analyze the increasing complexity of each test compared with the previous one. It is noted that these sizes are cumulative since the usage of Group 2 headers implies the usage of the header of Group 1 and so on and so forth.

The set of experiments indicated in Table 3 aims to validate the feasibility of the proposed traffic filtering mechanism to handle the traffic coming from thousands of NB-IoT devices in the core of a multitenant 5G-network simulating a low-rate DDoS attack that might send packets every 30s to keep connection/sessions open to collapse the target service. To this end, each of the experiment ranges exponentially (power 2) the number of filtering rules being loaded from (1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, and 32768) according to the packets per seconds that arrive to the filtering agent. In the worst case in terms of scalability, the administrator would need the most finest grain of details in the control of the traffic and thus considering one rule per each of the services being running in each of the NB-IoT devices of the infrastructure. Usually, one NB-IoT device hosts one service. Therefore, a 1:1 match correspondence between the number of filtering rules and devices can be assumed in the experiments. In summary, four different infrastructures are analyzed against three different complexities in the rules, and each of these scenarios will be ranged against the different number of rules previously described.

7.4. Hardware Infrastructure. The deployment presented in Figure 4 matches the deployment carried out in our premises with some assumptions. Firstly, one operator has been deployed in our infrastructure and one CU and DU pair is currently deployed. Second, two LTE-based sensors have been simultaneously connected to our testbed in order to reproduce a packet trace in the data path, completely accurate to the NB-IoT devices. This has been required since our LTE stack based on 5G OpenAirInterface currently does not provide support for the NB-IoT radio interface. However, the main limitations as indicated in the previous sections are in the radio access side and those assumptions do not affect the quality of the results presented herein since the packet encapsulation stack being evaluated is equivalent. The complete infrastructure has been deployed in OpenStack Mitaka to allow tenant isolation. This deployment has allowed us to gather Packet Captures (PCAP) files of the communication between the IoT device and the SGW and PCAPs of the four different infrastructures presented in the testbed section.

After the PCAP files have been gathered for one device, they have been processed in order to generate derivate PCAP files where there are as many flows as devices analyzed in the scenario. As mentioned, there is a 1:1 match between devices, flows, and rules. Then, these files are used in the experiments associated with the same number of rules/devices.

Therefore, a testbed has been set up to measure matching times and scalability of the filtering mechanisms proposed and to validate its feasibility into large-scale NB-IoT change deployments. Figure 8 shows a logical layout of the components deployed in the physical computer used to execute the experiments for performance evaluation. The testbed machine has installed an Ubuntu 16.01 Xenial 64-bit operating system and is equipped with 32 GiB RAM, a 16 core Intel Xeon CPU E5-2630 v4 @2.2GHz processor, and 2TB optical hard disk plus a 500GB solid-state hard disk. Each of the VMs employed for the deployment of vFirewall where the experiments have been carried out has been deployed KVM with 8Gb RAM, 2 vCores, and 40G HDD.

Firstly, the filtering agent receives an intent and uses the user-space tool provided by Netfilter (iptables) to load the filtering rules at two different points of the framework NF_IP_PRE_ROUTING and NF_IP_LOCAL_IN, hooking points 1 and 2, respectively (see A in Figure 8). Secondly, an external process replays the PCAP file generated for this experiment from the location indicated by label B in Figure 8. When the first packet from the network interface matches at hooking point 1 shown as label C shown in Figure 8, a timestamp value is produced and such information is saved. Finally, packets cross the hooking point 2 labeled as D in Figure 8 where all the set of rules are already deployed (from 1 to 131072, depending on the scenario). Only the last rule contains the filter predicate that perfectly matches the flow. All the rules are homogeneous in complexity and all the packets must be matched against all the rules. This approach is a very pessimistic approach since it assumes always the worst-case scenario. However, if scalability is proven for this worst-case scenario, it will continue to be valid for less extreme conditions. Once the first packet reaches the last rule, it matches the predefined predicate and produces a new timestamp. Therefore, the difference between timestamps gathered at hooking point 2 and hooking point 1 provides the time consumed of a packet crossing the kernel network space when different rules have been applied. This allows us to measure performance results related to the complexity and the number of the predicates in the rule, and how it affects the normal traffic of the network in terms of delay.

8. Performance Evaluation

This section evaluates the impact of dealing with complex rules that requires deep inspection of the packets to support nested encapsulation originated by multitenancy and mobility of 5G-enabled NB-IoT networks. The testbed estimates the advisable maximum number of complex filtering rules that can be enforced in one vFirewall without incurring packet loss, taking into account the finest grained conditions (i.e., one rule per NB-IoT device). In addition, it evaluates the scalability and performance in terms of jitter, overhead times, matching filtering rules times, and management times (flushing and loading of rules).

To this end, for each of those evaluations, the paper compares the performance of our traffic filtering mechanism for NB-IoT 5G service-aware networks with the performance achieved in traditional IP infrastructures, which can be handled with traditional traffic filtering methods. Figure 9 shows the four infrastructures previously presented in Section 7.3, which will be analyzed along this section.

(1) Classical IP Infrastructure.

(2) User-aware Multitenant Infrastructure, where there is tenant isolation and virtualization.

(3) 5G NB-IoT device mobility, multitenant infrastructure, where there is tenant isolation, virtualization and device mobility.

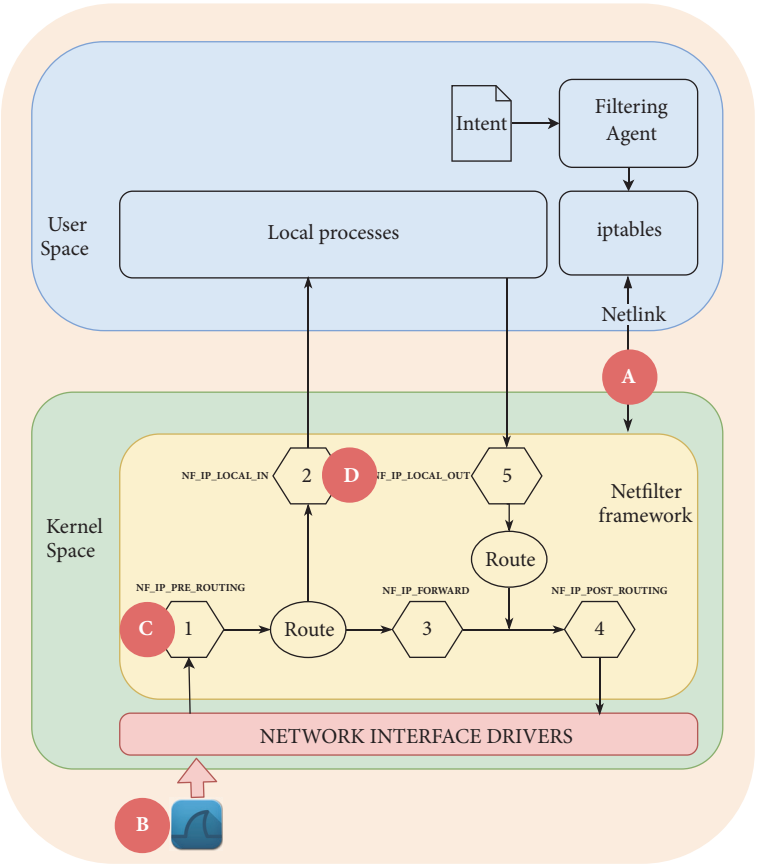

Figure 8: Performance evaluation testbed.

