Highly-scalable software firewall supporting one million rules for 5G NB-IoT networks

Matencio Escolar, Antonio; Alcaraz Calero, Jose M.; Wang, Qi

Published in:
ICC 2020 - 2020 IEEE International Conference on Communications (ICC)

DOI:
10.1109/ICC40277.2020.9149152

Published: 27/07/2020

Document Version
Peer reviewed version

Matencio Escolar, A., Alcaraz Calero, J. M., & Wang, Q. (2020). Highly-scalable software firewall supporting one million rules for 5G NB-IoT networks. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)* (IEEE Conference Proceedings). IEEE. https://doi.org/10.1109/ICC40277.2020.9149152

Highly-Scalable Software Firewall Supporting One Million Rules for 5G NB-IoT Networks

Antonio Matencio Escolar
University of the West of Scotland
Paisley, Scotland
antonio.matencio@uws.ac.uk

Jose M. Alcaraz Calero
University of the West of Scotland
Paisley, Scotland
jose.alcaraz-calero@uws.ac.uk

Qi Wang
University of the West of Scotland
Paisley, Scotland
qi.wang@uws.ac.uk

***Abstract*—There is a significant lack of software firewalls for 5G networks especially when the support for the Internet of Things (IoT) technologies such as NB-IoT are considered. The main contribution of this research work is an advanced software firewall based on the Open Virtual Switch (OVS), which is able to provide firewall capabilities over these 5G IoT devices. The proposed software firewall is able to significantly scale up the number of rules to fulfill the 5G Key Performance Indicator of controlling 1 million IoT devices per square kilometer. Intensive experimental results are achieved in this work, validating the suitability of the proposed architecture for this remarkable level of scalability. In the most demanding conditions, where more than 1 million of firewall rules are installed and 1 million NB-IoT devices are sending traffic, yielding a total of 4 Gbps, the system shows only 8% of packet loss and 4 ms delay.**

***Index Terms*—5G, NB-IoT, OpenVSwicth, Software Datapath, firewall**

I. Introduction

The maximum 5G speed in the New Radio (NR) interface reported by Huawei in October 2019 [1] is 3.67 Gbps, beating their previous world-wide mark of 2 Gbps. A more typical scenario using the same technology indicates 1 Gbps for the coverage of 1 square kilometer. In that coverage, a 5G NB-IoT (NarrowBand-Internet of Things) network is expected to provide access to 1,000,000 devices according to the 5G Key Performance Indicator (KPI) defined by 5G Public-Private Partnership (PPP). When combined with softwarization and virtualization, which are the cornerstone technologies in 5G architectures to reduce capital expenditure (CAPEX) and operational expenditure (OPEX) , it imposes a significant scalability challenge and performance overhead that need to be addressed to fulfill the ambitious 5G KPI.

This work was funded in part by the European Commission Horizon 2020 5G-PPP Programme under Grant Agreement Number H2020-ICT-2016-2/761913 (SliceNet: End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks). This work has been also supported by the UWS VP Fund - 5G Video Lab.

Currently, software firewalls are primarily designed to protect traditional IP networks. The support to protect overlay IP networks used by 5G NB-IoT architectures has not been sufficiently provided. Moreover, to the best of the authors' knowledge, there is no published software-based firewall solution that is able to deal with the level of scalability envisioned for the massive number of IoT devices. These gaps pose significant security challenges that need to be addressed.

This paper attempts to address these problems by providing a novel software firewall capability with support for 5G NB-IoT overlay networks. The software firewall exposes a significant increase in the scalability with respect to the number of rules, up to 5G expectations. The following list enumerates the main contributions of this work:

- Novel 5G software firewall architecture with advanced capabilities for 5G-enabled IoT networks.
- Significant enhancement of the scalability in terms of handling a large number of firewall rules for security proposes, being able to handle up to 1 million firewall rules per software firewall.
- Empirical validation of the scalability and performance of the proposed solution based on a prototypical implementation in a realistic testbed.

The rest of this paper is structured as follows. Section II outlines a state of the art on software firewall capabilities and firewall filtering in overlay networks. Section III describes the design and prototyping of the proposed scalable 5G IoT firewall architecture. Section IV presents the implementation of the proposed architecture. Section V validates the solution and provides a scalability analysis of the prototype. Finally, Section VI provides conclusions and future work.

