



SECURING AGAINST INTRUDERS AND OTHER THREATS
THROUGH A NFV-ENABLED ENVIRONMENT

[H2020 - Grant Agreement No. 700199]

Deliverable D5.2

Final demonstration, roadmap and validation results

Editor G. Gardikis (SPH)

Contributors A. Kapodistria, G. Kolonias (SPH), O.E. Segou, C. Xilouris, (ORION), C. Fernandez, B. Gaston (i2Cat), E. Trouva, A. Kourtis (NCSR-D) L. Jacquin (HPELB), M. De Benedictis, A. Lioy (POLITO), D. Papadopoulos, A. Litke (INFIL), F. Ferreira, R. Preto, L. Conceição (Ubiwhere), G. Dimopoulos (TALAIA), A. A. Pastor (TID).

Version 1.0

Date March 30th, 2019

Distribution PUBLIC (PU)



POLITECNICO
DI TORINO



Telefonica



Agenzia per l'Italia Digitale
Presidenza del Consiglio dei Ministri



Executive Summary

This document describes all the activities which were carried out over the last six months of the project and focused on the system verification and evaluation, through the various testing, demonstration and pilot activities. It also analyses the results which were derived from these activities and presents some proposals for the further SHIELD evolution (roadmap).

Following the implementation activities, which preceded in WP3 and WP4, the integrated SHIELD framework was deployed in two separate instances (with slightly different features) at the i2CAT (Barcelona) and ORION (Athens) premises. The two testbeds featured, among others, an OpenStack-based compute cluster as the main PoP for hosting the vNSF instances, a secondary cluster for hosting the SHIELD components, and an SDN-based network backbone, which connects to the external networks via a security appliance.

The system testing activity was executed on the SHIELD instances in the two above-mentioned testbeds. Its aim has been to verify the proper functionality of the SHIELD platform on end-to-end basis, against the envisaged operation as well as the system requirements laid out in D2.2. The test plan consists of various discrete test cases, each of which has been appropriately defined using a test case template. The tests verified that the SHIELD platform successfully implements all the foreseen functionalities, namely service creation and validation, full service lifecycle handling, incident detection, classification and mitigation, infrastructure and service attestation. These capabilities enable the full realisation of the three key system use cases (UC1, UC2 and UC3), as identified in D2.2.

The tested/verified system was used to implement a set of demonstrations, each of which corresponded to a specific usage scenario. The scenarios included detection and automated mitigation of several incidents, such as: ransomware propagation, slowloris (slow DoS) attack, cryptomining abuse and vNSF/network integrity breach.

These demos were implemented and showcased at the end of Y2, and further enhanced by the end of the project with additional features, such as multi-user support, billing, integrated attestation and alternative remediation options.

In addition to the above mentioned demos, which were carried out using synthetic replayed traffic, SHIELD organised also pilot activities, which were conducted on operational networks (in SPH and NCSR premises), in order to assess the operation of the system under realistic network topologies and conditions.

Following the lab testing and the demos/pilots, SHIELD pursued an extensive open demonstration campaign, towards reaching out to experts outside the project team. This campaign included the following events:

- Pilot workshops with TID and SPH experts outside the project team (mainly from cybersecurity and marketing units) to assess the results of the pilot trials.
- A pilot workshop with CESICAT (Catalan Cybersecurity Agency), focusing on the usage of the SHIELD system by CERTs.
- The project's final workshop ("Modern Network-based Security: Softwarized Networking, Trusted Computing, and Artificial Intelligence for Cybersecurity") and public demo at ICISSP conference in Prague, Czech republic.

- The project's winter school, as part of the NeCS Cyber Security Winter School 2019, in Fai della Paganella near Trento, Italy.

All these events helped to gather valuable feedback from various stakeholders of the cybersecurity community with respect to advantages and shortcomings of the SHIELD framework, further activities towards its evolution, as well as exploitation roadmaps.

This feedback, along with internal evaluations and discussions within the project team, contributed to producing a technical roadmap for SHIELD, identifying specific features which can be implemented in the short term after the end of the project. Such features were identified at component level (NSFO, vNSFs, Dashboard, Attestation framework, DARE), while system-wide evolutions are also discussed.

Judging from the feedback received from all the external experts who were engaged in the final evaluation and assessment activities, it can be deduced that SHIELD produced quite impactful results, and realised a definitive step towards next-generation managed security services, particularly suited for software-based networks (including 5G infrastructures).

Table of Contents

1. INTRODUCTION.....	7
2. THE SHIELD PLATFORM AND USE CASES.....	8
2.1. Platform overview	8
2.2. Use case overview	9
3. THE SHIELD SYSTEM DEPLOYMENT	11
3.1. ORION testbed	11
3.1.1. Overall architecture	11
3.1.2. Key components	11
3.2. i2CAT testbed	12
3.2.1. Overall architecture	12
3.2.2. Key components	13
4. SHIELD SYSTEM TESTING	15
4.1. Test plan approach.....	15
4.2. Overview of test cases	16
4.3. Test case execution summary and requirements fulfillment	17
5. YEAR TWO DEMONSTRATIONS.....	18
5.1. A 0-day worm attack.....	18
5.1.1. Scenario motivation.....	18
5.1.2. Scenario description	18
5.1.3. Scenario setup.....	19
5.1.4. Scenario results.....	20
5.2. A hidden type of DDoS – Slowloris.....	20
5.2.1. Scenario motivation.....	20
5.2.2. Scenario description	21
5.2.3. Scenario setup.....	22
5.2.4. Scenario results.....	25
5.3. Cryptomining abuse	25
5.3.1. Scenario motivation.....	25
5.3.2. Scenario description	25
5.3.3. Scenario setup.....	26
5.3.4. Scenario results.....	31
5.4. Attestation of the vNSFs, their hosts and the network.....	31

5.4.1. Scenario motivation	31
5.4.2. Scenario description	32
5.4.3. Scenario setup.....	33
5.4.4. Scenario results.....	33
5.4.4.1. Infrastructure administrator (ISP).....	33
5.4.4.2. Security-as-a-Service client (vNSF user)	35
6. FINAL PROJECT DEMONSTRATIONS	37
6.1. A 0-day worm attack.....	37
6.1.1. Scenario motivation	37
6.1.2. Scenario description	37
6.1.3. Scenario setup.....	37
6.1.4. Scenario results.....	37
6.2. Cryptomining abuse	38
6.2.1. Scenario description	38
6.2.2. Scenario setup.....	38
6.2.3. Scenario results.....	38
6.3. Infrastructure and service attestation	39
6.3.1. Scenario motivation	39
6.3.2. Scenario description	39
6.3.3. Scenario setup.....	39
6.3.4. Scenario results.....	39
7. OPEN DEMONSTRATION EVENTS.....	41
7.1. Final workshop	41
7.1.1. Event details.....	41
7.1.2. Demo setup / ETSI ENI PoC	41
7.1.3. Assessment and feedback.....	43
7.2. Winter school	43
7.2.1. Event details.....	43
7.2.2. Hands-on session description	44
7.2.3. Assessment and feedback.....	44
8. SHIELD PILOTS	46
8.1. ISP pilot – UC1	46
8.1.1. Data collection and pilot setup	46
8.1.2. Presentation and stakeholders engagement	47
8.1.3. Assessment and feedback.....	48

- 8.2. Enterprise pilot – UC2 49
 - 8.2.1. Data collection and pilot setup 49
 - 8.2.2. Presentation and stakeholders engagement 50
 - 8.2.3. Assessment and feedback 50
- 8.3. CERT pilot – UC3..... 51
 - 8.3.1. Data collection and pilot setup 51
 - 8.3.2. Presentation and stakeholders engagement 52
 - 8.3.3. Assessment and feedback 52
 - 8.3.3.1. Technical features 52
 - 8.3.3.2. Business features..... 53
 - 8.3.3.3. CERT specific features 53
- 9. SHIELD TECHNOLOGY ROADMAP 54**
 - 9.1. Methodology 54
 - 9.2. vNSF ecosystem evolutions 55
 - 9.2.1. Lessons learned and gap analysis 55
 - 9.2.1.1. NFVO 55
 - 9.2.1.2. vNSF Store 56
 - 9.2.1.3. Attestation framework 57
 - 9.2.1.4. NSs and vNSFs..... 57
 - 9.2.2. Future activities and required resources..... 57
 - 9.2.2.1. NFVO 57
 - 9.2.2.2. Store 58
 - 9.2.2.3. Attestation framework evolutions..... 58
 - 9.2.2.4. NSs and vNSFs..... 58
 - 9.3. DARE evolutions 59
 - 9.3.1. Lessons learned and gap analysis 59
 - 9.3.2. Future activities and required resources..... 60
 - 9.4. System-wide evolutions..... 61
- 10. CONCLUSIONS 64**
- 11. ANNEX A: SHIELD TEST CASES AND RESULTS 65**
 - 11.1. Platform tests 65
 - 11.2. Service tests 108
- REFERENCES 120**
- LIST OF ACRONYMS..... 122**

1. INTRODUCTION

The SHIELD project aims at delivering a next-generation Security-as-a-Service solution, based on virtual security infrastructures, especially tailored for software-driven networks. The SHIELD technical framework is quite complex and its successful operation relies on end-to-end coordination between the various components, installed in the service provider and customers' premises.

In this context, towards assessing the value of the SHIELD solution as a whole, the aim of WP5, and especially Task 5.2, is to deliver an integrated end-to-end SHIELD Demonstrator platform and evaluate its effectiveness and efficiency against realistic usage scenarios.

This document describes all the activities which were carried out over the last six months of the project and focused on the system verification and evaluation, through the various testing, demonstration and pilot activities. It also analyses the results which were derived from these activities and presents some proposals for the further SHIELD evolution (roadmap).

D5.2 draws inputs from the following deliverables:

- **D2.2 “Updated requirements, KPIs, design and architecture” (work in progress)** is the final, updated version of D2.2, which was drafted concurrently with this document. D5.1 maintains its alignment with D2.2., which currently under preparation.
- **D5.1 “Integration results of SHIELD HW-SW modules”** presents the early integrated prototype of the SHIELD framework as well as the initial demo activities achieved at the end of the first year of the project.
- **D3.3 “vNSF framework ready for experiments”** is the final prototype of the vNSF framework and the trust framework.
- **D4.3 “Information-driven engine ready for experiments”** is the final prototype of the Data Analysis and Remediation Engine, and the SHIELD Dashboard.

This document is organized as follows:

- **Chapter 1** (present chapters) serves as a basic introduction to this document, its scope and structure;
- **Chapter 2** provides an overview of the SHIELD technical framework and the main use cases behind it;
- **Chapter 3** describes the technical details of the two main lab testbeds which are hosting the SHIELD framework;
- **Chapter 4** presents the approach and methodology for the system testing and overviews the results of the tests;
- **Chapter 5** presents the demonstration scenarios implemented at the end of the second year of the project (shown, among others, in the second year review)
- **Chapter 6** presents the demonstration scenarios implemented at the end of the project (to be shown, among others, in the final project review)
- **Chapter 7** describes the two main open demonstration events carried out during the final month of the project, i.e. the final workshop and the winter school.

Last but not least, **Annex A** includes the details and the results of the various test cases of SHIELD.

2. THE SHIELD PLATFORM AND USE CASES

2.1. Platform overview

The mission of SHIELD is to create a next-generation cybersecurity platform for advanced SecaaS offerings tailored for software networks, exploiting state-of-the-art techniques such as Big Data analytics and infrastructure/service attestation.

To that end, the SHIELD platform, whose functional architecture is shown in Figure 1, (described in detail in Deliverable D2.2 but also briefly overviewed herein) brings together the following components:

Network infrastructure - The network infrastructure provides a trusted environment for supporting the execution of virtual Network Security Functions (vNSFs), implementing a Network Functions Virtualisation Infrastructure (NFVI) environment, according to the ETSI NFV specifications.

Virtual Network Security Functions (vNSFs) - vNSFs are software instantiations of security appliances that are dynamically deployed into the network infrastructure. vNSFs i) gather information about the network traffic and generate events sent to the DARE and ii) prevent attacks or mitigate vulnerabilities and threats.

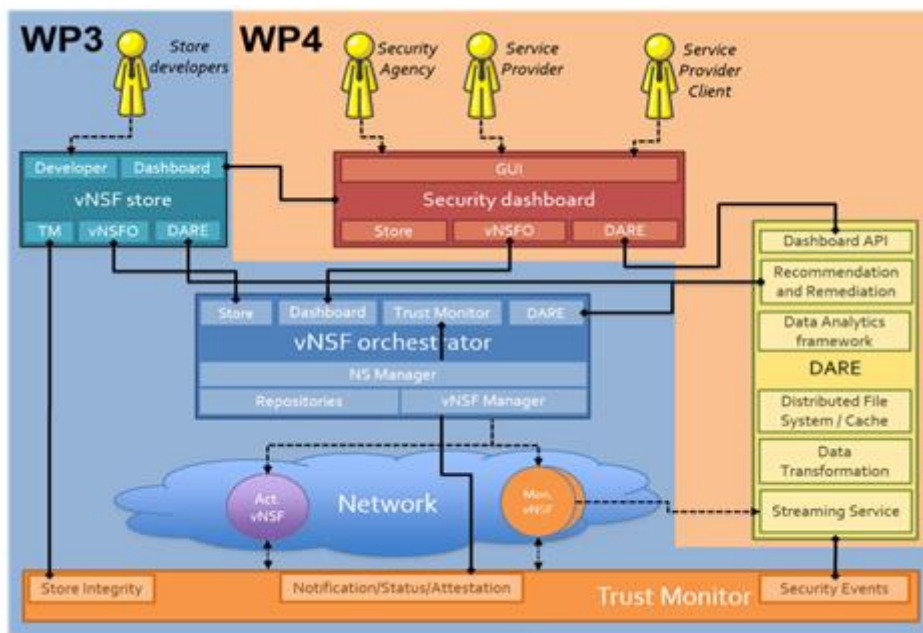


Figure 1. Functional architecture of the SHIELD platform

vNSF orchestrator (vNSFO) – it is responsible for managing the lifecycle of Network Services (NS), which are composed by one or more vNSFs. This allows to onboard packages for vNSFs and NSs, deploy (instantiate and place) NSs in specific points of presence within the network infrastructure, check the available and running services, execute actions on them, and so on.

vNSF store - it acts as a nexus between the vNSFO and third-party vNSF providers/developers, who can register and manage vNSFs in order to make them available through the SHIELD platform.

Trust Monitor – it is the component in charge of monitoring the trust of the SHIELD infrastructure. Integrity is checked periodically to detect compromised software and/or hardware and it is based on the Trusted Computing paradigm and its Remote Attestation workflow.

Data Analysis and Remediation Engine (DARE) – The DARE is an information-driven IDPS platform that stores and analyses heterogeneous network information, previously collected via vNSFs. It features cognitive and analytical components capable of predicting specific vulnerabilities and attacks. The processing and analysis of large amounts of data is carried out by using Big Data, data analytics and machine learning techniques. Furthermore, the DARE Remediation engine uses the analysis from the data analytics modules and is fed with alerts and contextual information to determine a mitigation plan for the existing threats.

Security dashboard and controller – Using the dashboard, operators have access to monitoring information showing an overview of the security status. The dashboard also allows operators as well as tenants to take actions and react to any detected vulnerability.

During Y1, all SHIELD components were hosted in VMs in the Athens testbed (private cloud computing infrastructure provided by ORION and hosted in NCSR), with the exception of the attestation components. The attack vectors utilised for the Y1 demonstration activities were also hosted in separate VMs and are described in Section 6. In Y2, specified DARE components will be migrated to the Barcelona testbed. After initial testing and validation of the remote attestation components, the related software will be integrated in the Athens and Barcelona (i2CAT) testbed in Y2. Some additional local testbeds by SHIELD partners will be used in Y2, for local development, functional and unit tests, such as the vNSF configuration and data collectors. One example is TID's Mouseworld Lab. This environment is responsible to generate synthetic network traffic (as close as possible to real traffic) tailored to Machine Learning needs in controlled environment (no production environments with privacy restrictions or lack of training labels). The Mouseworld Lab include a configurable generator of labelled network traffic datasets to be utilised during the training process of ML algorithms, VNFs for traffic capture and processing and visual dashboard. TID's plans for Y2 includes deploy a DARE engine based on Apache Spot to make functional and unitary test for ML algorithm in DARE and for vNSFs collector engine development.

2.2. Use case overview

Three dominant use cases have been identified for the SHIELD platform. These use cases reflect the deployment configurations that are supported by SHIELD and the various end-users. Both horizontal and vertical services can be envisioned as part of the three main SHIELD use cases:

Use Case 1: An ISP using SHIELD to secure their own infrastructure

In order to protect their own network infrastructure, ISPs have to deploy specific hardware which is very expensive since this hardware has to be updated and maintained by very specialised operators. The virtualization offered by SHIELD in this use case aims to dramatically reduce this cost by replacing specific hardware for vNSFs (virtual Network Security Functions), as well as providing a central interface (dashboard) to understand the gathered information and to act in the network. Apart from ISPs, the SHIELD platform can be internally used also by large enterprises, which operate NFV-capable corporate networks.

Examples of this use case might include DDoS detection and mitigation, blocking known malware Command and Control channels across the ISP network, tunneling detection to bypass roaming or data charges etc. This work is focused on providing network monitoring tools and cybersecurity for the ISP to on-board to their production environment.

Use Case 2: An ISP leveraging SHIELD to provide advanced SecaaS services to customers

SHIELD provides an ideal foundation for building enhanced SecaaS services, far beyond current offerings. Using this SecaaS paradigm, the complexity of the security analysis can be hidden from the client (either a company or an SME) who can be freed from the need to acquire, deploy, manage and upgrade specialised equipment.

In this UC, the ISP would be able to insert new security-oriented functionalities directly into the local network of the user, through its provided gateway or in the ISP network infrastructure.

Examples of this use case might include horizontal cybersecurity services (DDoS protection, Data exfiltration detection, Malware protection etc.) that can be offered as-a-Service from the ISP to their clients. This use case also includes vertical, tailor-made cybersecurity services to a variety of industries. Examples can range from IPR protection for streaming services (VPN detection, traffic management, etc), eGovernment (detection of comment bots on official government websites, phishing attacks etc), and other services.

Use Case 3: Contributing to national, European and global security

The DARE platform is able to export, upon request, threat models or data regarding acquired threat intelligence, to authorised third parties, for instance, public cybersecurity agencies. The secure SHIELD framework offers, in this manner, a way of sharing threat information with third-parties who wish to synchronise information and research on measures to be taken on recent attacks, suffered by others. Furthermore, using SHIELD, Cybersecurity agencies can establish agreements with the SP and deploy vNSF very fast and without cost in the infrastructure. Moreover the data is automatically accessible through the dashboard because the unification of the data treatment done in the data engine.

Examples of this use case might include notifying an authorised party of an identified anomaly that might not be classified as an attack indicator but could be suspect as a zero-day exploit, notify the authorities of a large scale or coordinated cybersecurity event and allow access to important threat information. This use case effectively showcases the scalability of SHIELD's proposed platform as well as the automated mitigation recommendations that can be attached to the threat data.

3. THE SHIELD SYSTEM DEPLOYMENT

During the integration, pilots and evaluation phase, the SHIELD system has been deployed into two testbeds, hosted by ORION (Athens) and i2CAT (Barcelona).

3.1. ORION testbed

3.1.1. Overall architecture

The ORION testbed is composed of a NFVI-PoP used for the vNSF's deployment, three ESXI servers that host the SHIELD project's core VMs and the network equipment that interconnects them. All of the key components are depicted in the figure below and described in the next section.

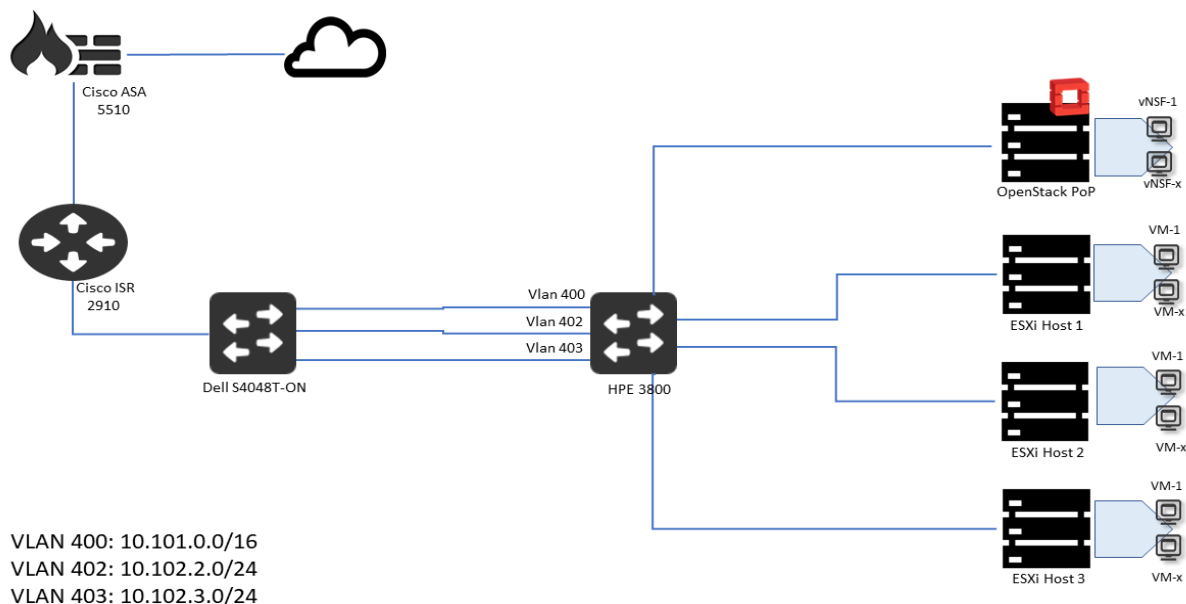


Figure 2. ORION testbed topology

3.1.2. Key components

The key components of ORION testbed could be divided into the following three main segments:

NFVI PoP:

The NFVI Point-of-Presence (PoP) in ORION's Athens site runs the Openstack Ocata distribution based on Centos 7.4.1708. The OpenStack controller and a compute node are situated on a single server, thus denoting this an "all-in-one" deployment. The PoP provides networking to the VNFs through Openstack's Neutron service. All the networking is therefore handled automatically by Openstack, provided that the required physical networks are established.

Additional networking infrastructure includes a Cisco 5500 Series Adaptive Security Appliance (integrating firewall, NAT and Intrusion Detection capabilities), a Cisco 2900 Series Integrated Services Router, and two switches, namely an SDN-enabled HPE Aruba 3800 and a Dell Switch. The NAT is configured either to be dynamic in order to allow all the hosts to reach internet or public addresses, or static NAT to allow also access to specific services from the inside networks to be reachable outside the firewall.

Additionally, all the partners are able to remotely connect to the infrastructure either via software SSL clients (Cisco AnyConnect, Openconnect etc). In order to increase security and also manage connectivity problems, each partner is provided with separate credentials and connection profile.

ESXi VMs cluster:

This cluster hosts the SHIELD's key components (apart from the vNSFs). The VMs are hosted in three baremetal servers running ESXi virtualization software. These VMs serve the core functionalities to the SHIELD platform. Following is the list of the software hosted by each of the VMs:

- Cloudera Manager (part of DARE platform)
- Cloudera Edge Node (part of DARE platform)
- Cloudera Worker (part of DARE platform)
- Dashboard
- HSPL Policy Engine
- Security Analytics Engine
- Trust Monitor
- Store
- NFVO (OSMr2)
- SDN Controller (ODL)

Networking components:

The network connectivity is supported by two switches (one HP3800 switch and one Dell S4048T-ON), one router and a Cisco ASA server. Both SHIELD PoP and ESXi Servers are connected to the Aruba switch while the Cisco ISR and the Cisco ASA server provide Internet access for the entire Infrastructure.

3.2. i2CAT testbed

3.2.1. Overall architecture

The i2CAT testbed is composed of a couple of computing clusters (hosting VMs for SHIELD logic and VNFs) and the network fabric that interconnects them. All of them are depicted in the architecture and described in detail in the key components' section.

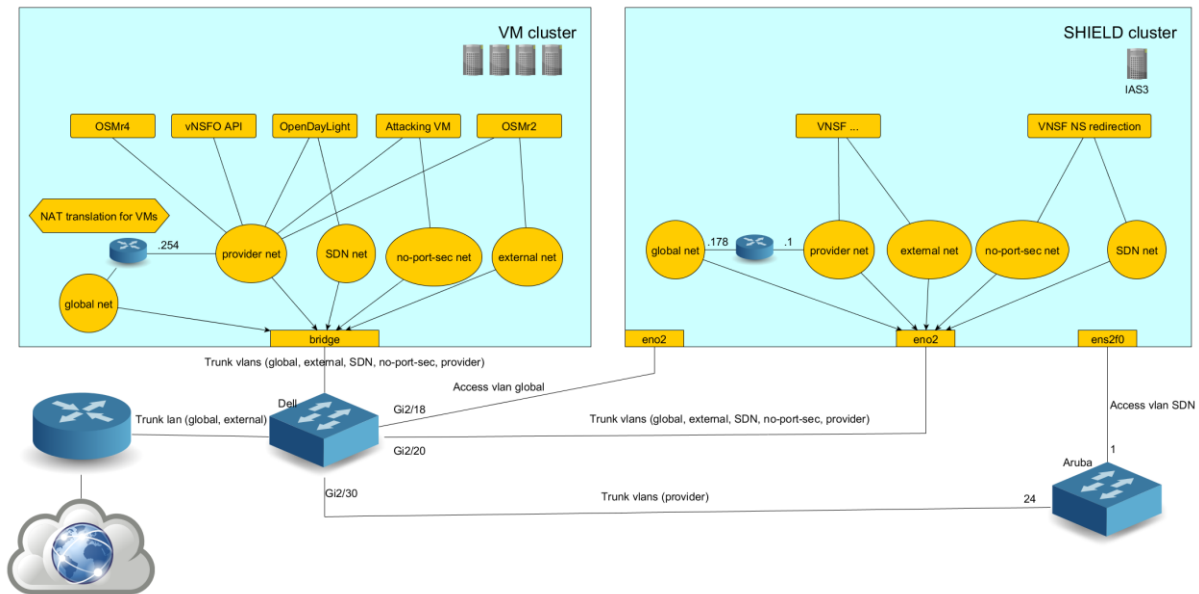


Figure 3. i2CAT testbed topology

3.2.2. Key components

The infrastructure is divided into the following basic segments:

- **NFV PoP:** cluster that hosts the OpenStack instance for the SHIELD VNFs. The SHIELD services run here and the traffic sent or received from them is isolated from the VMs that offer the SHIELD functionality (i.e., belonging to the SHIELD platform).
- **External VM cluster:** hosts VMs for specific SHIELD logic; either directly serving components belonging to the SHIELD platform or 3rd-party software components directly used or needed by them.
- **Switches and routers:** intermediate switching equipment providing VLANs to connect different services and connecting these between them and to the outer Internet. The different clusters (NFV PoP and external VM cluster) are connected by this network fabric using a VLAN per type of network and in different modes (trunk or access).

NFVI PoP

The OpenStack instance in Barcelona a production-enabled environment that runs the Ocata distribution on top of CentOS 7.5.1804. This instance was installed as an All-In-One (AiO) Openstack through Ansible scripts.

Before starting the installation, the partitions for root ("/"), swap, volumes ("/cinder") and other OpenStack data ("/openstack") must be defined. Then, its mounting point must be set.

Then, the configuration for OpenStack Ansible starts. Features like VLAN shall be enabled, bridges must be created (manually work best), the definition of the Swift and Nova loopback disks shall take place, as well as other data such as the bridge to link the virtual interfaces of the VMs deployed by OpenStack.

Finally, run the OpenStack bootstrap script, verify that the environment is according to the configuration defined before running such script and run the whole setup via its specific playbook.

After the deployment, the OpenStack server must be running a total of 23 LXC containers, each with its own specific service and have enabled different bridges for the management, storage, connectivity (VLAN, VxLAN and/or others, as defined in the previous configuration step). This is linked to the HP Aruba (3800) switch and to another switch, which is connected to a router for Internet access. Both of them are also connected to the internal servers where the SHIELD-functionality VMs are hosted. Partners accessing the infrastructure are to given access via their OpenVPN client and log-in with their given credentials.

External VM cluster

The external VM cluster is a group of other OpenStack instances. It is comprised of four different bare-metal servers which are hosting, among others, the VMs that serve functionality for the SHIELD platform: the SDN controller (OpenDayLight), different versions of the NFVO (OSMr2 and OSMr4), the vNSFO API and other ancillary VMs such as a VM to perform attacks from outside the infrastructure that supports the NFV environment.

This is connected to the core network of the i2CAT organisation and is also connected by L2 to the SHIELD PoP. This connectivity is achieved through the usage of trunk and access VLANs, one per network.

Switches and routers

The network fabric is made up of two switches and one router. The HPE Aruba is one of those two switches and it connects the SDN network of the NFV PoP with the other internal switch. On the other hand, the other internal switch connects the external VM cluster (and all other SHIELD-specific networks, such as “provider” for vNSF management, “external” for public IP assignment, etc) to the Internet.

4. SHIELD SYSTEM TESTING

4.1. Test plan approach

Following the components' unit tests which were carried out in the frame of WP3 and WP4, the aim of the system testing activity, as part of WP5, is to verify the proper functionality of the SHIELD platform on end-to-end basis, against the envisaged operation as well as the system requirements laid out in D2.2.

The test plan consists of various discrete test cases, each of which has been appropriately defined using a test case template (see table below).

The set of test cases has been mainly derived from the D2.2 requirements. Instead of having a separate test case for each requirement, the approach has rather been to group requirements of similar scope to a smaller set of integrated test cases.

Key Performance Targets (KPIs) – such as response time, throughput, data volume etc. - related to non functional requirements, have been mostly included as metrics in functional tests.

Table 1 below presents the template for the SHIELD test cases and includes an explanation of each field.

Table 1. SHIELD Test case template and explanation

Test Case ID	<i>[Unique ID per test case]</i>		
Description	<i>[Brief description of the scope of the test case]</i>		
Executed by	<i>[Partner responsible for executing]</i>	Date	<i>[Date on which the TC was executed]</i>
Purpose	<i>[Justification for the TC, i.e. high-level platform operation to be verified]</i>		
Associated Requirements	<i>[Requirements fulfilled via this TC, from D2.2 (only reference IDs)]</i>		
Components involved	<i>[SHIELD platform components involved in this TC]</i>		
Tools	<i>[Auxiliary hardware/software tools used in this TC, such as traffic generators, protocol analysers etc]</i>		
Metrics	<i>[KPIs to be measured, such as delay, throughput etc.]</i>		
Pre-test conditions	<i>[All the conditions which need to hold true before the TC sequence starts; assumptions for the tests]</i>		

Test Sequence	Step	Type	Description	Result
	1	Stimulus	<i>[Action/step taken, as many as necessary, to invoke an operation]</i>	
	2	Check	<i>[Observation, measurement etc.]</i>	
Evidence	<i>[Visualisation of results, e.g. screenshots, responses etc.]</i>			
Verdict	<i>[Success/Partial success/Fail]</i>			
Comments	<i>[If any]</i>			

4.2. Overview of test cases

Annex A details all the test cases of SHIELD and presents the details about their implementation as well as the results. As seen, the test cases have been grouped into two main categories:

- Platform tests refer to the end-to-end functionality of the SHIELD platform. They correspond to the “Platform” requirements, as listed in D2.2.
- Service tests refer to the security service capabilities, as exposed by the vNSFs. They correspond to the “Service” requirements, as listed in D2.2.

The platform test cases cover all mandatory operations of the platform, namely:

- Service onboarding and image validation
- Service lifecycle handling
- Data ingestion
- Security incident detection, classification and visualisation
- Multi-user support and billing
- Decision support and automatic mitigation
- Service attestation
- Infrastructure attestation

The ethics and regulatory requirements (ERC), which were introduced at a later stage in the project and did not correspond to the initial scope of work, were not mapped into specific test cases. Some of them were already covered by the existing TCs, while others constitute recommendations to be taken into account in a future operational SHIELD deployment – and mostly apply to the organisation(s) operating the SHIELD platform, rather than the platform itself.

4.3. Test case execution summary and requirements fulfillment

Annex A details the outcome and results of the platform- and service-focused test cases. It can be seen that the SHIELD platform successfully implements all the foreseen functionalities, namely service creation and validation, full service lifecycle handling, incident detection, classification and mitigation, infrastructure and service attestation. These capabilities enable the full realisation of the three key system use cases (UC1, UC2 and UC3), as identified in D2.2.

The following platform requirements were only partially fulfilled, since their full compliance was considered to require excessive effort and would deviate from the main targets of the project:

- *PF16 (History reports) and PF21 (Operation Traceability)*: the Dashboard already provides access to event logs, especially with respect to attestation and DARE incidents. Further work is required to enable full logging of all user and system actions.
- *PF18 (Service composition)*: it is currently possible to compose a complex vNSF service, yet the service chaining process requires several manual interventions. Fully automated chaining and deployment of a virtual network function service is currently an open issue in the NFV community and out of the scope of SHIELD.
- *PF22 (Management communication security)*: User access to the Dashboard, vNSFO and virtual infrastructure management is already encrypted and performed over TLS. Further work is required to secure all the communication channels among the management components.
- *NF03 (Scalability) and NF04 (Data volume)*: There have been no specific tests targeted to the scalability of the system. However, the SHIELD platform was observed to scale in a satisfactory manner, especially during the two pilot activities (see Chap.8), which involved a considerable amount of traffic, compared to the lab trials. In addition, it must be noted that SHIELD builds on well-established foundation technologies (OpenStack, Hadoop, Kafka, Hive, Spark), whose scalability has been extensively proven over the past decade.
- *NF08 (Deployment and support simplicity)*: Significant effort has been devoted to streamline the platform installation procedure, especially during its final release. Yet, it is still meant to be installed and configured by an ICT professional.
- *NF09 (vNSF hardening)*: Several security measures have been applied during vNSF implementation, such as e.g. secure management access and configuration of a minimum OS distribution, without unnecessary modules which would increase the attack surface. However, further hardening and extensive security auditing of vNSFs are required prior to their deployment in an operational environment.