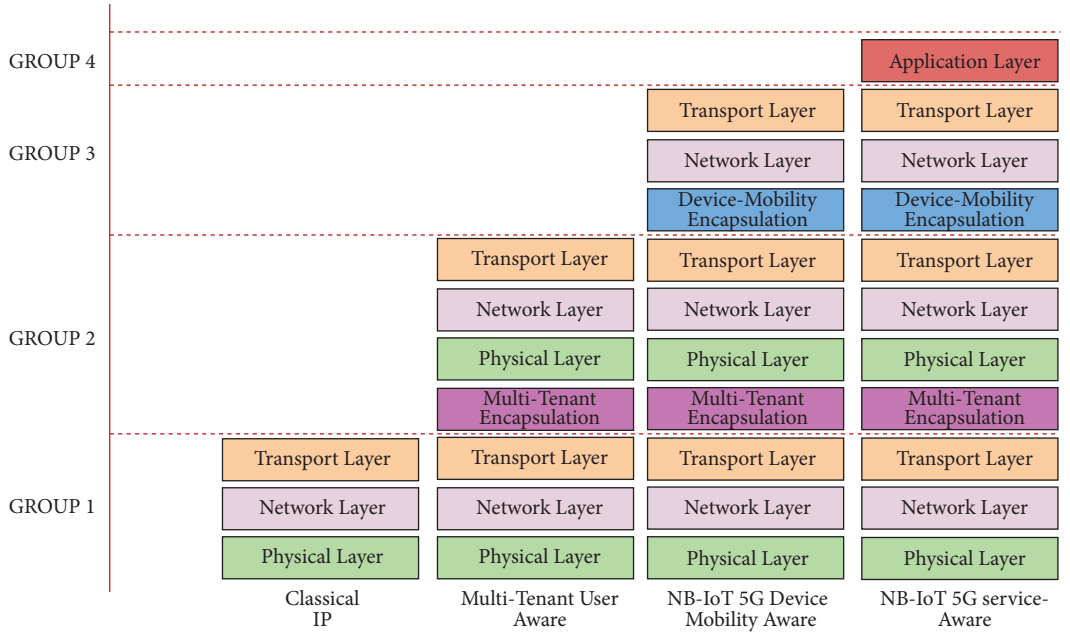

Figure 9: Analyzed infrastructures.

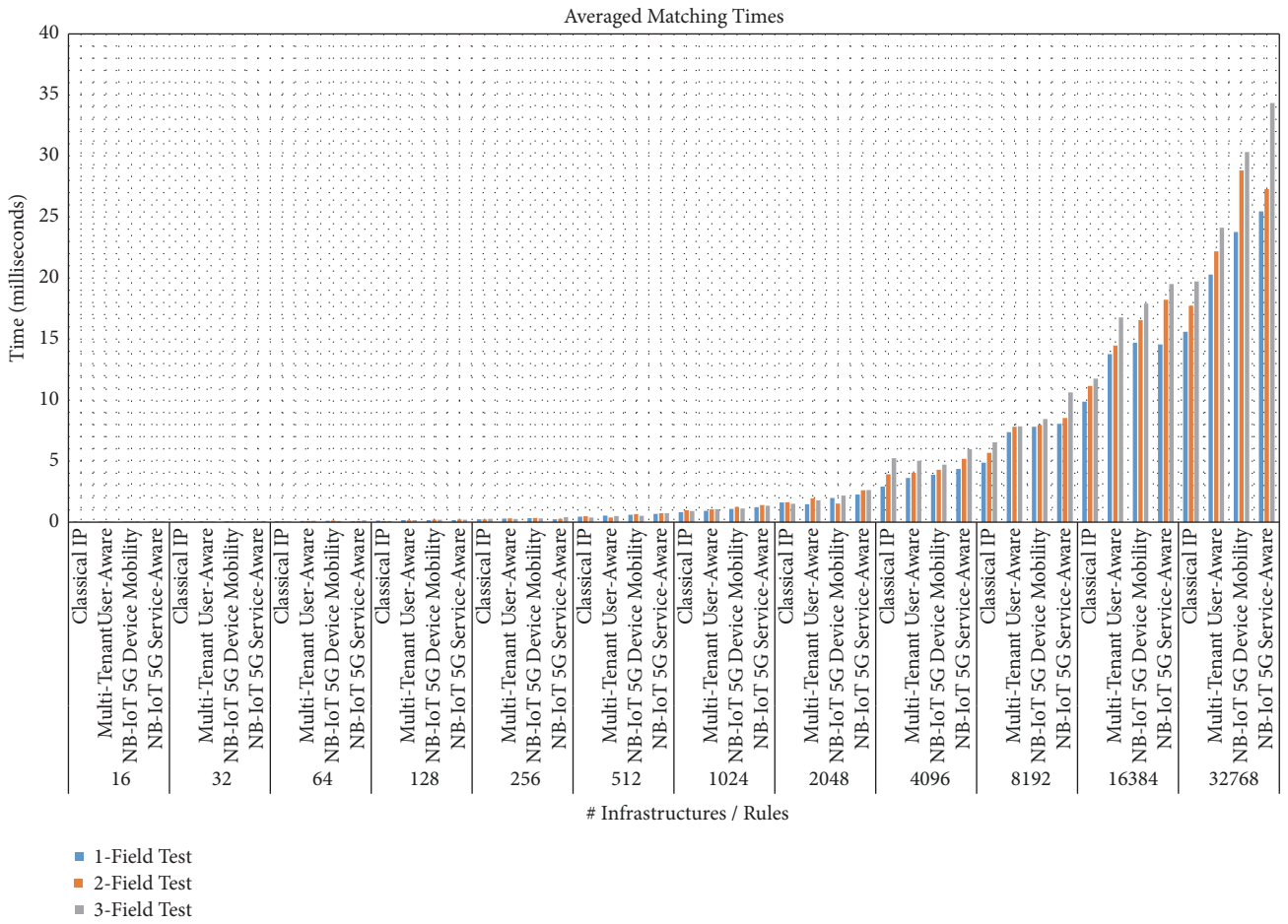

Figure 10: Average rule matching time per number of rules and analyzed infrastructure.