II. Related Work

The vast majority of open source and commercial software switches that could be extended to act as firewalls simply have not been designed to support overlay networks, and they merely work in traditional

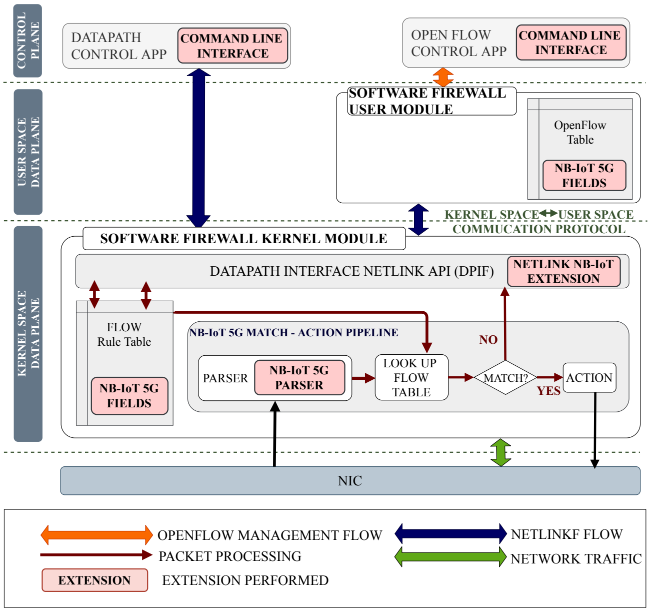

Fig. 1. Architectural of the proposed software firewall

IP networks. For example, Linux iptables, ebtables, ipcop, pfSense, ipFire, ufw, smoothwall and VyOS firewalls do not support any overlay network, including the GPRS Tunneling Protocol (GTP) used to implement 5G NB-IoT networks. Windows Firewall, Avast, AVS, TinyWall, GlassFire and many others also lack the same capability for the Windows operating system.

There is significant absence of solutions to address the lack of support of firewall policies over the GTP protocol, used in LTE, LTE-Advanced (LTE-A) and 5G and on their respective adaptions for cellular IoT networks, LTE-M and NB-IoT. In the hardware side, Ricart et al. [2] provided a hardware appliance with these novel capabilities able to work up to 3.67 Gbps and up to 1024 wildcard-enabled firewall rules against 512 flows (305 kilo packet per second - kPPS). In the software side, Salva et al. [3] indicated that the maximum rules support for Linux IP tables, with an extended version to support GTP traffic, in the most ideal conditions, are 512 rules when traffic is transferred at 1 Gbps against 512 flows (666 kPPS). Salva et al. [4], further investigated the support for NB-IoT traffic using IP tables-BPF integration, optimized for scalability on the number of rules, achieving a maximum of 4096 rules for 4096 simultaneous flows at 90 Mbps (60 kPPS). At higher speeds, packets drops and delay start to be unacceptable. Such level of scalability will work for normal cellular network end users although it is not able to deal with the significantly higher level of scalability envisioned for 5G IoT. In early 2019, FortiGate, a software appliance from Forninet [5], claimed to provide a carrier-grade firewall support for LTE, LTE-A, 5G and IoT. However, these capabilities are not reflected yet in their data sheets, no performance has been published and for their highest-end product (VM08), they claimed to provide support for up to 4 Gbps with a maximum of 40k firewall rules. Even that level of scalability in software appliances will not be suitable for 5G requirements. Another way to address this scalability is to perform the deployment of several virtual appliances in the same physical machine in order to use a distributed load-balancing approach to deal with scalability.

The lack of support for such advanced firewall capabilities in software solutions and the need to push the scalability boundaries to truly support for 5G networks has been the main motivation of this work.

