Software Defined Networks and Network Function Virtualisation Testbed within FIRE+

Third Wave of Experiments on the SoftFIRE Platform

NFV and SDN Experiments for 5G Networks

June 2018

SoftFIRE

Table of Contents

List of Figures

List of Tables

1 Introduction

SoftFIRE [1] has a modular and extensible middleware [2], which abstracts the complexity of underlying open-source software, i.e. the infrastructure controllers. This has made it possible to extend the middleware with software *manager* modules, each responsible for a certain group of functions, e.g. software-defined networking (SDN)[3], network functions virtualisation (NFV)[4], security enablement [5], physical device reservation [6], and experiment monitoring [7]. Thanks to this capability to support these various types of virtualisation functions, and its flexible and easy-to-use Experimenter Manager [8] middleware, the Project could support 13 experiments during its 3rd Wave of Experiments [9], reaching a total of 30 experiments, plus many more in its Final Challenge [10] and hackathon events [11][12][13].

In this white paper, the experiments that were successfully deployed on the SoftFIRE platform during its 3rd Wave of Experiments are briefly presented. In doing so, the intention is to present what was achieved by experimenters on the platform, and the types of NFV [14][15] and SDN [16] related 5G application/solution experiments that were executed on the platform. The 3rd Wave of Experiments included the following selected experiments:

- Experience
- 5G MEC Caching
- Breeze (Balancing Requests for Backend Zones)
- BotsOnFIRE (Mitigation of Botnets in Federated SDN/NFV Infrastructures for 5G)
- Slicemon (Slice QoS Monitoring in a vCDN environment)
- NFVOB (NFV Framework testing with OpenBaton)
- MONET (Monitoring Network Security in SDN/NFV environment)
- I-EVS (Enhanced Video Services for the Mobile Edge Computing in 5G environment)
- Flexnet (FLEXible NETworks)
- Softblast (SoftFIRE BLockchAin SDN Technology)
- Dune (Disconnecting Users from Network)
- HighPep (High Performance OneM2M Edge Processing)
- Inferno (IoT Interoperability over Federated SDN/NFV Domains)

The white paper presents summaries of the architecture, experimentation, and contributions of this set of experiments. Each experiment is presented in a separate section below.

2 EXPERIENCE

5G networks are to offer enhanced mobile broadband connections, whilst supporting low latency and increased Quality-of-Experience (QoE) for all users of the network, which are necessary requirements for immersive Augmented Reality (AR) applications. AR content includes new formats, such as stereoscopic, high dynamic range (HDR), and 360°, videos and 3D objects at increased resolutions (8K+) and higher framerates (90+ fps). Although basic implementations of these formats can be delivered through 4G networks, large-scale adoption of applications that use these formats will soon congest 4G network, thus rendering the user experience intolerable.

A fundamental but still common problem that almost all European companies that deliver mobile services face each time that a new product or feature is about to launch is testing the limits of such services. In the context of 5G and in particular AR applications, this inefficiency mostly refers to the lack of tests to evaluate the behaviour of the services under different conditions in terms of network capacity, latency, and uniform user experience. The situation becomes even worse when AR content is to be delivered to users since more computing power is needed while users demand minimal delay in content delivery. Hence, performance testing is a fundamental need for every new AR application. Currently, thorough and extensive testing is a missing point in the lifecycle of AR content and services.

The experiment EXPERIENCE conducted by the company Intellia ICT [17] aimed to analyse the performance of the company's provided virtual Augmented Reality (AR) solutions that run on the state-of-the-art virtualisation platform offered by SoftFIRE. By taking advantage of the software and hardware abstractions provided by SoftFIRE, the experiment revealed a set of best practices and adaptive strategies for optimal delivery of AR content.

The performance of different AR applications were measured on the SoftFIRE infrastructure. In doing so, the experiment aimed to facilitate the European Augmented Reality ecosystem by (*i*) providing the virtualized tools and services for testing AR applications over an existing European NFV/SDN/5G infrastructure, and (*ii*) offering a best-practice guide and recommendations for testing AR applications. Different configurations and application requirements were used to extract useful results and practical recommendations to the community of experimenters and practitioners.

2.1 Objectives

The main technical and scientific objectives of the EXPERIENCE experiment were the following:

- **Objective 1:** To validate the proper execution of NFV-powered AR applications via a set of VNFs and evaluate their performance over the SoftFIRE platform.
- **Objective 2:** To analyse the results of the experiment and provide a set of best practices and practical recommendations to 5G experimenters for the AR domain.

To achieve these objectives, the experiment involved the following technical tasks

- **Task 1:** To adapt and extend conventional Augmented Reality applications to comply with and fulfil the technical and operational requirements in the NFV platform and the mobile network testbed provided by SoftFIRE,

- **Task 2:** To integrate the AR applications with the SoftFIRE platform and especially with (*i*) the Experiment Manager environment, and (*ii*) the 5GIC testbed [18] in terms of orchestration, control, and virtualization capabilities.

2.2 Experiment

The physical network architecture of the experiment is provided in Figure 1. EXPERIENCE took advantage of the C-RAN equipment offered by SoftFIRE in the 5GIC component testbed, to reach the deployed VNFs. Specifically, the experiment used one femto cell that works on LTE Band 20 to access the SoftFIRE environment. In three of the four scenarios, multimedia content was retrieved from a web server and visualized on the respective mobile AR applications following the architecture of Figure 1. The architecture of the fourth scenario was almost the same as that of the other three, apart from the fact that a web application server was present instead of a multimedia one.

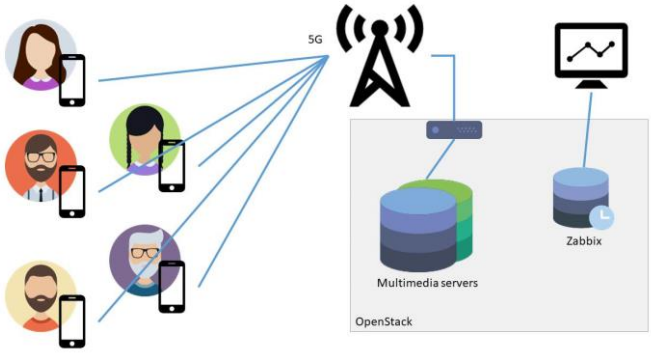

Figure 1. Architecture of the EXPERIENCE experiment.

The AR scenarios were decompiled as a series of VNFs (AR content, storage, execution, content delivery) that were accessed via the 5G user plane network slice dedicated to the experimenter. The experiment investigated the capability to insert these VNFs in an on-demand way, with respect to AR content storage, processing, and delivery, so as to achieve programmability in the network infrastructure. To achieve this, an external NFV controller software as well as a custom monitoring manager that includes a pre-configured Zabbix server were run by the experimenter. The AR applications were remotely configured, programmed and tested via the provided set of tools and platforms (e.g., NFN/SDN APIs) by SoftFIRE.

2.3 Deployed AR applications

In EXPERIENCE, several AR scenarios were developed, and assessed based on four different real-world AR applications that were customised to comply with the SoftFIRE infrastructure. Specific scenarios were tested in the following four application domains:

- Provision of multimedia content for historic monuments (cultural heritage and arts domain),
- AR-based training against fires (vocational training domain),
- 3D representation of our solar system (entertainment domain),
- Kids' learning (educational domain).

2.4 Key performance indicators

The following Key Performance Indicators (KPIs) were evaluated as part of the experiment.

2.4.1. Network bandwidth

The first parameter measured was network bandwidth. Measurements for this KPI were conducted on a single RAN access point. First, it was necessary to examine what the overall attainable network bandwidth was, how stable it was, and how the network behaviour was affected when multiple terminals (i.e. mobiles) were connected to the network. For this reason, stress tests were performed using different number of mobile phones and AR applications concurrently. Figure 2 demonstrates the measurement results of the stress tests using different numbers of mobile phones across a time period.