(4) NB-IoT Service-aware 5G Multitenant Infrastructure, where there is tenant isolation, virtualization and device mobility and NB-IoT service-level filtering support.

Figure 10 represents the empirical results in terms of performance times over analyzed infrastructures. The X-axis follows an exponential function, increasing the number of devices/rules in order to show how the proposed filtering approach scales according to the number of devices. It is noted that NB-IoT deployments should deal with thousands of devices. This is why the largest scenario analyzed uses this level of scalability to prove the feasibility of the proposed approach at production-grade. The Y-axis is the average time in milliseconds taken to process a packet that transverses all the rules loaded.

The three series represent the three different tests previously indicated. The rules matching time grows according to the increasing complexity of the infrastructures deployed and the services provided by the infrastructure. This further leads to more complex filtering rules that need to be applied. Thus, the Classical IP Infrastructure provides the fastest performance results while the NB-IoT Service-aware 5G Multitenant Infrastructure provides the slowest.

As can be seen in the graph, the number of rules deployed is the predominant and critical factor that affects the scalability. In addition, the complexity of the supporting infrastructure (scenario) also has an impact on time. For the largest deployment with 32,768 devices, the time consumed in a Classical IP Infrastructure scenario is around 20ms. This case can be considered as the reference best scenario as this is the simplest infrastructure. The other three scenarios analyzed are User-aware Multitenant Infrastructure (or simply referred to as Multitenant) and 5G NB-IoT device mobility/multitenant infrastructure (NB-IoT Multitenant) as well as 5G NB-IoT device mobility/multitenant infrastructure (Service-aware NB-IoT Multitenant) exposing an increasing overhead with respect to the classical IP scenario. Finally, the complexity of the rules available in each of the scenarios for each of the infrastructure seems to be a factor incurring less delay.

In addition, the classical filtering method by using just IP protocol inspection is around 37% for the 1-field test, 48% for 2-field test, and 69% for 3-fields faster than the Service-aware NB-IoT Multitenant scenario and when thousands of rules are applied in the vFirewall.

Moreover, the largest NB-IoT scenario with 32,768 rules/devices being filtered simultaneously requires 30ms for

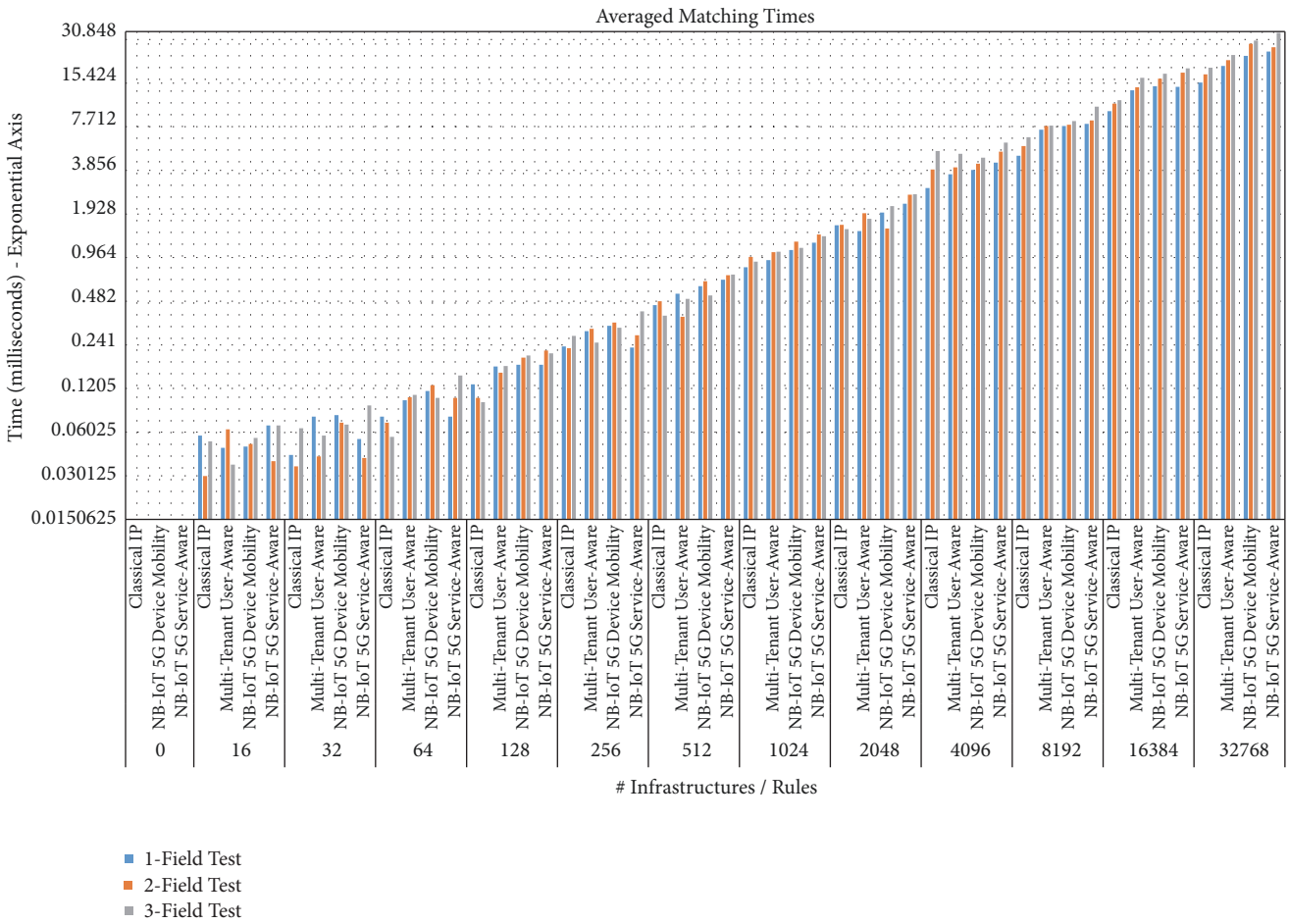

Figure 11: Performance of matching filtering rules using a logarithmic scale.

matching, which is clearly in the boundaries of the tolerable delays indicated by the NB-IoT and LTE-M standards.