III. The Proposed Architecture

Fig. 1 provides an overview of the proposed highly-scalable 5G NB-IoT software firewall architecture. It has been logically divided in three different planes. The kernel space module works at the maximum speed with hardware administrative privileges (execution ring 1). When a packet is received by the network interface card (NIC) driver, it is inserted into the match-action pipeline implemented in this kernel module. The match-action pipeline applies the firewall rules to the packets being received in the data path. To do so, the packets are parsed using the extracted metadata. An extension to the traditional IP packet parsing has been designed and prototyped to be able to extract information about the GTP protocol and also about the inner IP headers that are inside the tunneling protocol to be able to provide firewall capabilities not only to traditional IP traffic but also to 5G NB-IoT traffic. This parsing extension is explained later in subsection 2. The metadata extracted is matched against a firewall rule table where all the firewall rules are inserted for a lookup in the table. It is noted that the traditional design of a table of rules only considers fields of the traditional IP network. The proposed system extended the rule table with the new fields related to the GTP protocol in order to include the new expressions of the firewall rules for 5G NB-IoT. If there is a match in the rule table, then the action is taken. Different default policies are supported with deny or accept by default which is seen as the last implicit action to be applied if there is not any match against the rules. The key for speed is the design in the kernel space. Meanwhile, there is a noticeable trend to bypass the kernel and implement everything in the user space. Our approach is to make use of the kernel which is a much more optimized way when compared with

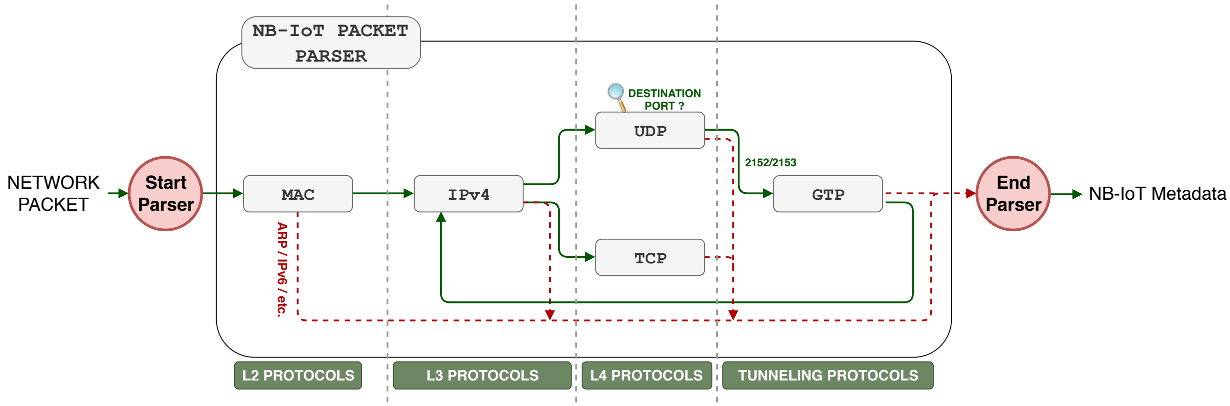

Fig. 2. Extended parser with 5G NB-IoT support

the traditional kernel modules, and thus a new kernel module is proposed. To help achieve scalability, we have taken a co-design approach between kernel and user spaces, and thus further introduced another new module implemented in the user space. This module keeps all the rules inserted into the firewall by the user in memory. The module employs OpenFlow and the architecture of tables defined in OpenFlow for this purpose. When the packets being processed in the kernel do not match any of the installed rules in the table of the kernel, a communication between kernel and user spaces takes place to perform a lookup in the user-space tables. If there is a match, the firewall rule in the user space is installed in the kernel module for performance purposes. This communication between user space and kernel module has been also extended in order to inter-exchange all the new metadata required to be copied between user and kernel spaces. Then, another key decision for scalability has been the fact that when rules are installed in the kernel, if they have not been looked up for a period of time, they are removed from the table, freeing up resources and thus enhancing scalability. When new packets come, the rules will be reinstalled into the kernel by means of the user space to kernel migration of rules mechanisms. This architectural philosophy is not radically new, and the Open Virtual Switch (OVS) [6] software switch, used for multiple purposes in softwarized networks, adopts a similar architecture. However, OVS does not provide firewall capabilities for 5G NB-IoT, which is addressed in this work. Our prototypes are a significant extension of the original OVS software to deal with an enhanced level of scalability.