All other platform requirements have been fully complied with.

In addition to the test cases, which verified step-by-step the proper system functionality, the behaviour of SHIELD as an integrated system was also demonstrated and assessed in the demo setups (which correspond to realistic usage scenarios), as described in Chap. 5 and 6, as well as the pilot deployments in operational networks, as described in Chap. 8.

5. YEAR TWO DEMONSTRATIONS

5.1. A 0-day worm attack

5.1.1. Scenario motivation

Wannacry was a nearly 0-day attack using ransomware (i.e. a malware that kidnaps / encrypts files and asks for a ransom) in May 2017. A zero day exploit is a cyber attack that occurs on the same day a weakness is discovered, at that point, it is exploited before a fix becomes available. Wannacry used a known vulnerability to infect around 200.000 computers across 150 countries with an expected losses of four billion dollars. The attack was finally stopped when a researcher accidentally discovered a hidden pattern. The overall loses were estimated to be around 4 billion dollars.

Currently, most of the Intrusion Detection Systems (IDS) are rule/signature based systems. This means that they can only stop an attack when there is a specific pattern to look for in the network traffic. This means that these kind of systems can not detect and prevent 0-day attacks.

In SHIELD, we propose an anomaly detection algorithm, using unsupervised learning, able to detect wannacry as an anomaly in around one minute. This would had been enough for a cybersecurity expert to stop wannacry before it propagated when wannacry was launched. Moreover, we have also added a supervised algorithm that works with the detected anomalies in order to classify them and hence, identify Wannacry as the correct attack.

5.1.2. Scenario description

Due to the high risk and high impact involved with the reproduction of an attack like wannacry in a real demo environment (with physical machines in the I2CAT testbed), the consortium decided to approach this problem by getting real wannacry data produced outside of the environment. This data has been modified to be congruent with the demo scenario (range of ips, etc.) and hence, it can be injected it in the network.

Using this approach, security of the infrastructure is assured (no real wannacry application running), but at the same time, the traffic is as real as the one produced by the infection. This means that from the data collection to the remediation, all the steps are as real as they should have been with a real infection.

The attack traffic corresponds to 6 machines working in the same LAN. After some normal activity, one of them starts sending Wannacry propagation traffic to the other machines. Due to this propagation three of the other machines are also infected and start their own process of propagation.

The following picture shows the developed components of the SHIELD framework that have been used in this demonstration:

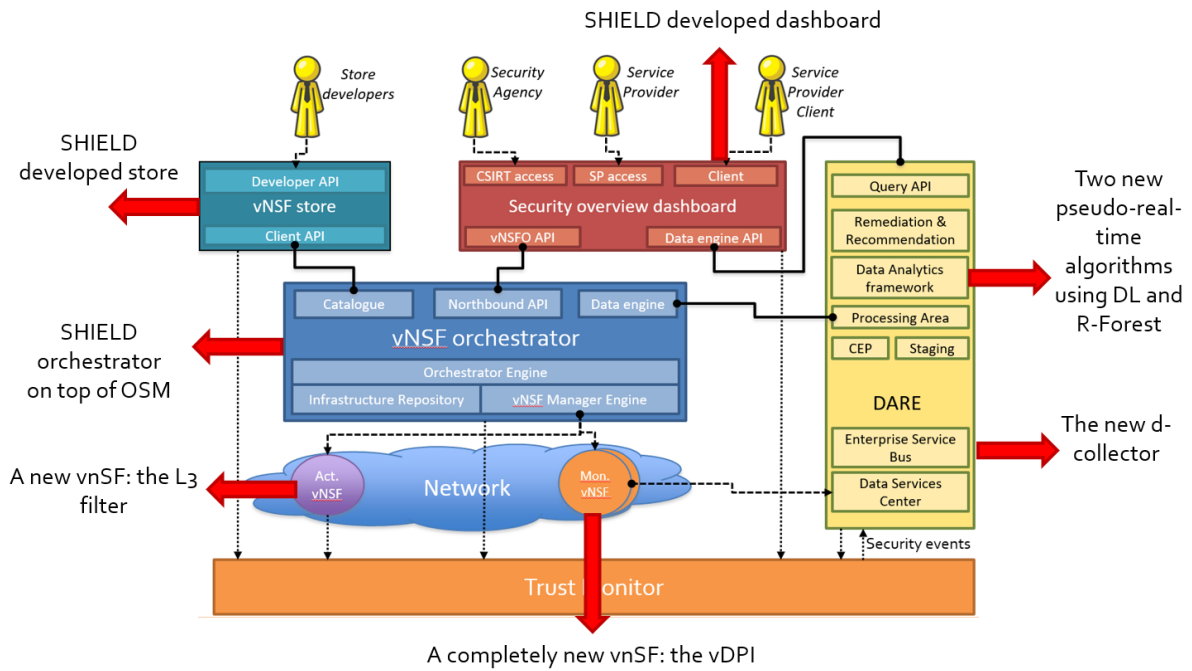


Figure 4. SHIELD components involved in the worm scenario

5.1.3. Scenario setup

The scenario is shown in the following picture and it is defined as follows:

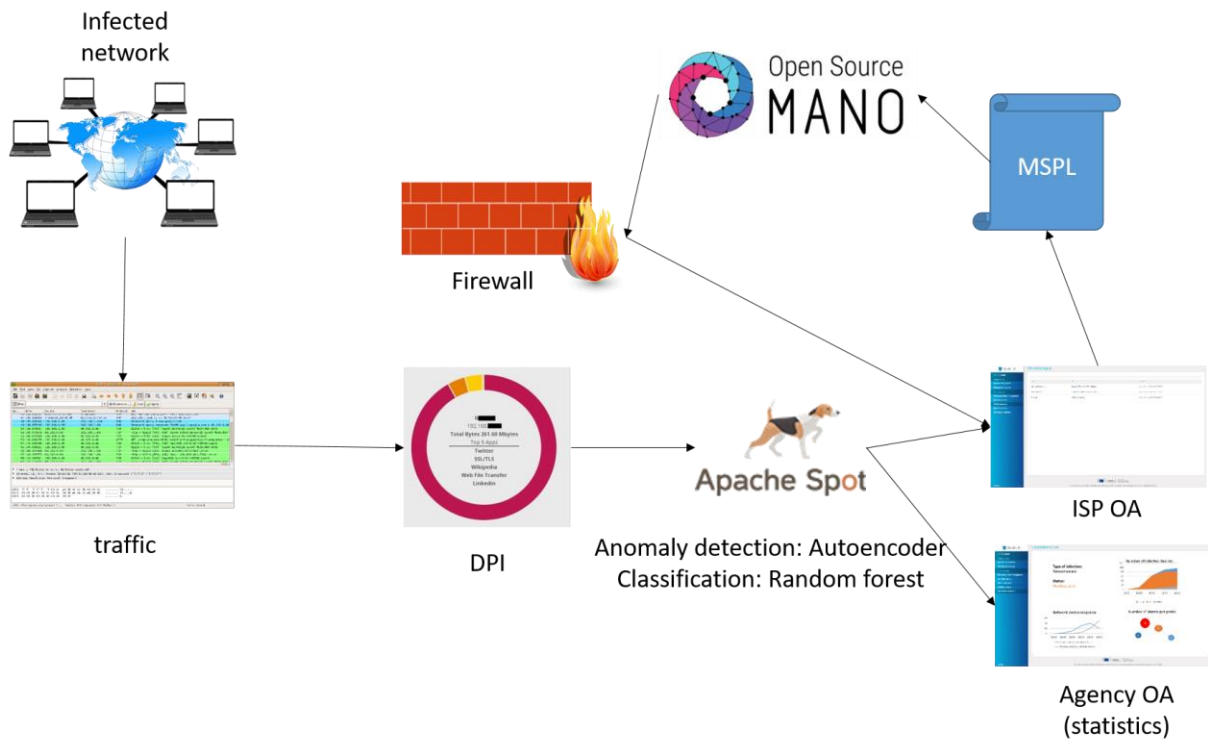


Figure 5. SHIELD system setup for the worm scenario

The scenario comprises the following steps:

1. A network infection with wannacry is replicated from one of these computers to the others (a total of 4 computers will be infected at the end of the demonstration)
2. Traffic is analysed by the vDPI and sent it to the DARE
3. The Deep Learning Autoencoder (unsupervised learning) detect the anomalous threats
4. The Classification algorithm (supervised) classifies them into the Wannacry attack
5. A recipe is created blocking the involved IPs
6. The statistics of the attack are shown in the Agency Operation Analytics dashboard
7. The recipe is shown in the ISP Operation Analytics dashboard and it is applied
8. The recipe (MSPL) is converted in configuration for the L3 filter vNSF.
9. The L3 filter blocks the IPs
10. The traffic of the infected IPs is not longer reaching the network.

5.1.4. Scenario results

The resulting outputs of the scenario are:

1. A classified set of detected IPs infected by Wannacry
2. A recipe for blocking these IPs
3. Statistics regarding the infected IPs and their propagation characteristics
4. A firewall configuration that can stop the attack

5.2. A hidden type of DDoS – Slowloris

5.2.1. Scenario motivation

The security attack that has been selected to be demonstrated at NCSR-ORION infrastructure is Slowloris. Slowloris is a type of denial of service attack that operates at Layer 7 (application layer). It exploits a design approach of many web servers, allowing a single machine to take down another machine's vulnerable web server with minimal bandwidth.

In particular, during a Slowloris attack the attacker opens as many connections to the target web server as possible. He/she holds the connections open as long as possible by sending a partial request and adding to it periodically (to keep the connection alive) but never completing it. Usually, servers use threads to handle each concurrent connection and have a limit on the total number of threads. Under a Slowloris attack, the pool of threads available to the server is consumed by the attacker. Thus, the affected server will deny connection attempts from legitimate users until at least some of the held connections are released.

The motivation behind the selection of Slowloris attack in the SHIELD demonstrations is two-fold: First, the selected scenario requires an end-to-end integrated run of many of the developed SHIELD components, demonstrating the developments within WP3 and WP4.

5.2.2. Scenario description

The scenario involves interactions between the following components:

- Two of the developed NSs and vNSFs and in particular the virtualised DPI and the L3 filter services
- The SHIELD vNSF store
- The SHIELD vNSF orchestrator
- The SHIELD Dashboard
- The DARE, including the distributed parallelizable ingestion of data (d-collector¹) and a new threat classification algorithm based on supervised learning (Random Forest²). The algorithmic implementation as well as the functionalities of these modules have been described thoroughly in D4.3 (sections 3.1, 3.2.1 respectively).

The following figure shows in detail the SHIELD components that are involved to this demonstration:

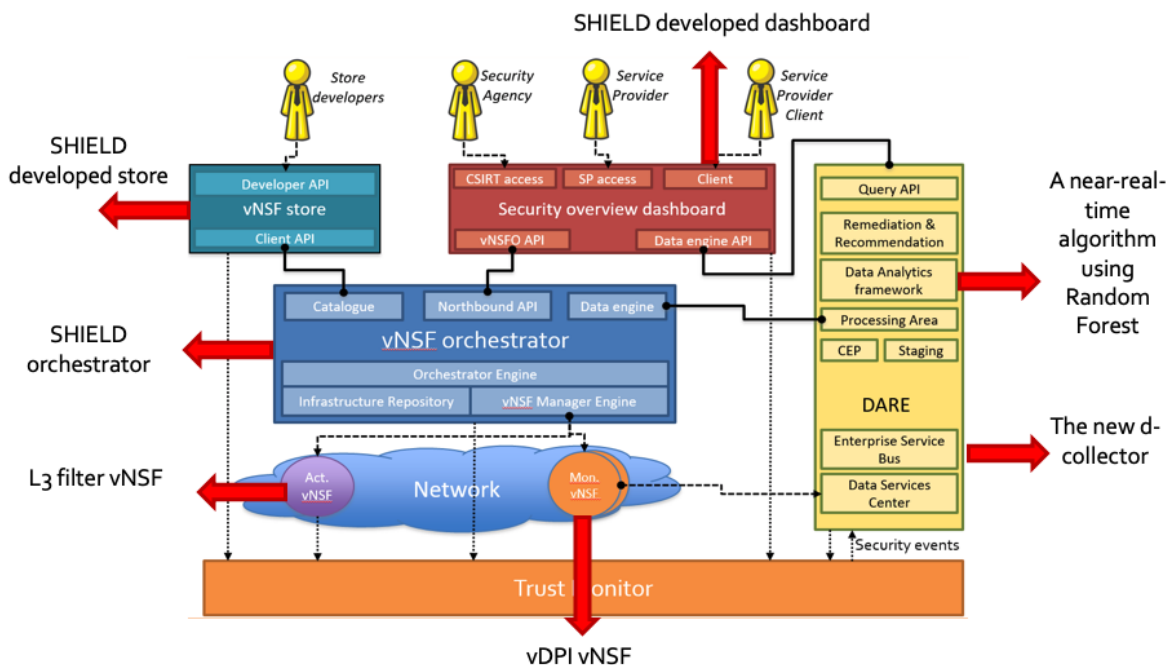


Figure 6: SHIELD components involved in the Slowloris scenario

The second motivation behind the selection is relevant to the significance of DDoS attacks today. DDoS attacks are amongst the most common and most serious cyber-attacks seen in today's networks. DDoS is an attack that disrupts operations and causes loss of reputation, productivity and/or revenue. This particular attack has been used in 2009 Iranian presidential election against sites run by the Iranian government and by the spam network River City Media to force Gmail servers to send thousands of messages en-bulk, by opening thousands of connections to the Gmail API. In addition, Slowloris is applicable against well-known web servers. Some versions of Apache (1.x and 2.x), Flask, dhttpd and other commercial servers are vulnerable to a Slowloris attack.

¹ <https://github.com/shield-h2020/vnsfs-collectors/tree/master/distributed>

² https://github.com/shield-h2020/dare/tree/master/classifier_ml

For the affected servers, no reliable configuration exists to prevent Slowloris attack. However, there are some measures that someone can take to reduce the impact of the attack. Common measures towards this direction are to increase the maximum number of clients the server will allow, limit the number of connections a single IP address is allowed to make, impose restrictions on the minimum transfer speed a connection is allowed to have, restrict the length of time a client is allowed to stay connected, set up reverse proxies, firewalls, load balancers or content switches.

SHIELD, based on the developed ML solution using supervised learning and the Big Data processing architecture and also on the NFV technology for monitoring traffic and mitigating the attacks, provides a solution on how attacks such as Slowloris could be identified mitigated in the future.

5.2.3. Scenario setup

The setup we used for this demonstration has 5 main components:

- a) The attacker's server, which is the machine that performs the Slowloris attack.
- b) The target server, which is the machine to which the attack is targeted to.
- c) The traffic generator server, which is a machine that generates traffic, simulating normal traffic that in a real scenario would be sent to the target server.
- d) The NFV Infrastructure Point of Presence (NFVI-PoP), which is the infrastructure in which we deploy our network functions, the vDPI and the L3 filter.
- e) The Cloudera Big Data cluster (Cloudera Distribution for Hadoop - CDH) that has been used for the deployment of SHIELD's DARE/Cognitive DA module which incorporates the machine-learning algorithms that have been developed as part of the project.

Figure 7 describes the Infrastructure setup that has been used for the demonstration in detail.

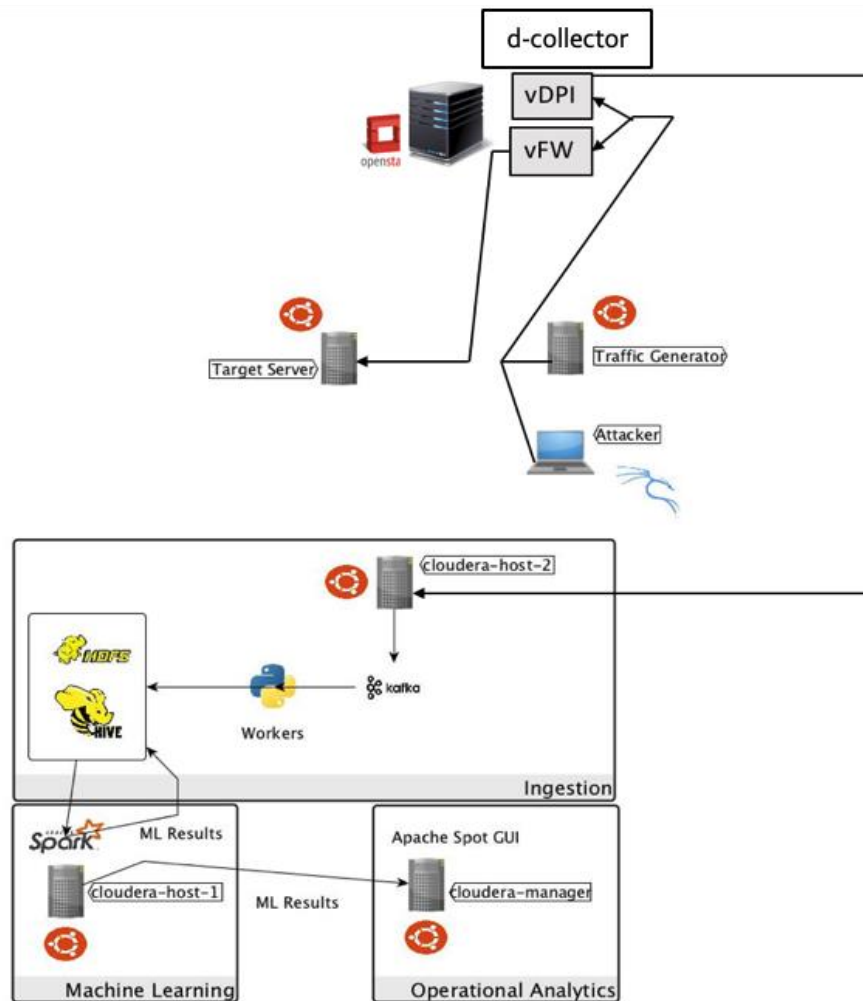


Figure 7: Slowloris scenario setup

The installation of the Cognitive DA module, as part of the DARE was made on a Cloudera Cluster version 5.11.1y. A minimum of three Cloudera nodes is required in order to deploy the Setup, Ingestion, Machine Learning and Visualisation components of the module. Three hosts were set up running Ubuntu Trusty 14.04 and the prerequisite components (HDFS, Hive, Impala, Kafka, Spark, Yarn, and Zookeeper). Our setup with the 3 Virtual Machines (VMs) was the following:

- Cloudera-host-1.shield.com: Configured as the Machine Learning (ML) Node, includes the Yarn Node Manager;
- Cloudera-host-2.shield.com: Configured as the Setup and Ingest Node, also known as the Edge Gateway (GW) node, includes Kafka Broker and HDFS NFS Gateway; and
- Cloudera-manager.shield.com: Operational Analytics (OA) node, includes Cloudera Manager.

For the setup of the target, traffic generator and attacker servers, three virtual machines were used. Specifically, the deployment and configuration for those were as follows:

- The target server: The victim server towards which the Slowloris attack was performed. Apache 2 on Ubuntu 16.04 Server was installed.

- The traffic generator: Replays files which contain normal, legitimate traffic. Ubuntu 16.04 Server was installed.
- The attacker: A Kali Linux VM, which performs the attacks against the victim webserver. Kali Linux 2017.2. An open-source Python implementation of Slowloris³ was used to perform the attack.

For the implementation of the NFVI-PoP, the OpenStack cloud computing software platform has been selected and installed on a server, following an “all-in-one” deployment scheme. Specifically, OpenStack Ocata release was deployed with the following components: Nova, Neutron, Glance, Heat, Horizon and Keystone. Also, RabbitMQ open source message broker was installed for communication between the OpenStack components and MariaDB server, as the database backend for OpenStack services. The re-direction of traffic through the different deployed network services was managed with Openflow rules on the Open VSwitch (OVS) internal to OpenStack, that was deployed to implement OpenStack networking.

The detailed implementation and interfacing of the virtualised DPI and the L3 filter services has been described in Deliverable D3.3. A modification that was made to the virtualised DPI for the implementation of this scenario was to deploy the developed distributed collector (collect-d) to allow the ingestion of the traffic towards the DARE.

Network traffic was recorded and anonymized from the network of the Media Networks Laboratory (at the Institute of Informatics and Telecommunications of NCSR Demokritos) and saved in pcap files. For the NetFlow Telemetry, the target server forwards the traffic from its network interface, using the fprobe tool, to cloudera-host-2, which is the server responsible for the data ingestion function of the Data Acquisition module. Cloudera-host-2 records the traffic in nfcapd files. The attacks are deployed through Kali Linux in real time (during the replay of the network traffic from the traffic generator). This way, they are also forwarded to cloudera-host-2 and saved in the aforementioned nfcapd files.

The sequence of steps that take place during the Slowloris demonstration are the following:

1. A client (attacker) is performing a Slowloris attack against a web server (target).
2. Traffic is intercepted within the NFVI-PoP deployment, analysed by the vDPI vNSF and sent it to the DARE.
3. The Classification algorithm (supervised) classifies the traffic into either Slowloris attack or background (normal) traffic.
4. A notification of the occurring attack is shown to the user through SHIELD Dashboard.
5. A recommendation (MSPL) is created to block the attacker’s IP address.
6. The user accepts the recommendation through the dashboard and a firewall rule is sent to the L3 filter vNSF to block the attacker’s IP address.
7. The L3 filter vNSF blocks the attacker’s IP address and so the attack is mitigated.

³ <https://github.com/gkbrk/slowloris>

5.2.4. Scenario results

The Cognitive DA module outputs the results of the analytics procedure in the form of a csv file. The csv file contains a list of the malicious traffic flows discovered by the Random Forest model. Furthermore, each entry provides extensive information like date and time, source and destination IP addresses, source and destination ports, network protocol, packet and byte counts, etc, following the netflow (nfcapd) protocol. These results are sent to the Dashboard via the RabbitMQ message broker. These results, apart from providing contextual info to the Dashboard are leveraged to generate an attack-specific remediation policy (MSLP) that will eventually be applied by the user to block the attack.

5.3. Cryptomining abuse

5.3.1. Scenario motivation

Cryptocurrencies [1] are forms of digital currency, that follow a decentralised architecture (as opposed to central banking systems). Bitcoin [2] is one of the most well-known decentralised cryptocurrencies and is often used as a global online payment system. Bitcoin, as well as other cryptocurrencies like Ethereum (from Ethereum) [3], Monero [4] etc., utilise blockchain [5] technology to record transactions. The existence of the blockchain ledger requires a network of communicating nodes that maintain and verify the blockchain. Thus, the decentralised architecture has led to the development of cryptocurrency mining (also known as cryptomining). Users may join the network of communicating nodes to complete tasks relating to the blockchain, offering their computational resources for a monetary reward (payable in cryptocurrencies).

Although cryptomining is per se not illegal, there are numerous cases where cryptomining abuses have been reported. On Jan 2018, software security firm Check Point issued a report [6] about a sharp increase in the prevalence of crypto-mining malware, stating that 55% of businesses worldwide are affected by the attacks. The report declared Coinhive [7] to be the number 1 “Most Wanted Malware.” Coinhive was used in a large-scale malware campaign [8] and has been found on more than 300 academic and government websites, news sites, etc. The use of malicious cryptomining software hidden in legitimate websites has also been known as “cryptojacking”. A similar case was revealed on YouTube [9]. According to the report, anonymous hackers have managed to run ads on YouTube that consumed the visitors’ CPU power and electricity in order to mine cryptocurrencies for the attackers. In some cases of malware like Loapi [10], mining overheated mobile devices and caused significant damage, through the constant use of the device’s processor at maximum load.

5.3.2. Scenario description

Cryptocurrency mining is commonly performed via:

- A solo user downloading the related client software on their systems: This method is almost deprecated since the chance of being assigned a block to resolve is very small and too random.

- A user joining a cryptocurrency mining pool: Mining pools are platforms where miners can pool their resources and split the bitcoin reward proportionally. The earnings are small but more guaranteed than solo mining, hence it is taking over as the most popular means of mining.
- Mining scripts inserted in webpages and using the (sometimes unsuspecting) visitors' resources to mine bitcoin.

Although cryptocurrency mining is legal, there exist situations where it is not allowable, i.e. in enterprise networks. Malicious scripts/malware can also high-jack a system's resources while the user remains unawares, and cause significant performance loss or in some cases, damage to their devices.

5.3.3. Scenario setup

Figure 8 illustrates the main SHIELD components⁴ that are used in this demonstration. The security analytics engine is in charge of monitoring network traffic, as shown in Figure 9. If an anomaly is detected (Figure 10), its data are relayed to the remediation engine that provides remediation suggestions (Figure 11). The remediation options are then sent to the dashboard via a RabbitMQ messaging system, where the user may select whether to apply them (Figure 12). This demonstration also features an additional VNF that is utilised as an intrusion detection and prevention (IDPS).

In the first part of the demo, an employee is using an organisation's resources to mine cryptocurrencies. Although mining itself is not illegal, it can be considered as a misuse of an organisation's assets and may potentially damage them. SHIELD's DARE (and specifically, the security analytics engine) detects the mining traffic as an anomaly (Figure 9). This is a challenge as the Stratum [11] protocol that is most commonly utilised for cryptomining operations (i.e. communication with the mining pool) is non-standardised and often uses randomised ports. Once it is detected, the remediation engine provides a recipe to block outgoing packets towards the mining pool. The dashboard receives the remediation recipe and may choose whether to enforce it or not. This is a major feature that shows how SHIELD can detect a non-standardised protocol like Stratum.

⁴ The detailed implementation and interfacing of components that are utilised in the demo are included in past documents. Specifically, you may refer to SHIELD deliverables D3.3 and D4.3 for a setup and installation guide, whereas the technical specifications are provided in SHIELD deliverables D3.2 and D4.2.

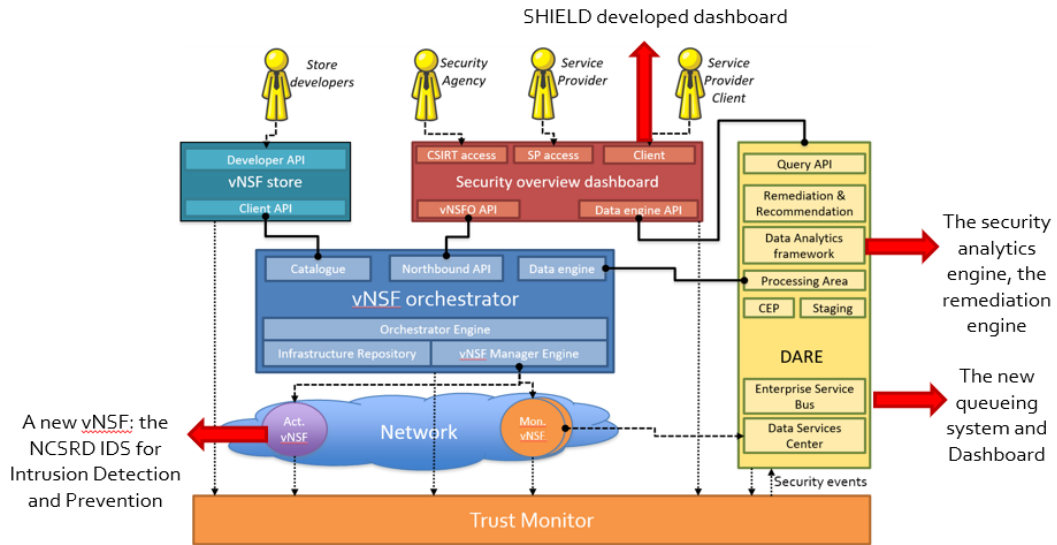


Figure 8. Main SHIELD components utilised in the cryptomining abuses demonstration.

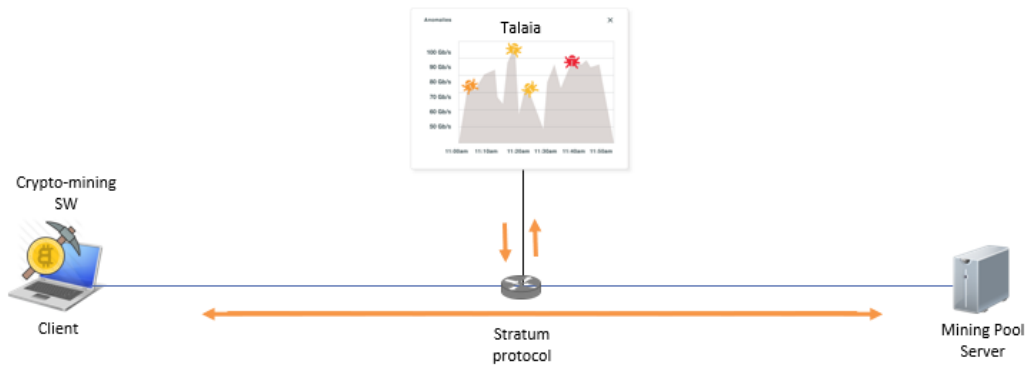


Figure 9. Setup for the cryptomining detection demonstration.



(a)

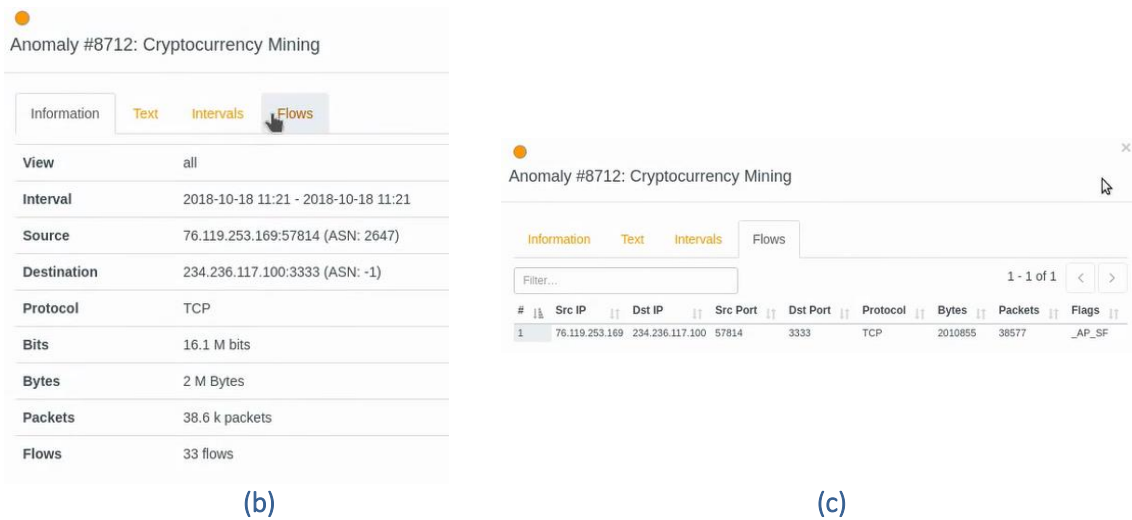


Figure 10. Detection of cryptomining traffic in SHIELD’s DARE. (a) The anomaly is detected, (b) and (c) overview of the anomaly characteristics.

```

tail: mspls.dump: file truncated
----- file got truncated mspls.dump 2018/10/18 12:24:21-----
<mspl-set xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://security.polito.it/shield/mspl">
  <context>
    <severity>2</severity>
    <type>Cryptomining</type>
    <timestamp>2018-10-18T12:21:00</timestamp>
  </context>
  <it-resource id="l3filter_vnfd">
    <configuration xsi:type="filtering-configuration">
      <default-action>accept</default-action>
      <resolution-strategy>FMR</resolution-strategy>
      <rule>
        <priority>1</priority>
        <action>drop</action>
        <condition>
          <packet-filter-condition>
            <direction>inbound</direction>
            <source-address>76.119.253.169</source-address>
            <source-port>57814</source-port>
            <destination-address>234.236.117.100</destination-address>
            <destination-port>3333</destination-port>
            <protocol>TCP</protocol>
          </packet-filter-condition>
        </condition>
      </rule>
    </configuration>
  </it-resource>
</mspl-set>
    
```

Figure 11. The remediation recipe for the cryptomining anomaly.

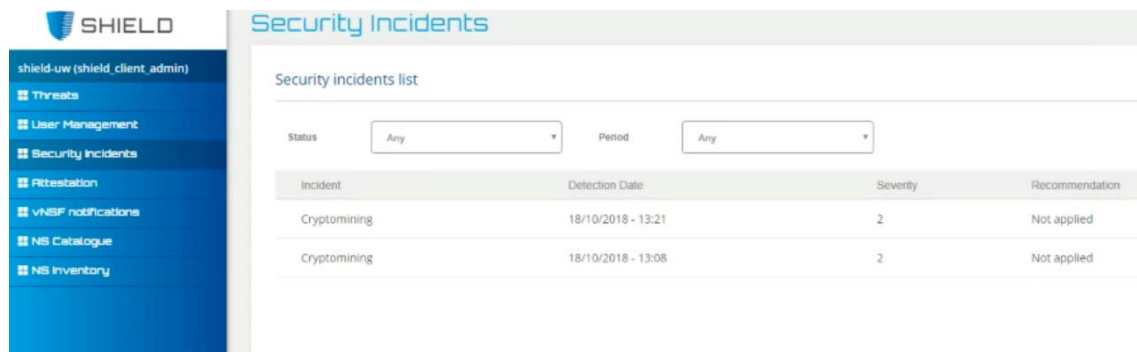


Figure 12. Cryptomining anomalies shown on the user dashboard.