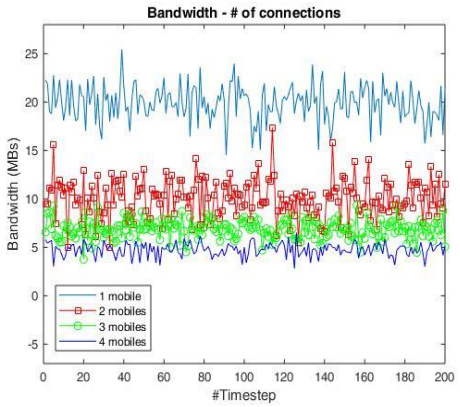

Figure 2. Network bandwidth measured using different number of mobile phones.

2.4.2. Latency

The second parameter measured was network latency. Four different mobile phones were used, each running one of the four EXPERIENCE AR applications. The network was observed to be stable and could serve the connected users resulting in an average latency of between 22ms and 28ms in all scenarios. This result is relevant for demanding applications, such AR applications, and especially those that aim to offer real-time 3D model visualization and video streaming. As shown in Figure 3, the measured latency did not have major differences across the different scenarios.

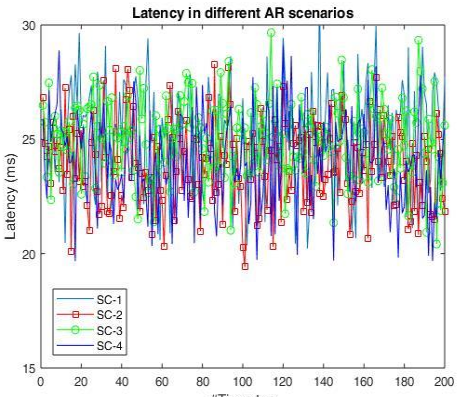

Figure 3. Network latency. (SC: Scenario)

Regarding the quality of experience received by the users, no significant difference was noted on different mobile phones. In particular, the overall network performance did not suffer from large performance variations, since the total bandwidth of the network was equally shared among the active users. The network was quite stable and the latency was similar on all mobile phones.

2.4.3. Average application storage requirement on mobile equipment

A major limitation of typical AR applications is their size, since they are mostly based on storage demanding resources such as multimedia material, and 3D animations and models. This usually results in applications of huge size (i.e. hundreds of MBs). By taking advantage of the cloud-based storage capability offered by SoftFIRE, EXPERIENCE aimed to reduce the

overall size of a typical AR application. In all application scenarios, reductions between 40% and 80% were obtained, in comparison to the initial original application size.

2.4.4. Number of simulated users

The number of concurrent users that a solution supports is a key indicator of its performance. Maximising this metric means that the system has the capacity to serve more users with less resources thus giving the advantage of lowering the cost of an installation. The simulations that were performed in order to emulate a real world scenario showed that the SoftFIRE infrastructure could significantly raise this number beyond initial expectations. This indicates that NFV platforms are highly beneficial as they can provide scalability and flexibility to AR applications.

3 5G MEC Caching

Latency is one of the most important metric as an indicators of the performance of a mobile network. In particular, high delay caused by long transmission distances in the Internet and congested backhaul links have a large impact on the quality of service of multimedia applications in mobile networks.

CityPassenger's [19] 5G MEC caching experiment proposed an extended architecture to support a software defined MEC platform, which targeted at reducing latency in a mobile network environment. The experiment implemented this solution in the mobile network running the Open5GCore [20] virtual mobile core provided by SoftFIRE in its FOKUS component testbed [21]. The system reduced the latency and increased the bandwidth to the subscriber by serving content from a local caching server. The traffic on the backhaul links was greatly reduced (which provides bandwidth for traffic of other non-edge users).

3.1 Architecture

5G MEC caching experiment proposed an extended architecture (Figure 4) to support a software-defined MEC platform. The experimenters implemented a solution called 5GMEC, as illustrated in Figure 4, which had an interface for collecting Multi-access Edge Computing (MEC) Radio Access Network (RAN) Information (MRI) between MEC applications and the mobile network.

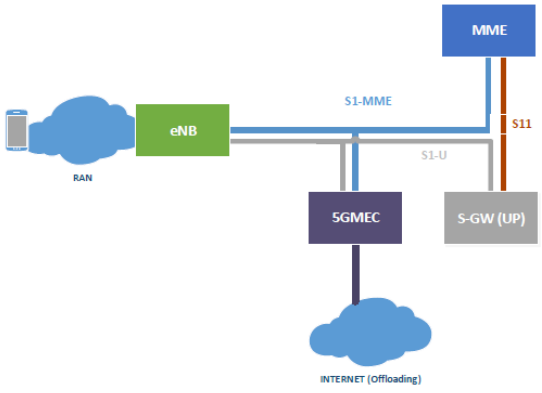

Figure 4. 5G MEC Caching Experiment Setup.

A specific software component (Stream Control Transmission Protocol (SCTP) monitor) intercepted the S1-MME messages between eNB and Mobility Management Entity (MME) and populated a Radio Network Information Service with User Equipment (UE) specific information. The minimal set of data required to uniquely identify the traffic of a subscriber is

International Mobile Subscriber Identity (IMSI), Tunnel endpoint identifier (TEID) (both for Uplink and Downlink Data), GPRS Tunneling Protocol (GTP) tunnel endpoints, IP addresses used by UE, MME and Service Gateway (S-GW).

The gathered information was then used to uniquely identify the traffic of a mobile subscriber, for the traffic between the eNB and the Serving Gateway (S-GW). Based on this information the traffic was redirected to a caching server that served cached content directly to the subscriber in the local area.

1.1 Key performance indicators

The following is the KPI measurement results of the 5G MEC experiment.

1.1.1 Packet loss rate

The first KPI measured system reliability, measured as the packet loss rate of the main components of the implemented solution. A packet lost rate of 0% was achieved by all components, thanks to the highly reliable architecture of traffic sniffing in MRI layer and due to the selection of reliable open-source components, such as open virtual switch (OVS) [22] and Squid [23]. Linux kernel IP forwarding is also reliable and can easily scale to large traffic volumes and data rates.

3.1.1. Latency imposed by system components

The average latency of each main system component was less than 1 ms. This result demonstrated that a service operator that deploys this approach would not experience a negative effect on non-HTTP traffic flows.

EXECUTION	MRI	MEC App Platform	MEC App
1	0.519	1.37	0.26
2	0.089	0.576	0.106
3	2.02	0.527	0.173
4	0.58	0.593	0.099
5	0.493	0.505	0.172
6	0.058	0.595	0.131
7	0.138	0.487	0.148
8	0.055	0.669	0.106
AVERAGE	0.494	0.66525	0.1493

Figure 5. Latency (ms) of 5GMEC system components.

1.1.2 Benefit of caching

This KPI quantified the decrease in the average Round Trip Time (RTT) and the reduction in backhaul data volume achieved by caching content at the 5G MEC caching server. The comparison of average RTT for cached results and for RTT without caching, as shown in Figure 5, demonstrates the substantial decrease of average RTT (94%) for well-known sites with static content.

RTT without caching

	site A	site B	site C	site D	site E
1	72	36	328	126	396
2	74	36	287	135	346
3	73	37	327	134	375
4	75	37	328	137	388
5	73	36	325	133	353
6	75	39	328	135	375
7	84	36	325	133	355
8	74	37	326	134	355
9	73	36	328	135	356
10	72	36	325	226	346
AVERAGE	74.5	36.6	322.7	142.8	364.5

RTT with caching

	site A	site B	site C	site D	site E
1	13	3	9	12	3
2	2	3	2	12	3
3	2	2	3	13	3
4	2	2	7	11	3
5	3	3	21	29	4
6	2	4	19	12	2
7	3	7	2	11	2
8	3	4	22	18	2
9	4	3	24	10	2
10	2	3	13	13	4
AVERAGE	4	3	12	14	3

Figure 6. Benefit of caching, as observed by the 5G MEC experiment.