Figure 11 illustrates the same results plotted in Figure 10 yet in an exponential scale (log2 base, Y-axis) in terms of time overhead and an exponential scale in the number of devices (X-axis). Despite the exponential nature of both scales, the plot shows a close to linear trend as the number of devices grows, which validates the good scalability performance of the proposed approach when scaling the number of rules/devices.

After the above analysis of the behavior of all the different rule complexities, a deeper analysis has been carried out by stressing the infrastructure to gather results of the system regarding scalability in terms of the number of simultaneous devices performing a low-rate DDoS attack. To this end, Figures 12, 13, and 14 focus on the most complex scenario where all the headers' fields are matched (3-field test). These figures analyze the behavior of the system when 4,096, 8,192 and 16,384 devices are sending low-rate packets every 30s, which simulates the behavior of a low-rate DDoS attack. It leads to 137, 274, and 546 packets/s, respectively, as indicated in the legend of these figures.

Figure 12 shows how the system is stable for the three different number of devices analyzed until 4,096 rules where the overhead time is close to seconds. Beyond that point, the system is still stable for 8,196 rules for a scenario with 274 PPS (Packets Per Second) (i.e., 8196 devices) for all infrastructures analyzed, including NB-IoT 5G service-aware infrastructure where the average overhead time is around 100ms. However, the system becomes unstable when 8,196 rules are facing a higher volume of attack, 546 PPS (i.e., 16384 devices). This threshold determines the boundaries of the scalability in terms of devices supported by a given virtual firewall.

Figure 13 shows an analysis of the packet loss and reassures the previous results shown in Figure 12. There is no packet loss for the three different number of devices analyzed until 4,096 rules. Then, the system is still close to 0% packet loss for 8,196 rules for a scenario with 274 PPS (i.e., 8196 devices). After this point, the number of lost packets starts to increase dramatically, showing a similar behavior to the one plotted in Figure 12. The Service-aware NB-IoT Multitenant scenario with 16,384 devices (546 PPS) and 8,192 rules shows an unacceptable packet loss rate of around 50%.

Figure 14 shows the behavior of the jitter when ranging the number of devices and rules. Jitter is almost insignificant and close to 0us, up to 4,096 rules. Then, it is still acceptable for a scenario with 274 PPS (i.e., 8,196 devices) where the jitter is around 100-300us, depending on the infrastructure

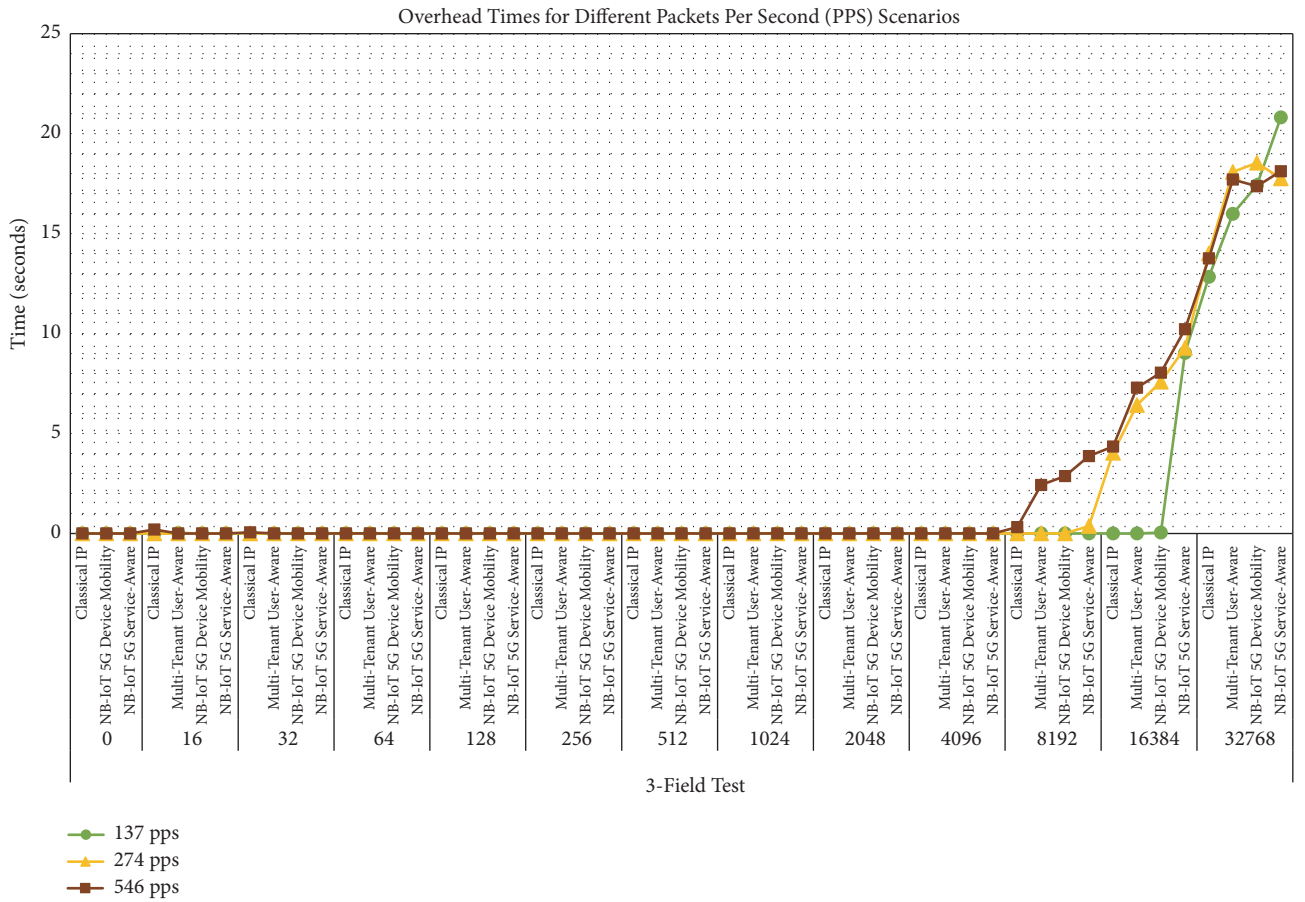

Figure 12: Overhead time with respect to a 0-rule scenario according to both number of rules and infrastructure analyzed.