A. NB-IoT Parsing Extension

Fig. 2 shows the proposed extension over the traditional parsing of IP packets. When a network packet arrived at the parser, it visits different metadata extractors. For example, in the MAC processing, the source and destination MAC addresses and the Ethernet type field are extracted in order to allow later on creating firewall rules based on the metadata. Analogously, source and destination IP addresses and key attributes available in the IP header are extracted, including transport protocol, Differentiated Services Control Protocol (DSCP) field, time-to-live (TTL) field, among others. In UDP and TCP, the source and destination ports are extracted. For TCP, all the TCP flags are also extracted to allow key security firewall rules to be applied and to deal with the tracking of the connection to support both stateful and stateless operational modes on the firewall. Then, when the GTP protocol is detected, the Tunnel Endpoint Identification (TEID) is extracted. This field is used to uniquely identify a 5G NB-IoT device across all the antennas of the network. This tunneling protocol has the IP packets generated by the NB-IoT devices (inner traffic) whereas the outer traffic is the 5G infrastructure traffic required to deal with NB-IoT connectivity, control and mobility (if supported). Thus, the inner traffic need to be parsed again. To do so, it takes a re-entrance on the previously visited parsing steps. It will allow extracting now the information related to the NB-IoT devices including, specifically, the source and destination IP addresses, ports and other key firewall information. This extension in the parsing provides the number capabilities to deal with NB-IoT firewall rules, which is one of the main motivations of this paper.

The following excerpt of code is an example of a firewall rule supported now thanks to our extension (rule in JSON format). A flow from inner source port 16500 of a given NB-IoT device with source IP address 10.10.10.1 going to outer destination port 2152 is identified in the 5G network with the TEID 2001 using GTP as the tunneling protocol, and it will be dropped once mapped:

```
{"rule": {
  "firewall": "fw0",
  "action": "add-rule",
  "table" : 0,
  "cookie": "0x100",
  "priority": "0xFF",
  "match": [
      {"outer_destination_port": 2152 },
      {"tunnel_protocol": "GTP" }
      {"tunnel_key": 2001 },
      {"inner_source_ip": "10.10.10.1" },
      {"inner_source_port": 16500 }
    ],
   "action": "drop"
   }
}
```

As it can been observed, each firewall rule has a field termed "cookie" which purpose is to uniquely identify a rule. Besides, Each rule is inserted into a table and a priority is set for it.

IV. Implementation

The proposed architecture has been implemented as extensions in OVS (version 2.9.2) using the C language in both user and kernel spaces. More details of the original OVS can be found in [7]. To be concrete, the first extension has been carried out over the kernel module of OVS (openvswitch.ko) to support parsing of 5G NB-IoT protocol and new fields in the expressions of firewall rules. The second extension has been applied on the Netlink communication between the openvswitch.ko module and the OVS user-space software daemon (vswitchd). The command line tools have also been extended, including both ovs-dpctl and ovs-ofctl in order to allow the administrator to insert the rules in both OpenFlow tables and kernel tables directly on demand. These prototypical extensions were introduced following exactly the design presented in Section III, and the results achieved in this prototype are described in Section V.

V. Experiment Results

This section validates the suitability of the proposed NB-IoT firewall and analyses the scalability achieved in the number of rules and the performance achieved in terms of delay, jitter, packet loss and throughput.

A. Testbed Description

Experiments have been conducted on a host computer testbed for the proposed OVS-based 5G NB-IoT software firewall, with the following specification: Dell T5810 with 1xIntel Xeon E5-2630 v4 CPU (10 cores with hyper-threading), 32GB RAM, and 512GB SSD HDD. Fig. 3 depicts the setup of the testbed deployed to empirically validate the proposed firewall. Experiments are handled by the *experiment controller script* that customizes the configuration of each experiment with different parameters that allow analyzing the behaviour of the firewall in different scenarios. These parameters and their range of values are explained in Subsection V-B.