3.2 Conclusion

The 5G MEC experiment demonstrated the benefits of deploying a MEC caching server as close as possible to the radio network. Round trip time for cached content can be decreased up to 99% for specific cases and the decrease on RTT was observed to be 94% on the average. A significant decrease on backhauling data usage was also achieved by this approach. With optimal configuration of Squid's object data store, all web pages can be cached (at least for HTTP use case) and provide 100% offloading from the mobile packet core. Furthermore, even with dynamic content, Squid proxy could offload and retrieve web content from local break-out, removing CPU and memory resource usage load from S-GW. Finally, there was no significant performance degradation on signalling and data traffic flows since the implemented architecture and software components add less than 1ms latency.

4 Breeze

The "*Balancing Requests for Backend Zones*" (BREEZE) experiment by Infotech [24] was motivated by the need to build a pseudo-private content delivery network (CDN) in order to provide a more robust mechanism for the Infotech API which is exposed to customers (and hence to malicious users). Infotech product is available to customers as a security-as-a-service (SaaS) system. It is designed for the wholesale hotel booking market, and hence has to be available 24/7 due to its nature. The system provides back-end services to medium/large enterprises that use the system to book hotel rooms.

The experiment used SDN features to balance requests on a number (N) of system servers distributed in multiple locations. The numerical results showed that the SDN layer was able to almost linearly scale on N servers. The benefits of this outcome are (*i*) a better overall performance that can scale up to N times, and (*ii*) a more robust distributed infrastructure with the servers located in different sites. Hence, the experiment deployment showed that it is possible to extensively improve the scalability of the Infotech SaaS solution in a "multi-tenant" environment.

4.1 Deployment on SoftFIRE

The BREEZE experiment implemented different scenarios on the SoftFIRE platform, on its Ericsson [25], FOKUS and 5GIC component testbeds. In the following figure, the configuration with three servers (two on FOKUS, and one on 5GIC) is shown.

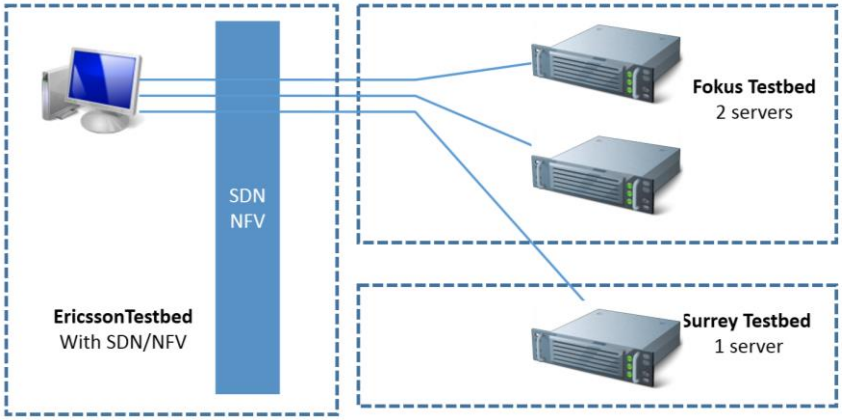

Figure 7. BREEZE experiment deployment with N=3 servers.

5 BotsOnFIRE

The experiment called "Mitigation of Botnets in Federated SDN/NFV Infrastructures for 5G", which was referred to as BostOnFIRE, was carried out by Universidad de Murcia [26] on the SoftFIRE platform. The aim was to develop a detection and mitigation system for *botnets*[1] in a federated virtualisation testbed, by combining NFV and SDN capabilities.

Botnets are regarded as one of the most powerful cyber threats affecting continuity and delivery of network services; not only in current networks, but also in future networks like those expected with the advent of 5G. Within this context, the BotsOnFIRE experiment was built by following a network self-protection approach, fully feasible for 5G networks, where effective detection and mitigation capabilities are achieved by tight coupling of SDN and NFV technologies in order to protect users and infrastructures from botnet cyber-attacks.

Two components were developed for (*i*) botnet detection phase and for (*ii*) attack mitigation phase. During the first phase, a VNF with IDS capabilities based on Snort [27] was deployed, which ran deep traffic flow analysis between possible bots (deployed as VNFs) and the command and control (C&C). Once a bot was identified, the second component that included a Honeynet [28] VNF reacted by isolating the bot and emulating its behaviour, i.e. sending fake responses to the C&C. In this way, the C&C server was unaware of the compromised bot and it was possible to stop advanced security threats, such as Distributed Denial of Service (DDOS) attacks. To let traffic flows reach the Snort VNF and to isolate bot VNFs, an SDN application was developed to enable mirroring and diversion of traffic during the detection and mitigation

[1] A botnet is a network of devices, the bots, controlled by a server, the C&C, which sends commands to them in order to execute malicious activities.

phases. Furthermore, Self-Organizing Network (SON) capabilities were supported by the experiment in order to detect and adapt to the C&Cs' behaviour changes.

To demonstrate the viability of the scenario in SoftFIRE testbed, the experiment evaluated performance indicators directly related to the time needed to reconfigure and provide security SDN and NFV applications.

5.1 Key performance indicators

The KPIs of the BostOnFIRE experiment were related with security issues:

5.1.1. Configuration time of Snort for a new detection rule

The measured time for the proper reconfiguration of the Snort VNF with a new detection rule was found to be less than 60 secs. The results achieved in this measurement for several detection rules were better than initial expectations, e.g. average configuration time was 1.57 seconds.

5.1.2. Configuration time of Honeynet for a new fake bot

The measured time for the proper reconfiguration of the Honeynet VNF with the execution of a new fake bot was found to be less than 60 secs. In this case, the resolution time was shorter than expected, with an average configuration time of 1.67 seconds.

5.1.3. Time to apply mirroring and diversion rules

These rules were applied to virtual switches. The measured time from starting the reconfiguration of mirroring and diversion rules to their application into the corresponding virtual switches to properly forward the suspicious/malicious traffic was found to be less than 20 seconds.

Table 1. BOTSONFIRE Performance times (in secs) when applying mirroring and diversion.

Network flow mirroring	Total Time	VNF Time	SDN Time
Method 1: Downloading ODL-Inventory	8.172387	0.754000	**7.418387**
Method 2: ODL-Inventory locally stored	1.752459	0.801000	**0.951459**
Network flow diversion	Total Time	VNF Time	SDN Time
Method 1: Downloading ODL-Inventory	8.006425	0.310000	**7.696425**
Method 2: ODL-Inventory locally stored	2.096475	0.264000	**1.832475**

6 Slicemon

The experiment called *Slice QoS Monitoring in a vCDN environment* (Slicemon) by Ubiwhere deployed and monitored a virtual Content Distributed Network (vCDN) function in the form of a VNF on the SoftFIRE platform. This vCDN does not employ a proxy methodology, but a methodology closer to a primitive version of a commercial CDN, where a client delivers a set of content to be fully managed by the service. The experiment also included a central service deployed outside the platform.

6.1 Key Performance Indicators

The following are the KPIs measured in the Slicemon experiment.

6.1.1. Downstream bandwidth

This KPI measured the increased bandwidth between the user and the edge caching element. During the experiments, a higher performance was obtained, compared to a scenario in which content was downloaded from a remote site.

6.1.2. Upstream bandwidth

This KPI measured the available bandwidth between the vCDN VNF and the content provider. Experimentations showed that an increase in the upstream bandwidth occurs only when the media content is requested by the clients. This increase in upstream bandwidth was clearly observed when the experimenter performed a higher sized image test (Reaches to 3.27 Mbps).