In the second part of the demo, a malicious cryptojacking script is injected in a page (Figure 13). Coinhive was selected, specifically due to its popularity in being used maliciously. When a user visits the page, the script hijacks CPU resources for mining. A simple user interface is utilised for the demo purposes, to show the mining process.

The vIDPs has been configured to run in an intrusion prevention mode to prevent the users from loading the cryptomining script on their browser. Figure 14 illustrates the snort rule utilised to block the mining script. An extension of the existing implementation of the vIDS was developed to push the Snort alerts to the SHIELD dashboard for notifying the user for identified threats and/or mitigation actions. The overall setup for this part of the scenario is illustrated in Figure 15. Alerts are logged although they also appear as a pop-up notification, so that the dashboard user is always notified (Figure 16). To implement this functionality, the occurring Snort alerts were reformatted to an agreed pre-defined JSON format and were sent to through an exchange to a RabbitMQ message broker.



Figure 13. A website running a Coinhive script.

```
1- {
2-   "snort_rule":{
3-     "action": "drop",
4-     "message": "Observed DNS Query to public CryptoMining pool domain",
5-     "content": "authedmine",
6-     "protocol": "udp",
7-     "source_ip": "any",
8-     "source_port": "any",
9-     "destination_ip": "any",
10-    "destination_port": "53",
11-    "classtype": "cryptomining-event"
12-   }
13- }
```

Figure 14. The snort rule used to drop traffic towards the cryptomining pool domain.



Figure 15. Immediate effects of the cryptojacking mitigation rule.

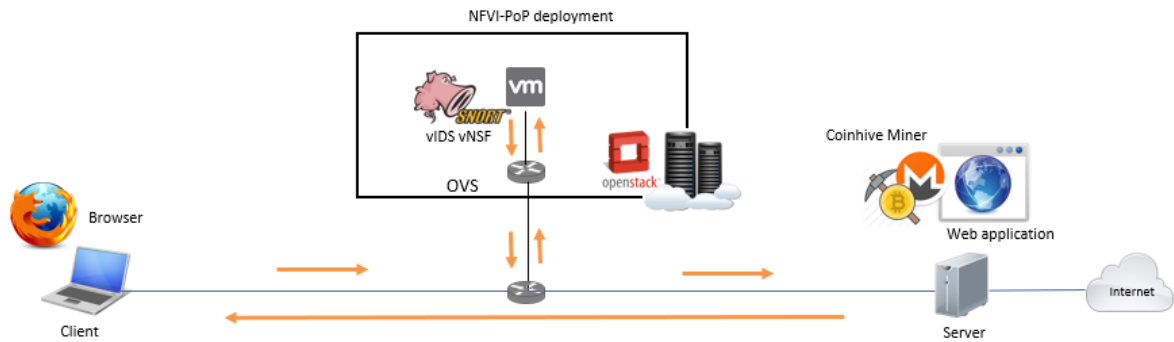


Figure 16. Demonstration setup for detection of cryptojacking scripts.

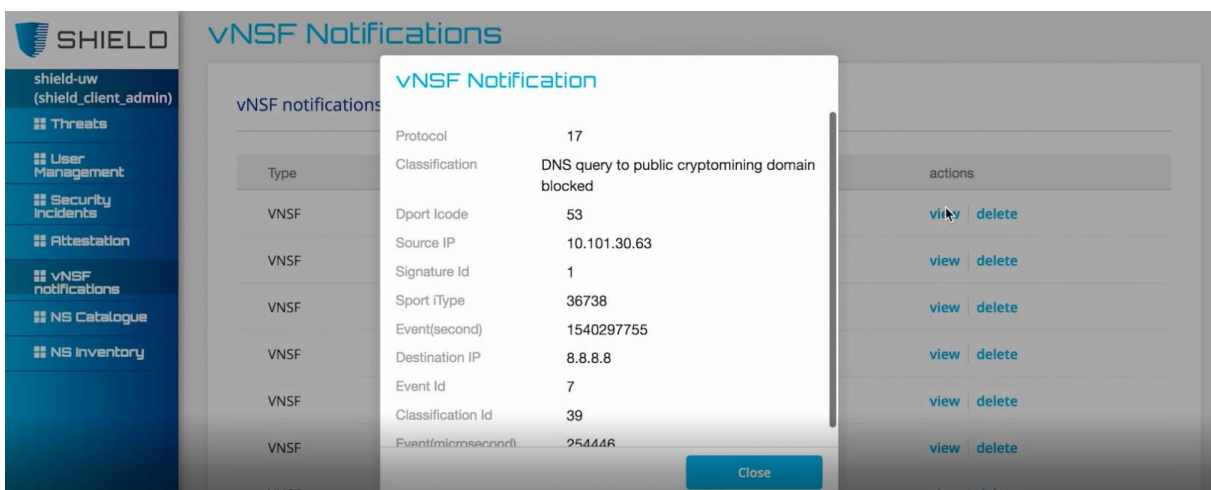


Figure 17. vNSF notification and pop-up.

5.3.4. Scenario results

In the first part of the demo, SHIELD showed how to detect and block mining traffic. Network traffic to known mining pools and online wallets can be blocked in the service level, based on a blocklist. Detection of unknown mining pools can be challenging, as the related protocols (e.g. Stratum) are not standardized and often use randomized ports. The SHIELD security analytics engine was shown to detect Stratum traffic as an anomaly. This helps an enterprise secure its assets and devices against misuse by employees.

In the second part of the demo, SHIELD showed how to detect and block cryptojacking scripts. Using intrusion detection countermeasures deployed as network services, SHIELD blocks a malicious javascript that is intended to mine resources from an unsuspecting visitor to a website. This service provides an umbrella of protection to all devices in the network (computers, laptops, mobile devices, IoT etc.) without the need to configure individual protection measures (e.g. antivirus, browser extensions etc.) across all devices. The malicious *.js is blocked before it is served to the user and the administrator is notified through a message to the Dashboard. Therefore, the remediation does not require the SHIELD DARE components. The administrator may choose to roll-back a remediation action if necessary and unblock a script, in case of a false detection.

5.4. Attestation of the vNSFs, their hosts and the network

5.4.1. Scenario motivation

The network infrastructure of SHIELD is a critical part of the overall system as it is responsible for forwarding the packets and executing the different vNSFs to the traffic. In addition to the requirement of having the network infrastructure behave as expected, for evident reasons such as quality of service or service level agreement, it is also one of the easiest attack surfaces of SHIELD: any traffic flowing needed to be handled by the vNSFs will necessarily go through some network elements (e.g. switches) and compute nodes (e.g. servers) in addition to the required vNSFs. Ensuring the integrity of the network infrastructure is thus paramount to ensure the correctness of SHIELD as a whole. The Trust Monitor is responsible for that verification in SHIELD, as illustrated in Figure 18.

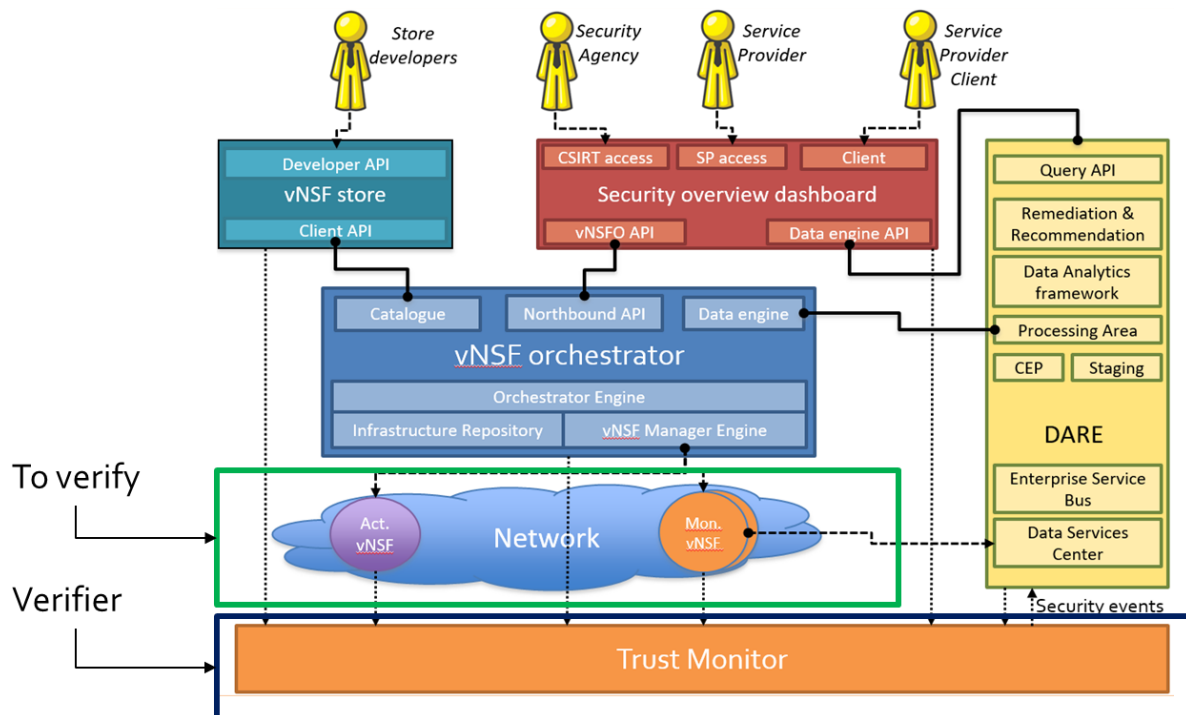


Figure 18. The Trust Monitor in SHIELD

5.4.2. Scenario description

To achieve that, SHIELD relies on the Protect, Detect and Recover approach for the cyber-resiliency of the network infrastructure:

- Protect refers to the effort put in place by the SHIELD's network infrastructure stakeholder to harden the different components (e.g. static code analysis of the vNSFs, access control management for the servers and switches, etc.);
- Detect is the ability to identify unauthorized behaviours of the network infrastructure's components, usually due to an attack or to a misconfiguration;
- Recover is the action of getting back the network infrastructure to a trusted state – that is a state where each component behave as expected.

This demonstration focuses on the detection capability of the Trust Monitor to identify misbehaving components of the network infrastructure. It also shows the different stakeholders view of the network infrastructure: the network administrator and the end-user (vNSF client) views. One of the main addition of this demonstration compared to the previous one is the integration of the Trust Monitor with the other components of SHIELD: the Security Dashboard, the vNSF Orchestrator, the vNSF Store and the DARE, shown on Figure 19.

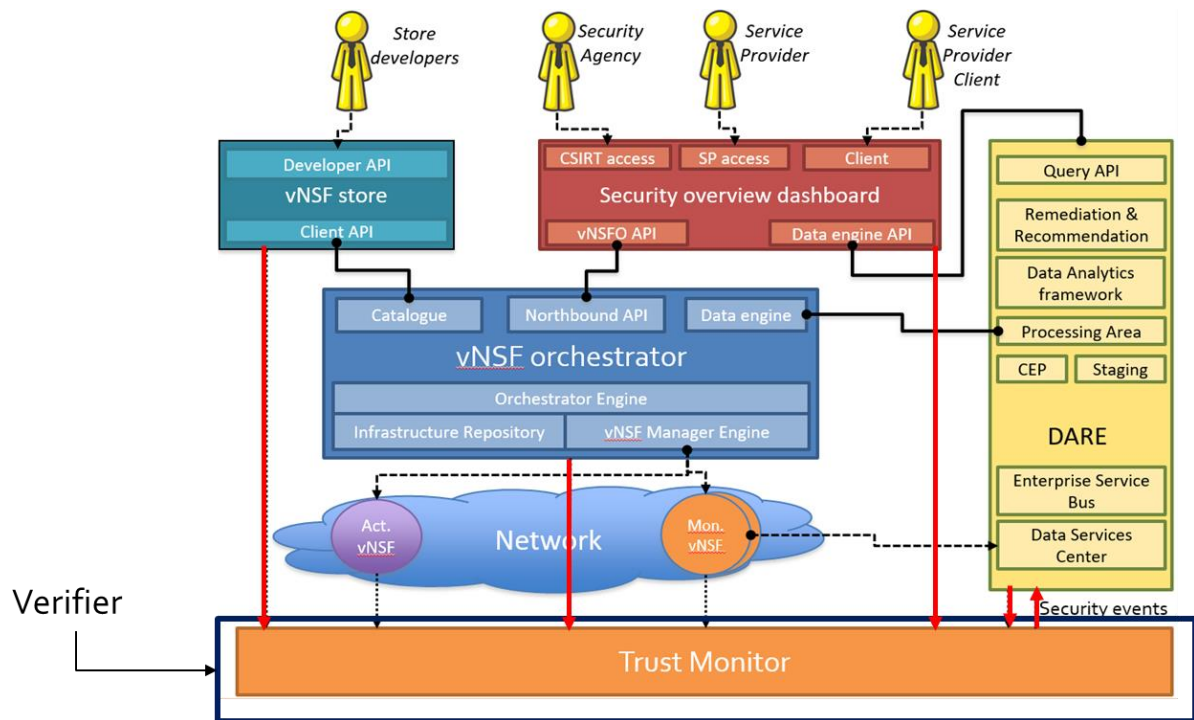


Figure 19. The interaction of the Trust Monitor with the other SHIELD's components

5.4.3. Scenario setup

The proposed scenario is based on the interaction between the following components:

- Trust Monitor application running a periodic attestation task on the NFVI infrastructure;
- NFVI compute host configured to run Docker containers and equipped with a Trusted Platform Module (TPM);
- SDN-enabled switch for forwarding traffic in the NFVI PoP and equipped with a TPM;
- vNSF Orchestrator, which manages the vNSF lifecycle and provides the list of running vNSFs to the TM;
- vNSF Store, which stores the NS and vNSF descriptors and provides the vNSF Security Manifests to the TM;
- Dashboard, which shows notifications on the trustworthiness of the infrastructure from the TM.

5.4.4. Scenario results

5.4.4.1. Infrastructure administrator (ISP)

The ISP administrator view in the Dashboard shows the attestation result for the NFVI physical infrastructure, as shown in Figure 5-13. This includes the trust level for both the NFVI compute host and the SDN switch. The TM periodically queries the vNSF Orchestrator to retrieve the list of physical nodes belonging to the NFVI, and runs the remote attestation procedure for each of them. Whenever a new attestation result is obtained, a notification pops up in the ISP administrator Dashboard. The ISP administrator can also browse the history of past attestation, illustrated in Figure 20.

Regarding the SDN network elements, the ISP can leverage the Trust Monitor to verify that only authorised configurations are used by the different switches. The ISP administrator configures the authorised configuration for each network element in the Trust Monitor. Whenever a different configuration is being used by a network element, the Trust Monitor notifies the ISP administrator in the SHIELD dashboard. This capability is particularly useful to detect any insider threats and it is worth noting that this covers both the stored configuration as well as the current running configuration.

The screenshot shows the SHIELD dashboard's 'Attestation' view. On the left is a blue sidebar with the SHIELD logo and a list of menu items: Default (admin), Threats, User Management, Security Incidents, Attestation (highlighted), VNSF notifications, SecaaS Client Management, VNSF Catalogue, NS Catalogue, Billing, and Onboard validations. The main content area is titled 'Attestation' and features a blue button 'Attest NFV Infrastructure' in the top right. Below the title is a 'Type' dropdown menu set to 'any'. A table displays the attestation records:

Type	Node	Driver	Trusted	Recommendation	Time	Actions
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-27 17:43:52.620125483 +0000 UTC	View History Attest
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:43:59.292322 +0000 UTC	View History Attest

At the bottom right of the table, it says '< > Showing 1 to 2 of 1'. Below the table, there are logos for the European Commission and Horizon 2020 European Union Funding for Research & Innovation, along with the text: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199'. The bottom left of the sidebar shows 'admin' with a dropdown arrow.

Figure 20. ISP administrator attestation view

The screenshot shows the SHIELD dashboard with a sidebar menu on the left containing options like Threats, User Management, Security Incidents, Attestation, vNSF notifications, SecaaS Client Management, vNSF Catalogue, NS Catalogue, Billing, and Onboard validations. The main content area is titled 'Attestation History' and displays a table of attestation results for 'nfvi-node'. A 'Back' button is located in the top right corner of the table area.

Type	Node	Driver	Trusted	Recommendation	Time	Actions
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:43:59.292322 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:38:51.551505 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:33:41.807939 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:28:31.903898 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:23:24.255623 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:18:14.496205 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:13:06.655816 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:07:58.699900 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 17:02:51.762851 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 16:57:44.032587 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 16:47:34.834282 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 16:42:28.144277 +0000 UTC	View
hosts	nfvi-node	OAT	true	Not Applied	2019-02-27 16:37:20.175720 +0000 UTC	View

Figure 21. Historical log of attestation results

5.4.4.2. Security-as-a-Service client (vNSF user)

The SecaaS client view in the Dashboard shows the attestation result for each vNSF belonging to the tenant, as shown on Figure 22. Once the NS comprising the target vNSF is instantiated via the Dashboard, the next attestation result shows that the vNSF is trusted. The TM periodically queries the vNSF Orchestrator to retrieve the list of vNSFs running in the NFVI, and runs the remote attestation procedure for each of them. At each vNSF attestation, the TM queries the vNSF Store to retrieve the white-list of reference measurements included in the vNSF Security Manifest. Whenever a new attestation result is obtained, a notification pops up in the SecaaS client Dashboard view. Once the vNSF image is manipulated by running a binary that is not part of the vNSF Security Manifest, the next attestation result shows that the vNSF is in an untrusted state.

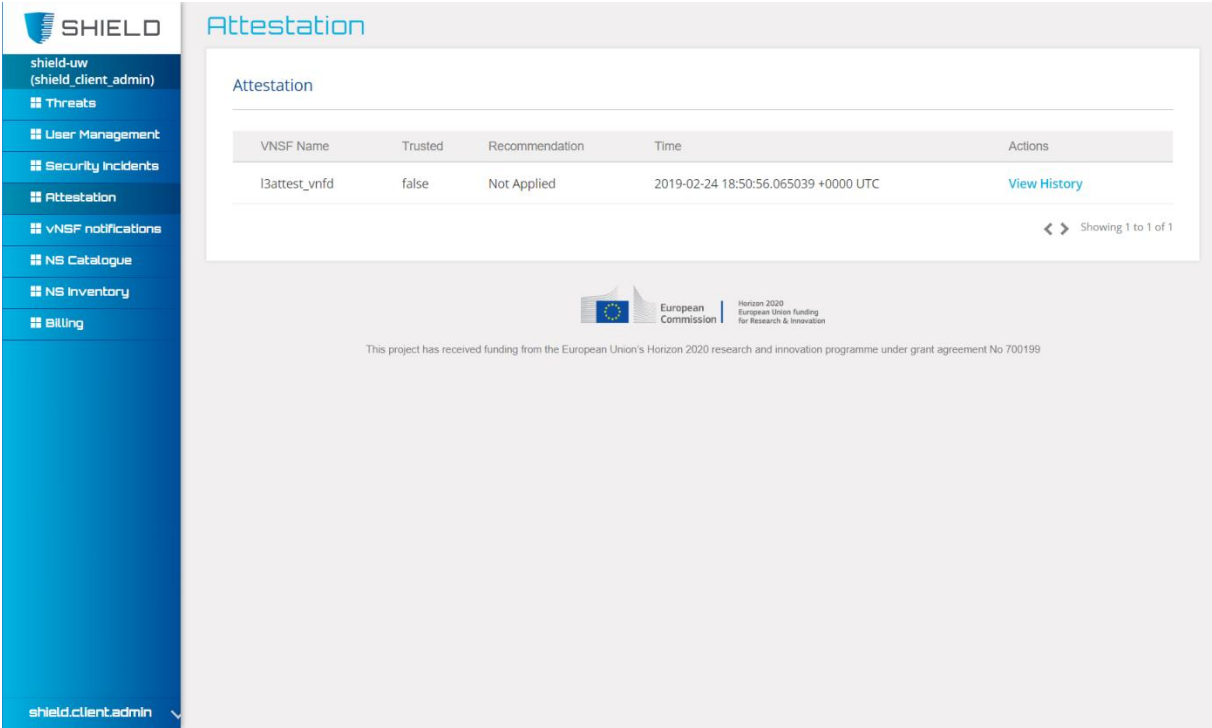


Figure 22. SecaaS client attestation view

6. FINAL PROJECT DEMONSTRATIONS

The demonstrations done at the end of the project are based on the Y2 scenarios (see Chap.5), properly enhanced in order i) to assess the scenarios under more realistic conditions and ii) to showcase the additional features which were implemented.

6.1. A 0-day worm attack

6.1.1. Scenario motivation

The motivation of the scenario is the same one as explained in subsection 5.1.1. The main change is the goal to run the attack and the mitigation in a more close to market scenario, under more realistic conditions.

6.1.2. Scenario description

The data used for this demo is no longer based on a controlled Lab traffic but based on the SHIELD platform behaviour against real traffic. The traffic was captured in the Media Networks Laboratory (MNL) of NCSR “Demokritos”, naturally generated during the day-to-day operations of the lab, which currently employs 15 full-time researchers (see Section 8.1.1 for details).

6.1.3. Scenario setup

The algorithm is trained with the traffic corresponding to 24 hours of the 22th of January 2019. During this day, no attack was performed so we consider all the traffic “normal”. This traffic is composed by more than 5 million flows and more than 1500 different IPs.

During the operational hours of the 27th of February 2019, the Wannacry traffic was reinjected in the NCSR network changing the ips from the original ones to ones in the range on the NCSR network.

The analysis of the traffic corresponds to 5 hours of this day, including the hours when Wannacry was launched.

6.1.4. Scenario results

The anomaly detection is able to detect the 4 infected IPs together with more than 2251 other IPs. Here, we observe a decreasing of the accuracy, precision and recall over the Lab scenario. This behaviour is mainly due to the following reasons. Firstly, the algorithm was trained using data from a different day, so probably increasing the amount of data used to train (at least one week) could increase these results since the algorithm would be more used to different traffic behaviours. Secondly, because we do not control all the elements that are part of this scenario, mainly attacks or threats that are not artificially introduced but part of the scenario, and that the algorithm is detecting. Thirdly, because the amount of “normal” traffic is much bigger than the wannacry one, this unbalanced testing dataset may impact in the result.

Nevertheless, the classification algorithm is able to isolate the 4 infected IPs from the anomalies detected, which confirms the robustness of the solution. Hence, the designed combination of anomaly detection plus classification demonstrates its efficiency and good results in different situations.

6.2. Cryptomining abuse

With respect to the cryptomining abuses there are no changes to this demonstration since Y2 (as described in subsection 5.3). The Y2.5 demonstration aims to showcase additional SHIELD features such as billing, multiple users and the logging/visualisation of threat information on the dashboard.

6.2.1. Scenario description

In terms of the cryptomining abuses there are no updates since Y2 (please also refer to subsection 5.3 for a full description).

The billing model is a new feature introduced in this demonstration, showcasing different billing contexts, according to the user role in the platform. From the SecaaS client perspective, the billing panel provides a monthly discrimination of the fees applied to each network service instance, including the month usage period and applied fee percentage for that period. From the Platform administrator perspective, a more comprehensive monthly billing analysis is shown, which includes the network service instances per each SecaaS Client and the used vNSFs, including the expense costs towards the Developers.

Regarding the multi user functionality, each SecaaS Client has a different NS Inventory which contains its network services and instances. The NS Inventory view, including the network service instances, is segregated and belongs to specific SecaaS Client, whereas only the Platform administrator has the ability to see the number of network service instances and the relation to their SecaaS Clients.

The dashboard logs the activity of users. Each task performed by users is registered in the activity panel, identifying the date, user, SecaaS Client and task description.

6.2.2. Scenario setup

The basic setup of the scenario has not been altered since the Y2 version. Please refer to subsection 5.3 for more information.

6.2.3. Scenario results

In addition to the detection and mitigation of cryptomining abuses, this demo illustrated how billing can be introduced within SHIELD. A simple flat-rate scheme was utilised in this case. The granularity of the flat-rate billing model is one day, i.e. each network service instance is charged for at least one day since the moment of its instantiation. The same logic applies to the expenses towards developer users, inherent to the usage their vNSFs.

6.3. Infrastructure and service attestation

6.3.1. Scenario motivation

The motivation of the infrastructure and service attestation demonstration is identical to the previous attestation demonstrations (see Sec. 5.4): the Trust Monitor should continually monitor the integrity of the network infrastructure and notify the SHIELD operator when a security breach is detected.

6.3.2. Scenario description

For this demonstration, we assume that an attacker managed to compromise an administrator laptop, thus having access to the credentials to connect to the different platform of the SHIELD network infrastructure. The attacker will use those credentials to hide its use of DNS tunnelling for exfiltrating data.

6.3.3. Scenario setup

The proposed scenario is based on the interaction between the following components:

- Trust Monitor application running a periodic attestation task on the NFVI infrastructure;
- NFVI compute host configured to run Docker containers and equipped with a Trusted Platform Module (TPM);
- SDN-enabled switch for forwarding traffic in the NFVI PoP and equipped with a TPM;
- vNSF Orchestrator, which manages the vNSF lifecycle and provides the list of running vNSFs to the TM;
- vNSF Store, which stores the NS and vNSF descriptors and provides the vNSF Security Manifests to the TM;
- Dashboard, which shows notifications on the trustworthiness of the infrastructure from the TM.

This demonstration does not require a scenario-specific testbed and was showcased using the SHIELD's testbeds, with their existing topologies.

6.3.4. Scenario results

The initial attack, once the administrator credentials have been retrieved, consists of a connection to the SDN controller inside the vNSF Orchestrator, in order to inject SDN rules and bypass the DNS capturing vNSF. This is detected by the Trust Monitor, which also recommends to push the correct rules to the vNSF Orchestrator; once the correct SDN rules are put back in place, the attestation succeeds again.

Given the failure to reroute its DNS traffic, the attacker then tries to replace the DNS capturing vNSF by a custom vNSF that does not capture DNS traffic. He connects to a NFVI node and modify the vNSF image; then he tries to instantiate the vNSF through the Dashboard. Since the

Trust Monitor attests the vNSFs before they are used, the vNSF Orchestrator is notified of the unknown vNSF and terminates it; a notification is sent to the Dashboard.

Finally, the attacker wants to deploy a vulnerable software on the NFVI node in order to get a permanent access. The Trust Monitor detects the modification during the next periodic attestation and the Dashboard can be used to inspect the attestation audit trail.

7. OPEN DEMONSTRATION EVENTS

7.1. Final workshop

7.1.1. Event details

The SHIELD consortium has organised a tutorial entitled “Modern Network-based Security: Softwarized Networking, Trusted Computing, and Artificial Intelligence for Cybersecurity” on February 24, 2019 in Prague (Czech Republic) as a final workshop. This has been realised as part of the 5th International Conference on Information Systems Security and Privacy (ICISSP 2019), held in Prague from February 23rd to February 25th.

The workshop has introduced to the audience the key pillars of the SHIELD platform, comprising modern network technologies (SDN, Software Defined Networking, and NFV, Network Function Virtualization), Trusted Computing (TC), Machine Learning (ML), and Artificial Intelligence (AI) techniques. The workshop has focused on describing how the conjunct use of these technologies can effectively counter cyberattacks in a trusted softwarised domain. The technical aspects have been complemented by a market and economical analysis, to evaluate benefits versus costs of the SHIELD approach.

Finally, the SHIELD consortium has held a booth during the three days of the event to present the project mission and achievements to the conference’s attendees.

7.1.2. Demo setup / ETSI ENI PoC

Experiential Networked Intelligence Industry Specification Group (ENI ISG [13]) is defining a Cognitive Management architecture for the network, using Artificial Intelligence (AI) techniques, such as machine learning, and context-aware policies to adjust network based on user needs, environmental conditions and business goals.

The ETSI ENI PoC [14] had the objective to demonstrate how the DARE component developed in SHIELD, can cover some of the functionalities of the ENI system, including some external interfaces still in definition by ETSI ENI.

The PoC called “Securing against Intruders and other threats through a NFV-enabled Environment” was presented during the ICISSP event and next figure shows the global scheme.

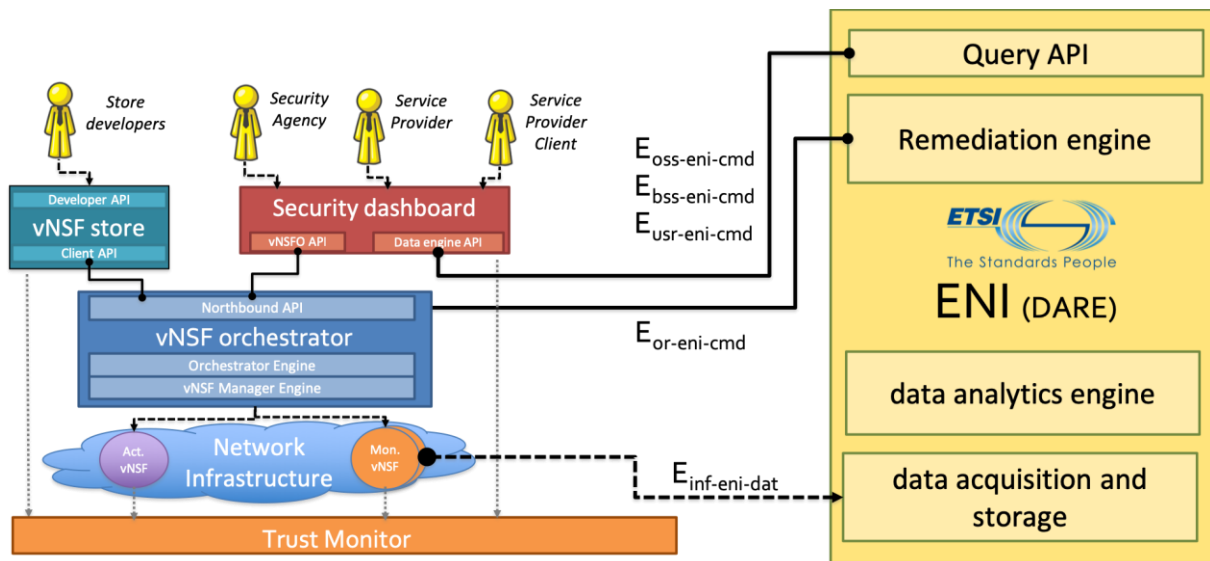


Figure 23. SHIELD instantiation as an ETSI ENI PoC

The DARE components and interfaces needed for the ETSI ENI PoC were :

- Data acquisition and storage: Using apache spot d-collector capacity, network traffic including malware (wannacry) was captured and stored in netflow format within the distributed Hadoop system. This module correspond with “Ingestion and Normalization” ENI functional block using the $E_{inf-eni-dat}$ interface.
- Data analytics engine: Serialisation of two machine Learning algorithms was demonstrated. First the anomaly detection, that filters out normal traffic and, at second stage, the classification algorithm, pre-trained in wannacry traffic, which identifies the malicious flows. These modules correspond to the “Cognition Framework” ENI functional block.
- Remediation engine and Query API: The cyber topology engine was able to identify the threats, derive the correct recipe to mitigate the attack, and finally provide an intent-based security policy, presented as a suggestion to the Human operator. This module corresponds to the “Denormalisation and Output generation” ENI functional block using some of the external interfaces available ($E_{oss-eni-cmd}$, $E_{bss-eni-cmd}$, $E_{usr-eni-cmd}$).

Additionally, as part of the PoC goals, an additional demonstration related to remote attestation technology was shown. It presented the idea of how to avoid the tampering of VNFs monitoring and data collecting processes, that could affect the ENI system reliability.

7.1.3. Assessment and feedback

The workshop (organized under ICISSP as a dedicated, 3-hour tutorial session) had a total of 26 attendees, which provided valuable feedback on the project proposal, current status and future iterations of the platform. With respect to anomaly detection, the audience proposed to focus on Layer 7 information in the DARE (e.g. Web logs) in order to better detect application-layer incidents, and stated the need to benchmark and compare the SHIELD Machine Learning performance with commercial closed-source solutions. Moreover, the audience stressed the importance of retraining of ML algorithms and questioned whether the SHIELD platform could be extended to provide automated (and periodic) retraining through Apache Spot. With respect to privacy, GDPR compliance was discussed as a crucial issue for the viability of the service. Regarding platform trust, the runtime attestation (already part of the Trust Monitor workflow) is considered particularly relevant to protect the whole life-cycle of VNFs. Finally, the audience proposed to consider alternative SDN deployments with SDN controller redundancy to avoid having a single point of failure at the network control.



Figure 24. SHIELD final workshop (tutorial session)

7.2. Winter school

7.2.1. Event details

SHIELD has co-organised the NeCS Cyber Security Winter School 2019, held in cooperation with the C3ISP EU project, AEGIS, and the CINI Cyber Security National Lab, from 18th to 22nd February 2019 in Fai della Paganella near Trento, Italy.

The European Network for Cybersecurity (NeCS) was formed in response to the increased need for highly qualified experts. It addresses the issues of training and development of talented junior researchers as indicated in the European Cyber-security strategy and highlighted in the EC's Digital Agenda.