For each experiment, the *experiment controller script* injected the rules into the NB-IoT 5G firewall through a dedicated management interface (See 1 in Fig. 3). After that, the *traffic generator agent* generated several pcap files that were sent in parallel by the *traffic sender agent* (See 2 and 3 in Fig. 3). The pcap files generated are compliant with the NB-IoT protocol. The number of flows generated by each NB-IoT device was fixed to 1 flow, and the number of NB-IoT devices sending traffic was always matched with the number of rules. Thus, a rule always produced a match to a given flow leading to dropping or passing. This is a way to ensure that the experiments were fair and there was not any kind of artificial acceleration due to the synthetic generation of the pcaps. In fact, each NB-IoT device available in the pcap had different source and destination IP address, different source and destination ports, and even different GTP tunnel ID. This traffic represented the pattern and behaviour of a 5G node allocated at the edge of the network, where all these NB-IoT devices were connected to the radio interface managed by the edge node. The traffic was then received by the 5G NB-IoT firewall, which processed it through the NB-IoT 5G Match-Action pipeline. The outgoing traffic was sent back to the virtual machine where it was captured by the *traffic receiver agent* and saved in a pcap file. In a final step, the *results analyzer agent* compared both sent and received pcap files to gather the experiment results (delay, jitter, packet loss and throughput). In the experiments, the receiver received the whole traffic originally sent since there is an "drop-by-default" firewall policy and each flow has a "pass/accept" action associated.

B. Experiments

Table I lists the range of values of the different parameters that configured each of the experiments conducted to validate and evaluate the proposed 5G NB-IoT software firewall. As shown in the table, NB-IoT traffic consisting of 1500 byte MTU was transmitted by every IoT device. Scalability was analyzed in two different dimensions: the number of IoT devices sending traffic and amount of traffic sent by each of

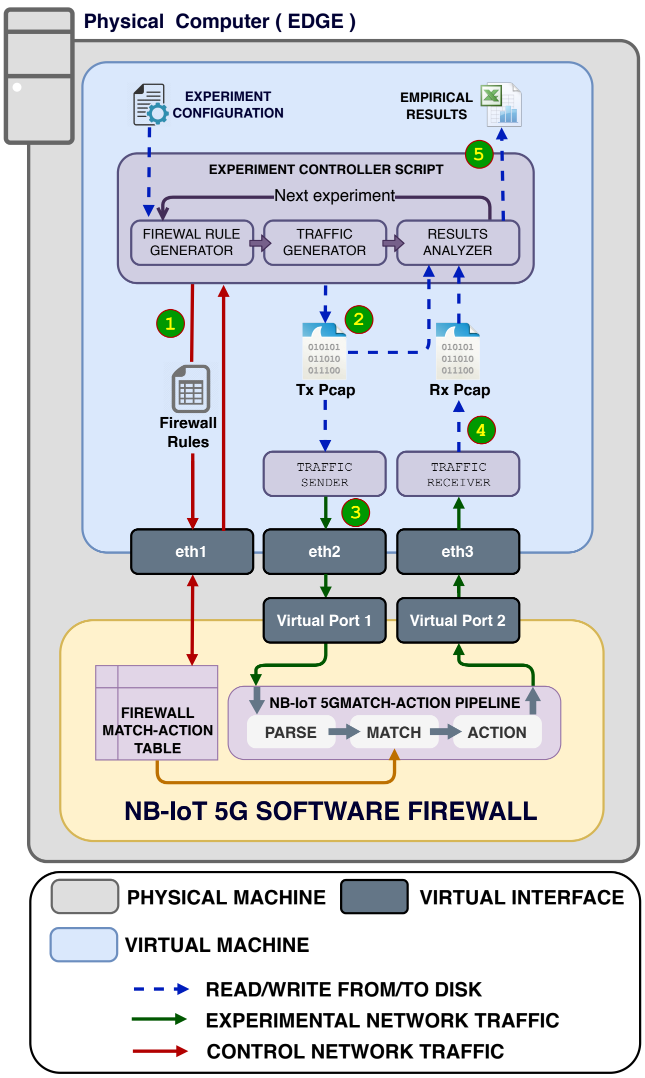

Fig. 3. Testbed deployed for the proposed OVS-based 5G NB-IoT software firewall

these devices. As mentioned, the number of NB-IoT devices matched the number of rules inserted in the firewall. The traffic sent by each of the IoT devices was classified and processed in an isolated way with respect to the rest of the traffic and each of the rules was tailored for that purpose. Through this scheme, an ultra-fine grained firewall control mechanism was provided for the NB-IoT traffic in a 5G architecture. For simplicity, for each different configuration executed in the testbed, all IoT devices transmitted traffic with the same bandwidth. This bandwidth was the result of dividing the total available bandwidth by the number of connected IoT devices.