6.1.3. IOPS / Response time

Request Number	Client location	Response Time (ms)
1	Local	0,072
	Remote	0,164
2	Local	0,057
	Remote	0,154
3	Local	0,05
	Remote	0,154

Figure 8. Slicemon edge server response times.

This KPI measured the impact of read/write operations to the caching file system in terms of the time taken to load a media file from the central node or the edge server. Results showed a decrease in response time when an edge server is deployed near the client (0.08 ms for the client deployed near the edge (local) server (as VNF) in SoftFIRE's Fokus component testbed, and 1.17 ms for a remote client). Figure 8 shows the measurement results obtained in three experiments (requests for content).

6.1.4. Disk usage

This KPI measured the impact of writing the content cached by the vCDN VNFs to disk. When a scale-out operation was triggered, a new vCDN VNF instance for handling new incoming requests was dynamically deployed. This operation was performed using the Open Baton auto-scaling engine system. The experiment showed that an increase in disk usage occurs only when the client requests a new media file, so that the edge server is required to register on the central node and requests this new content. Using the auto-scaling mechanism, the available disk size was kept to be larger than 87%.

6.2 Conclusion

The experiment demonstrated that, running a local virtual content delivery client can help achieve lower response times on loading media content. It was possible to validate the orchestrator monitoring system and its interaction with its auto scaling system. The experiment also allowed the development and *know-how* acquisition on working with monitoring systems external to the provided orchestrator and how to interact with it.

7 MONET

Monitoring can provide the knowledge necessary to assure a network's QoS and security. Existing legacy security solutions (e.g., SIEM, IDS, IPS, FW) need to be adapted and correctly controlled since they are meant mostly for physical networks and do not allow fine-grained analysis sometimes needed by SDN/NFV networks (i.e. the basics of the future 5G networks). Thus, for these types of networks, the monitoring function needs to be highly adaptable and distributed to deal with their inherent dynamicity and heterogeneity, and offer mechanisms for finding the right balance between the cost of monitoring (w.r.t performance, scalability, latency) and the potential risks that could be encountered [30].

There are currently no commercial security network monitoring solutions for SDN and NFV mostly because there is yet no dedicated consistent standards and interfaces. Existing monitoring solutions, such as Ceilometer, Monasca (open source), and Applications Manager (commercial) only monitor performance, and target at the requirements of cloud operators for infrastructure metering (e.g. resource consumption for billing). However, these solutions are not for the tenants and their services/applications running in the virtualized environment since they do not allow for fine-grained service/application-level or guest system monitoring.

The experiment titled "*Monitoring Network Security in SDN/NFV environment*" (MoNet) by Montimage [31] had the objective to deploy two security monitoring solutions (MMT and Suricata) on the SoftFIRE platform and demonstrate how they can detect different security incidents both in the control and data planes of a virtualized network. Different attack scripts were implemented and launched, permitting to gather different key performance indicators (KPIs) assessing the detection and reaction capabilities of these security monitoring tools.

7.1 Technical challenge and solution

The main technical challenge that needed to be solved by the system was to have a security monitoring solution that is easily deployable and configurable by any NFV MANO solution. All the network traffic need to be mirrored to this tool to detect potential security incidents and react accordingly.

Montimage developed a flexible and remotely configurable monitoring tool called MMT that consists of distributed monitoring probes and a centralized monitoring operator. This prototype were conceived in the DOCTOR [32] and SENDATE [33] collaborative research projects. MMT uses DPI/DFI (Deep Packet and Flow Inspection) techniques to analyse the network traffic using rules based on temporal logic (sequence of events occurring in time). The probes need to be installed in the VMs (Virtual Machines) collocated with the NFV functions or

in a separate VM receiving aggregated traffic mirrored by the SDN controller. To assess the detection and reaction capabilities of MMT, a comparison was made to the Suricata IDS provided by the SoftFIRE platform.

7.2 Architecture

The architecture of the experiment is illustrated in Figure 9. In this figure, the following elements are shown:

- An SDN controller that configures a virtual switch to copy all the traffic from one switch (Ubuntu Server) to another switch (Ubuntu Probe).

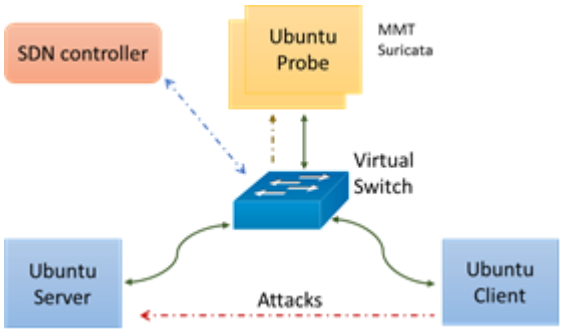

Figure 9. MoNet experiment architecture.

- The Ubuntu Client that simulates several attack scenarios targeting the Ubuntu Server.
- The Ubuntu Probe where MMT and Suricata are installed. It analyses the network traffic to detect the generated attacks and reacts accordingly.

7.3 Key performance indicators

The following KPIs for monitorability and security were defined.

KPIs	Description
Detection time (ms)	The delay between the attack and the notification of the alarm for different attacks.
False Positive rate	The accuracy of detecting different attacks. A legitimate traffic should not trigger an alarm.
False Negative rate	The accuracy of detecting different attacks. A malicious traffic should trigger at least one alarm.
Reconfiguration time	The capability of the monitoring system to adapt to changing situations.

Table 2. MoNet KPIs assessing the security monitoring performance.

It was possible to detect several security attacks using MMT and Suricata. Detection time was less that 10ms and the false positive and false negative rates were less than 5%. Furthermore, MMT was able to react after the detection of a security incident. This reaction is configurable and depends on the severity of the attack.

8 I-EVS

Italtel [34] performed the i-EVS experiment (*Enhanced Video Services for the Mobile Edge Computing in 5G environment*). i-EVS is a framework developed for edge-computing-based immersive video applications. The experiment was based on an application which was designed to supply intensive media transcoding, data caching, and content forwarding, which enables both high processing and massive broadband access at the network edge.

In particular, the experiment was aimed to demonstrate how to provide Immersive Video Services during a flash event, or in crowded locations using the i-EVS VNF. To achieve

Immersive Video Services, the network infrastructure has to overcome several challenges namely:

- the high density of user devices (more than tens of thousands per km^2),
- high data-rates to enable HD video streaming (at least 7 Mbps per user),
- low latency (in the range 10-50 ms) to guarantee the expected level of perceptual quality and user experience (QoS/QoE).

Moreover, HD video services require large IT resources, both in terms of compute and storage capabilities at the network's edge for reducing delay and preventing data streams from going through the network core when not necessary. To fulfil these requirements, i-EVS leveraged mobile edge computing capabilities with potential of NFV. Figure 10 shows the i-EVS experiment architecture.