In the scope of this event, the SHIELD consortium has held several courses related to the project's activities in the following lectures:

- “SDN, NFV and AI: the SHIELD approach to network-based cybersecurity” by prof. Antonio Lioy (Politecnico di Torino);
- “Verifying the integrity of software-defined infrastructures” by Ludovic Jacquin (Hewlett Packard Enterprise Labs);
- “Machine learning for cybersecurity” by Bernat Gaston (i2CAT foundation).

Moreover, a live showcase of Y2 demonstrations as described in Section 5.1 and 5.4 has been performed in a session of the Winter School.

7.2.2. Hands-on session description

In the context of the lecture “Machine learning for cybersecurity”, a one-hour coding session was also scheduled by Dimitris Papadopoulos (Infili Technologies PC), aiming at demonstrating how scalable analytics can be exploited in the cybersecurity domain. During this hands-on session, attendees were introduced to one of DARE's underlying distributed computing frameworks, namely Apache Spark. The session's objectives were to provide a high-level understanding of Spark's basic components and functionalities, to explore its advantages for production-level analytics tasks over other commonly used frameworks and finally to implement a machine-learning model (Random Forest classifier) in PySpark, that assigns threat labels to network traffic records. The code presented to the audience was a slightly modified version of the (supervised) classification algorithm used in Y2 demonstrations for the detection of worm and DDoS attacks.

The audience had the option to code along with the presenter, by downloading a pre-configured virtual machine that contained all the necessary prerequisites to train a classifier model using a publicly available dataset.

7.2.3. Assessment and feedback

The audience from the Winter School was mainly composed by PhD students, young researchers and by experts in cybersecurity. Questions have addressed the scalability of the attestation framework and the protection of the Whitelist Database by insider threats. In this regard, the Trust Monitor is designed as a stateless application, whose only stateful sub-component is represented by the Whitelist Database. Because of this, replicas could be instantiated to scale up in case of several target nodes for attestation. With respect to the Whitelist Database, the audience has commented that it should be protected against insider threats to not invalidate the integrity verification phase. Moreover, the audience has suggested to include attestation logs as part of the anomaly detection logic in the DARE in a future iteration of the platform, so that they can be leveraged to identify certain type of attacks in the ISP infrastructure. Regarding the machine-learning models that were introduced as part of the DARE predictive analytics capabilities, most of the questions were targeted at the utilized technologies and frameworks, the detection efficiency of the implemented algorithms, as well as the availability of public datasets to be used for training similar models.



Figure 25. Winter school – course on SHIELD integrity verification mechanisms



Figure 26. Winter school – attendees

8. SHIELD PILOTS

The aim of the SHIELD pilot activities have been to assess the efficiency of the SHIELD solution under near-operational conditions, as well as to engage stakeholders from outside the project team. Three pilot workshops were organised, corresponding to the three basic use cases of the system (see Chap. 2)

8.1. ISP pilot – UC1

8.1.1. Data collection and pilot setup

The data used for the ISP pilot were based on the SHIELD platform behaviour against real traffic. The traffic was captured in the Media Networks Laboratory (MNL) of NCSR “Demokritos”, naturally generated during the day-to-day operations of the lab, which currently employs 15 full-time researchers. In this manner, the project technical partners were offered the opportunity to assess SHIELD under real usage conditions, rather than using replays of synthetic traffic.

The setup of the SHIELD pilot deployment at NCSR is depicted in Figure 27 below. The traffic information (in NetFlow v.9) format was captured from the Lab’s central firewall, which connects the researchers’ PCs with the lab computing infrastructure and to the backbone network of NCSR “Demokritos” – and from there to the Internet. The firewall was configured to send the traffic information in real time to a NetFlow collector vNSF, which in turn fed the DARE storage using the distributed collector pipeline.

To preserve the anonymity of the personnel and, more importantly, allow the usage of the generated dataset outside NCSR premises by third parties without severe GDPR implications, the collected information was anonymised. For this purpose, an anonymiser component was introduced in the ingest chain, exploiting the CryptoPAN library⁵ developed by GeorgiaTech in order to convert the lab’s internal IP addresses into other ones, in an 1-to-1 manner. The external IP addresses were not affected.

A dedicated PC was installed outside the lab’s perimeter to periodically launch targeted attacks against the lab devices. Depending on their nature, some of these attacks were blocked by the firewall, others were not. Yet all traffic in any case (either blocked or allowed) was reported to the DARE.

The incident detection and classification results, as well as the mitigation proposal, was visualised in the SHIELD dashboard.

The pilot operated for two whole weeks (21/1/2019 – 4/2/2019), capturing and processing traffic without interruption, on 24/7 basis.

⁵ <https://www.cc.gatech.edu/computing/Networking/projects/cryptopan/>

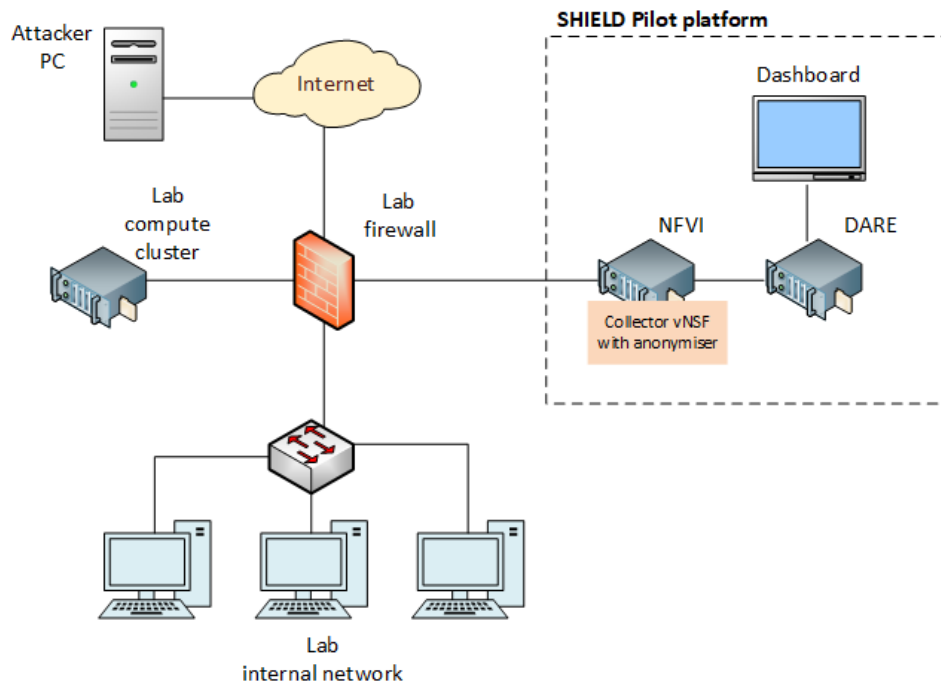


Figure 27. SHIELD pilot deployment at NCSR lab

8.1.2. Presentation and stakeholders engagement

TID held an internal workshop on 7th March 2019, in their main premises (District T, Madrid) during a morning session (10h-13h CET). The workshop was presented by TID's personnel involved in SHIELD project (A. Pastor, J. Nuñez).

A total of 10 people attended to the workshop coming from different departments and business units not involved in SHIELD project. People were carefully selected, to cover different point of view:

- Digital security Telefonica Spain, in charge of developing internal ISP security services for Spain business units.
- Security Operational staff in Spain, that manage day by day security threats.
- GCTIO (Global Chief Technical and Information Office) department in charge of the network technical evolution at Telefonica worldwide, including techno-economics analysis.

The goal of the workshop was to present the SHIELD Project results and run a real exercise, based on data collection and demo setup executed in NCSR, to demonstrate the performance and main functionalities of the SHIELD framework. Additionally, the workshop included a general description of the project SHIELD and their main components, the standardization activities in progress and other demonstrations, such as attestation capacity from Trust Monitoring module.



Figure 28. ISP pilot workshop

8.1.3. Assessment and feedback

The last part of the workshop included a round of questions from the attendees and an open debate, as evaluation/assessment. As a summary, the feedback received from technical and operational staff was positive, especially in how we are able to combine different ML algorithms in DARE to improve the performance, or the selection of standards protocols such as Netflow or DNS, widely used by ISPs and supported by vendors, to obtain the data needed. The findings/recommendations can be summarized as follows:

- NFV technology is in process of adoption by Telefónica and its use by SHIELD for security solutions is very relevant and aligned to the company's commercial roadmap.
- Related to the use cases presented, Telefonica sees two main exploitation scenarios. First, for its own infrastructure protection (UC1). Second, SME security services through the deployment in the customer premises or inside its own network (UC2).
- Telefonica prefers a recommendation dashboard (using MSPL), instead of automatic response, especially from operational staff point of view. Present audience highlighted the capacity of SHIELD to offer alternatives instead of only one mitigation recipe (feature available in last version of SHIELD).
- A recommendation was made related to investigate how it can affect in the DARE analytics and detections when ISP works with "Netflow sampling". Netflow sampling is commonly applied by network nodes with high throughput, processing only one randomly selected packet out of n sequential packets to generate the flows statistics. Distributed collector (d-collector) developed in SHIELD was mentioned as an alternative when NFV technology is available.
- The DARE performance demonstrated with NCSR data was positively evaluated, especially if SHIELD focuses in customers' routers or if SHIELD supports netflow sampling.

- The capability of integrating with existing security appliances deployed by Telefonica could be a valuable feature. For example, with an anti-DDoS high performance appliance. The OSM community is in process of incorporating PNFs (physical Network Functions), so it can be adopted by vNSFO in the future.
- Another additional feature suggested is the integration in the DARE of information from external security feeds, also commonly known as IOC (Indicators of Compromise), to improve the detection accuracy with publicly available data.
- Standardization efforts done in SHIELD are very important for Telefonica to integrate vendors, including the DARE interfaces for security policy enforcement and remote attestation extension in virtualized environments.
- It would be interesting to extend the Trust Monitoring module to update and track “golden” measurements in the verifier, beyond the vNSF descriptor content, for instance using API interfaces, decoupling from the onboarding process.

8.2. Enterprise pilot – UC2

8.2.1. Data collection and pilot setup

The data used for the enterprise pilot were captured under real conditions from inside the SPH corporate network, naturally generated during the day-to-day operations of the company personnel (~300 employees). This fully corresponds to the TSS scenario, where the traffic of a medium-size enterprise is analysed –and responded to- in real time.

The setup of the SHIELD pilot deployment at SPH is depicted in Figure 29 below. Similarly to the NCSR D pilot deployment (see Sec. 8.1.1), the traffic information (in NetFlow v.9) format was captured from the company’s central firewall, which interconnects the company internal network with the DMZ and also to the Internet. The firewall was configured to send the traffic information in real time to a NetFlow collector vNSF, which in turn fed the DARE storage using the distributed collector pipeline.

Unlike the NCSR D case, no anonymization procedure was applied in the SPH pilot, as the traffic was analysed strictly within the company by the IT and security personnel, who are already authorised to inspect and analyse the company’s traffic.

The incident detection and classification results, as well as the mitigation proposal, was visualised in the SHIELD dashboard.

The pilot operated for one whole week (14/2/2019 – 21/2/2019), capturing and processing traffic without interruption, on 24/7 basis, in an average of ~300 new network flows per second.

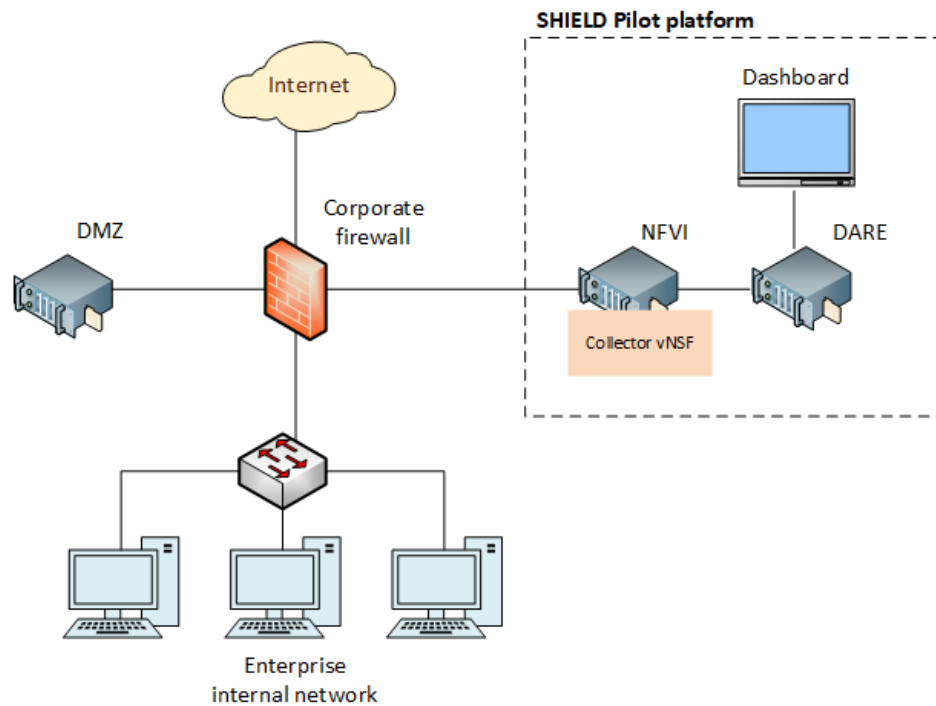


Figure 29. SHIELD pilot deployment at SPH corporate network

8.2.2. Presentation and stakeholders engagement

The internal pilot workshop was held on Monday, 4/3/2019, 09.30 – 11.00 EET. The meeting was organized in a dedicated meeting room and included the following:

- i) general introduction to the SHIELD project and discussion about its objectives,
- ii) technical description of the pilot setup,
- iii) overview of the detection results in the Dashboard,
- iv) hands-on session and
- v) Q&As and discussion.

The presentation was given by two persons from the SHIELD project team (G. Gardikis, A. Kapodistria), while the evaluation panel consisted of seven experts outside the project team, with no previous involvement in the project, including Security engineers, Big Data & Machine Learning engineers, as well as representatives from the company management, with special emphasis on commercialization aspects.

8.2.3. Assessment and feedback

The assessment and discussion was mostly focused on the DARE part and the incident detection capabilities.

From the technical point of view, it was observed that, indeed, many of the traffic flows which were detected by DARE as suspicious, diverted from the normal network behavior and were deserving further investigation (yet not necessarily corresponding to security incidents).

Overall, it was agreed that the DARE part consists a very useful tool, which can be exploited in the short term (only a few months time-to-market) and can assist network administrators and security engineers in their daily activities, always acting complementarily with traditional existing network security appliances (firewalls, IDS/IPSs, content filters etc.). Apart from the in-house deployment and the telco-driven (TSS) model, DARE service could be outsourced as a managed security service from the enterprise to a trusted security provider/data analyst. In this case, though, legal and GDPR issues need to be carefully taken into account.

The following specific recommendations for the further evolution of SHIELD were recorded:

- The processing time window (default was one day for the LDA algorithm) should be configurable. The system should be able to work with microbatches (especially for pre-trained algorithms) in order to improve the response time.
- The detection threshold should be configured and fine-tuned per customer during operation.
- Flows with zero byte count have been probably already dropped/filtered by the firewall, so they should be excluded from suspicious connects analysis.
- We should investigate cluster management solutions other than Cloudera – e.g. Ambari [14] in order not to depend on a commercial solution and overcome licensing issues.
- IP addresses should be explicitly marked as internal or external in the Dashboard view.
- The incident/suspicious flows view should have more capabilities of sorting and filtering out information.
- The output of the ML algorithms, in addition to be visualized in the Dashboard, can be fed to a third-party data indexing and analytics system, such as Splunk[15] or the Elastic stack[16]. This would allow better visualization, data aggregation and correlation from multiple sources (also outside SHIELD feeds) and also faster and more advanced querying.
- In an operational environment, depending on the deployment scenario, the integrity of the data collection and DARE modules should be also verified, in order to prevent attacks against the analytics subsystem.

It must be noted that items (3), (4) and (7) have been identified as of prime importance, and SPH personnel are already working on these extensions, beyond the duration and the scope of the project.

8.3. CERT pilot – UC3

8.3.1. Data collection and pilot setup

The data used for the CERT pilot was derived from the pilot deployment at NCSR, as described in Sec. 8.1.1.

8.3.2. Presentation and stakeholders engagement

The cybersecurity agency pilot of SHIELD has been carried out for CESICAT, the centre for Cybersecurity of the Catalan Government, in charge of the private network and applications of the Government.

i2CAT is the technical partner leading the Pilot while CESICAT evaluated the performance and features of the project, and provided feedback that can harmonise the outcomes with their needs, in order to better direct the project towards an envisioned real deployment in CESICAT facilities.

The pilot session has involved three i2CAT employees. Two technical persons that will presented the pilot, and one business development person that helped to identify possible opportunities for SHIELD as well as main needs from the stakeholder.

CESICAT was represented by 3 persons, the Director of Strategy, one member of the technical team (Security Operations Control - SOC) and the Head of the innovation office from the business perspective.

The pilot workshop had the following schedule:

- SHIELD Introduction – 20 min
- Architecture & components – 20 min
- SHIELD demos – 20 min
- SHIELD pilot – 15 min
- Round table & questions - 40 min
- Close & next – 5 min

8.3.3. Assessment and feedback

Several feedback was produced during the presentations, the piloting and the discussion. This feedback can be summarized in the following aspects:

8.3.3.1. Technical features

The trusted computing part is quite impressive, maybe we should consider to add the 0-trusted concept in SHIELD. In 0-trusted environments, you consider any device to be malicious by default, and you must consider protocols to trust the messages that they sent.

One main technical issue is the need to train with a dataset that represents the normal behaviour of the network. This involves considering that the training traffic, which corresponds to day-to-day operation, is clean (which is not the case on most of the networks and may lead to false negatives). An important feature will be the possibility that the expert behind the solution is able to mark a detected anomaly as normal, and so the solution does not show further alarms for that anomaly (reduce false positives). In the same way, a feature that allows the solution to retrain when an attack that was not detected by the solution is found to be a threat (false negative reduction) would be important.

In general we have to take into account that most of the failures in a Security Operations Control (SOC) are human errors, so we have to be tolerant to these and permit the undoing of operations.

8.3.3.2. Business features

In order to commercialize the solution, we need to show an operator use-case, so that we can show the added value of the solution. In addition, we should prepare a protocol to manage the change i.e. how the organization is going to manage the adoption of this solution across all the departments (specially operations and security).

Maybe we should focus the product as a solution for autonomous decision systems or even better, for automatization engineers (a role that does not currently exist in CERTs but that are completely necessary)

8.3.3.3. CERT specific features

The provided features with the dashboard are interesting. Nevertheless, we have to take into account that a security manager is basically a risk manager. Hence, we should be able to share information with the current software platforms that they use (i.e. Mitre) that basically use STIX as data sharing protocol. If we can use these protocols to share information that is able to be added in the risk management system, we will provide a very valuable information for CERTs. For example, taking into account metrics as the propagation time, damage of the attack or frequency of occurrence provide a risk factor indicator.

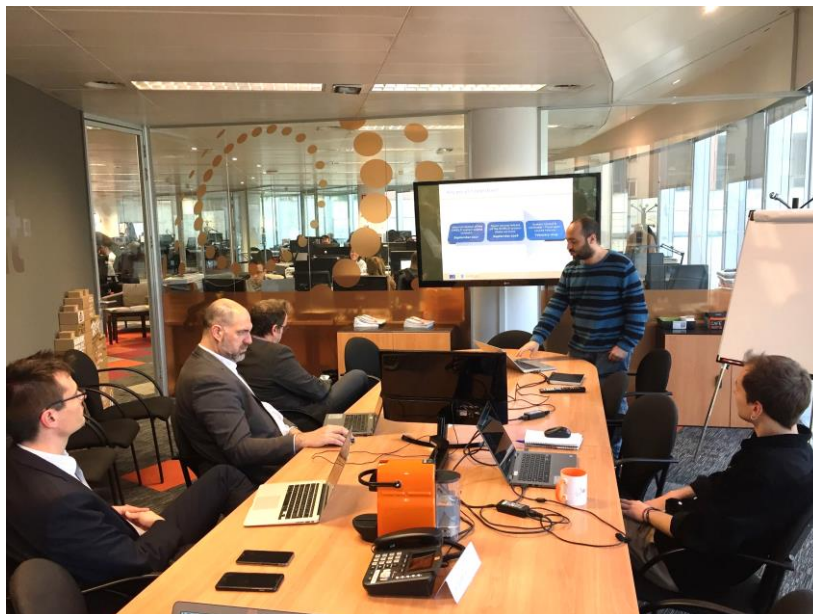


Figure 30. Pilot workshop with CESICAT

9. SHIELD TECHNOLOGY ROADMAP

9.1. Methodology

A technology roadmap (also known as TRM) can be thought of as a methodical approach to the analysis and projections of future developments and outcomes, leading to improved or new products or environments [12]

Most roadmapping approaches follow similar structured phases, that include collection of information, synthesis and validation or the final roadmap. There is no single, standardised approach to providing a TRM, since its creation is inherently needs-driven. A TRM is therefore created to serve a specific purpose and fulfil different needs. Key decisions that need to be addressed prior to the definition of a roadmapping methodology include:

- The definition of its purpose: A scientific roadmap might target to unearth and fill specific research gaps whereas a technical roadmap might target outcomes of higher TRL levels.
- The definition of a timeline: i.e. if a medium- or long-term TRM is needed.
- The definition of a starting/ending point: Exploratory approaches (Figure 31) try to foresee different future outcomes based on an extensive analysis of the current state. Normative roadmaps (Figure 32) explore possible known/desirable future outcomes and trace back to the current state.

According to JRC [12]:

“Exploratory methods are outward bound. They begin with the present as the starting point, and move forward to the future, either on the basis of extrapolating past trends or causal dynamics, or else by asking “what if?” questions about the implications of possible developments or events that may lie outside of these familiar trends.”

In SHIELD this would apply by analyzing SoA & Known gaps in SHIELD technical areas based on our experience, as well as on current trends to synthesize new use cases & developments for future research.

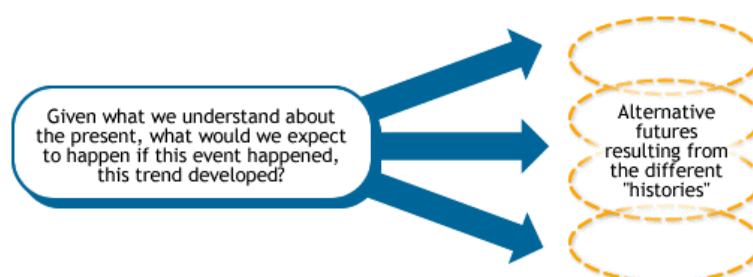


Figure 31. Exploratory roadmap overview.

On the other hand, normative approaches follow a different path [12]:

“Normative methods are, by contrast, inward bound. They start with a preliminary view of a possible (often a desirable) future or set of futures that are of particular interest. They then work backwards to see if and how these might futures might or might not grow out of the

present – not how they might be achieved, or avoided, given the existing constraints, resources and technologies.”

This would be an appropriate methodology for studies on how SHIELD can apply to different verticals, or future use cases (e.g. 5G, MEC, IoT etc.), by first analysing the desired future states and tracing back to how the SHIELD components (and platform as a whole) can be improved.

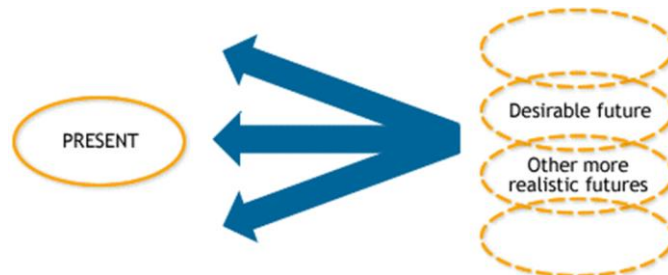


Figure 32. Normative roadmap overview.

Normative approaches are usually more effective when a common shared vision or common goal exists (e.g. when considering 5G) and when foresight helps construct a viable vision of the future. In other cases, however, normative approaches can be considered as inadequate and subjective, not allowing for a “bigger picture”. Hence, SHIELD has selected the exploratory approach, as it allows more flexibility with respect to future goals.

In any case, the roadmapping methodology needs to integrate the view of multiple stakeholders in order to be credible. Sufficient documentation and comprehensive analysis are cornerstones of the roadmapping process. As discussed in the following subsections, SHIELD applies the roadmapping methodology to analyse short-term to long-term plans for the further development of both individual products and SHIELD as a whole. This work complements the evolution of business and exploitation plans provided by WP6.

9.2. vNSF ecosystem evolutions

9.2.1. Lessons learned and gap analysis

9.2.1.1. NFVO

In the early phases of the development we assumed, and later learned, that developing a layer on top of the NFVO (OSM) was more adequate for us than directly modifying the component's code. There are some reasons for this:

1. Less complexity. A development from scratch and the limited set of features at such layer greatly simplifies the work.
2. Better maintainability. A smaller and less complex project requires less effort for further troubleshooting and addition of new features. Furthermore, and even more important for the project, separating the SHIELD-specific logic from that of the NFVO makes it considerably easier to upgrade the NFVO over time -- where we were concerned with the backwards compatibility they offer to the external developers rather than also merging our previous work into newer versions.

3. More modular. The new functionality is separated from the core of the service (here, the NFVO). Identifying where each feature or service is running is easier this way.
4. More tailored configuration. The development can benefit from its isolation due to the fact that specific SHIELD configuration or features can be directly integrated.

Therefore, the original plan of directly modifying the code of the NFVO was discarded and we provided an extra logic instead for SHIELD purposes. We noticed one possible drawback; as the new features are not directly available in the NFVO and any possible integration into the upstream code (i.e., for collaboration within the project) must be done as a separate work. Nevertheless that is not a blocker; given that it is feasible and even more adequate to cherry-pick specific changes in the code prior to contributing to the community.

On the other hand, some of the initial planned developments and integrations were not carried out in the end within the orchestration side:

- Verification of policies (MSPL) sent for the vNSF configuration -- as this is typically generated by the Trust Monitor, trusted and inside the SHIELD platform;
- Built-in NS/vNSF catalogue -- as that is already available in the store (keeping also SHIELD-specific information) and in the NFVO itself;
- Providing network topology information to the DARE; where such component would be aware of the topology and therefore would suggest placement of specific NSs at specific locations. This was changed and only the vNSFO and NFVO have idea of the locations -- as otherwise there would be duplication of information and logic across components whose function might be less related

Whilst others were incorporated later on:

- Secure communication with its clients -- as the HTTPS-based communication is expected for the different components in the project;
- Endpoints to interact with the Trust Monitor for manual or automatic registration of physical and virtual nodes (to add the physical nodes from the infrastructure and those virtual services added on-the-fly, respectively) -- as the vNSFO API is the one that intercepts the requests for instantiating the virtual nodes;
- VDU registration from the vNSFO API -- even though not required, it was understood as a nice-to-have feature to upload the VDU image from the same IP, prior to the vNSF instantiation
- Endpoints to provide granular information on the status of the attested physical & virtual nodes

9.2.1.2. vNSF Store

The Store module was designed to be agnostic to any type of Orchestrator, supporting multiple types of descriptors with respect to different types of Orchestrators. The underlying architecture of this component supports the onboarding of different types of descriptors by abstracting the validation and extraction of descriptors through a generic adaptor which routes the descriptor analysis to the specific Orchestrator processor. Given the time and resource constraints of the

project, a specific Orchestrator had to be chosen, namely the OSM. As a result, the gap of using the Store module with promiscuous packages for different types of Orchestrators was not filled.

9.2.1.3. Attestation framework

As soon as SHIELD started, the partners involved in the Trust Monitor identified a couple of technical risks associated with the underlying technologies used:

1. The mainstream open-source vTPM implementation was found out to be bugged, which rendered it unusable. Furthermore, the community supporting the implementation decided not to fix the issue. This meant that SHIELD would not be able to support the attestation of multiple VMs and the host platform, thus the consortium decided to focus on attestation container-based vNSFs.
2. OpenAttestation⁶ and OpenCIT⁷: SECURED project was based on OpenAttestation, which uses TPM1.2⁸. Early in SHIELD, the partners considered a migration to OpenCIT, which can be seen as the evolution of OpenAttestation and that relies on TPM2.0⁹. It turned out that OpenCIT and OpenAttestation were too different and the consortium could not migrate to OpenCIT.

Those two risks, related to open-source components outside of SHIELD, led to the following task being reduced:

1. Between the unavailability of a vTPM and the immaturity of container support in the NFVO, only prototype container-based vNSFs have been implemented, deployed and tested in SHIELD.
2. SHIELD stuck to using TPM1.2 instead of TPM2.0 for the testbeds, the demonstrations and the pilots. Some design and proof-of-concept work has been carried out with TPM2.0.

9.2.1.4. NSs and vNSFs

The encapsulation of NSs and vNSFs in SHIELD packages allows additional information to be provided to the Store module, the entity that is responsible to onboard the “striped” packages to the appropriate Orchestrator. As a consequence of the gap identified in the previous Store section, the possibility of having packages driven to multiple Orchestrator types was not implemented.

9.2.2. Future activities and required resources

9.2.2.1. NFVO

Possible improvements on the NFVO and network side relate to:

⁶ <https://01.org/openattestation>

⁷ <https://01.org/opencit>

⁸ <https://trustedcomputinggroup.org/resource/tpm-main-specification/>

⁹ <https://trustedcomputinggroup.org/resource/tpm-library-specification/>

- The integration of the service chaining as implemented in the NFVO (OSMr4)
- The network slice features implemented in newer versions (OSMr5)

Each of the improvements might take around 2 months of work (2 PMs), considering the proper configuration of the infrastructure and the integration of such features with the vNSFO API.

On the security side, for NFVO, some other improvements are:

- Provide basic auth security for the northbound API; if requested from a client outside the SHIELD platform
- Implement authentication through certificates for the northbound API, when used internally to the platform (that is, communicating with other SHIELD components)

These efforts require days and weeks, respectively.

Finally, some other enhancements relate to providing more granular data on the instantiation and configuration of NSs, filtering of specific data and other modifications of the northbound API to make it even more useful by external clients. That kind of work typically would take days up to a week.

9.2.2.2. Store

Regarding the current developed billing model, it can be considered rather simplistic as it may come insufficient to deal with complex rating contracts. For instance, the flat-rate model limits the possibility of specifying different rates to date intervals or even provide benefits associated with usage volumes. To develop this feature, a comprehensive study of the market would be mandatory and would also require the design and implementation of a complex model. For the analysis and implementation of this project it is estimated 4 months of work (6 PMs).

9.2.2.3. Attestation framework evolutions

Given the gap in a vTPM implementation and the new features of TPM2.0 versus TPM1.2, POLITO and HPELB identified the need for a vTPM2.0. Although it was not planned in SHIELD, both partners started a collaboration on researching and designing a secure vTPM2.0. Given the complexity of the new TPM2.0 features, this task is estimated to require at least one year of development, with an estimated effort of 18 PMs.

9.2.2.4. NSs and vNSFs

Concerning the significant effort related to the development of NSs and respective vNSFs, a modeling tool to perform the implementation and offline testing of these descriptors would ease and expedite the whole process. Therefore, a descriptor composer tool, such as a Service Development Kit (SDK) to help the development of NSs and vNSFs, would be an important addition to the project. To implement this module at least 3 months (2 PMs) of work are estimated.

9.3. DARE evolutions

9.3.1. Lessons learned and gap analysis

The development of a successful, information-driven cybersecurity engine that aims at providing Security-as-a-Service functionalities even at ISP-level workloads, requires a holistic mindset shift from traditional IDS/IDPS systems, not only in terms of architectural design, but also in terms of hardware infrastructure and specifications. The DARE, as one of the three central innovation pillars of SHIELD along with the vNSF ecosystem and the hardware attestation, materialises this shift by combining state-of-the-art security analytics techniques, in order to produce useful inference and exchange that information with all the other components of the solution.

When properly leveraged, security analytics can offer a significant boost in securing network environments, usually higher than most legacy/local services can currently provide. A framework that combines a set of such services, should ideally be designed with scalability and fault-tolerance in mind, while also taking efficiency and real-time performance into consideration, terms which can often be contradictory. It is thus evident that many important decisions had to be taken early in the development process in order to ensure that the optimal trade-off between these contrasting notions would be selected. From a technical perspective, the following decisions played a major role in the development process:

Multi-layer architecture: An analytics-based cybersecurity engine that handles network traffic should be considered as an integrated system. This system has layers, components, interfaces and interactions, which can be thought as interconnected parts of a unified central design. To this end, the DARE has implemented a 3-layer architectural approach, starting from data acquisition to data analysis and finally to threat remediation. As data is passed from one layer to another, it is being further processed and enriched, ultimately leading to the accumulation of the necessary information to mitigate a network threat.

Distributed computing: The aforementioned multi-layer design facilitates the implementation of a multi-node system as well, with each node serving a discrete role. This configuration is indeed considered as the best practice for every computationally intensive application, since the processing needs for cybersecurity analytics at scale impose the use of distributed computing architectures that can balance the workload, while adding to the fault-tolerance of the overall infrastructure. The DARE has exploited the functionalities of an open-source distributed computing framework (Apache Spark) to be able to train machine-learning and deep-learning models using large network datasets, without suffering from the shortcomings of other popular frameworks (e.g. out-of-memory errors, limitations in workload balancing, latency in disk I/O).