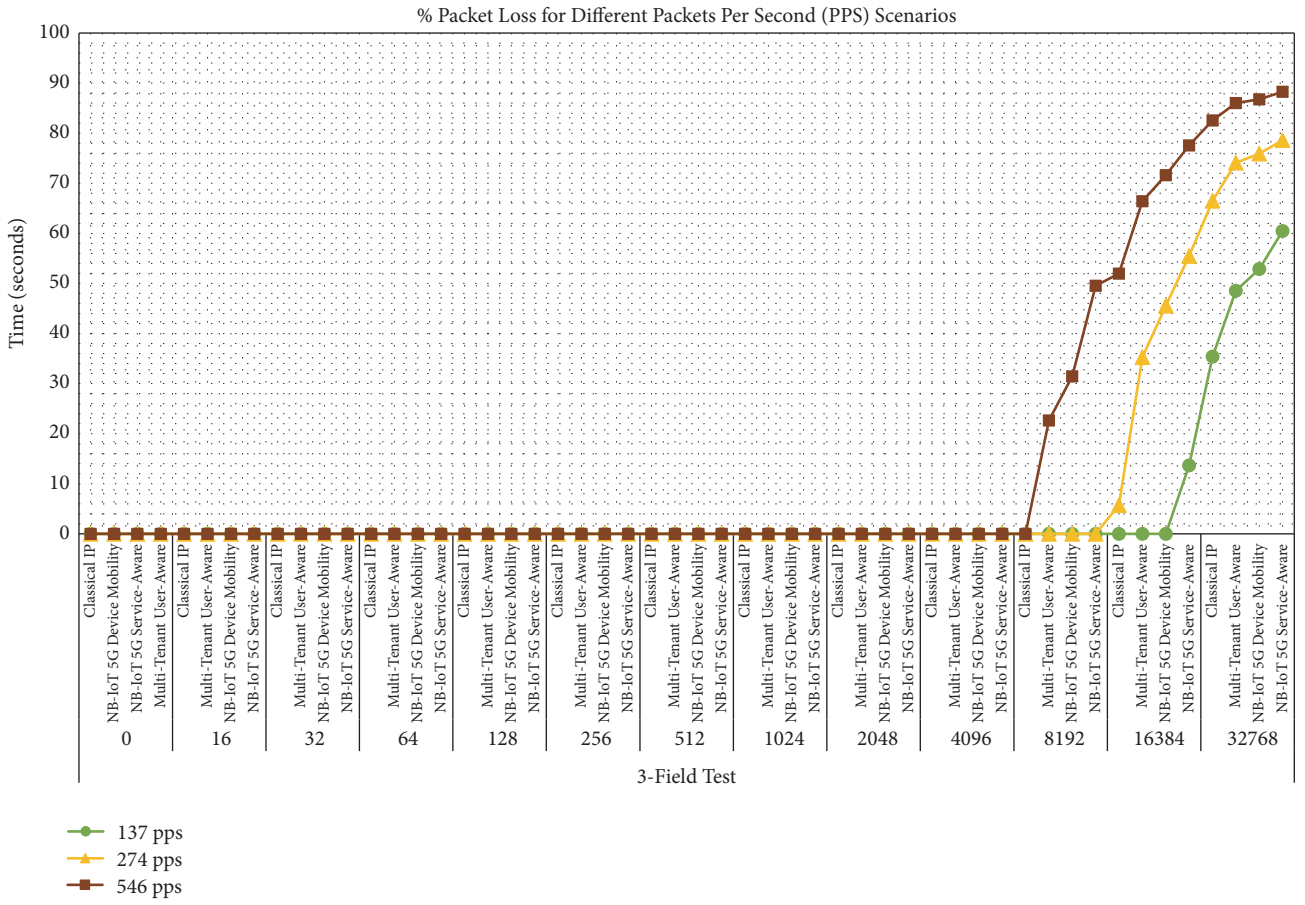

Figure 13: Packet loss according to both number of rules and infrastructure analyzed.