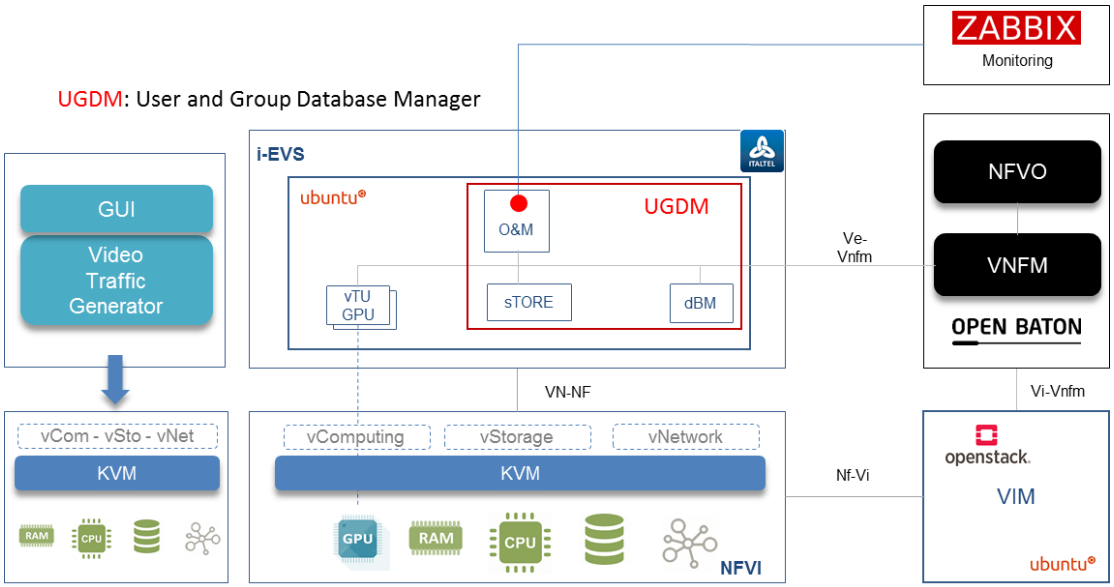

Figure 10. High-level i-EVS architecture on the SoftFIRE platform.

The experiment VNF was deployed in the 5GIC component testbed of SoftFIRE, and its tester, which emulates end-devices, was deployed in the FOKUS component testbed. Various real-time video streaming flows were generated and received concurrently.

8.1 Experiments

The i-EVS experiment was run in “CPU-only” mode. In experiments, a dedicated VM was designed for emulating the video traffic, and the following two tests were performed:

8.1.1. Video content upload

A user is connected to the i-EVS system. The user originates a high definition (1080p) video content through the i-EVS App. Then, using the i-EVS App, the originated video content is uploaded to the i-EVS local storage system. The content is uploaded to the i-EVS system and is made available to the other users in the same group.

8.1.2. Video content sharing performance

A user is connected to the i-EVS through Wi-Fi. The user originates a high definition (1080p) video content through the i-EVS App. At least one other user within the same i-EVS group and in the same site shares in real time the originated content, at the same resolution.

9 FLEXNET

The *FLEXible NETworks* (FLEXNET) experiment by Tglobal [46] aimed at providing a mechanism to apply per-user added value services accordingly to specific profiles from a BSS/OSS platform. In particular, the solution was designed to map specific profiles and services from the BSS/OSS architecture on top of the SDN/NFV SoftFIRE testbed in order to apply firewall-like rules on the platform so that the network is able to drop packets towards specific ports or services, such as telnet/ssh, ftp, etc. This aspect allows Internet service providers (ISP) to enable the filter only to those customers who do not request to keep such ports open and who are autonomously managing their own firewalls; e.g. end users do not configure a firewall but at the same time they expect that the ISP is defending them from malicious attacks, viruses, etc.

The experiment was deployed and conducted on the SoftFIRE testbed and in particular on the Ericsson component testbed.

9.1 Objective

The main objective of the FLEXNET experiment was to build a distributed firewall service that can block the access to certain customer devices. Instead of using commercial firewalls at the customer's premises the network operator can sell the firewall protection as a service using SDN/NFV technologies.

9.2 Experiment setup

The experiment set up followed a general architecture where a customer can subscribe to the firewall service. In particular, protection of the customer's devices from SSH access was performed. SSH attacks, as a test use case, was chosen for their simplicity to be shown as a test attack yet the setup was generic and could support other attack types.

For the FLEXNET basic experiment, the following architecture, which was made of three virtual machines and the SDN/NFV layer, was implemented.

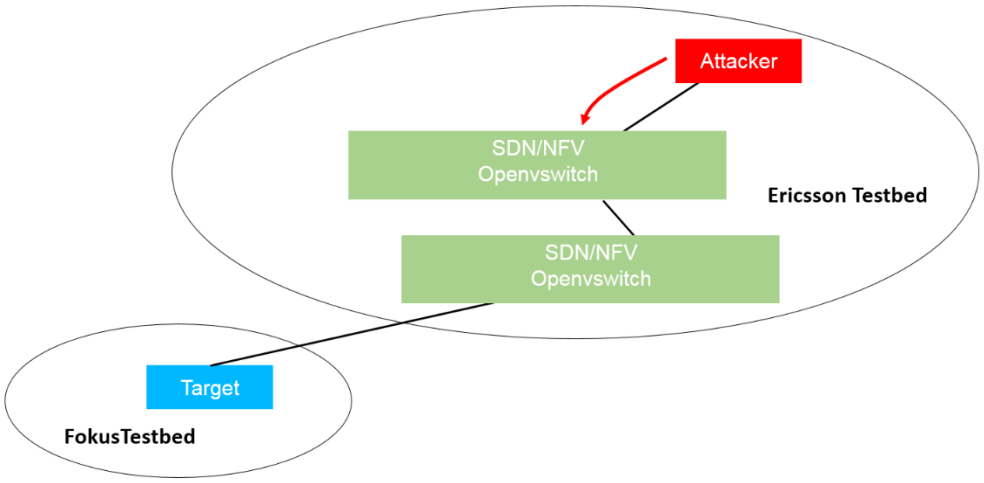

Figure 11. FLEXNET attack scenario.

In this first scenario, the goal is to show that the FLEXNET experiment is able to block traffic, even in a complex scenario where multiples SDN/NFV devices are used. There was one attacker and one target. The blocking function was implemented in the Ericsson SDN/NVF infrastructure, in the middle between the attacker and the target, using the available OpenDaylight controller that was used to add flow rules to open virtual switches. The attacker sent ssh connection attempts (tcp SYN packet on port 22) using hping3 towards the target. Experiments were run for 120 seconds, with 20 connection attempts per second. It was observed that 99% of packets could be effectively blocked, and the time for an SDN command to be operational was measured to be around 2 seconds.

The experiment also included a mobility scenario, as depicted in Figure 3. To simulate the handover two terminals were used, simulating the IP migration from one testbed (Ericsson) to another testbed (FOKUS). At the end of the handover procedure, the new terminal on the FOKUS testbed is protected.

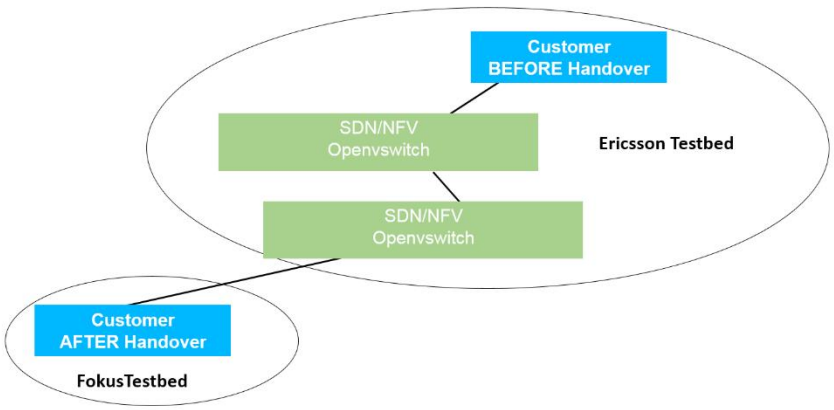

Figure 12. FLEXNET mobility scenario.

This mobility scenario is also related with cases in which the customer is to be protected from the SSH attack even when an IP change occurs. In such cases, protection rules are migrated

from the old IP to the new IP of customers. The migration time needed to reapply policies to the SDN controller located in the target testbed was measured to be around 8 seconds.

9.3 Conclusion

The FLEXNET experiment has shown that with the right architecture for SDN/NVF, it is valuable to build new added value services and monetise the infrastructure of the ISP/telecom operator. In particular, the experiment has shown that a distributed firewall approach can be implemented on an ISP network in order to protect customer devices.