Data availability: One of the most crucial decisions of every information-driven solution is related to the way it stores and transfers the data to the processing stage(s). Data needs to be considered in all its elemental forms (at rest, in motion and in use) and must be available at all times. That implies that, although a distributed architecture approach has been implemented, each node of the DARE should have access to all of the accumulated traffic, otherwise any inference produced by the analytics components will be limited by the latent information that resides only in the portion of data provided to it, severely reducing the efficiency of the

implemented machine-learning and deep-learning models. To solve this issue, the DARE has exploited a popular distributed filesystem (HDFS) that is designed to simulate centralised storage by providing high throughput access to application data and is suitable for applications that process large datasets.

Cluster configuration: While it is possible to install and configure all the prerequisite services related to storage, communication and processing of data to each node individually, this can be achieved far more easily and more efficiently with the help of a centralized interface. In this context, CDH (Cloudera Distribution for Hadoop) was leveraged as a bundle of open-source services, specifically optimised for end-to-end Big Data workflows. CDH can be considered as a superset of the DARE's prerequisite software specifications, as it includes a set of storage, processing, analytics, load-balancing and messaging features that can be configured and controlled with the help of a convenient web interface.

Efficiency of cybersecurity analytics: The analytics stage of the DARE leverages two different ML-based modules (the Cognitive DA module and the Security DA module) as a means of maximising the detection efficiency of the SHIELD solution. In each module, a two-stage approach has been implemented, following the basic principles of anomaly detection and threat classification. During the last two years, we experimented with a large number of different machine-learning and deep-learning models, evaluating them in terms of detection efficiency, scalability, speed and ease of training. We derived that it is possible to create an analytics-based solution that offers detection capabilities equivalent of or even superior to traditional rule-based IDSs, however not without any limitations. These limitations stem mainly from the fact that there is a lack of publicly available, representative network traffic which is necessary for training our models. This, along with the fundamental limitations inherent in machine learning techniques -which are mainly probabilistic in nature and thus can be prone to false positives/negatives under certain conditions- led us to the conclusion that a hybrid solution, combining rule-based and analytics-based detection can yield the optimal results. This approach is indeed followed by SHIELD, as the NS Store offers a set of rule-based monitoring and actuating vNSFs to enhance the security of the protected infrastructure.

9.3.2. Future activities and required resources

The DARE is a complex analytics engine, offering a multitude of different services for accumulating network traffic, predicting malicious attacks and mitigating detected threats. It implements a multi-layer cybersecurity solution based on Big Data infrastructure and scalable analytics. The emergence of these technologies as core business disruptors in the digital age has led to the existence of a plethora of different approaches, models and tools, which we could not have possibly exhausted over the 30-month span of this project. However, there are activities for improvement that we would prioritise over others:

- Both the Cognitive DA module and the Security DA module of the DARE support the netflow traffic protocol (with some early models of the former also supporting packet captures). This was a conscious decision, since netflow provides a metadata-based view of the network activity which is far more manageable in terms of storage compared to full packet capture, which continuously records a complete record of all network activity, including the actual data (packet payload). However, packet capture is encapsulating information that netflow is lacking. Therefore, analysing actual packets

may offer a significant improvement in the efficiency of our analytics models, which can be converted to handle this type of traffic relatively easily. This improvement would not require more than 5 months of work (5 PMs).

- The aforementioned DARE modules are currently providing their findings in an aggregated manner, which is visualised via the Dashboard. A potential improvement would be to merge these findings to single combined result, based on the confidence level of each module (which can be also communicated in the same dashboard view). This work should be feasible, given that both modules exploit similar analytics techniques and would require approximately 6 months of work (6 PMs).
- In its current form, the DARE relies on the iterative analysis of network logs which occurs automatically in time intervals. Recent advancements in data accumulation and processing have enabled the further exploitation of data streaming techniques, allowing the analysis of each network log individually (almost upon arrival), thus significantly reducing detection times. The current underlying framework supports this improvement and it is estimated that it would require 12 months of work (12 PMs) to implement it.

9.4. System-wide evolutions

The exploitation of SHIELD (as a whole) requires its evolution to fit the needs of the current market towards 2030. For example, the IHS Cyber Security Market study [18] for the Europe, Middle East and Africa (EMEA) region for 2016-2020 reports that although standalone products hold the biggest share in the current market, Managed Security Services (MSS) in Europe show the most growth with an 8.8% CAGR (growing from \$9.2 Billion in 2015 to \$14.2 Billion in 2020), which is an appropriate model for the evolution and exploitation of SHIELD as a whole. On the technical side, it is important to predict what kind of system-wide evolutions will help SHIELD reach this potential.

As also defined in WP6 deliverable D6.2, Ovum [19] and Forrester [20] have defined the important characteristics of Managed Security Services as:

- **protecting the business, instead of solving network point of view problem, adding consulting, analytics, data science, threat hunting, incidents response, and remediation:** SHIELD can focus on improving the performance and scalability of current components (data science, incidents response, remediation) and develop additional mechanisms to improve billing and provide mechanisms that monitor for service-level agreement (SLA) breaches. This would be an important aspect towards protecting businesses.
- **featuring autonomic cyberhealth through integration and automatic orchestration of customers' security tools:** A future evolution would enable the DARE to work together with customers' existing cybersecurity tools. This requires the use of common APIs or data formats that can easily be ingested by DARE. Another future development would be to automate the instantiation/placement of cybersecurity services with minimal input from the human operator. Proper interfaces should still be developed, so as to keep the dashboard user properly informed in a highly dynamic service ecosystem.
- **Bear weight in prevention and detection solutions rather than focus solely in mitigation with special interest in the analytics:** SHIELD takes into account preventative/mitigation

measures on the level of the network, as cybersecurity vNSFs. One such case was also demonstrated with the IDPS where detection and mitigation of the malicious script was instantaneous. Further work could focus on enhancing the performance and scalability of SHIELD services. Dedicated rulesets for SHIELD vNSFs (e.g. IDPS, DPI, firewalls etc.) can be developed and even monetised (per client, per vertical etc.).

- **use augmented technology as a complement to technical staff expert in cybersecurity:** SHIELD uses powerful Machine Learning algorithms in its DARE, that show how advanced technologies can complement the work of a dedicated technical staff. Additional work should provide a better understanding of the limitations of ML (with respect to false positive/negative detections), the improvement of its detection capabilities as well as the study of ML behaviours under adversarial conditions (e.g. when a cybercriminal is attacking the ML itself).

Gartner¹⁰ has gone further to define the integrated threat intelligence and response capabilities in a single flow, as SOAR (Security Orchestration, Automation and Response). According to Gartner, the **share of organizations with security teams larger than five people** will turn to integrated SOAR frameworks rather than individual products, for orchestration and automation. Gartner states that “most of the drivers have existed for as long as enterprise and government SOCs have existed — for decades, not years. However, SOAR tools only appeared in mid-2010s” and estimates that the market share for SOAR will rise from 1% to 15% by 2020. It also states that “as the security skills shortage persists, alert numbers and attack vectors grow, and product proliferation continues, more complex organizations will consider SOAR solutions to unlock the full potential of both their analysts and security product suite”.

In order to evolve SHIELD towards a SOAR solution, it is necessary not only to consider performance and scalability (also addressed in D2.2), but also focus on usability and the development of the user interfaces to combat gaps in emergency response that are related to the quality and quantity of information that reaches their security teams. In the case of a SHIELD evolution towards SOAR, the dashboard as well as the individual DARE and vNSF user interfaces need to be improved to provide concise information and combat “alert fatigue”. Based on ENISA’s report on “Exploring the opportunities and limitations of current Threat Intelligence Platforms”[21] some of the limitations of current systems that SHIELD should overcome are:

- **Time-to-live for shared intelligence:** Apart from UI improvements to enhance the responders’ capacity to handle a cyber incident, there is a need to report time-to-live information for shared intelligence. This added feature will help responders prioritise remediation based on the time window as well as the severity of the incident.
- **Limited technology enablement in threat triage and relevancy determination:** SHIELD’s classification algorithms and security analytics perform a basic “triage” in the sense that they assign a severity level to the detected threats. This work should be prioritised to ensure that the information that reaches the dashboard is critical, actionable and relevant.
- **Diverse data models and formats used:** Multiple standards are being currently used to share threat information, like STIX 1.x/2.x, MISP JSON and others, although there are

¹⁰ Anton Chuvakin, Augusto Barros, “Preparing your security operations for Orchestration and Automation tools”, Gartner, February 2018.

concerns that they are being underutilized and threat information are not properly conveyed. SHIELD has adopted MSPL/HSPL as it enabled the project to include information fields not available in other formats (e.g. the type of vNSF that should handle a remediation recommendation). While this functionality was important for SHIELD, it can still improve its support for well-known standards.

10. CONCLUSIONS

D5.2 marks the completion of the technical work of the SHIELD project. Judging from the feedback received from all the external experts who were engaged in the final evaluation and assessment activities, it can be deduced that SHIELD produced quite impactful results, and realised a definitive step towards next-generation managed security services, particularly suited for software-based networks (including 5G infrastructures).

The fact that most project deliverables were publicly released and most software was open-sources further contributes to reinforce the impact of SHIELD in the cybersecurity landscape. It is worth to mention that 5G Americas, the leading 5G industry coalition for the whole American continent, in its February 2019 white paper (“The status of open-source for 5G”)¹¹ lists SHIELD as the only open-source cybersecurity framework currently available, tailored for 5G networks.

Following the official ending of the project, most SHIELD partners already continue the technical work, either on external or even own funding, towards further enhancing the project results and exploiting the SHIELD components in next-generation cybersecurity solutions and services.

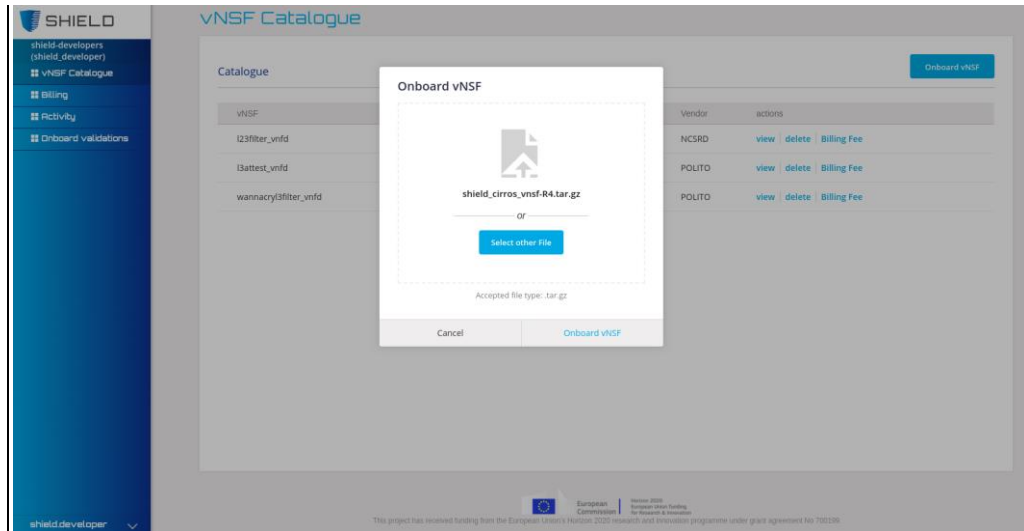
¹¹

http://www.5gamericas.org/files/9815/5189/9562/5G_Americas_White_Paper_The_Status_of_Open_Source_for_5G_Feb_2019.pdf

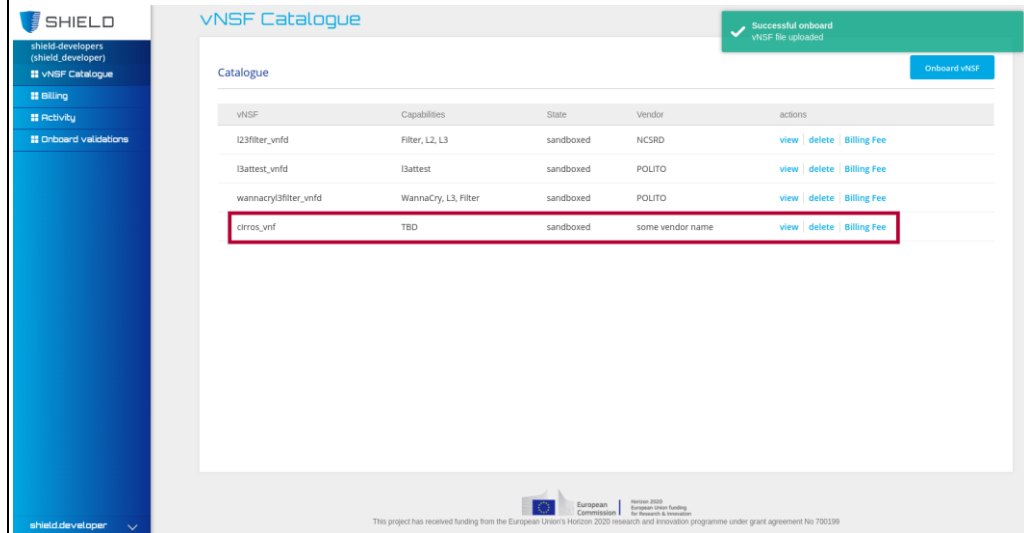
11. ANNEX A: SHIELD TEST CASES AND RESULTS

11.1. Platform tests

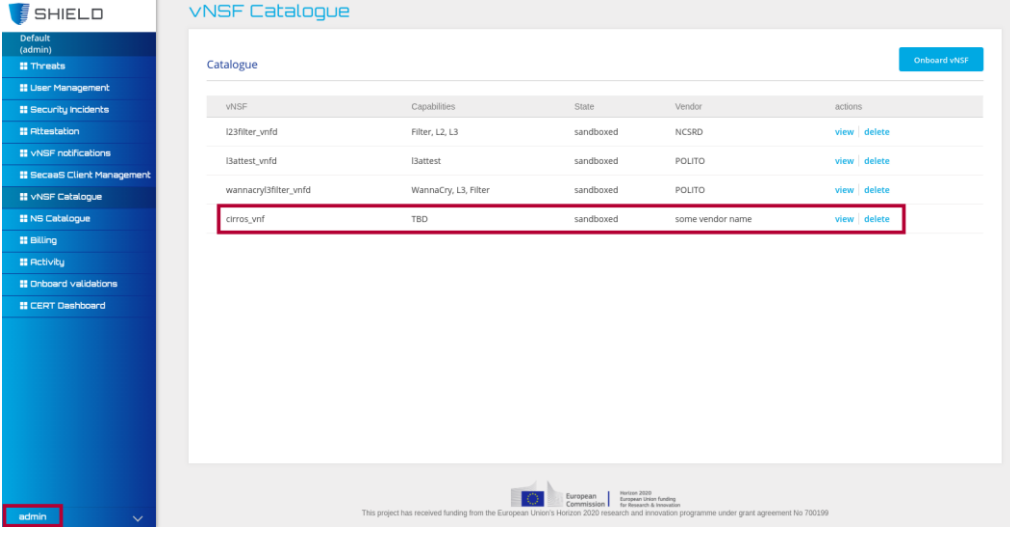
Test Case ID	TC_P01			
Description	NS/vNSF onboarding (image& descriptor)			
Executed by	UBI	Date	10/3/2019	
Purpose	Establish the ability of the SHIELD platform to allow and enable users under the “developer” role to submit and onboard vNSF packages.			
Associated Requirements	PF01, PF02, PF08, PF15			
Components involved	Dashboard, Store, NFVO			
Tools	None			
Metrics	None			
Pre-test conditions	SHIELD packages are generated The VM images used by the onboarded vNSFs are registered and stored in the VIM.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	As a Developer, onboard a vNSF package using the Dashboard interface	
	2	Check	The package is correctly onboarded and it becomes available in the “NS Catalogue” section	OK
	3	Stimulus	As the SHIELD platform administrator, verify if I have access to the onboarded vNSF	
	4	Check	The package is available in the “NS Catalogue” section of the platform administrator.	OK
Evidence	#1 Onboard a vNSF package			



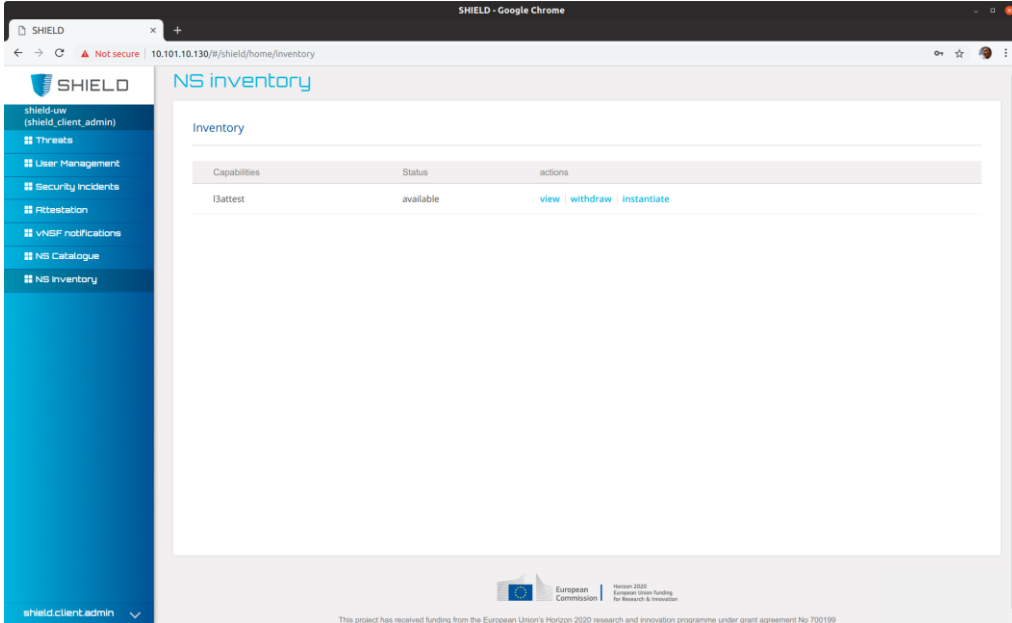
#2 Check that the package is available from the NS Catalogue



#3 As platform administrator verify that the onboarded vNSF is available from the NS Catalogue

	
Verdict	Success
Comments	<p>The onboarding process of a SHIELD package involves the opening and validation of the SHIELD package itself as well as the validation of the underlying orchestrator package (in this case OSM), and finally the submission of the nested orchestrator package to the orchestrator itself. These tasks are mainly handled by the Store component which also communicates with other components to complete all these tasks. A slight delay was observed but it is negligible and does not affect the user experience.</p>

Test Case ID	TC_P02		
Description	NS/vNSF control and lifecycle management		
Executed by	I2CAT	Date	20/2/2019
Purpose	The user-facing components of the SHIELD platform are capable of providing the basic management operations so that a developer can upload its service to the platform and an operator can deploy or remove it, acting as a response to mitigate a threat in the network.		
Associated Requirements	PF02, PF03, PF15, NF05, NF06		
Components involved	NFVO, Store, Dashboard		
Tools	None		
Metrics	<p>Response time</p> <p>Impact on user traffic (additional delay by vNSF)</p>		

Pre-test conditions	<p>Packages are generated for the correct version of OSM (here, R4).</p> <p>Packages are wrapped for SHIELD.</p> <p>The vNSF images are properly registered in the VIM.</p>																														
Test Sequence	<table border="1"> <thead> <tr> <th>Step</th> <th>Type</th> <th>Description</th> <th>Result</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Stimulus</td> <td>Onboard vNSF and NS packages from the Store</td> <td></td> </tr> <tr> <td>2</td> <td>Check</td> <td>Both packages are correctly onboarded and are available for instantiation via the Dashboard</td> <td>OK</td> </tr> <tr> <td>3</td> <td>Stimulus</td> <td>Instantiate NS from the Dashboard as a reaction to the perceived threat by the DARE</td> <td></td> </tr> <tr> <td>4</td> <td>Check</td> <td>NS is properly instantiated in OSM, vNSFs run as expected in the VIM, vNSFs are configured as expected by the MSPL</td> <td>OK</td> </tr> <tr> <td>5</td> <td>Stimulus</td> <td>Remove NS via the Dashboard</td> <td></td> </tr> <tr> <td>6</td> <td>Check</td> <td>NS and corresponding VMs are deleted from OSM and the VIM</td> <td>OK</td> </tr> </tbody> </table>	Step	Type	Description	Result	1	Stimulus	Onboard vNSF and NS packages from the Store		2	Check	Both packages are correctly onboarded and are available for instantiation via the Dashboard	OK	3	Stimulus	Instantiate NS from the Dashboard as a reaction to the perceived threat by the DARE		4	Check	NS is properly instantiated in OSM, vNSFs run as expected in the VIM, vNSFs are configured as expected by the MSPL	OK	5	Stimulus	Remove NS via the Dashboard		6	Check	NS and corresponding VMs are deleted from OSM and the VIM	OK		
Step	Type	Description	Result																												
1	Stimulus	Onboard vNSF and NS packages from the Store																													
2	Check	Both packages are correctly onboarded and are available for instantiation via the Dashboard	OK																												
3	Stimulus	Instantiate NS from the Dashboard as a reaction to the perceived threat by the DARE																													
4	Check	NS is properly instantiated in OSM, vNSFs run as expected in the VIM, vNSFs are configured as expected by the MSPL	OK																												
5	Stimulus	Remove NS via the Dashboard																													
6	Check	NS and corresponding VMs are deleted from OSM and the VIM	OK																												
Evidence	<p>#1 Enroll a Network Service in the “NS Inventory” view (as shield.client.admin)</p>  <p>#2 Onboard the vNSF and NS from the “vNSF Catalogue” and “NS Catalogue”, respectively (as admin)</p>																														

SHIELD vNSF Catalogue

Successful onboard vNSF file uploaded

Capabilities	State	Vendor	Actions
Filter, L2, L3	sandboxed	NCSR	view delete
l3attnst	sandboxed	POLITO	view delete

admin

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

#3 Instantiate the NS from the “NS Inventory” view (as shield.client.admin)

SHIELD NS Inventory

Capabilities	Status	actions
Filter, L3	available	view withdraw instantiate
l3attnst	configuring	view

shield.client.admin

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

#4 Verify that the NS is instantiated at the NFVO (OSMr4)

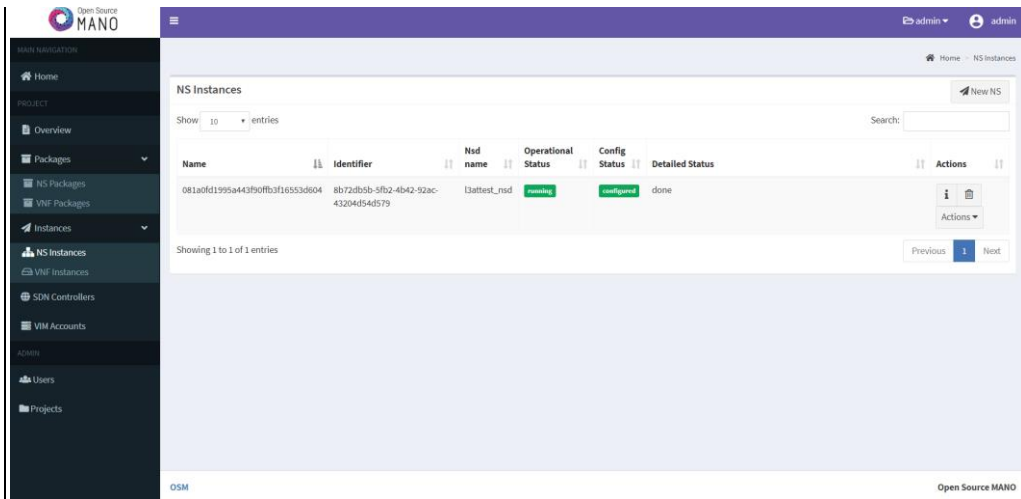
Open Source MANO NS Instances

Name	Identifier	Nsd name	Operational Status	Config Status	Detailed Status	Actions
081a0fd1995a443f90fb3f16553d604	8b72db5b-5fb2-4b42-92ac-43204d54d579	l3attnst_nsd	OK	OK	Waiting ns ready at RO. RO_id=678f2782-4209-48bb-a662-14eb3de93d49; VMs: 0/1, networks: 0/3	i trash Actions

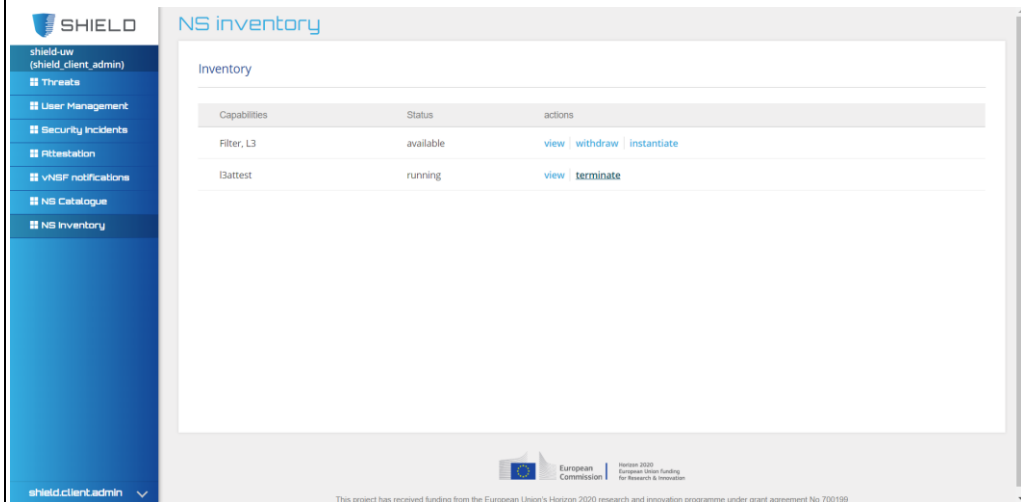
Showing 1 to 1 of 1 entries

OSM

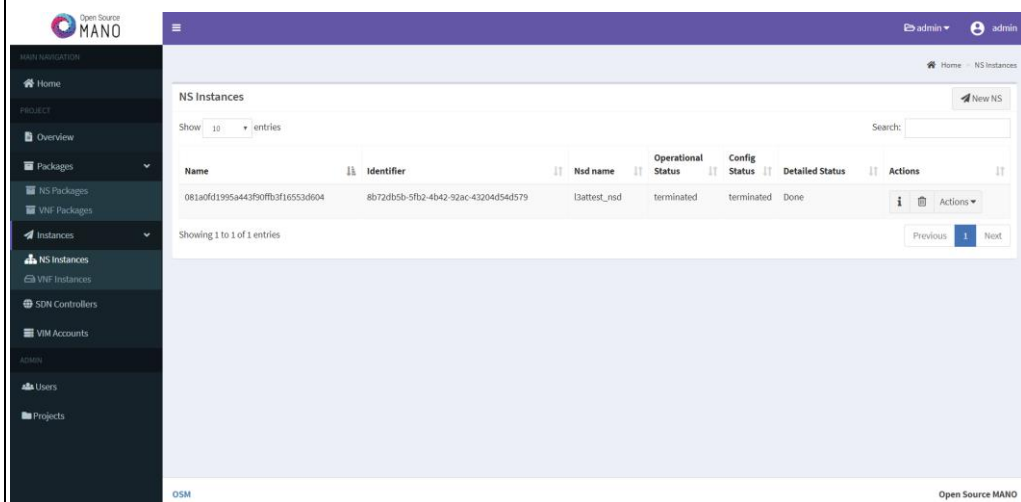
Open Source MANO



#5 Terminate the NS from the “NS Inventory” view (as shield.client.admin)



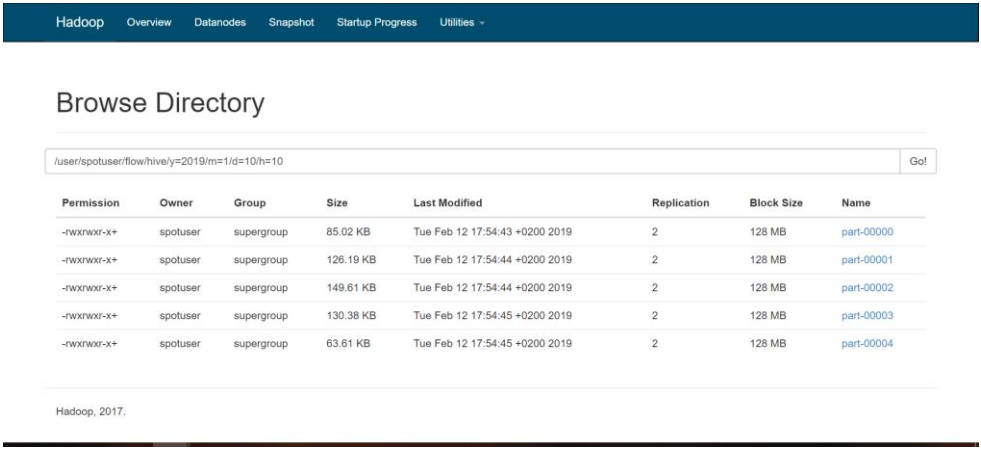
#6 Verify that the NS is terminated at the NFVO (OSMr4)



Verdict	Success
Comments	The time taken for onboarding does introduce a slight delay to the end-user when comparing to the onboarding time directly to the NFVO. This is expected, as SHIELD-


	specific operations are performed during the onboarding and different endpoints must be contacted. Such delay is minimum and does not affect the user experience.
--	---

Test Case ID	TC_P03			
Description	Security data monitoring			
Executed by	SPH	Date	12/2/2019	
Purpose	To verify the proper collection and ingestion of network data to the DARE.			
Associated Requirements	PF04, NF01, NF07, NF02			
Components involved	vNSF (flow collector), DARE			
Tools	Tcpreplay (https://tcpreplay.appneta.com/)			
Metrics	Delay from capture to storage			
Pre-test conditions	The flow collector vNSF has been deployed and the user traffic is directed through it. The DARE is up and running and ingest engine initialised.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Activate the distributed collector (d-collector) in the vNSF	
	2	Stimulus	Replay a previously captured traffic dump using tcpreplay, via the vNSF	
	3	Check	Verify that the traffic flows have been properly recorded in the Hive table in the DARE	Success Flow data were communicated and recorded within ~15 sec from the replay time

Evidence	<p>Collector initialisation and operation:</p> <pre> 2019-02-12 17:53:42,373 INFO SHIELD.DC.COLLECTOR Initializing Distributed Collector process... 2019-02-12 17:53:42,373 INFO SHIELD.DC.FILE_WATCHER Schedule watching "/home/spotuser/incubator-spot/traffic/flow" directory. 2019-02-12 17:53:42,374 INFO SHIELD.DC.FILE_WATCHER Supported filenames: "nfcapd.*" 2019-02-12 17:53:42,374 INFO SHIELD.DC.FILE_WATCHER The search in sub-directories is enabled. 2019-02-12 17:53:42,374 INFO SHIELD.DC.COLLECTOR Use directory "/tmp/_DC.SgIM7E" as local staging area. 2019-02-12 17:53:42,380 INFO SHIELD.DC.COLLECTOR Master Collector will use 4 parallel processes. 2019-02-12 17:53:42,380 INFO SHIELD.DC.COLLECTOR Initialization completed successfully! 2019-02-12 17:53:42,380 INFO SHIELD.DC.COLLECTOR Start "Flow" Collector! 2019-02-12 17:53:42,382 INFO SHIELD.DC.COLLECTOR Signal the "FileWatcher(Thread-1)" thread to start. 2019-02-12 17:53:58,749 INFO SHIELD.DC.FILE_WATCHER ----- New File Detected! ----- 2019-02-12 17:53:58,749 INFO SHIELD.DC.FILE_WATCHER File "/home/spotuser/incubator- spot/traffic/flow/nfcapd.201901101046" added to the queue. </pre> <p>Overview of data in Hadoop:</p>  <table border="1" data-bbox="384 1328 1321 1496"> <thead> <tr> <th>Permission</th> <th>Owner</th> <th>Group</th> <th>Size</th> <th>Last Modified</th> <th>Replication</th> <th>Block Size</th> <th>Name</th> </tr> </thead> <tbody> <tr> <td>-wxrwxr-x+</td> <td>spotuser</td> <td>supergroup</td> <td>85.02 KB</td> <td>Tue Feb 12 17:54:43 +0200 2019</td> <td>2</td> <td>128 MB</td> <td>part-00000</td> </tr> <tr> <td>-wxrwxr-x+</td> <td>spotuser</td> <td>supergroup</td> <td>126.19 KB</td> <td>Tue Feb 12 17:54:44 +0200 2019</td> <td>2</td> <td>128 MB</td> <td>part-00001</td> </tr> <tr> <td>-wxrwxr-x+</td> <td>spotuser</td> <td>supergroup</td> <td>149.61 KB</td> <td>Tue Feb 12 17:54:44 +0200 2019</td> <td>2</td> <td>128 MB</td> <td>part-00002</td> </tr> <tr> <td>-wxrwxr-x+</td> <td>spotuser</td> <td>supergroup</td> <td>130.38 KB</td> <td>Tue Feb 12 17:54:45 +0200 2019</td> <td>2</td> <td>128 MB</td> <td>part-00003</td> </tr> <tr> <td>-wxrwxr-x+</td> <td>spotuser</td> <td>supergroup</td> <td>63.61 KB</td> <td>Tue Feb 12 17:54:45 +0200 2019</td> <td>2</td> <td>128 MB</td> <td>part-00004</td> </tr> </tbody> </table>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	-wxrwxr-x+	spotuser	supergroup	85.02 KB	Tue Feb 12 17:54:43 +0200 2019	2	128 MB	part-00000	-wxrwxr-x+	spotuser	supergroup	126.19 KB	Tue Feb 12 17:54:44 +0200 2019	2	128 MB	part-00001	-wxrwxr-x+	spotuser	supergroup	149.61 KB	Tue Feb 12 17:54:44 +0200 2019	2	128 MB	part-00002	-wxrwxr-x+	spotuser	supergroup	130.38 KB	Tue Feb 12 17:54:45 +0200 2019	2	128 MB	part-00003	-wxrwxr-x+	spotuser	supergroup	63.61 KB	Tue Feb 12 17:54:45 +0200 2019	2	128 MB	part-00004
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name																																										
-wxrwxr-x+	spotuser	supergroup	85.02 KB	Tue Feb 12 17:54:43 +0200 2019	2	128 MB	part-00000																																										
-wxrwxr-x+	spotuser	supergroup	126.19 KB	Tue Feb 12 17:54:44 +0200 2019	2	128 MB	part-00001																																										
-wxrwxr-x+	spotuser	supergroup	149.61 KB	Tue Feb 12 17:54:44 +0200 2019	2	128 MB	part-00002																																										
-wxrwxr-x+	spotuser	supergroup	130.38 KB	Tue Feb 12 17:54:45 +0200 2019	2	128 MB	part-00003																																										
-wxrwxr-x+	spotuser	supergroup	63.61 KB	Tue Feb 12 17:54:45 +0200 2019	2	128 MB	part-00004																																										
Verdict	Success																																																
Comments	-																																																

Test Case ID	TC_P04
Description	Security incident detection, classification and visualization.