With Regard to the methodology applied, each experiment was executed 10 times. To avoid outlier values, the best and worst results were ignored and the final outcomes shown in this paper are the arithmetic mean

TABLE I
Range of values for each parameter analyzed in the experiments

Parameter	Range of Values
Packet size (MTU)	1500 Bytes
Bandwidth (Mbps)	1000, 2000, 3000, 4000
Number of Devices/Rules	1, 2, 4, 8, 16 ... 1048576 (2^n,n in [0,1,..,20])
Type of Traffic	NB-IoT Traffic (GTP)
Type of Rule	Matching inner source ip address and inner destination port

of the remaining 8 experiments.

C. Results

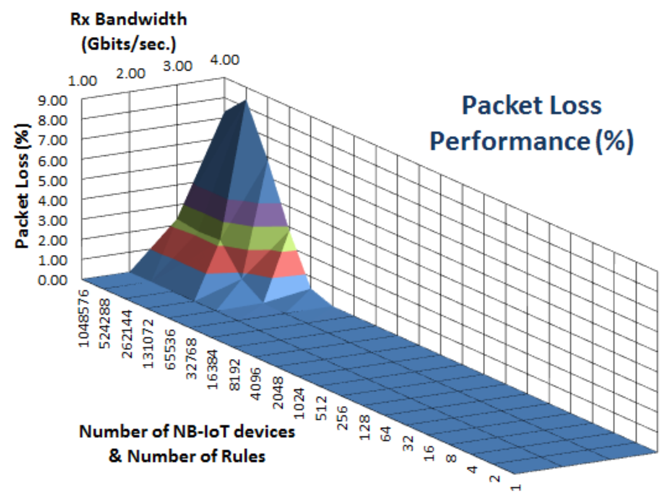

Fig. 4. Scalability analysis of packet loss

Fig. 4 shows the evolution of packets loss when ranging exponentially both the number of firewall rules and devices in 1:1 ratio and also when ranging linearly the total bandwidth for transmission per corresponding number of devices. It is worth mentioning that 1 Gbps there was no packet loss when 1 million of flows and rules were installed in the firewall. For the case of 4 Gbps, speeds currently aligned with the maximum performance achieved in the new radio interface up to date, there was no packet loss either up to 32k rules. Beyond these boundaries of system stability, packet loss started to appear and in the most stressful condition, at 4 Gbps, with 1 million rules and 1 million devices, a very reasonable 8.2% packet loss ratio was achieved. These numbers are remarkably advantageous especially considering highly demanding nature of the IoT communications. It is noted that in all the scenarios tested, the transmission throughput was exactly the same as the reception throughput with the variations associated to the percentage of packet loss. Thus it has been decided not to include this graph although these facts are critical to understand that all the graphs presented next are accurate.

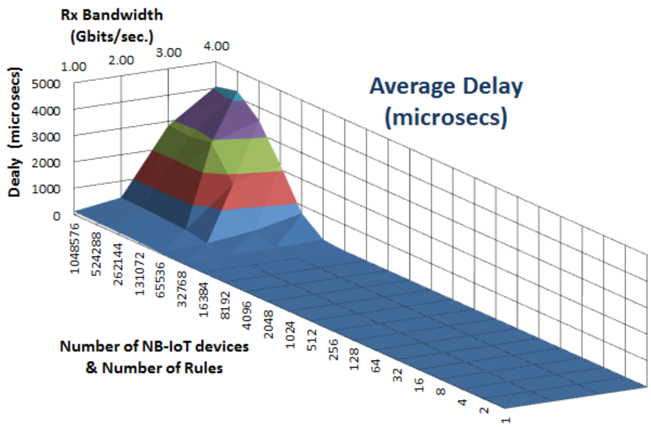

Fig. 5. Scalability analysis of delay

Fig. 5 shows the evolution of the delay incurred by the software firewall when ranging the same parameters. The behaviour of the graph is similar to that of packet loss. The system was stable despite the delay added by the number of rules up to 32k rules. After this threshold, the system behaved very decently at 1 and 2 Gbps. When the throughput were scaled to 3 and 4 Gbps, the delay was increased exponentially. It is very relevant to indicate that at 1 Gbps, the delay inserted when there was 1 million rules was around 0.1 ms and at 4 Gbps, the delay added was around 4 ms. These numbers are far beyond better than the acceptance boundaries defined for 5G NB-IoT traditionally associated to delay-tolerant applications. These results show that the firewall will be suitable even for delay-sensitive 5G NB-IoT use cases.