10 Softblast

The SoftBLAST (*SoftFIRE BLockchAin SDN Technology*) experiment by Intrasoft Intl [37] focused on testing and verifying a blockchain network. The aim was to design a blockchain network that benefits from network programmability with NFV.

The experiment solution was built on an open-source framework called Hyperledger Fabric, which can be used/modified to deploy blockchain networks. The framework was extended to support scaling of the network and enable monitoring of key performance metrics. The solution employs Docker containers which facilitate the creation and management of lightweight nodes that participate in the blockchain network.

The experiment framework provided a modular architecture where nodes that participate in the network play a specific role to enable and execute smart contracts. Smart contracts in the experiment setup belong abstractly to an entity that is called a *chaincode*. The chaincode is a chain of information that can be used, accessed, and verified by the network participants. The Hyperledger Fabric creates a blockchain network that consists of *peer* nodes, *orderer* nodes, *client* nodes and Certificate Authorities (root and intermediate).

- *Peer nodes:* These nodes execute a chaincode, access ledger data, endorse transactions and interface with applications. They belong to (or are associated to) a Client node or Organization.
- *Orderer nodes:* (aka Organizations) These nodes ensure the consistency of the blockchain and deliver endorsed transactions to the peers of the network.
- *Client Nodes:* (or Organizations): These are the entities that act on behalf of end users.
- *Certificate Authorities:* The are the nodes in the blockchain network that represent organizations that run and execute the authorisation services.

By leveraging the use of Dockers as lightweight containers of nodes that constitute the network and by introducing an automated way to describe the participant nodes by using Docker Compose [38][39] (to describe, define and run the network), SoftBLAST employed a simplified way to scale out the blockchain network. The solution offers a monitoring service that captures and collects key performance metrics of the network during its operation.

10.1 Experiment setup

Figure 13 presents the SoftBLAST experiment setup in the SoftFIRE testbed. The figure demonstrates a blockchain network that was instantiated in the SoftFIRE testbed. Dockers create lightweight containers of multiple nodes that participate in the blockchain network. The network was instantiated in a VM of 8 Virtual CPUs at the FOKUS component testbed of SoftFIRE, and the network ran inside this VM. Scaling was achieved by leveraging Docker Compose to define and run multiple container nodes inside the blockchain network. Key performance metrics were collected and sent to the local machine where the experimenter could observe and monitor the performance of the network.

Figure 13. SOFTBLAST Experiment: Creating a blockchain network in SoftFIRE.

The following figure illustrates the architecture of the SoftBLAST experiment. The Blockchain network consists of many virtualised lightweight software Docker containers that constitute the nodes of the network. A blockhain is instantiated in this network as an application inside a container. All peer nodes firstly join the channel where the aforementioned blockchain have been instantiated. Organisation nodes that are pertinent to the blockchain install this blockchain. Transactions are being invoked by the nodes and validation of the information is signed by the participant nodes in the blockchain network. Collection of the metrics are presented in a dashboard on the local machine, as shown in Figure 13, where the experimenter is in the position to monitor the performance and manage the network. Blockchain network processes are captured and execution time is measured for each procedure and blockchain node.

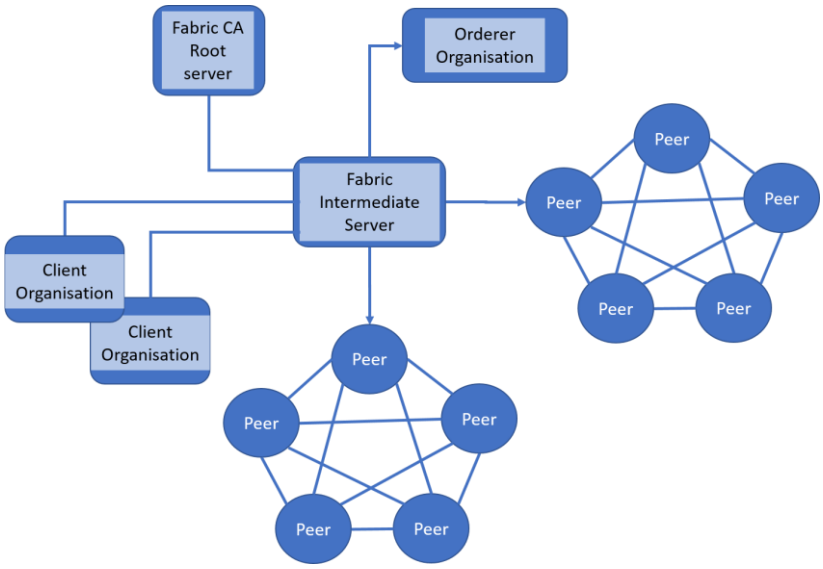

Figure 14. SOFTBLAST experiment architecture.

10.2 Performance measurements

Intrasoft developed tools that can collect measurements from Docker-based services by integrating free open source software, i.e. grafana, caddy, node exporter, prometheus, and cadvisor.

Key Performance Indicators (KPIs) of the solution were:

- Number of participant nodes in the blockchain network: The maximum number of nodes supported and used in the experimentation and proved to be successful. This was observed to be around 70 nodes.
- Data rate for signing the network packets for the blockchain network: 99.36 packets per sec (Packet size was 64 Bytes and average measured rate for all nodes in the blockchain network measured 6.21 KBytes ~ 99.36 packets per sec).
- Time duration for the network to converge and accept a transaction on the blockchain network. This is a set of KPIs which were measured for two different network configurations:
    - *Small Network Configuration:* 1 Orderer Organisation, 2 Client Organisations, 2 Peer nodes per Client (4 Peers in total), and 2 Certificate Authorities (CA) per Client (4 CAs in total), 2 nodes running the experiment and the blockchain; hence a total of 13 nodes.
    - *Large Network Configuration:* 1 Orderer Organisation, 5 Client Organisations, 5 Peer nodes per Client (25 Peers in total), and 5 Certificate Authorities (CA) per Client (25 CAs in total), 2 nodes running the experiment and the blockchain; hence a total of 58 nodes.

Table 3. KPI measurements in SoftBlast experiment.

Network configuration	Time for a peer node to join (seconds)	Time for a peer node to install the chaincode (seconds)	Time for a peer node instantiate the chaincode (seconds)	Time to query the chaincode (seconds)
Small	0.576	3.4	24.3	25.23
Large	0.976	4.3	61.02	64.7

11 Dune

The *Disconnecting Users from Network* (DUNE) experiment was conducted by Nemo srl [40]. The experiment aimed to solve a quite common business issue in the telecommunications market: customers who do not pay their monthly fees, which should be warned automatically, rather than being completely disconnected and removed from the operator's network. Such functionality would avoids unnecessary operations at the operator's side, especially for those instances when the customer simply miss a payment unintentionally.

The proposed solution is based on a simple but effective idea to simply add the customer to a blacklist and divert all the customer's connections to a "fake internet" that always displays a page (on the customer's web browser) notifying the use of the account suspension. When the user pays the outstanding balance, their account is unblocked and the customer can again surf the Internet. The mechanism operates in real time, i.e. it permits to add and delete the user from the blacklist in real time.

The DUNE experiment was set up on the FOKUS component testbed of SoftFIRE. This experiment included and implemented SDN mechanisms in order to divert the traffic for customers with bill issues. It was adapted to be compatible with the OpenSDNCore [41] SDN controller available at this testbed. The experiment scenario consisted of virtual machines:

- DNS server,
- DNS server for the fake Internet, diverting user traffic to a warning page,
- A web server with the "pay the bill" web page,
- A client that simulates a customer's terminal.

Table 4 shows the KPIs and the achieved results.

Table 4. KPIs for the DUNE experiment.