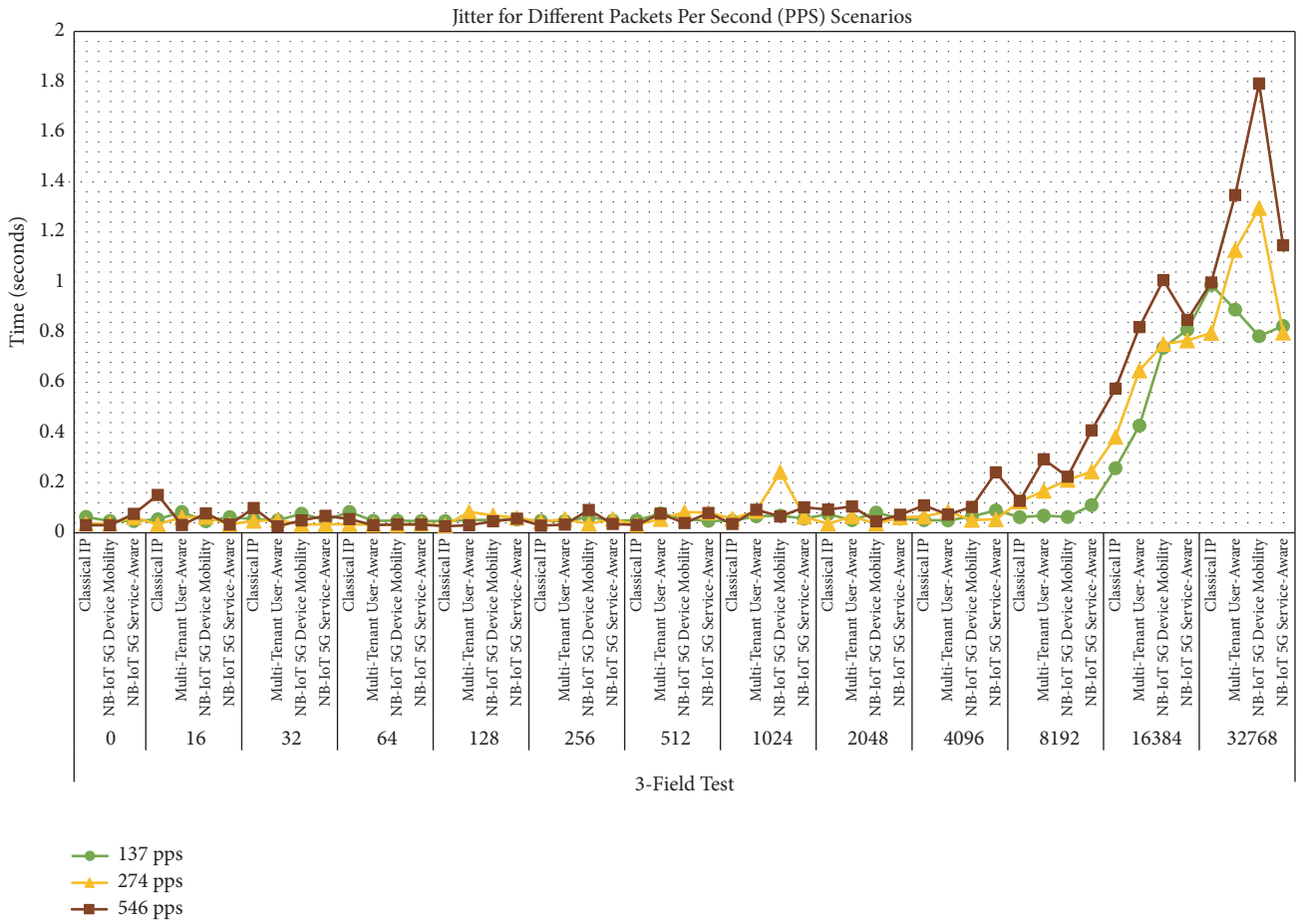

Figure 14: Jitter according to both number of rules and infrastructure analyzed.

analyzed. Beyond that number of devices, the jitter increases significantly.

It can be concluded that the scalability boundaries of the proposed architecture are set to 8,192 NB-IoT devices per virtual firewall, where each of such devices has associated a rule to control an NB-IoT service. Our testbed has the capability to run eight VMs for 65,536 devices, which is far beyond the expected 52.547 per cell indicated in the NB-IoT standard [6]. This result successfully validates the suitability of the proposed approach.

Figures 15 and 16 illustrate results on flushing and loading times of rules, respectively, in order to analyze the management plane of the proposed approach. From the management plane's perspective, flushing, and loading times gives critical information about how long it takes for the management system to reset the vFirewall when suddenly around up to 8,192 devices attack the system simultaneously. Cleaning the last configuration and loading a completely new one would take about 0.6ms and 6s, respectively. In other words, the proposed system would be ready for controlling thousands of completely different NB-IoT devices in 6s. It should be noticed that these fine-grain rules are NB-IoT service-aware, and it is worth clarifying that, by using a more generic rule such as filtering by tenant or using a mask in the IP addresses, multiple flows could be stopped just by using one rule and all the processes including flushing, loading, and matching will be significantly reduced from the times specified above.

9. Conclusions

This paper has described a novel autonomic security framework featured with an efficient network traffic filtering system for virtualized and multitenant 5G-enabled NB-IoT networks, which relies on a cognitive management framework for delivering autonomic self-protection capabilities to the network.

Our proposed security framework and filtering system are ready for mitigating an attack by deploying and loading dynamically, thousands of filtering rules in the vFirewall, corresponding to thousands of NB-IoT devices. The filtering mechanism is able to process encapsulated 5G network traffic in the core and in the edge of the virtualized 5G network simultaneously, with multitenancy, mobility, and DPI support. The complex filtering rules, capable of handling such traffic, are evaluated in the kernel space by our filtering agent with minimal overhead in the vFirewall, which demonstrates the feasibility and performance of the proposed security framework and filtering system.

As future work, we plan to investigate mechanisms that increase the intelligence of our management framework to

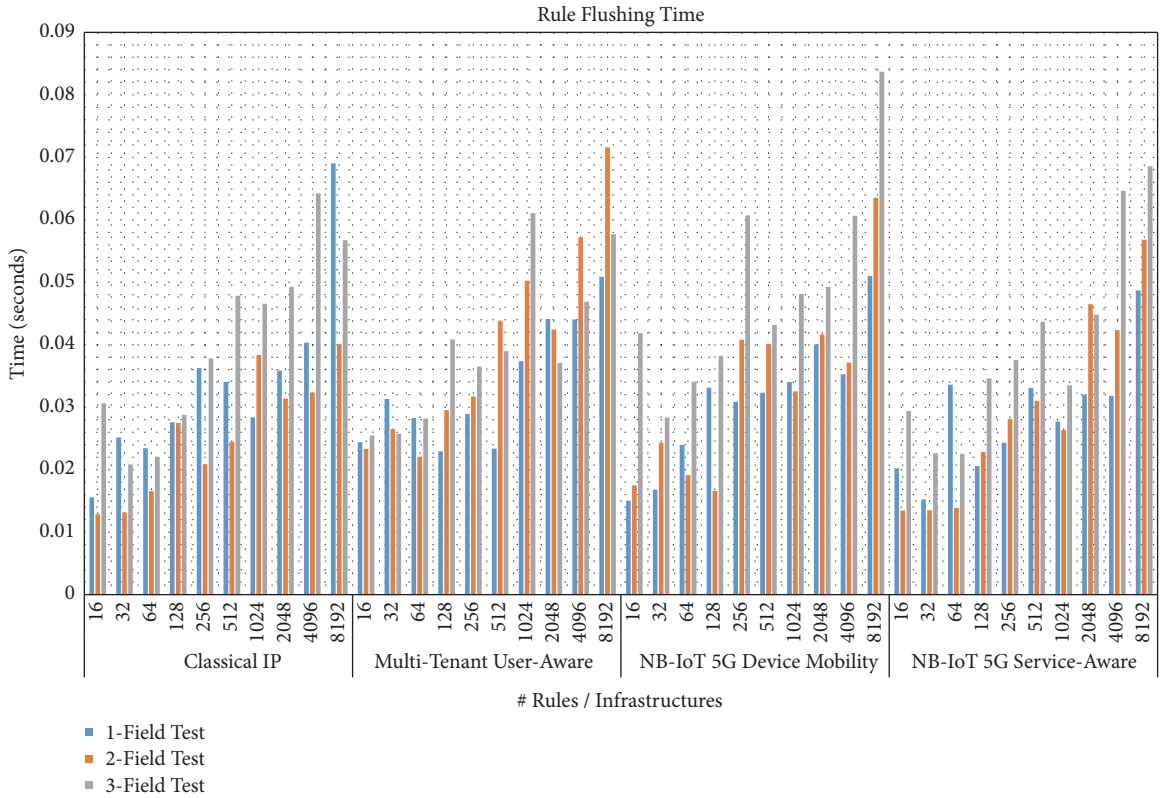

Figure 15: Flushing times of cleaning NB-IoT Service-aware 5G Multitenant Infrastructure rules.