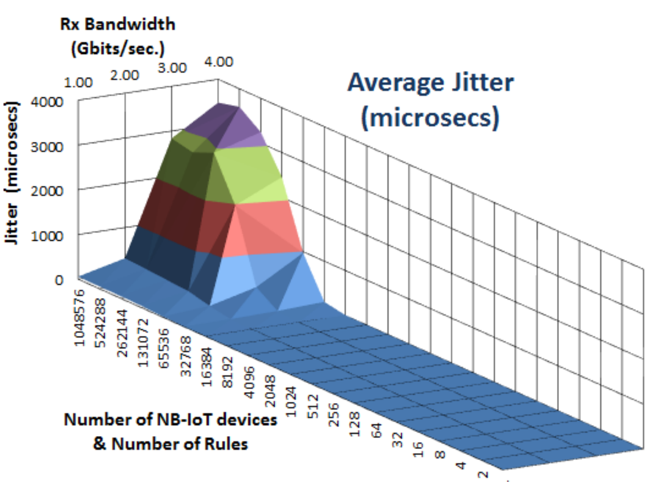

Fig. 6. Scalability analysis of jitter

Regarding the analysis of jitter (i.e., variance in packet delays), Fig. 6 shows a very similar trend with respect to the delay graph presented in Fig. 5. The stable boundaries were similar and then an exponential increase appeared following a similar pattern. This time, in the most stressed scenario (at 4 Gbps, 1 millions flows and 1 million rules), the jitter was about 3.5 ms. When combined with the delay, the worst case maximum delay would be lower than 8 ms.

These results have validated the suitability of the proposed architecture for the scalability and performance envisioned in 5G networks.

VI. Conclusion

This paper proposes a highly scalable software firewall architecture that provides support for NB-IoT devices in 5G networks and beyond. The architecture has been validated with the extreme use case of massive machine-type communications composed by traffic of 1 million devices, totalling 4 Gbps, with 1 million of firewall rules installed. Experiment results have shown that the solution has very promising performance of around 8% of packet losses and just 4 ms delay under these extreme conditions where all these devices are connected to the same radio interface and the software firewall is deployed in the edges of the network. The level of scalability is compliant with the 5G KPI expectation.

In future work, this level of high scalability will be explored in the context of network slicing and network slice management where different types of actions need to be applied over the same type of traffic. In addition, the compatibility to other IoT protocols will also be investigated.

References

[1] "Huawei 5g wireless network planning solution white paper," https://www-file.huawei.com/-/media/corporate/pdf/white\paper/2018/5g_wireless_network_planing_solution_en.pdf?la=en-ch, Huawei Technologies Co., Ltd., 2018.

[2] R. Ricart-Sanchez, P. Malagon, J. M. Alcaraz-Calero, and Q. Wang, "Netfpga-based firewall solution for 5g multi-tenant architectures," in *2019 IEEE International Conference on Edge Computing (EDGE)*, July 2019, pp. 132–136.

[3] 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.

[4] P. Salva-Garcia, J. M. Alcaraz-Calero, Q. Wang, J. B. Bernabe, and A. Skarmeta, "5G NB-IoT: Efficient Network Traffic Filtering for Multitenant IoT Cellular Networks," *Security and Communication Networks*, vol. 2018, pp. 1–21, dec 2018.

[5] (2019) The Fortinet website. [Online]. Available: https://www.fortinet.com/solutions/mobile-carrier.html

[6] (2019) The Open vSwitch website. [Online]. Available: https://www.openvswitch.org/

[7] B. Pfaff and et al., "The design and implementation of open vswitch," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 117–130.