KPI	Measured value
Time to fully execute the policy after a user has been added to the block list (i.e., all packets blocked).	Less than 15 seconds
Time to fully execute the policy after a user has been deleted from the block list	Less than 20 seconds
Time to show and notify the blocking policy.	Less than 15 seconds

12 HighPep

With the increasing number of devices and hence increasing network traffic, it is required to bring part of the intelligence to the network edge so as to reduce diverse impact of high traffic load on the network backhaul. The *High Performance oneM2M Edge Processing* (High-PEP) experiment by Easy Global Market [42] aims at combining state-of-the-art technologies, i.e. NFV, Multi-Access Edge Computing and oneM2M, showcasing the these technologies to build IoT slices deployed to perform edge computing for IoT services. The experiment deploys a oneM2M [43] gateway as a VNF on the SoftFIRE infrastructure in order to realise the oneM2M edge capabilities over 5G.

Three KPIs were evaluated in the experiment:

- *Latency:* To achieve effective edge computing mechanisms, there is a need for a low-latency aggregation point to manage the various protocols, distribution of messages and analytics processing. The objective with this KPI is to evaluate baseline performance of oneM2M edge VNFs by measuring the latency between two oneM2M endpoints that were elastically deployed.
- *Scalability:* IoT data networks are extremely dynamic where the number of devices changes very quickly and with a large magnitude. Scalability support is thus essential for IoT networks, so that an acceptable level of quality of service can be provided to the customers of IoT services. The objective of this KPI is to measure the impact of the number of simultaneously connected devices.
- *Edge processing capability with security support:* For security reasons, messages from IoT devices are typically encrypted. To enable edge data processing, data decryption and encryption are hence mandatory for IoT services. The objective of this KPI is to evaluate evolution of baseline latency as well as scalability of the oneM2M network service while handling security processing.

As part of the experiment, two VNFs were deployed to execute the experiment: *(i)* a oneM2M server which plays the role of the oneM2M gateway (MN-CSE), and *(ii)* a message transceiver which sends messages to the gateway and receives the messages that the gateway forwards. The delay between the moment a message is sent and the moment when this forwarded message is received is measured to evaluate latency performance.

The main observation of the experiment was that edge processing components can be demanding for their virtualisation resources requirements, when the number of devices changes quickly and with large magnitudes. To minimize the impact of limited resources on one VNF component in such IoT scenarios, dynamic VNF scale-in/out mechanisms are mandatory to maintain a certain quality of service level. These observations are input for establishing a deployment reference guide in terms of performance for similar future virtual oneM2M implementations.

13 Inferno

The *IoT Interoperability over Federated SDN/NFV Domains* (INFERNO) experiment by INFOLYSiS [44] aimed to face IoT interoperability challenge within 5G networks through the agility brought by the combination of SDN/NFV, which allow network services to be automatically deployed and programmed. By exploiting the INFOLYSiS IoT mapping functions (VNFs), the main objective of this SDN/NFV-enabled IoT experiment was to examine the interoperability of the proposed IoT mapping VNFs over different SoftFIRE federated domains in order to assess an interoperable unified management of a large number of diverse smart objects that currently operate by utilizing a variety of different IoT protocols. The experiment was deployed on multiple SoftFIRE component testbeds (FOKUS and Ericsson).

As part of the experiment, IoT proxy VNFs were developed by INFOLYSiS (i.e. mapping functions (proxies)) of popular IoT data protocols (such as MQTT and CoAP) to generic data protocols (such as UDP). These VNFs were deployed and instantiated in order to provide an interoperable layer via the INFOLYSiS interoperable IoT virtual gateway (GW) solution. These mapping VNFs (i.e. proxies) utilised a set of TOSCA files on demand via the SoftFIRE Experiment Manager to be deployed on the relevant SoftFIRE domains/testbeds. A set of emulated IoT sensors with historical/offline datasets were also deployed for the needs of the experiment in order to provide monitoring data to their virtual IoT GWs/aggregators of the specific domain/testbed where they were deployed.

The key findings of the INFERNO experiment are as follows:

- The software-based IoT mapping functions developed by INFOLYSiS (which are also offered as standalone HW-based functions) can be successfully provided as a service without any limitation to the expected performance.
- The VNF formulation of the software-based IoT mapping functions can further facilitate the Software as a Service (SaaS) business model , since the INFERNO experiment showed that successful deployment and instantiation can be achieved via the SoftFIRE experimenter.
- The integration of SDN/Openflow rules in the virtualisation environment can further upgrade the provision of the INFOLYSiS interoperable services following the SaaS/PaaS model, considering that service isolation can be achieved with SDN. Thus, the lesson learnt is that the INFOLYSiS solution if it is provided in a SDN/NFV environment, can support multi-tenancy.
- Finally, thanks to INFERNO, INFOLYSiS successfully tested its product for IoT interoperability. Results showed that the solution maintains its functionality in a distributed mode as well, where each function/block of the solution is deployed at a different domain/testbed. Thus, by utilising two SDN/NFV-enabled testbeds, it was proven that the INFOLYSiS solution remains functional and capable of simultaneously serving two different IoT services/tenants, whilst reassuring their isolation and independent management and coordination.

14 NFVOB

The "*NFV Framework testing with OpenBaton*" (NFVOB) experiment by IS-wireless [46] aimed to deploy and validate the company's proprietary virtualization framework with ETSI NFV MANO compliant orchestrator Open Baton, which is the orchestrator used in SoftFIRE. The framework is essentially a VNF implementing functionalities that are common to large software projects. The framework provides the ability to use the APIs for common software libraries over the network and hence allow for rapid prototyping and building of more complex VNFs.

14.1 Virtualisation framework

One of the main domains of current interests of IS-Wireless is development of Software-Defined Radio Access Network (SD-RAN) for 4G and 5G. SD-RAN is an NFV-compliant base station which is deployable as a set of network functions, where the flexible RAN architecture allows using the components either as physical or virtual network functions. Hence it allows to provide not only a traditional physical network hardware-based solution, but also a cloud-based infrastructure, capable of running on a Mobile Edge Computing (MEC) server.

The virtualization framework – besides a RAN controller and its 3GPP stack implementation – is the main part of IS-Wireless' SD-RAN, as shown in Figure 15. It is a VNF that mediates between the operating system (VNF-MANO) and the protocol stack, and it can be treated as an abstraction layer between these two components. The deployed framework contributes to the network virtualization and NFV/5G ecosystem by:

- allowing the virtualization of network services,
- allowing to decouple RAN functionality from physical network infrastructure,
- providing the functionality to create network functions in RAN, hence the MANO management is possible up to the level of smallest eNB functionality.

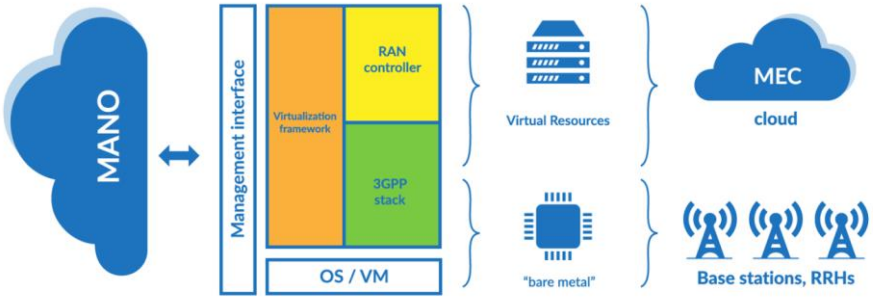

Figure 15. IS-Wireless' Software-Defined RAN for 4G and 5G.