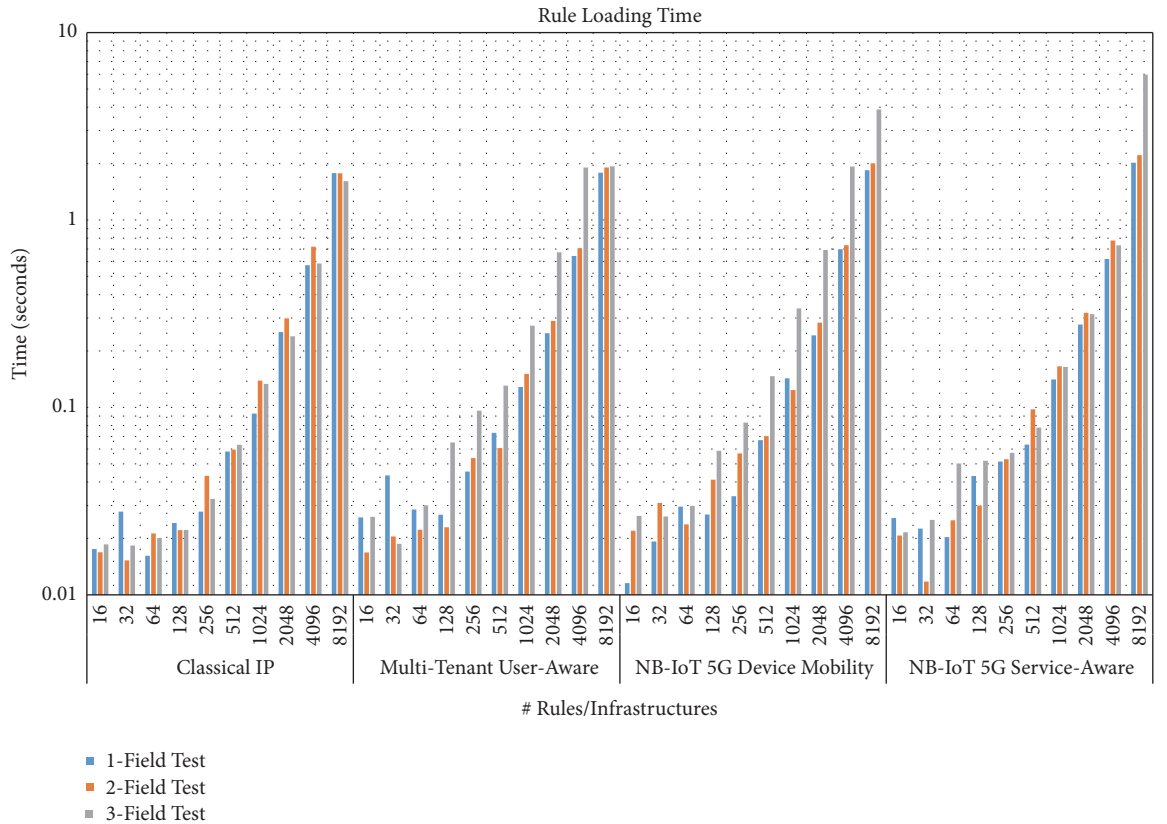

Figure 16: Loading time of adding NB-IoT Service-aware 5G Multitenant Infrastructure rules.

increase scalability and speed up, even more, the performance of the traffic filtering mechanism. We also envisage extending the security capabilities of the framework, by exploring the autonomic deployment and management of other kinds of virtual Network Security Functions, such as vAAA or vChannelProtection, to cope with the challenging requirements of IoT scenarios.

Data Availability

No external data were used to support this study. All network packet's captures and derived data sets have been generated in our infrastructure.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been partially funded by "Fundacion Seneca-Agencia de Ciencia y Tecnologia de la Region de Murcia", under the program "Jimenez de la Espada de Movilidad Investigadora, Cooperacion e Internacionalizacion" stay (20177/EE/17). The paper is the result of a joint collaboration partially funded by two EU H2020 projects: SLICENET, Grant Agreement H2020-ICT-2016-2/761913 and EU ANASTACIA, Grant Agreement no. 731558. In addition, it has been supported by a postdoctoral INCIBE grant "Ayudas para la Excelencia de los Equipos de Investigación Avanzada en Ciberseguridad" Program, with Code INCIBEI-2015-27363.

References

[1] M. Hatton, *Machina research predicts 10 million 5g internet of things connections in 2024*, 2016, https://machinaresearch.com/report/5g-will-account-for-10-million-cellular-iot-connections-in-2024/.

[2] M. De Donno, N. Dragoni, A. Giaretta, and A. Spognardi, "DDoS-Capable IoT Malwares: Comparative Analysis and Mirai Investigation," *Security and Communication Networks*, vol. 2018, pp. 1–30, 2018.

[3] Y. Gao, Y. Peng, F. Xie et al., "Analysis of security threats and vulnerability for cyber-physical systems," in *Proceedings of 2013 3rd International Conference on Computer Science and Network Technology*, pp. 50–55, October 2013.

[4] *Major DDoS Attacks Involving IoT Devices*, 2016, https://www.enisa.europa.eu/publications/info-notes/major-ddos-attacks-involving-iot-devices.

[5] Y. E. Wang, X. Lin, A. Adhikary et al., "A Primer on 3GPP Narrowband Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 117–123, 2017.

[6] Technical Specification Group GSM/EDGE Radio Access Network, "Cellular system support for ultra-low complexity and low throughput Internet of Things (CIoT) (Release 13)," Tech. Rep. 45.820 V13.1.0, 3rd Generation Partnership Project, January 2015.

[7] L. Zhou, M. Liao, C. Yuan, and H. Zhang, "Low-Rate DDoS Attack Detection Using Expectation of Packet Size," *Security and Communication Networks*, vol. 2017, 2017.

[8] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (CoAP)," *IETF*, Article ID RFC 7252, 2014.

[9] A. El-Atawy, T. Samak, E. Al-Shaer, and L. Hong, "Using online traffic statistical matching for optimizing packet filtering performance," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 866–874, Anchorage, Alaska, USA, 2007.

[10] A. El-Atawy and E. Al-Shaer, "Adaptive early packet filtering for defending firewalls against DoS attacks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '09)*, pp. 2437–2445, Rio de Janeiro, Brazil, 2009.

[11] Z. Trabelsi, L. Zhang, and S. Zeidan, "Dynamic rule and rule-field optimisation for improving firewall performance and security," *IET Information Security*, vol. 8, no. 4, pp. 250–257, 2014.