Executed by	INFILI	Date	20/2/2019	
Purpose	The DARE is capable of identifying network anomalies, classifying them as specific threats and communicating with the Dashboard to report that issue to the user.			
Associated Requirements	PF05, NF01			
Components involved	vDPI, DARE (Cognitive DA module), Dashboard			
Tools	-			
Metrics	Cognitive DA module (two-stage ML) performance Detection speed			
Pre-test conditions	Packet capture (.pcap) containing wannacry worm acquired from publicly available sources.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	A packet capture (.pcap) containing worm traffic is being replayed, simulating a wannacry attack.	
	2	Check	This traffic is being converted to netflow logs and is being ingested by the DARE.	The vDPI displays relevant network activity.
	3	Stimulus	The anomaly detection procedure of the Cognitive DA module is initialized.	
	4	Check	DARE logs are being displayed, as the ML procedure analyses the ingested logs.	The anomaly detection procedure reports a number of malicious logs.
	5	Stimulus	The threat classification procedure of the Cognitive DA module, analyses the malicious logs to assign threat labels.	
	6	Check	DARE logs are being displayed, as the ML procedure analyses the malicious logs.	The threat classification procedure reports a number of flows with the label "wannacry"
	7	Stimulus	The reported threats are being forwarded to the dashboard via the RabbitMQ message broker.	
	8	Check	The Dashboard receives the reported threats.	The threats are displayed on the Dashboard and a

		mitigation action is proposed.
Evidence	<p>1. The vDPI displays relevant network activity.</p>  <p>2. The anomaly detection procedure reports a number of malicious logs.</p> <pre> spotuser@ClusterOrchestrator:~ - 168x43 spotuser@ClusterOrchestrator:~\$./sim_anom_class_pub.sh 20180918 Starting anomaly detection for: 20180918 18/10/19 08:31:00 INFO client.RMPProxy: Connecting to ResourceManager at ClusterOrchestrator/172.27.0.139:8032 18/10/19 08:31:00 INFO yarn.Client: Requesting a new application from cluster with 3 NodeManagers 18/10/19 08:31:00 INFO yarn.Client: Verifying our application has not requested more than the maximum memory capability of the cluster (22347 MB per container) 18/10/19 08:31:00 INFO yarn.Client: Will allocate AM container, with 11264 MB memory including 1024 MB overhead 18/10/19 08:31:00 INFO yarn.Client: Setting up container launch context for our AM 18/10/19 08:31:00 INFO yarn.Client: Setting up the launch environment for our AM container 18/10/19 08:31:00 INFO yarn.Client: Preparing resources for our AM container 18/10/19 08:31:01 WARN yarn.Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME. 18/10/19 08:31:03 INFO yarn.Client: Uploading resource file:/tmp/spark-0f5678a4-876e-4066-9102-aaecb7884790/_spark_libs_127556829986290572.zip -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/_spark_libs_127556829986290572.zip 18/10/19 08:31:05 INFO yarn.Client: Uploading resource file:/home/spotuser/demo_2018/ShieldDL/dependencies/zoo/lib/analytcs-zoo-bigdl_0.6.0-spark_2.2.0-0.2.0-jar-with-dependencies.jar -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/analytcs-zoo-bigdl_0.6.0-spark_2.2.0-0.2.0-jar-with-dependencies.jar 18/10/19 08:31:07 INFO yarn.Client: Uploading resource file:/home/spotuser/demo_2018/ShieldDL/dependencies/zoo/bin/venv.zip -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/venv.zip 18/10/19 08:31:09 INFO yarn.Client: Uploading resource file:/home/spotuser/demo_2018/ShieldDL/deep_learning_zoo.py -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/deep_learning_zoo.py 18/10/19 08:31:09 INFO yarn.Client: Uploading resource file:/home/spotuser/demo_2018/ShieldDL/dependencies/zoo/lib/analytcs-zoo-bigdl_0.6.0-spark_2.2.0-0.2.0-python-api.zip -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/analytcs-zoo-bigdl_0.6.0-spark_2.2.0-0.2.0-python-api.zip 18/10/19 08:31:09 INFO yarn.Client: Uploading resource file:/home/spotuser/demo_2018/ShieldDL/transformers_deep_learning.py -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/transformers_deep_learning.py 18/10/19 08:31:09 INFO yarn.Client: Uploading resource file:/tmp/spark-0f5678a4-876e-4066-9102-aaecb7884790/_spark_conf_3804139281588991488.zip -> hdfs://ClusterOrchestrator:8020/user/spotuser/.sparkStaging/application_1539882589280_0004/_spark_conf_3804139281588991488.zip 18/10/19 08:31:09 INFO spark.SecurityManager: Changing view acls to: spotuser 18/10/19 08:31:09 INFO spark.SecurityManager: Changing modify acls to: spotuser 18/10/19 08:31:09 INFO spark.SecurityManager: Changing view acls groups to: 18/10/19 08:31:09 INFO spark.SecurityManager: Changing modify acls groups to: 18/10/19 08:31:09 INFO spark.SecurityManager: SecurityManager: authentication disabled; ucl acls disabled; users with view permissions: Set(spotuser); groups with view permissions: Set(); users with modify permissions: Set(spotuser); groups with modify permissions: Set() 18/10/19 08:31:09 INFO yarn.Client: Submitting application application_1539882589280_0004 to ResourceManager </pre> <p>3. The threat classification procedure reports a number of flows with the label “wannacy”</p>	

```

spotuser@ClusterOrchestrator: ~ 168x43
18/10/19 08:32:29 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-0f5678a4-876e-4866-9102-aaecb7884790

real    1m33.273s
user    0m22.488s
sys     0m3.456s
Starting threat classification for: 20180918
Classification phase selected. Starting Spark job.

Reading flows for classification...
root
 |-- trtime: string (nullable = true)
 |-- tryear: integer (nullable = true)
 |-- trmonth: integer (nullable = true)
 |-- trday: integer (nullable = true)
 |-- trhour: integer (nullable = true)
 |-- trmin: integer (nullable = true)
 |-- trsec: integer (nullable = true)
 |-- td: integer (nullable = true)
 |-- sa: string (nullable = true)
 |-- da: string (nullable = true)
 |-- sp: integer (nullable = true)
 |-- dp: integer (nullable = true)
 |-- pr: string (nullable = true)
 |-- ipkt: integer (nullable = true)
 |-- lbytt: integer (nullable = true)
 |-- opkt: integer (nullable = true)
 |-- obyt: integer (nullable = true)
 |-- prob: double (nullable = true)

Encoding categorical variables to numerical...
Dropping rows containing null values...
Loading the trained Random Forest model

Threat classification complete! Saving results file...
('Successfully created aggregated results folder:', 'results_flow_results.csv')
('Successfully created aggregated threatresults folder:', 'threatresults_flow_results.csv')

real    0m21.691s
user    1m58.184s
sys     0m9.124s
Removing old transmissions...
Fetching threat file from HDFS...
    
```

```

spotuser@ClusterOrchestrator: ~ 168x43

 |-- trsec: integer (nullable = true)
 |-- td: integer (nullable = true)
 |-- sa: string (nullable = true)
 |-- da: string (nullable = true)
 |-- sp: integer (nullable = true)
 |-- dp: integer (nullable = true)
 |-- pr: string (nullable = true)
 |-- ipkt: integer (nullable = true)
 |-- lbytt: integer (nullable = true)
 |-- opkt: integer (nullable = true)
 |-- obyt: integer (nullable = true)
 |-- prob: double (nullable = true)

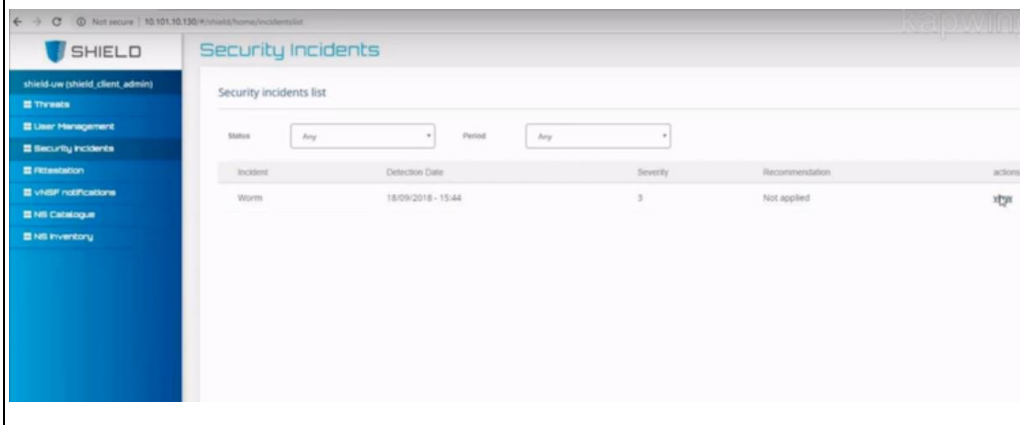
Encoding categorical variables to numerical...
Dropping rows containing null values...
Loading the trained Random Forest model

Threat classification complete! Saving results file...
('Successfully created aggregated results folder:', 'results_flow_results.csv')
('Successfully created aggregated threatresults folder:', 'threatresults_flow_results.csv')

real    0m21.691s
user    1m58.184s
sys     0m9.124s
Removing old transmissions...
Fetching threat file from HDFS...
Starting transmission:
[1] Sent: 20181819 High wannacry start 2018-09-18 14:44:08 2018 9 18 14 44 0 0 192.168.116.150 192.168.116.149 445 4
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:17 2018 9 18 14 44 17 0 192.168.116.138 192.168.116.149 1411 4
9391 TCP 4 426 0 0 1.8983204051745591 2018 9 18 14 44 09 0 192.168.116.143 192.168.116.149 445 4
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:17 2018 9 18 14 44 17 0 192.168.116.143 192.168.116.149 445 4
45 TCP 6 492 0 0 2.2575216343773126 2018 9 18 14 44 09 0 192.168.116.149 192.168.116.143 49566 4
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:09 2018 9 18 14 44 09 0 192.168.116.149 192.168.116.143 49566 4
9558 TCP 4 542 0 0 1.8943389450446284 2018 9 18 14 44 09 0 192.168.116.149 192.168.116.143 49566 4
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:09 2018 9 18 14 44 09 0 192.168.116.149 192.168.116.143 49566 4
45 TCP 4 312 0 0 2.2591884991425983 2018 9 18 14 44 10 0 192.168.116.254 192.168.116.143 67 6
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:10 2018 9 18 14 44 10 0 192.168.116.254 192.168.116.143 67 6
8 UDP 1 328 0 0 1.315597792300766 2018 9 18 14 44 20 0 192.168.116.172 192.168.116.149 49384 4
[1] Sent: 20181819 High wannacry 2018-09-18 14:44:20 2018 9 18 14 44 20 0 192.168.116.172 192.168.116.149 49384 4
45 TCP 7 668 0 0 2.241190812489771 2018 9 18 14 44 20 0 192.168.116.172 192.168.116.149 49384 4
[1] Sent: 20181819 High wannacry stop

Transmission completed.
spotuser@ClusterOrchestrator:~$
    
```

4. The threats are displayed on the Dashboard and a mitigation action is proposed.



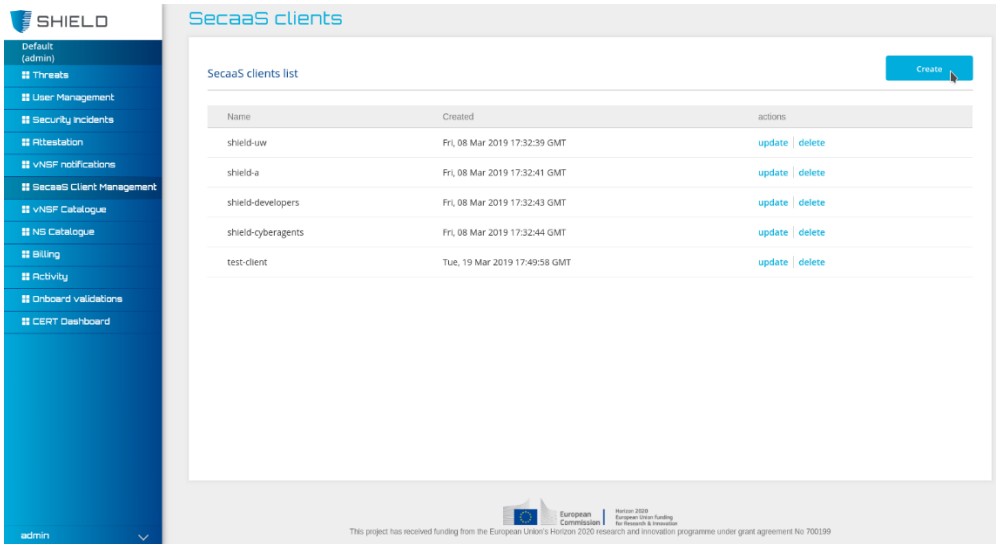
<p>Verdict</p>	<p>The DARE was able to ingest a batch of network traffic, detect and label any included malicious activity and report the attacker IPs to the user via the Dashboard, in an automated manner and in near-real-time (few minutes). The two-stage ML procedure of the Cognitive DA module was capable of detecting all the malicious IPs involved in the wannacry attack.</p>
<p>Comments</p>	<p>During this demonstration, the Cognitive DA module presented optimal detection results. In general, detection efficiency depends on the type of attacks, as well as on the amount and quality of the ingested traffic. The small time delay between ingestion and reporting is introduced by the automated network transmission and data processing, and is considered normal in scalable distributed computing frameworks.</p>

<p>Test Case ID</p>	<p>TC_P05</p>			
<p>Description</p>	<p>Access control and multi-role support</p>			
<p>Executed by</p>	<p>UBI</p>	<p>Date</p>	<p>5/3/2019</p>	
<p>Purpose</p>	<p>Verify that the SHIELD Dashboard allows the creation of multiple SecaaS Clients and multiple users with different roles for each SecaaS Client. Ensure that each user has the proper functionalities available and scoped authorization for its role.</p>			
<p>Associated Requirements</p>	<p>PF06, PF09</p>			
<p>Components involved</p>	<p>Dashboard</p>			
<p>Tools</p>	<p>-</p>			
<p>Metrics</p>	<p>-</p>			
<p>Pre-test conditions</p>	<p>The Dashboard is running after a clean setup, no SecaaS Clients or users are available. The only user available is the platform administrator.</p>			
<p>Test Sequence</p>	<p>Step</p>	<p>Type</p>	<p>Description</p>	<p>Result</p>

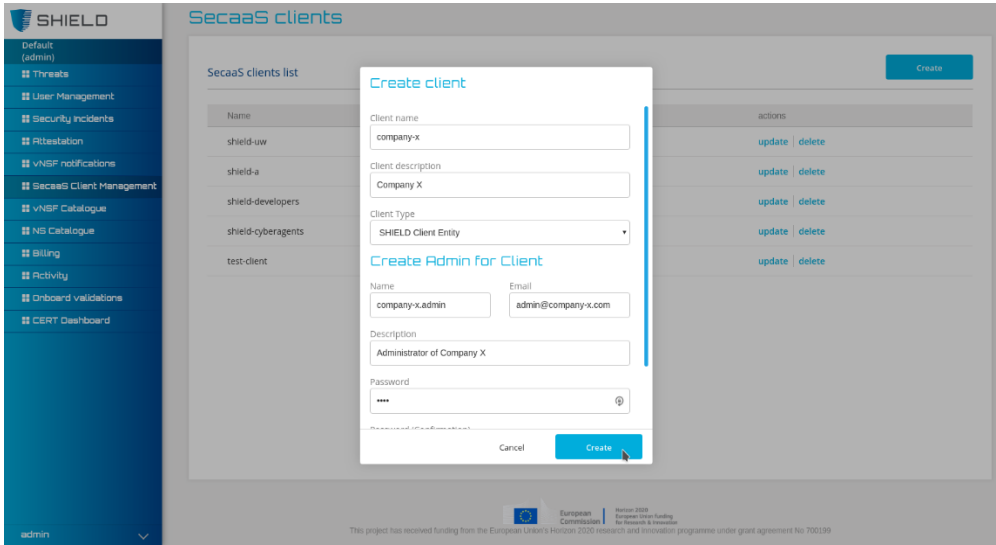
	1	Stimulus	Create a new SecaaS Client	
	2	Check	Verify that a new SecaaS client is listed.	OK
	3	Stimulus	Impersonate the created SecaaS client administrator by signing in as such.	
	4	Check	Verify that the SecaaS client has the proper authorization to access all the features meant for its role.	OK

Evidence

#1 Create a new SecaaS Client



The screenshot shows the SHIELD web interface. On the left is a navigation menu with 'SecaaS Client Management' selected. The main area displays 'SecaaS clients' with a table listing existing clients: shield-uw, shield-a, shield-developers, shield-cyberagents, and test-client. A 'Create' button is visible in the top right corner of the client list area.



The second screenshot shows the 'Create client' modal form open over the client list. The form includes fields for 'Client name' (filled with 'company-x'), 'Client description' (filled with 'Company X'), and 'Client Type' (set to 'SHIELD Client Entity'). Below this is a 'Create Admin for Client' section with fields for 'Name' (filled with 'company-x.admin'), 'Email' (filled with 'admin@company-x.com'), and 'Description' (filled with 'Administrator of Company X'). There is also a 'Password' field. 'Cancel' and 'Create' buttons are at the bottom of the modal.

#2 Verification that the new SecaaS Client is created as well as its administration user

SecaaS clients

SecaaS clients list Create

Name	Created	actions
shield-uw	Fri, 08 Mar 2019 17:32:39 GMT	update delete
shield-a	Fri, 08 Mar 2019 17:32:41 GMT	update delete
shield-developers	Fri, 08 Mar 2019 17:32:43 GMT	update delete
shield-cyberagents	Fri, 08 Mar 2019 17:32:44 GMT	update delete
test-client	Tue, 19 Mar 2019 17:49:58 GMT	update delete
company-x	Wed, 27 Mar 2019 09:47:00 GMT	update delete

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

Users

Users list

Name	SecaaS client	Role	actions
shield.client.admin	shield-uw	Client administrators	update delete
shield.client.a.admin	shield-a	Client administrators	update delete
shield.client.user	shield-uw	Client users	update delete
shield.client.a.user	shield-a	Client users	update delete
shield.developer	shield-developers	VNSF Developers	update delete
shield.cyberagent	shield-cyberagents	Cyberagency users	update delete
test.client.admin	test-client	Client administrators	update delete
company-x.admin	company-x	Client administrators	update delete

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

#3-4 Login as administrator of the new SecaaS Client and verify that the left panel contains all the required functionalities for its role

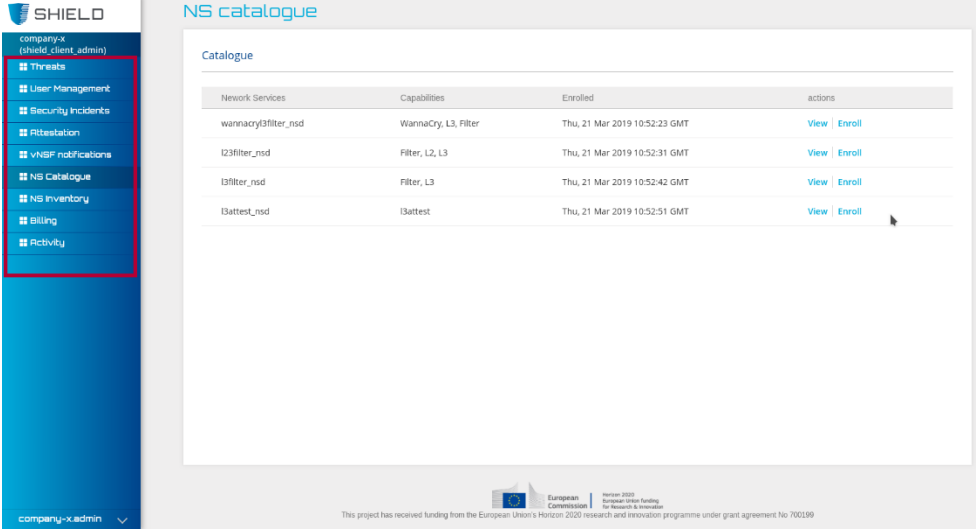
SHIELD

company-x.admin

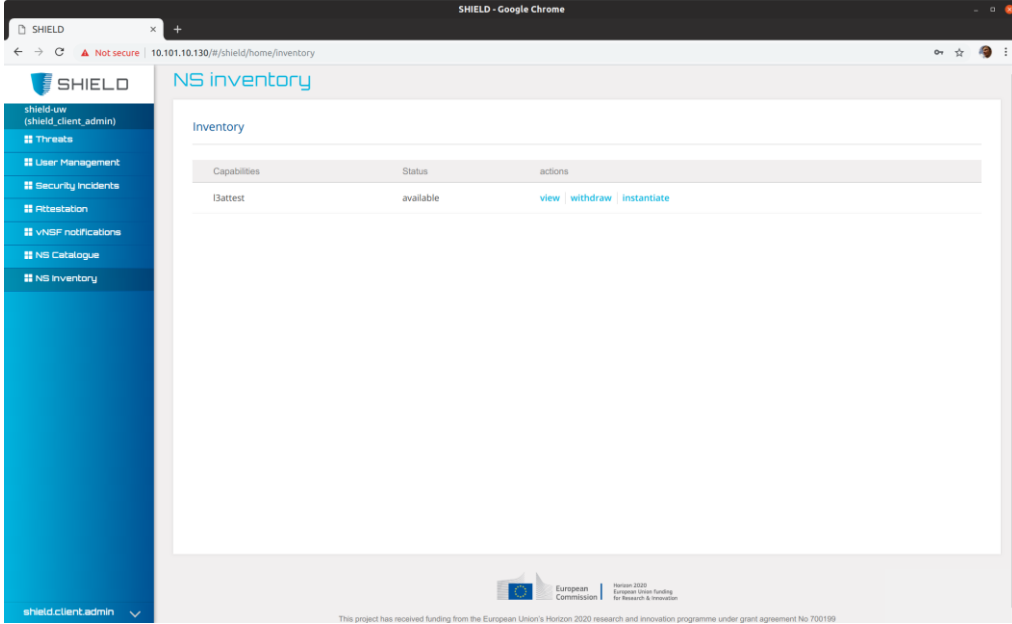
.....

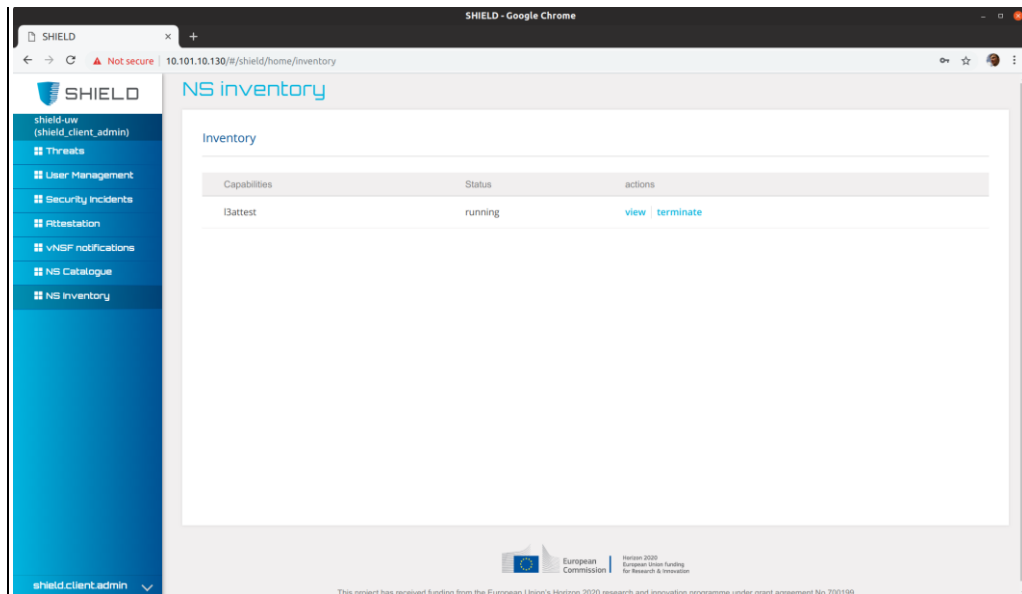
company-x

Login

	
Verdict	Success
Comments	-

Test Case ID	TC_P06			
Description	vNSF attestation			
Executed by	POLITO	Date	20/2/2019	
Purpose	The Trust Monitor is capable of detecting a manipulation on a running vNSF. The manipulation is represented by a custom application which is not part of the vNSF Security Manifest.			
Associated Requirements	PF10. PF11			
Components involved	Trust Monitor, Dashboard, NFVO, NFVI compute host			
Tools	None			
Metrics	Attestation delay			
Pre-test conditions	vNSF and NS packages are correctly onboarded in the vNSF Store and the vNSF image has not been tampered beforehand.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Instantiate NS including the vNSF from Dashboard by selecting it from the NS catalogue	
	2	Check	The vNSF is correctly instantiated by the NFVO on the NFVI compute host	OK

	3	Stimulus	Run attestation check on the Dashboard from the ISP client view	
	4	Check	The Dashboard shows a notification on the attestation check, with successful result	OK
	5	Stimulus	The vNSF is tampered by loading a custom application (e.g. Bash script) that is not part of the vNSF Security Manifest, by manually accessing the NFVI compute host	
	6	Check	The vNSF runs the custom application, and its output is shown on the NFVI compute host console	OK
	7	Stimulus	Run attestation check on the vNSF from the ISP client view	
	8	Check	The Dashboard shows a notification on the attestation check, with failure	OK
	9	Stimulus	The ISP client checks the attestation result from the Dashboard view	
	10	Check	The Dashboard view shows that the custom application was launched in the tampered vNSF	OK
Evidence	<p>#1 Enroll a Network Service in the SecaaS client view</p>  <p>#2 Run the Network Service in the SecaaS client view</p>			

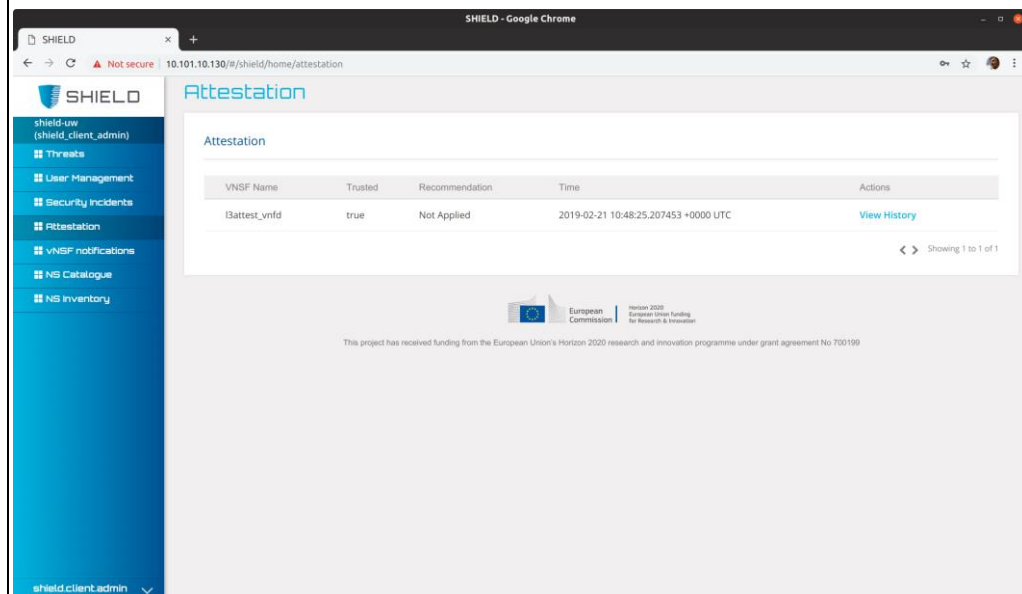


The screenshot shows the SHIELD web interface in Google Chrome. The browser address bar displays "10.101.10.130/#shield/home/inventory". The page title is "NS inventory". On the left, a blue sidebar menu lists various options, with "NS inventory" selected. The main content area, titled "Inventory", contains a table with the following data:

Capabilities	Status	actions
l3attest	running	view terminate

At the bottom of the page, there is a footer with the European Union logo and text: "European Commission | Horizon 2020 European Union funding for Research & Innovation. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199".

#3 Verify that the NS is trusted in the Attestation view



The screenshot shows the SHIELD web interface in Google Chrome. The browser address bar displays "10.101.10.130/#shield/home/attestation". The page title is "Attestation". On the left, the same blue sidebar menu is visible, with "NS inventory" selected. The main content area, titled "Attestation", contains a table with the following data:

VNSF Name	Trusted	Recommendation	Time	Actions
l3attest_vnfd	true	Not Applied	2019-02-21 10:48:25.207453 +0000 UTC	View History

At the bottom of the page, there is a footer with the European Union logo and text: "European Commission | Horizon 2020 European Union funding for Research & Innovation. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199".

#4 Inject a script in the vNSF by manipulating the container

```
torsec@nfvi-node:~/y2demo
File Edit View Search Terminal Help
[torsec@nfvi-node y2demo]$ ls
attack_vnf.sh injected_script.sh unmeasured_script.sh
[torsec@nfvi-node y2demo]$ ./attack_vnf.sh
Script injection in vNSF. Success!!!
[torsec@nfvi-node y2demo]$
```

#5 Verify that the vNSF is untrusted at the next attestation refresh

SHIELD - Google Chrome

SHIELD

shield-usr (shield_client_admin)

- Threats
- User Management
- Security Incidents
- Attestation
- vNSF notifications
- NS Catalogue
- NS Inventory

Attestation

VNSF Name	Trusted	Recommendation	Time	Actions
l3attest_vnfd	false	Not Applied	2019-02-21 10:49:31.797065 +0000 UTC	View History

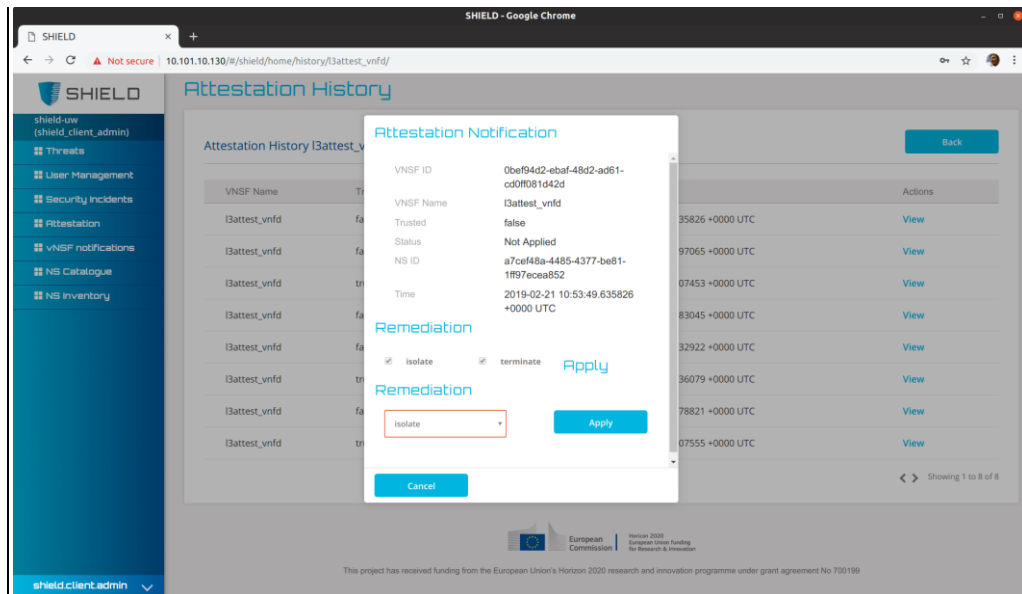
Showing 1 to 1 of 1

European Commission | Horizon 2020
European Union Funding
for Research & Innovation

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

shield_client_admin

#6 Show details of failed attestation in the Dashboard



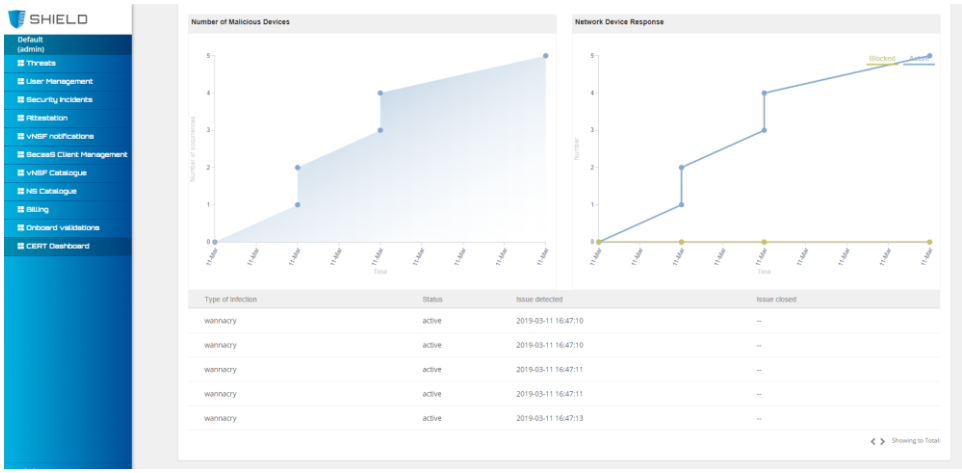
Trust Monitor API to attest the vNSF

```
REQUEST: curl -X GET -k https://<TRUST_MONITOR_BASE_URL>/nfvi_pop_attestation_info/?node_id=nfvi-node
RESPONSE: {"hosts":{"node":"nfvi-node","status":0,"time":"2019-02-08 10:47:24.218939 +0000 UTC","remediation":{"terminate":false,"isolate":false},"vnsfs":[],"trust":true,"driver":"OAT","extra_info":{"n_digests_valid":465,"n_packages_valid":135,"list_digests_fake_lib":[],"n_packages_not_security":0,"n_packages_unknown":1,"n_packages_security":0,"list_digests_not_found":[],"n_digests_not_found":0,"n_digests_fake_lib":0}},"sdn":[""],"trust":true,"vtime":"2019-02-08 10:47:24.219215 +0000 UTC"}
```

The attestation delay to verify the vNSF (which comprises underlying NFVI compute host attestation) is 12,231s seconds.