The main development work was to enable a number of modules of the framework created by IS-Wireless to become compatible with OpenBaton, as used by SoftFIRE. Initial state involved several binaries running on POSIX-like system. As a results, five VNF modules were developed, listed as follows:

- *Dummy Server* – An example implementation instrumented for debugging and learning purposes; clients can send empty requests, which can be traced through the network, and will receive equally empty and traceable responses,
- *Logging Server* – A centralised logging facility; clients can send log messages to the Logging Server, which backs them up on some non-volatile memory, for example a file,
- *Settings Server* – A central location holding and propagating settings to clients; clients can request a list of settings known to them and receive a response with a list of values for those settings (strings, floats, or Boolean values),
- *Time Server* – The VNF that is responsible for time-keeping, using a user-defined time unit; a client can subscribe to those notifications and whenever the time unit changes, a message is pushed to all clients notifying the change. It was observed that it took on average 500 microseconds for time updates to reach clients,
- *Test Client* – An example client implementation connecting to all the above servers instrumented for debugging and learning purposes. This test client sent 8 messages, 1 request to the Dummy Server, another to the Settings Server, and 6 requests to the Logging Server. This process was observed to take 200 ms.

System measurements showed that the time it takes for clients to receive responses to their requests was on average 1ms. Furthermore, the Logging Server was able to process 8000 messages per second.

15 Concluding Remarks

Various experiments were executed on the federated virtualisation testbed provided by SoftFIRE. This white paper presents the experiments that deployed NFV and SDN solutions on the SoftFIRE platform during its 3rd Wave of Experiments.

SoftFIRE Middleware helped experimenters define different types of experiments, each provided by the platform as virtualised resources. Thanks to its Middleware, the project supported 13 experiments during its 3rd Wave. The Project's main achievement during this experimentation wave is its modular middleware, enabling services for various 5G applications. SMEs and academic organisations benefitted from the SoftFIRE platform, by testing their solutions in a virtualised test environment. The white paper presents all these applications and solutions, which are examples of near-future virtualised services that the industry and the technology market will benefit from.

Bibliography

[1] EU SoftFIRE project, https://www.softfire.eu/

[2] SoftFIRE Middleware, http://docs.softfire.eu/softfire-middleware/

[3] SoftFIRE SDN Manager, http://docs.softfire.eu/sdn-manager/

[4] SoftFIRE NFV Manager, http://docs.softfire.eu/nfv-manager/

[5] SoftFIRE Security Manager, http://docs.softfire.eu/security-manager/

[6] SoftFIRE Physical Device Manager, http://docs.softfire.eu/pd-manager/

[7] SoftFIRE Monitoring Manager, http://docs.softfire.eu/monitoring-manager/

[8] SoftFIRE Experiment Manager, http://docs.softfire.eu/experiment-manager/

[9] Third Wave of Experiments on the SoftFIRE platform, https://www.softfire.eu/open-calls/third-open-call/

[10] SoftFIRE Challenge, https://www.softfire.eu/open-calls/softfire-challenge/

[11] First SoftFIRE Hackathon, https://www.softfire.eu/events/first-hackathon/

[12] Second SoftFIRE Hackathon, https://www.softfire.eu/events/second-hackathon/

[13] SoftFIRE Innovation Hackathon, https://www.softfire.eu/events/innovation-hackathon/

[14] "Network Function Virtualisation: State-of-the-Art and Research Challenges", Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, Raouf Boutaba, *IEEE Communications Surveys & Tutorials*, vol 18, no 1, 236-262, September 2015,

[15] "Network Functions Virtualisation— Introductory White Paper", ETSI, 22 October 2012, retrieved 20 June 2013.

[16] "Software-Defined Networking: A Comprehensive Survey", Diego Kreutz, Fernando M. V. Ramos, Paulo Esteves Veríssimo, Christian Esteve Rothenberg, Siamak Azodolmolky, Steve Uhlig, *Proceedings of the IEEE*, vol 103, no 1, pp 14-76, January 2015,

[17] Intellia ICT, http://www.intellia.gr/

[18] 5G Innovation Centre, University of Surrey, http://www.surrey.ac.uk/5gic

[19] CityPassenger, https://www.citypassenger.com/

[20] Open5GCore, https://www.open5gcore.org/

[21] Fraunhofer FOKUS, FUSECO Playground, https://www.fokus.fraunhofer.de/go/en/fokus_testbeds/fuseco_playground

[22] Open Virtual Switch, http://openvswitch.org/

[23] Squid, http://www.squid-cache.org/

[24] InfoTech srl, https://www.infotechsrl.it/

[25] Ericssion RMED CloudLab, https://www.ericsson.com/portfolio/services-and-solutions/learning-services/education-centers/rmed

[26] Universidad de Murcia, http://www.um.es/

[27] Snort, https://www.snort.org/

[28] The Honetnet project, https://www.honeynet.org/about

[29] Ubiwhere, https://www.ubiwhere.com/en/

[30] Edgardo Montes de Oca, Wissam Mallouli. "Security aspects of SDMN" in "Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture", edited by Madhusanka Liyanage, Andrei Gurtov and Mika Yliantilla. ISBN-13: 978-1118900284 ISBN-10: 1118900286 Edition: 1st. Published on August 17, 2015
[31] Montimage, https://www.montimage.com/
[32] The Doctor project: http://doctor-project.org/index.htm
[33] The Sendate project: https://www.celticplus.eu/project-sendate
[34] Italtel, http://www.italtel.com/
[35] Tglobal srl, http://www.tglobal.it/
[36] OpenDaylight, https://www.opendaylight.org/
[37] Intrasoft, https://www.intrasoft-intl.com/
[38] Docker container, https://www.docker.com/what-container
[39] Docker Compose, https://docs.docker.com/compose/
[40] Nemo srl, http://www.nemo.it/
[41] OpenSDNcore, http://www.opensdncore.org/
[42] Easy Global Market, http://www.eglobalmark.com/
[43] oneM2M, http://www.onem2m.org/
[44] Infolysis, http://www.infolysis.gr/
[45] OpenFlow, https://www.opennetworking.org/sdn-resources/openflow
[46] IS-wireless, https://www.is-wireless.com/

List of Acronyms and Abbreviations

Acronym	Meaning
3GPP	Third Generation Partnership Project
4G	Fourth Generation Mobile Network
5G	Fifth Generation Mobile Network
5GIC	5G Innovation Centre
API	Application Programming Interface
AR	Augmented Reality
C&C	Command and Control
CDN	Content Delivery Network
CPU	Central Processing Unit
C-RAN	Cloud Radio Access Network
DFI	Deep Flow Inspection
DNS	Domain Name System
DPI	Deep Packet Inspection
eNB	evolved NodeB
ETSI	European Telecommunications Standards Institute
EU	European Union
FW	Firewall
GW	Gateway
GPRS	General Packet Radio Service
GTP	GPRS Tunnelling Protocol
HD	High Definition
HTTP	Hypertext Transfer Protocol
HW	Hardware
IDS	Intrusion Detection System
IOPS	Input/output Operations Per Second
IoT	Internet of Things
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IPS	Intrusion Prevention System

ISP	Internet Service Provider
KPI	Key Performance Indicator
MANO	Management and Orchestration
MB	Megabyte
MEC	Mobile Edge Computing
MME	Mobility Management Entity
MMT	Montimage Monitoring Tool
MRI	Multi-access Edge Computing (MEC) Radio Access Network (RAN) Information
NFV	Network Function Virtualisation
OVS	Open Virtual Switch
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
RAN	Radio Access Network
RTT	Round Trip Time
SaaS	Security as a Service
SCTP	Stream Control Transmission Protocol
SDN	Software Defined Network
SD-RAN	Software-Defined Radio Access Network
SGW	Serving Gateway
SIEM	Security Information and Event Management
SON	Self-Organised Network
SSH	Secure SHell
TEID	Tunnel Endpoint ID
TCP	Transport Control Protocol
TOSCA	Topology and Orchestration Specification for Cloud Applications
UDP	User Datagram Protocol
UE	User Equipment
vCDN	virtual Content Delivery Network
VM	Virtual Machine
VNF	Virtual Network Function

SoftFIRE

Disclaimer

SoftFIRE has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 687860.