[12] Z. Trabelsi, S. zeidan, M. M. Masud, and K. Ghoudi, "Statistical dynamic splay tree filters towards multilevel firewall packet filtering enhancement," *Computers & Security*, vol. 53, pp. 109–131, 2015.

[13] S. Ziegler, A. Skarmeta, J. Bernal, E. E. Kim, and S. Bianchi, "Anastacia: Advanced networked agents for security and trust assessment in CPS IoT architectures," in *Proceedings of the Global Internet of Things Summit, GIoTS 2017*, pp. 1–6, Switzerland, 2017.

[14] P. Neves, R. Calé, M. R. Costa et al., "The SELFNET Approach for Autonomic Management in an NFV/SDN Networking Paradigm," *International Journal of Distributed Sensor Networks*, vol. 12, no. 2, 2016.

[15] P. Salva-Garcia, J. M. Alcaraz-Calero, R. M. Alaez, E. Chirivella-Perez, J. Nightingale, and Q. Wang, "5G-UHD: Design, prototyping and empirical evaluation of adaptive Ultra-High-Definition video streaming based on scalable H.265 in virtualised 5G networks," *Computer Communications*, vol. 118, pp. 171–184, 2018.

[16] R. Ricart-Sanchez, P. Malagon, P. Salva-Garcia, E. C. Perez, Q. Wang, and J. M. Alcaraz Calero, "Towards an FPGA-Accelerated programmable data path for edge-to-core communications in 5G networks," *Journal of Network and Computer Applications*, vol. 124, pp. 80–93, 2018.

[17] A. Serrano Mamolar, Z. Pervez, J. M. Alcaraz Calero, and A. M. Khattak, "Towards the transversal detection of DDoS network attacks in 5G multi-tenant overlay networks," *Computers & Security*, vol. 79, pp. 132–147, 2018.

[18] M.-A. Kourtis, H. Koumaras, G. Xilouris, and F. Liberal, "An NFV-Based Video Quality Assessment Method over 5G Small Cell Networks," *IEEE MultiMedia*, vol. 24, no. 4, pp. 68–78, 2017.

[19] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "IoT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT," in *Proceedings of the IEEE International Conference on Distributed Computing Systems, ICDCS 2017*, pp. 2177–2184, USA, 2017.

[20] H. Hsieh, J. Chen, and A. Benslimane, "5G Virtualized Multi-access Edge Computing Platform for IoT Applications," *Journal of Network and Computer Applications*, vol. 115, pp. 94–102, 2018.

[21] W. Meng, "Intrusion detection in the era of IoT: Building trust via traffic filtering and sampling," *The Computer Journal*, vol. 51, no. 7, pp. 36–43, 2018.

[22] N. Pandeeswari and G. Kumar, "Anomaly Detection System in Cloud Environment Using Fuzzy Clustering Based ANN,"

Mobile Networks and Applications, vol. 21, no. 3, pp. 494–505, 2016.

[23] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[24] A. V. Aho and M. J. Corasick, "Efficient string matching: an aid to bibliographic search," *Communications of the ACM*, vol. 18, pp. 333–340, 1975.

[25] R. S. Boyer and J. Strother Moore, *A Fast String Searching Algorithm*, Palo Alto Research Center, Palo Alto, California, USA, 1976.

[26] A. N. M. E. Rafiq, M. W. El-Kharashi, and F. Gebali, "A fast string search algorithm for deep packet classification," *Computer Communications*, vol. 27, no. 15, pp. 1524–1538, 2004.

[27] S. Kumar, S. Dharmapurikar, F. Yu, P. Crowley, and J. Turner, "Algorithms to accelerate multiple regular expressions matching for deep packet inspection," *SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 339–350, 2006.

[28] R. Smith, C. Estan, S. Jha, and S. Kong, "Deflating the big bang: Fast and scalable deep packet inspection with extended finite automata," *SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 207–218, 2008.

[29] T. Bujlow, V. Carela-Español, and P. Barlet-Ros, "Independent comparison of popular DPI tools for traffic classification," *Journal of Network and Computer Applications*, vol. 76, pp. 75–89, 2015.

[30] ONF: Open networking foundation website, https://www.opennetworking.org/.

[31] B. Pfaff and O. v. S. Commiter, http://www.openvswitch.org/support/slides/p4.pdf.

[32] W. c. C. Tu, "Offloading OVS flow processing using eBPF," http://www.openvswitch.org/support/ovscon2016/7/1120-tu.pdf.

[33] W. Stearns, "Netfilter extensions HOWTO: New netfilter matches," http://www.netfilter.org/documentation/HOWTO/netfilter-extensions-HOWTO-3.html.

[34] S. McCanne and V. Jacobson, "The bsd packet filter: A new architecture for user-level packet capture," in *Proceedings of the USENIX Winter 1993 Conference Proceedings on USENIX Winter 1993 Conference*, pp. 2-2, USENIX Association, 1993.

[35] "Harald Welte, P.N.A.: netfilter/iptables project homepage - the netfilter.org "iptables" project," http://www.netfilter.org/projects/iptables/index.html.

[36] J. Schlienz and D. Raddino, "Narrowband internet of things whitepaper," *IEEE Microwave Magazine*, vol. 8, no. 1, pp. 76–82, 2016.

[37] C. Mysirlidis, A. Lykoyrgiotis, T. Dagiuklas, I. Politis, and S. Kotsopoulos, "Media-aware proxy: Application layer filtering and L3 mobility for media streaming optimization," in *Proceedings of the IEEE International Conference on Communications, ICC 2015*, pp. 6912–6917, UK, 2015.

[38] A. Stanciu, T.-C. Balan, C. Gerigan, and S. Zamfir, "Securing the IoT gateway based on the hardware implementation of a multi pattern search algorithm," in *Proceedings of the International Conference on Optimization of Electrical and Electronic Equipment, OPTIM 2017 and Intl Aegean Conference on Electrical Machines and Power Electronics, ACEMP 2017*, pp. 1001–1006, Romania, 2017.

[39] J. Kima, D. Kimb, and S. Choi, "3gpp sa2 architecture and functions for 5g mobile communication," *ICT Express*, vol. 3, no. 1, pp. 1–8, 2017.

[40] A. Molina Zarca, J. Bernal Bernabe, I. Farris, Y. Khettab, T. Taleb, and A. Skarmeta, "Enhancing IoT security through network softwarization and virtual security appliances," *International Journal of Network Management*, vol. 28, no. 5, 2018.

[41] O. Standard, "OASIS advanced message queuing protocol (AMQP) version 1.0".