Verdict	Success
Comments	The delay to attest a vNSF includes the delay to attest the underlying NFVI compute host, as the vNSF attestation process is part of the host integrity verification workflow.

Test Case ID	TC_P07		
Description	Threat data sharing		
Executed by	I2CAT	Date	15/3/2019
Purpose	Share statistic data regarding incidents with CERTs so that they can have visibility on aspects like number of malicious devices, blocked devices and propagation speed.		

Associated Requirements	PF12, PF17, PF22 (encryption)																											
Components involved	DARE, dashboard																											
Tools	None																											
Metrics	Visualization delay																											
Pre-test conditions	Run one or more attack that generates reports																											
Test Sequence	Step	Type	Description	Result																								
	1	Stimulus	Run an attack that generates an alert	Alert																								
	2	Check	Check if the Dashboard shows the metrics	True																								
Evidence	Pilots and year 2.5 demo. The following picture shows the result of the dashboard after the wannacry attack																											
	 <p>The screenshot shows the SHIELD dashboard interface. On the left is a blue sidebar with navigation items: Details (id:010), Threats, User Management, Security Incidents, Mitigation, VMBP Notifications, SecaaS Client Management, VMBP Catalogue, M3 Catalogue, Billing, Onboard Validations, and CERT Dashboard. The main area contains two line graphs. The first graph, 'Number of Malicious Devices', shows a steady increase from 0 to 5 over time. The second graph, 'Network Device Response', shows a step-wise increase from 0 to 5, with the final point labeled 'Blocked'. Below the graphs is a table with the following data:</p> <table border="1"> <thead> <tr> <th>Type of Infection</th> <th>Status</th> <th>Issue detected</th> <th>Issue closed</th> </tr> </thead> <tbody> <tr> <td>wannacry</td> <td>active</td> <td>2019-03-11 16:47:10</td> <td>--</td> </tr> <tr> <td>wannacry</td> <td>active</td> <td>2019-03-11 16:47:10</td> <td>--</td> </tr> <tr> <td>wannacry</td> <td>active</td> <td>2019-03-11 16:47:11</td> <td>--</td> </tr> <tr> <td>wannacry</td> <td>active</td> <td>2019-03-11 16:47:11</td> <td>--</td> </tr> <tr> <td>wannacry</td> <td>active</td> <td>2019-03-11 16:47:13</td> <td>--</td> </tr> </tbody> </table>				Type of Infection	Status	Issue detected	Issue closed	wannacry	active	2019-03-11 16:47:10	--	wannacry	active	2019-03-11 16:47:10	--	wannacry	active	2019-03-11 16:47:11	--	wannacry	active	2019-03-11 16:47:11	--	wannacry	active	2019-03-11 16:47:13	--
Type of Infection	Status	Issue detected	Issue closed																									
wannacry	active	2019-03-11 16:47:10	--																									
wannacry	active	2019-03-11 16:47:10	--																									
wannacry	active	2019-03-11 16:47:11	--																									
wannacry	active	2019-03-11 16:47:11	--																									
wannacry	active	2019-03-11 16:47:13	--																									
Verdict	Success																											
Comments	-																											

Test Case ID	TC_P08		
Description	Automatic Incident Mitigation		
Executed by	POLITO	Date	20/2/2019
Purpose	The Recommendation and Remediation Engine of the DARE correctly parses an anomaly report (generated by the Data Analytics engine in the DARE) and creates a mitigation proposal starting from one or more pre-defined recipes for the specific type of attack		
Associated Requirements	PF13		

Components involved	DARE, Dashboard			
Tools	None			
Metrics	MSPL creation latency.			
Pre-test conditions	An incident has been detected, so that the Data Analytics engine in the DARE can generate an incident report			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	The Data Analytics engine pushes an incident report to the Recommendation and Remediation engine	
	2	Check	The Recommendation and Remediation engine logs the incoming incident report entries	OK
	3	Stimulus	The Recommendation and Remediation engine parses the incident report, selects the pre-defined recipes depending on the type of attack and generates the high-level policies in the HSPL language. The Recommendation and Remediation engine translates the HSPL policies of each recipe into MSPL policies.	
	4	Check	The Recommendation and Remediation engine logs the HSPL and MSPL policies for each recipe.	OK
	5	Stimulus	The Recommendation and Remediation engine forwards the MSPL policies to the Dashboard	
	6	Check	The Dashboard shows a notification regarding a new mitigation proposal to address an incoming security threat	OK
	7	Stimulus	The Dashboard user applies one of the presented mitigation proposals to address the incoming security threat	
	8	Check	The Dashboard shows that the mitigation has been pushed to the NFVO so that it will be applied by a running vNSF	OK
Evidence	#1 Configuration of recipe for DoS attack (packet filtering, rate limiting)			

```
polito@polito: ~/dare-sec-topo/cybertop/recipes
polito@polito: ~/dare-sec-topo/cybertop/recipes
polito@polito: ~/dare-sec-topo/cybertop/recipes
?xml version="1.0" encoding="UTF-8"?
<recipe-set xmlns:="http://security.polito.it/shield/recipe" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://security.polito.it/shield/recipe ../xsd/recipe.xsd" >

  <recipe>
    <name>limit packets</name>
    <action>limit</action>
    <object-constraints>
      <any-port>true</any-port>
    </object-constraints>
    <traffic-constraints>
      <max-connections>20</max-connections>
      <rate-limit>100/s</rate-limit>
    </traffic-constraints>
  </recipe>

  <recipe>
    <name>drop packets</name>
    <action>drop</action>
    <object-constraints>
      <any-port>true</any-port>
    </object-constraints>
  </recipe>
</recipe-set>

1,1 All
```

#2 Begin of incident report parsing (total of 1000 lines)

```
root@polito: /var/log/cybertop
root@polito: /var/log/cybertop
2019-02-08 13:44:02,166 - cybertop - DEBUG - Callback from event in directory
2019-02-08 13:44:02,328 - cybertop - INFO - Parsed an attack of type 'DoS' with severity 3 and containing 1000 events.
2019-02-08 13:44:02,329 - cybertop - INFO - Landscape with 1 IT resources read.
2019-02-08 13:44:02,330 - cybertop - DEBUG - Found 3 suitable recipes.
2019-02-08 13:44:02,330 - cybertop - INFO - 3 recipes chosen.
2019-02-08 13:44:02,417 - cybertop - DEBUG - <recommendations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
security.polito.it/shield/hspl">
  <hspl-set>
    <context>
      <severity>3</severity>
      <type>DoS</type>
      <timestamp>2017-10-23T15:28:00</timestamp>
    </context>
    <hspl>
      <name>drop packets #1</name>
      <subject>10.101.20.230:80</subject>
      <action>drop</action>
      <object>110.14.254.131:*</object>
      <traffic-constraints>
        <type>TCP</type>
      </traffic-constraints>
    </hspl>
    <hspl>
      <name>drop packets #2</name>
      <subject>10.101.20.230:80</subject>
      <action>drop</action>
      <object>1.152.233.144:*</object>
      <traffic-constraints>
        <type>TCP</type>
      </traffic-constraints>
    </hspl>
  </hspl-set>
</recommendations>
"cybertop.log" 29584L, 863681C
1,0-1 Top
```

#3 End of MSPL generation for DoS incident

```

root@polite: /var/log/cybertop
File Edit View Search Terminal Tabs Help
root@polite: /var/log/cybertop
<protocol>TCP</protocol>
</packet-filter-condition>
<traffic-flow-condition>
  <max-connections>20</max-connections>
  <rate-limit>100/s</rate-limit>
</traffic-flow-condition>
</condition>
</rule>
<rule>
  <priority>16</priority>
  <action>reject</action>
  <condition>
    <packet-filter-condition>
      <direction>inbound</direction>
      <source-address>100.114.244.0/25</source-address>
      <source-port>*</source-port>
      <destination-address>10.101.20.230</destination-address>
      <destination-port>80</destination-port>
      <protocol>TCP</protocol>
    </packet-filter-condition>
  </condition>
</rule>
</configuration>
</it-resource>
</mspl-set>
</recommendations>

2019-02-08 13:44:04,025 - cybertop - INFO - Connected to the dashboard at 10.101.10.130:5672
2019-02-08 13:44:04,025 - cybertop - INFO - Pushing the remediation to the dashboard
2019-02-08 13:44:04,025 - cybertop - DEBUG - Dashboard RabbitMQ exchange: shield-dashboard-exchange topic: shield.dare.policy
2019-02-08 13:44:04,025 - cybertop - INFO - Remediation forwarded to the dashboard
2019-02-08 13:44:04,047 - cybertop - INFO - Connection with the dashboard closed
29584,1 Bot
  
```

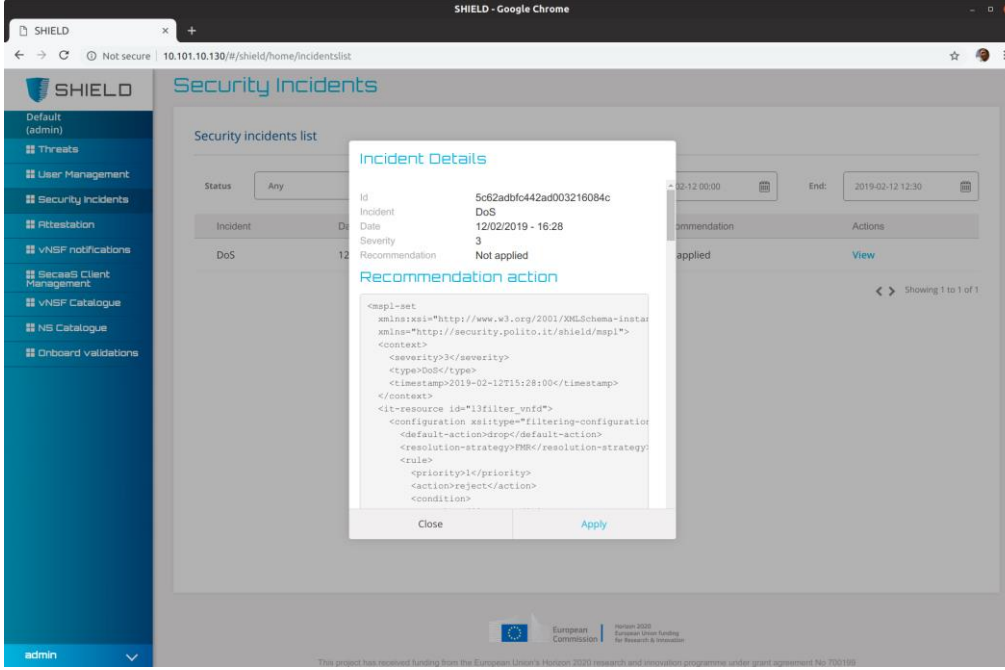
#4 Security incident entry on Dashboard (logged as ISP admin)

The screenshot shows the SHIELD dashboard interface in a Google Chrome browser. The page title is "Security Incidents" and the URL is "10.101.10.130/#/shield/home/incidentslist". On the left, there is a navigation menu with options like "Default (admin)", "Threats", "User Management", "Security Incidents", "Attestation", "vNSP notifications", "SecaaS Client Management", "vNSP Catalogue", "NS Catalogue", and "Onboard validations". The main content area displays a "Security incidents list" with a table containing one entry:

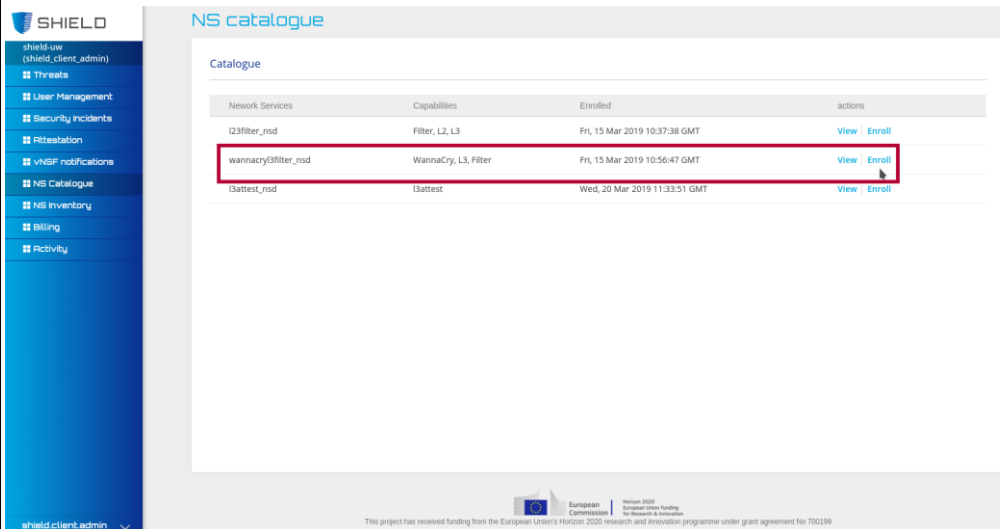
Incident	Date	Severity	Recommendation	Actions
DoS	12/02/2019 - 16:28	3	Not applied	View

At the bottom of the dashboard, there are logos for the European Commission and Horizon 2020 European Union Funding for Research & Innovation, along with a note: "This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199".

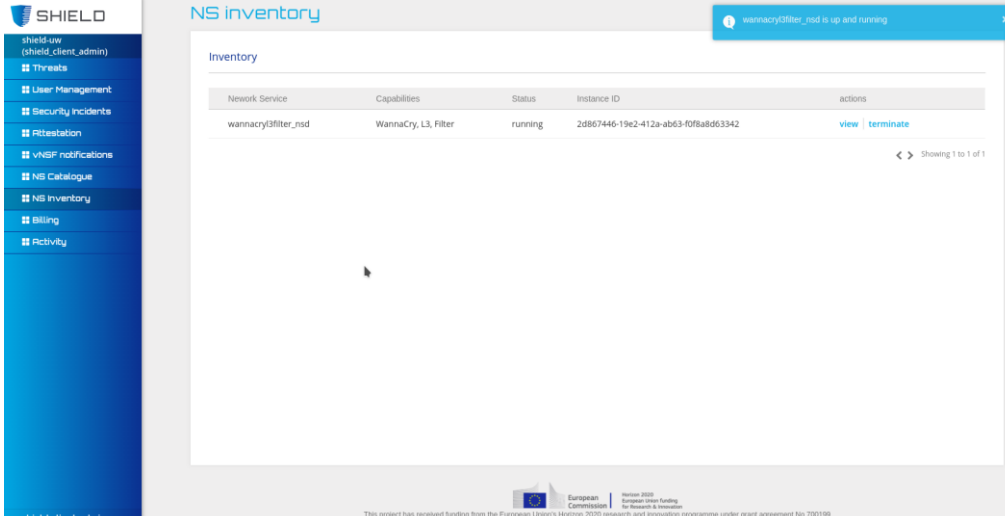
#5 MSPL-based recommendation action details on the Dashboard (logged as ISP admin)

	 <p>The MSPL creation latency, when parsing an incident report of type DoS comprising 1000 lines in order to create a recommendation based on two recipes (packet filtering, rate limiting) is 1,881 seconds.</p>
Verdict	Success
Comments	None

Test Case ID	TC_P09			
Description	Multi-user			
Executed by	UBI	Date	2/3/2019	
Purpose	Establish a separate visualization of Network Services and related security incidents between different SecaaS Clients.			
Associated Requirements	PF14			
Components involved	Dashboard			
Tools	-			
Metrics	Separation of authorization between SecaaS Client users			
Pre-test conditions	A Network Service instantiated by a SecaaS Client administrator will trigger security incidents which will not be visible by an administrator of a distinct SecaaS Client			
Test Sequence	Step	Type	Description	Result

	1	Stimulus	A SecaaS Client administrator enrolls/adds a Network Service to its NS Inventory. The added Network Service is instantiated.	
	2	Check	The instantiated Network Service gets instantiated, after a while, showing the Instance ID and its status marked as as “running”.	OK
	3	Stimulus	The SecaaS Client will view the list of Security Incidents related to the instantiated Network Service.	
	4	Check	A list of Security Incidents related to the instantiated Network Service is shown.	OK
	5	Stimulus	As SHIELD platform administrator, a new SecaaS Client is created, as well as its administrator user.	
	6	Check	Verify that the new SecaaS Client is available, its administrator user is available and that the platform allows its login.	OK
	7	Stimulus	Access the NS Inventory of the new SecaaS Client to verify it doesn't have any enrolled Network Services nor any instantiated Network Service.	
	8	Check	The NS Inventory should be empty.	OK
	9	Stimulus	Access the Security Incidents	
	10	Check	The Security Incidents should be empty.	OK
Evidence	#1 Enrollment and instantiation of a Network Service  <p>The screenshot displays the SHIELD NS catalogue interface. On the left is a navigation menu with options like Threats, User Management, Security Incidents, etc. The main content area shows a table titled 'Catalogue' with columns for Network Services, Capabilities, Enrolled, and actions. A red box highlights the row for 'wannacry3filter_rsd', which has capabilities 'WannaCry, L3, Filter' and was enrolled on 'Fri, 15 Mar 2019 10:56:47 GMT'. Other rows include 'I23filter_rsd' and 'I3attest_rsd'.</p>			

#2 Instantiation was successful and Instance details are shown

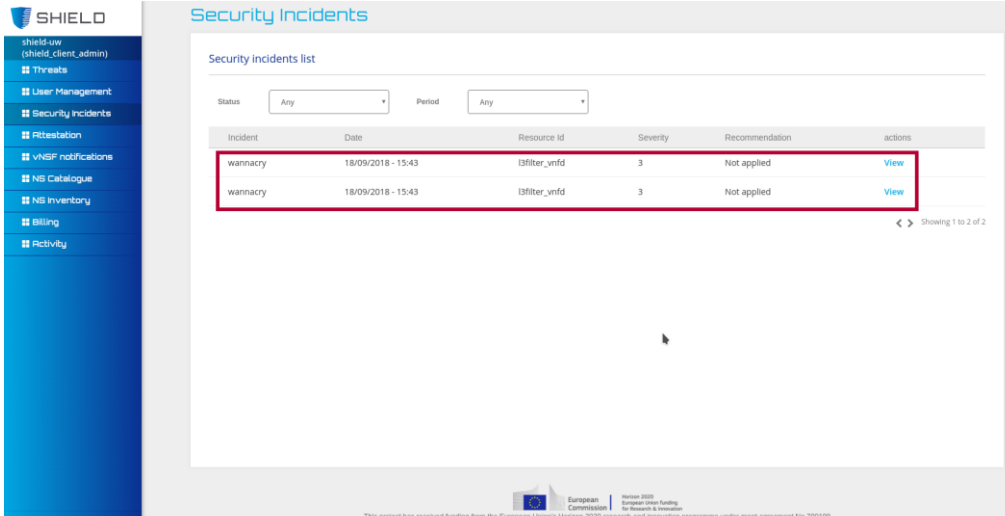


The screenshot shows the SHIELD web interface. On the left is a navigation menu with options like Threats, User Management, Security Incidents, etc. The main content area is titled 'NS inventory' and displays a table with the following data:

Network Service	Capabilities	Status	Instance ID	actions
wannacry3filter_nsd	WannaCry, L3, Filter	running	2d867446-19e2-412a-ab63-f0f8a8d63342	view terminate

At the bottom of the page, there is a footer with the European Commission logo and text: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199'.

#3-4 List of Security Incidents related to the instantiated Network Service.

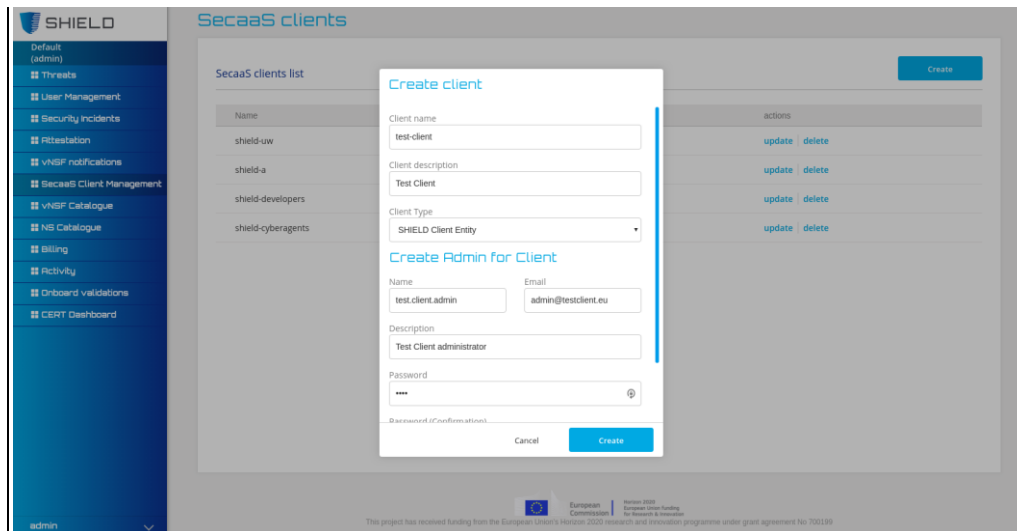


The screenshot shows the SHIELD web interface with the 'Security Incidents' page. The page has filters for 'Status' and 'Period', both set to 'Any'. Below the filters is a table with the following data:

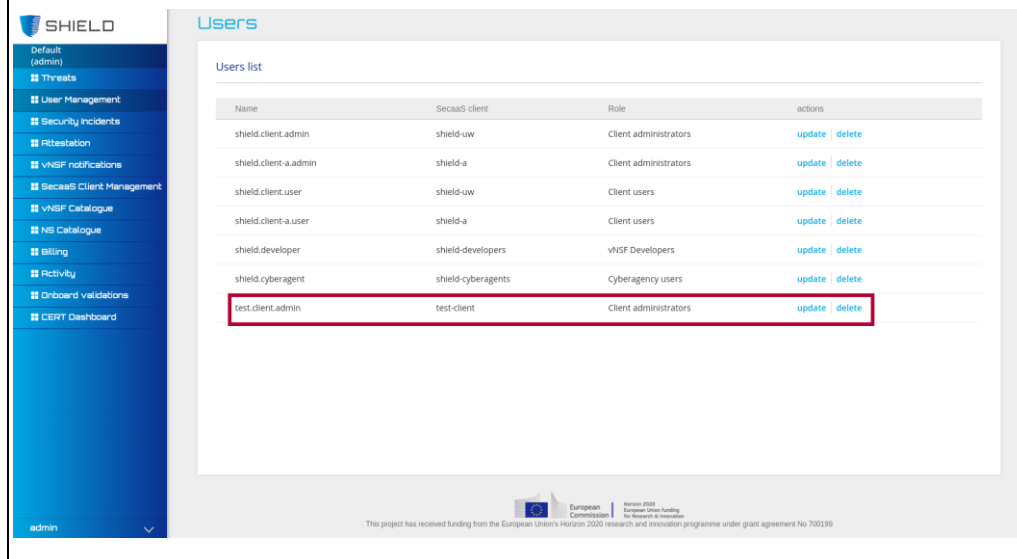
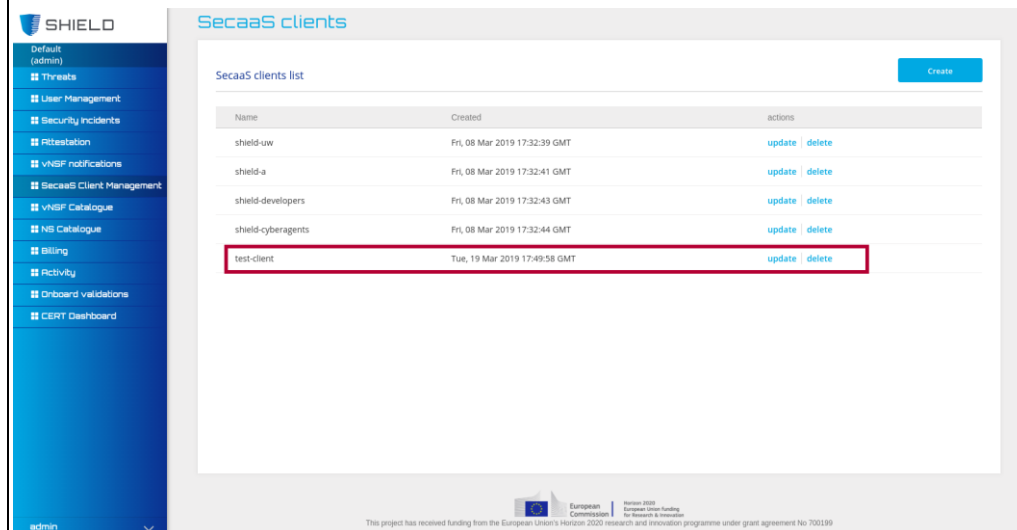
Incident	Date	Resource Id	Severity	Recommendation	actions
wannacry	18/09/2018 - 15:43	l3filter_vmfd	3	Not applied	View
wannacry	18/09/2018 - 15:43	l3filter_vmfd	3	Not applied	View

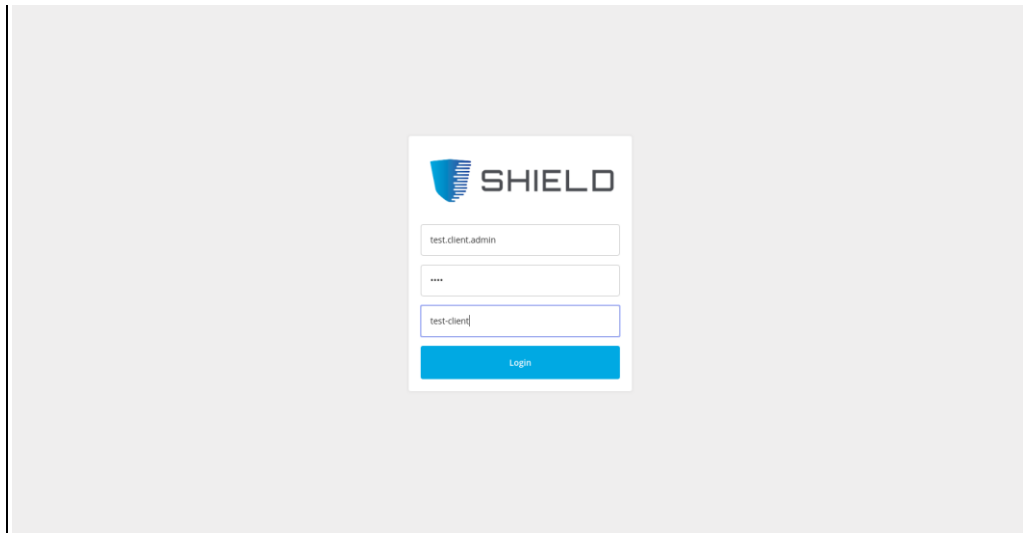
The table is highlighted with a red border. At the bottom of the page, there is a footer with the European Commission logo and text: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199'.

#5 The platform administrator creates a new SecaaS Client as well as its administrator user.

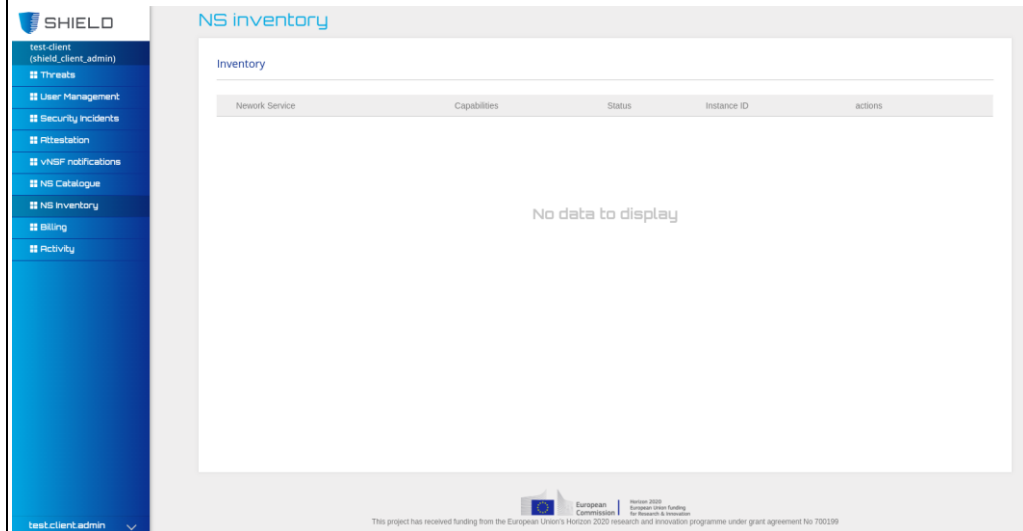


#6 The new SecaaS Client is available, its administrator user is available and that the platform allows its login.

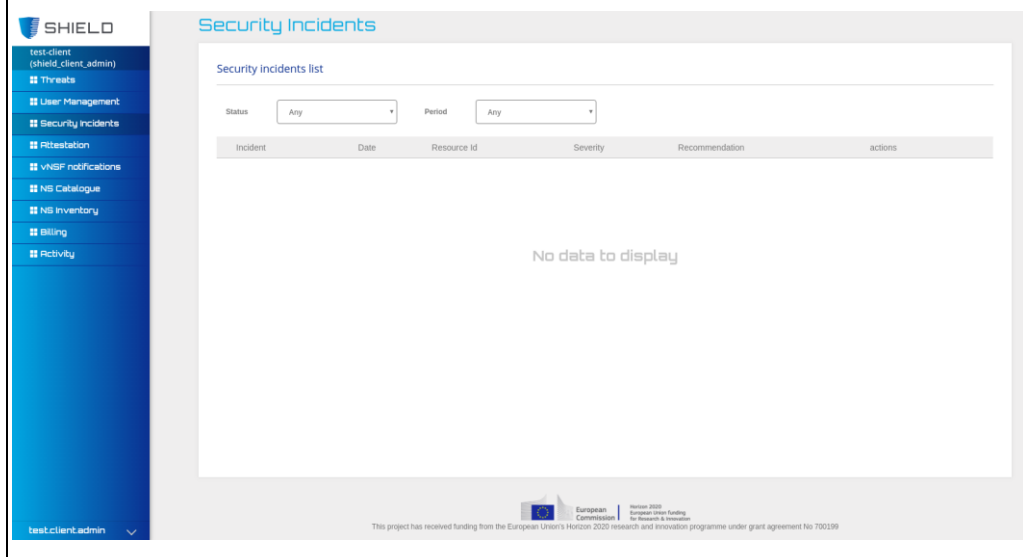




#7-8 The NS Inventory of the new SecaaS Client doesn't have any enrolled Network Services nor any instantiated Network Service.



#9-10 The Security Incidents of the new SecaaS client is empty



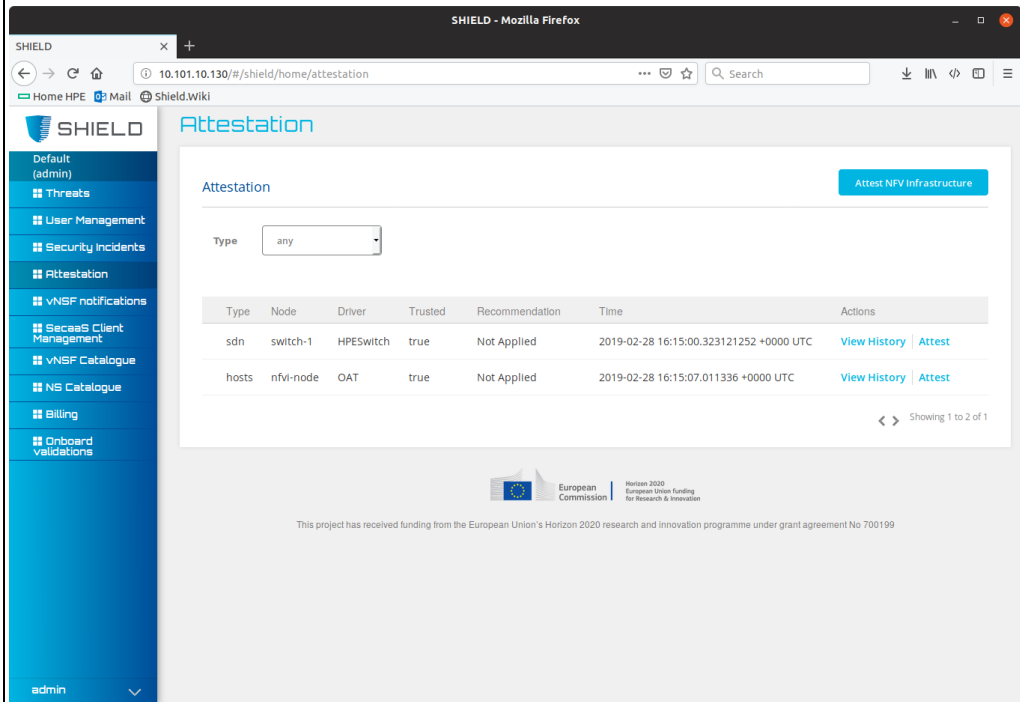
Verdict	Success
Comments	This test was intended to show the clear separation of the owned Network Service instances between SecaaS Clients as well as the related Security Incidents and Attestations performed by their running services.

Test Case ID	TC_P10			
Description	Network infrastructure attestation and remediation			
Executed by	HPE	Date	26/02/2019	
Purpose	The Trust Monitor is capable of detecting a manipulation on a NFVI network element. The manipulation is represented by the use of an unauthorised configuration, which is not part of the Trust Monitor Whitelist Database.			
Associated Requirements	PF19			
Components involved	Trust Monitor, Dashboard, Network Infrastructure			
Tools	None			
Metrics	Attestation delay			
Pre-test conditions	The NFVI network elements are running and its configuration has not been tampered beforehand.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Run attestation check on the Dashboard from the ISP admin view.	
	2	Check	The Dashboard shows a notification on the attestation check, with successful result.	OK
	3	Stimulus	One attacker (or unauthorised administrator) modifies the running configuration of one network element.	
	4	Check	The Dashboard shows a notification on the attestation check, with a failure result and a remediation recommendation of re-configuring the network element.	OK
	5	Stimulus	One authorised administrator restores the correct running configuration of the untrusted network element.	

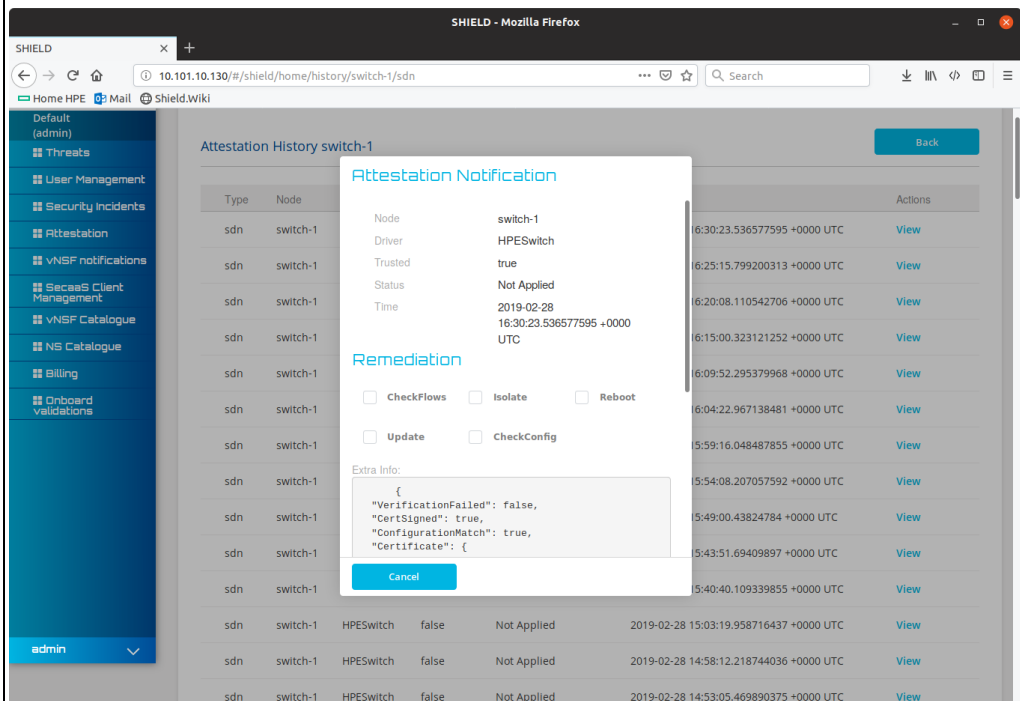
	6	Check	The Dashboard shows a notification on the attestation check, with successful result.	OK
--	---	-------	--	----

Evidence

#1 Attestation view in Dashboard (logged as ISP admin user)



#2 Attestation details for NFVI network element



#3 Manual modification of the network element configuration

```

localadmin@odl: ~
File Edit View Search Terminal Help
no ip address
exit
vlan 402
name "vlan402"
untagged 13-15
ip address 10.102.2.3 255.255.255.0
exit
vlan 403
name "vlan403"
untagged 16-18
ip address 10.102.3.3 255.255.255.0
exit
no tftp server
tftp server listen data
no autorun
no dhcp config-file-update
no dhcp image-file-update
password manager
password operator

Orion-sw# conf term
Orion-sw(config)# vlan 666
Orion-sw(vlan-666)# name "Rogue VLAN"
Orion-sw(vlan-666)#

```

#4 Notification of subsequent periodic attestation in Dashboard (logged as ISP admin user)

The screenshot shows the SHIELD dashboard interface in a Mozilla Firefox browser. The page title is 'SHIELD - Mozilla Firefox' and the URL is '10.101.10.130/#/shield/home/attestation'. A notification banner at the top right says 'TM Notification: New attestation data available' with a button 'Attest NFV Infrastructure'. The main content area is titled 'Attestation' and features a dropdown menu for 'Type' set to 'any'. Below this is a table with the following data:

Type	Node	Driver	Trusted	Recommendation	Time	Actions
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 16:35:31.189538915 +0000 UTC	View History Attest
hosts	nfvi-node	OAT	true	Not Applied	2019-02-28 16:35:37.912146 +0000 UTC	View History Attest

At the bottom of the dashboard, there is a footer with the European Commission logo and text: 'Horizon 2020 European Union Funding for Research & Innovation. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 7001199'.

#5 Detailed of attestation result, including the remediation recommendation (logged as ISP admin user)

The screenshot displays the SHIELD web interface in a Mozilla Firefox browser. The main content area shows the 'Attestation History' for a switch-1 node. A modal dialog titled 'Attestation Notification' is open, providing details for a specific attestation event. The modal includes a 'Remediation' section with several checkboxes and an 'Extra Info' section containing a JSON object.

Type	Node	Driver	Trusted	Status	Time	Actions
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 16:35:31.189538915 +0000 UTC	View
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 15:54:08.207057592 +0000 UTC	View
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 15:49:00.43824784 +0000 UTC	View
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 15:43:51.69409897 +0000 UTC	View
sdn	switch-1	HPESwitch	false	Not Applied	2019-02-28 15:40:40.189338855 +0000 UTC	View

Attestation Notification

Node: switch-1
 Driver: HPESwitch
 Trusted: false
 Status: Not Applied
 Time: 2019-02-28 16:35:31.189538915 +0000 UTC

Remediation

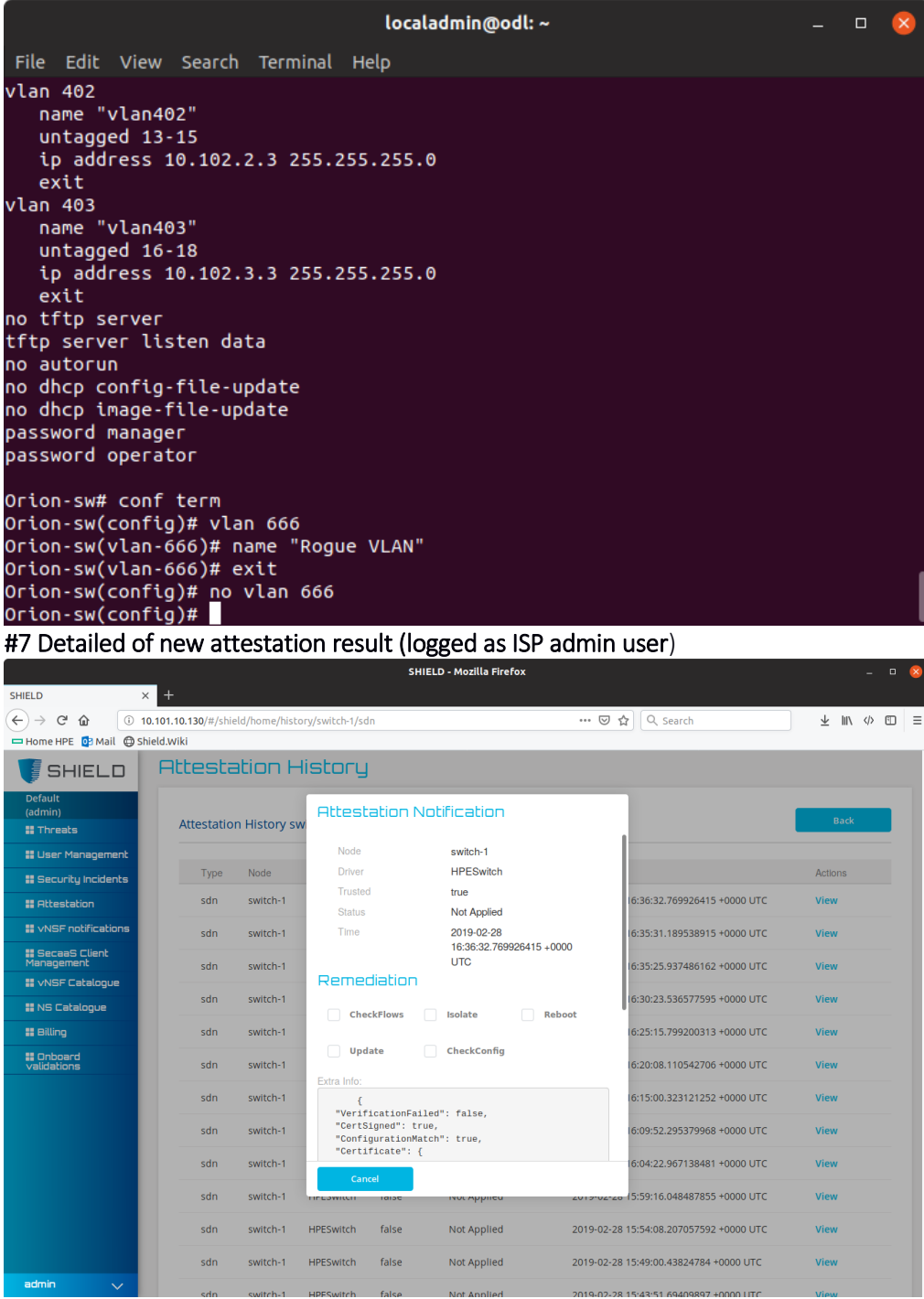
CheckFlows Isolate Reboot
 Update CheckConfig

Extra Info:

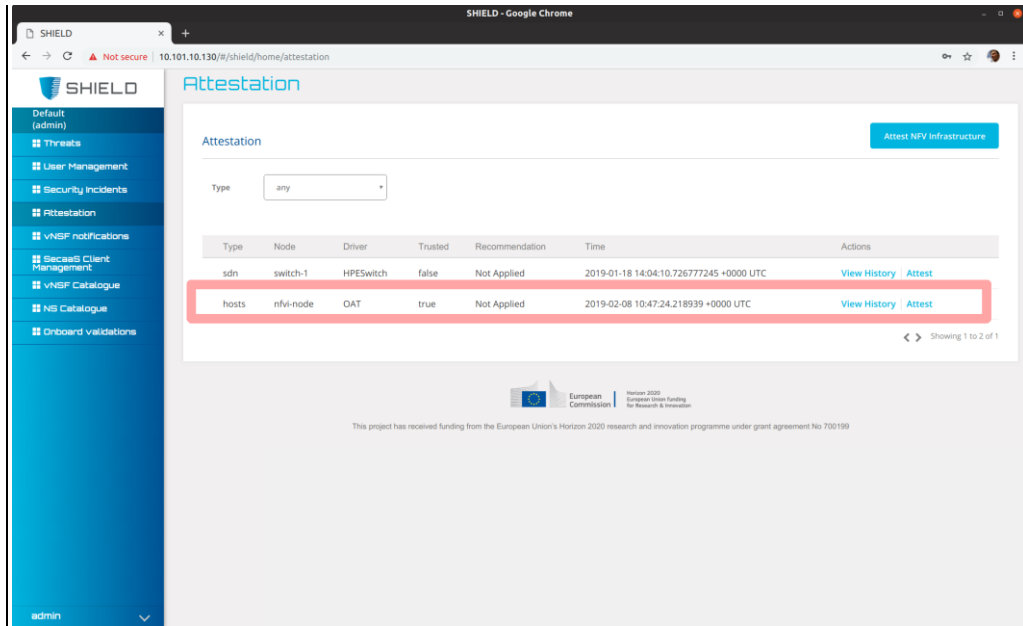
```
{
  "VerificationFailed": false,
  "CertSigned": true,
  "ConfigurationMatch": false,
  "Certificate": {
```

[Cancel](#)

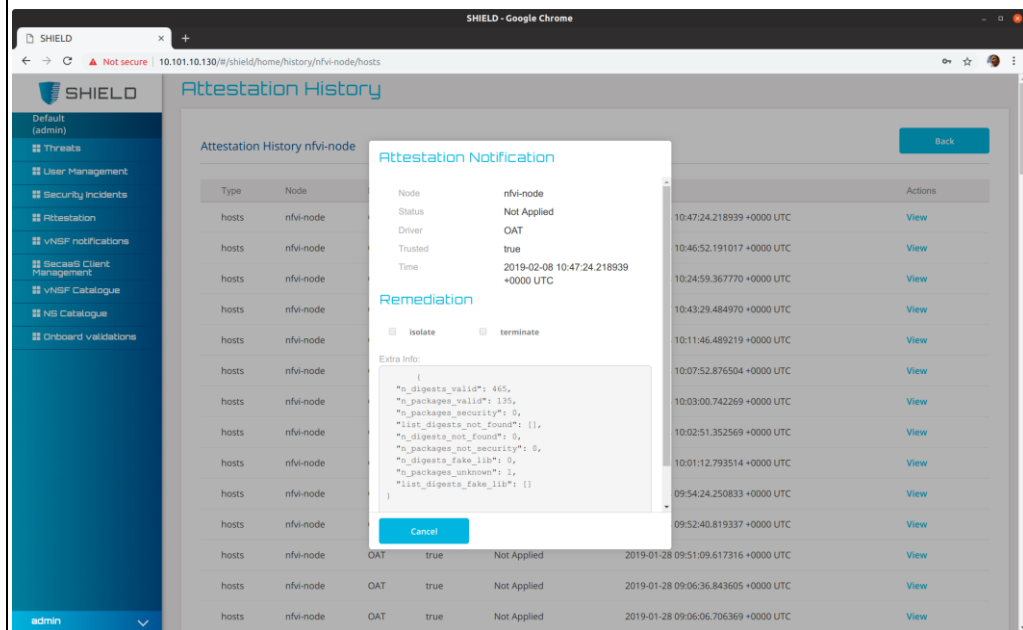
#6 Manual modification of the network element configuration

	 <p>The image shows a terminal window with the following configuration commands:</p> <pre> vlan 402 name "vlan402" untagged 13-15 ip address 10.102.2.3 255.255.255.0 exit vlan 403 name "vlan403" untagged 16-18 ip address 10.102.3.3 255.255.255.0 exit no tftp server tftp server listen data no autorun no dhcp config-file-update no dhcp image-file-update password manager password operator Orion-sw# conf term Orion-sw(config)# vlan 666 Orion-sw(vlan-666)# name "Rogue VLAN" Orion-sw(vlan-666)# exit Orion-sw(config)# no vlan 666 Orion-sw(config)# </pre> <p>Below the terminal is a screenshot of the SHIELD web interface. The main page is 'Attestation History' showing a table of events. A 'Remediation Notification' popup is displayed over the table, showing details for a specific event on 'switch-1'.</p> <p>Remediation Notification Details:</p> <ul style="list-style-type: none"> Node: switch-1 Driver: HPESwitch Trusted: true Status: Not Applied Time: 2019-02-28 16:36:32.769926415 +0000 UTC <p>Remediation Options:</p> <ul style="list-style-type: none"> <input type="checkbox"/> CheckFlows <input type="checkbox"/> Isolate <input type="checkbox"/> Reboot <input type="checkbox"/> Update <input type="checkbox"/> CheckConfig <p>Extra Info:</p> <pre> { "VerificationFailed": false, "CertSigned": true, "ConfigurationMatch": true, "Certificate": { } } </pre> <p>The background table in the screenshot has columns for Type, Node, Driver, Trusted, Status, Time, and Actions.</p>
<p>Verdict</p>	<p>Success</p>
<p>Comments</p>	<p>For the final validation, only the configuration modification has been tested; modification of the firmware, software, SDN rules and SDN controller has been demonstrated during the year 1 review and the test sequence is similar.</p>

Test Case ID	TC_P11			
Description	Compute infrastructure attestation			
Executed by	POLITO	Date	20/2/2019	
Purpose	The Trust Monitor is capable of detecting a manipulation on a NFVI compute host. The manipulation is represented by a custom application which is not part of the Trust Monitor Whitelist Database.			
Associated Requirements	PF19			
Components involved	Trust Monitor, Dashboard			
Tools	None			
Metrics	Attestation delay			
Pre-test conditions	The NFVI compute host is running and its configuration has not been tampered beforehand.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Run attestation check on the Dashboard from the ISP admin view	
	2	Check	The Dashboard shows a notification on the attestation check, with successful result	OK
	3	Stimulus	The NFVI compute host is manually tampered by loading a custom application (e.g. Bash script) that is not part of the Trust Monitor Whitelist Database	
	4	Check	The NFVI compute host runs the custom application, and its output is captured	OK
	5	Stimulus	Run attestation check on the NFVI infrastructure from the ISP admin view	
	6	Check	The Dashboard shows a notification on the attestation check, with failure	OK
	7	Stimulus	The ISP client checks the attestation result from the Dashboard view	
	8	Check	The Dashboard view shows that the custom application was launched in the NFVI compute host	OK
Evidence	#1 Attestation view in Dashboard (logged as ISP admin user)			



#2 Attestation details for NFVI compute host



#3 Manual execution of unmeasured script in NFVI compute host

```

torsec@nfvi-node:~/y2demo$ chmod +x unmeasured_script.sh
torsec@nfvi-node:~/y2demo$ ./unmeasured_script.sh
Unmeasured script is launched on NFVI-node!
torsec@nfvi-node:~/y2demo$

```

#4 Notification of subsequent periodic attestation in Dashboard (logged as ISP admin user)

SHIELD - Google Chrome

Attestation

TM Notification
New attestation data available

Attest NVF Infrastructure

Type: any

Type	Node	Driver	Trusted	Recommendation	Time	Actions
sdn	switch-1	HPESwitch	false	Not Applied	2019-01-18 14:04:10.726777245 +0000 UTC	View History Attest
hosts	nfvi-node	OAT	false	Not Applied	2019-02-08 11:02:30.312573 +0000 UTC	View History Attest

Showing 1 to 2 of 1

European Commission
Horizon 2020
European Union Funding
for Research & Innovation

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199

Trust Monitor API to attest the NFVI compute host

```

REQUEST: curl -X GET -k
https://<TRUST_MONITOR_BASE_URL>/nfvi_pop_attestation_info/?node_id=nfvi-node

```

```

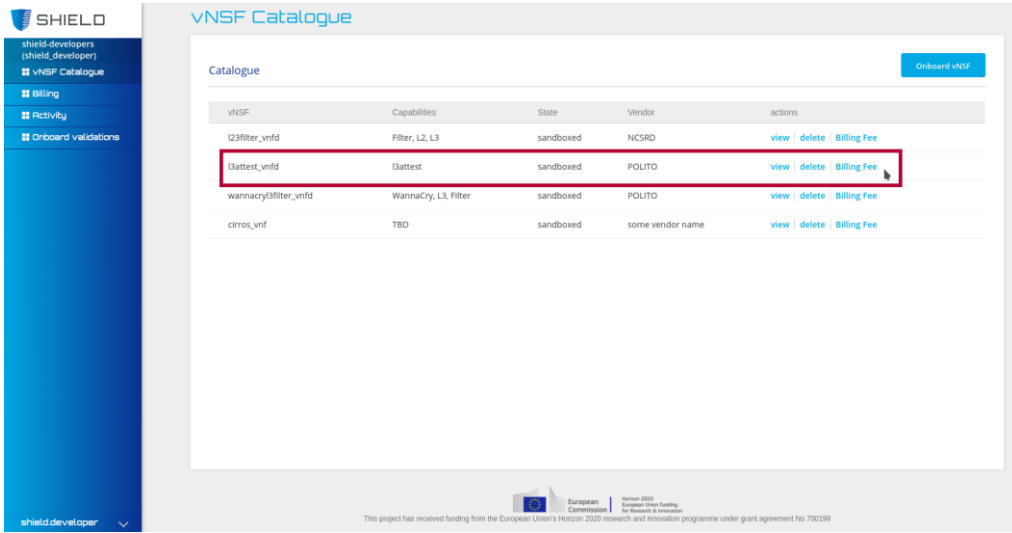
RESPONSE: {"hosts": [{"node": "nfvi-node", "status": 0, "time": "2019-02-08 10:47:24.218939 +0000 UTC", "remediation": {"terminate": false, "isolate": false}, "vnsfs": [], "trust": true, "driver": "OAT", "extra_info": {"n_digests_valid": 465, "n_packages_valid": 135, "list_digests_fake_lib": [], "n_packages_not_security": 0, "n_packages_unknown": 1, "n_packages_security": 0, "list_digests_not_found": [], "n_digests_not_found": 0, "n_digests_fake_lib": 0}}, {"sdn": [], "trust": true, "vtime": "2019-02-08 10:47:24.219215 +0000 UTC"}]}

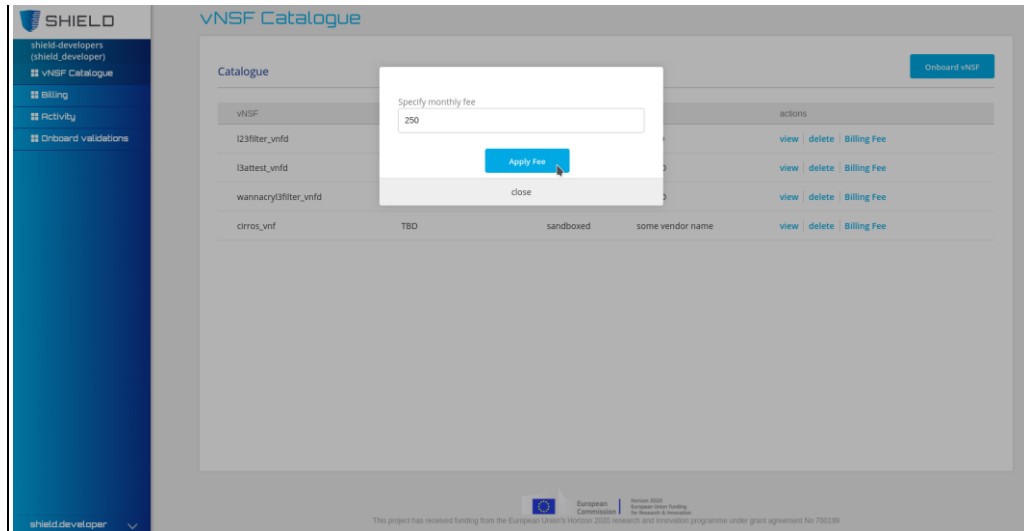
```

The attestation delay to verify the NFVI compute host is 11,567s seconds.

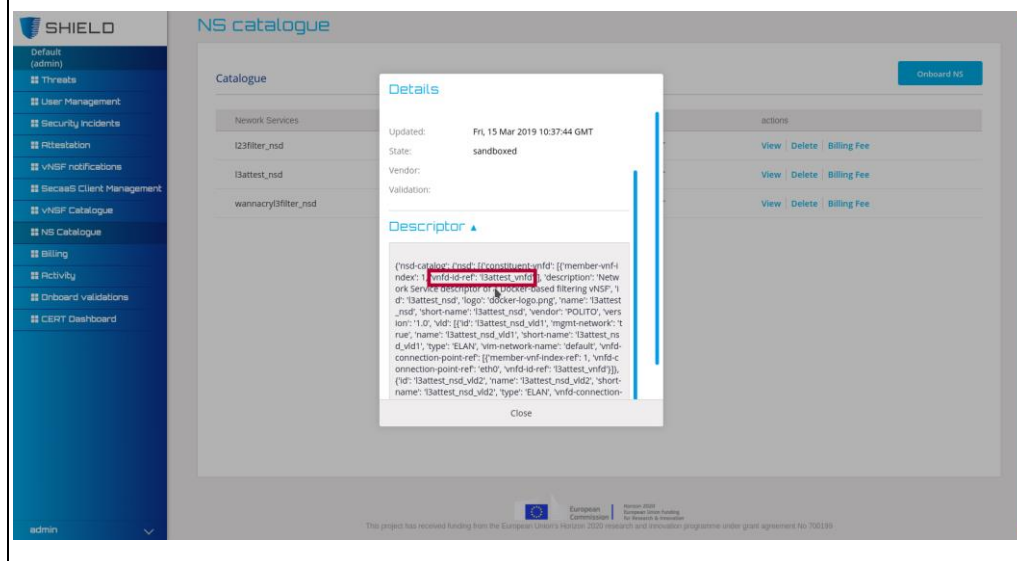
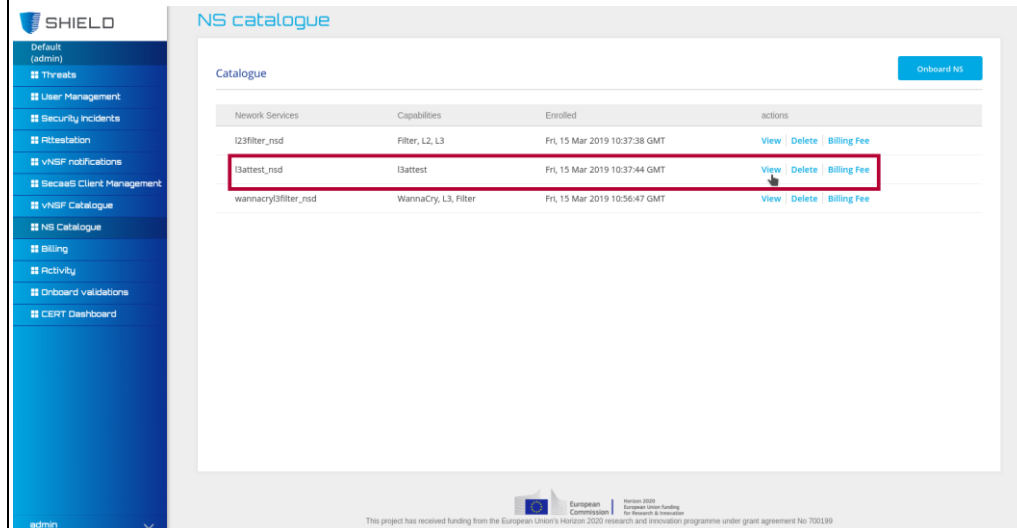
Verdict	Success
Comments	None

Test Case ID	TC_P12			
Description	Billing			
Executed by	UBI	Date	16/3/2019	
Purpose	Analyse the billing model flow by defining billing fees for vNSFs (provided by Developers), Network Services (provided by Platform administrators, e.g. ISP). The billing costs of Network Service instantiations should be supported by SecaaS Clients, whereas the billing costs of the vNSFs should be supported by the Platform/ISP.			
Associated Requirements	PF20			
Components involved	Dashboard			
Tools	-			
Metrics	Billing fee of a vNSF, Billing fee of a Network Service instance, Overall profit by the Platform/ISP			
Pre-test conditions	A vNSF is already onboarded. A Network Service is also already onboarded, which is constituted with the former onboarded vNSF.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	[As a Developer user] Having an onboarded vNSF in place, define its monthly billing fee.	
	2	Check	Check that the established billing fee is defined correctly.	OK
	3	Stimulus	[As Platform Administrator] Having an onboarded NS which uses/references the previous vNSF, define its monthly billing fee.	
	4	Check	During the definition of the NS billing fee it is possible to use a simple billing simulator to provide a general idea of potential profits. Establish that the NS billing fee is correctly defined.	OK
	5	Stimulus	[As SecaaS Client] Add/Enroll the Network Service to the client catalogue (NS Inventory) and trigger the creation of a new instance.	

	6	Check	The Billing panel should provide a resume on a monthly basis of the billable costs inherent to its Network Service instances. Particularly on this test, the details of the current month should report the usage of the new Network Service instance and display its associated costs, for the usage period, according to the established billing fees of the Platform/ISP.	OK																									
	7	Stimulus	[As Platform Administrator] The billing panel should have a resume on a monthly basis of the profits or losses obtained by Network Service instantiations. The details of each month should specify the details of all Network Service instantiations profits as well as the details of all vNSFs expenses, translating the overall profit/expense balance.																										
	8	Check	Regarding this particular test case, the Network Service instance billing fee as well as the used vNSF fee should be discriminated along with their usage dates and billing fees.	OK																									
	9	Stimulus	[As Developer user] Verify the billing information about the usage of its vNSF																										
	10	Check	Establish that the profits obtained resulting from the Network Service instantiation of the SecaaS client are accurate.	OK																									
Evidence	<p>#1-2 Specification of a vNSF billing fee</p>  <p>The screenshot displays the SHIELD vNSF Catalogue interface. On the left is a navigation menu with options like 'Shield developers', 'vNSF Catalogue', 'Billing', 'Activity', and 'Onboard validations'. The main content area shows a table titled 'Catalogue' with the following data:</p> <table border="1"> <thead> <tr> <th>vNSF</th> <th>Capabilities</th> <th>State</th> <th>Vendor</th> <th>actions</th> </tr> </thead> <tbody> <tr> <td>l23filter_vnfd</td> <td>Filter, L2, L3</td> <td>sandboxed</td> <td>NCSR</td> <td>view delete Billing Fee</td> </tr> <tr> <td>l3tattest_vnfd</td> <td>l3tattest</td> <td>sandboxed</td> <td>POLITO</td> <td>view delete Billing Fee</td> </tr> <tr> <td>wannacry3filter_vnfd</td> <td>WannaCry, L3, Filter</td> <td>sandboxed</td> <td>POLITO</td> <td>view delete Billing Fee</td> </tr> <tr> <td>cirros_vnfd</td> <td>TBD</td> <td>sandboxed</td> <td>some vendor name</td> <td>view delete Billing Fee</td> </tr> </tbody> </table> <p>The 'Billing Fee' link for the 'l3tattest_vnfd' instance is highlighted with a red box in the original image. At the bottom of the interface, there is a footer with the European Commission logo and text: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700399'.</p>				vNSF	Capabilities	State	Vendor	actions	l23filter_vnfd	Filter, L2, L3	sandboxed	NCSR	view delete Billing Fee	l3tattest_vnfd	l3tattest	sandboxed	POLITO	view delete Billing Fee	wannacry3filter_vnfd	WannaCry, L3, Filter	sandboxed	POLITO	view delete Billing Fee	cirros_vnfd	TBD	sandboxed	some vendor name	view delete Billing Fee
vNSF	Capabilities	State	Vendor	actions																									
l23filter_vnfd	Filter, L2, L3	sandboxed	NCSR	view delete Billing Fee																									
l3tattest_vnfd	l3tattest	sandboxed	POLITO	view delete Billing Fee																									
wannacry3filter_vnfd	WannaCry, L3, Filter	sandboxed	POLITO	view delete Billing Fee																									
cirros_vnfd	TBD	sandboxed	some vendor name	view delete Billing Fee																									



#3-4 Specification of a NS billing fee, which references the previous vNSF, using the simulator tool assistant



The screenshot shows the 'Assign Billing for NS' dialog box in the SHIELD NS catalogue. It features two input fields: 'Total monthly expense fee' with the value '250' and 'Specify monthly fee per instance' with the value '410'. Below these fields, it indicates 'Inherited from vNSF billing fee' and 'Specification of NS billing fee'. A 'Simulate' button is present. Below the form is a table with the following data:

Parameter	Description	# NS Instances	Fee
Balance per instance	Balance of a single instance (expense fee minus specified fee)	1	160
Active instances usage	Current active instances usage (assumed for active for the entire month)	0	0
Required instances	Minimum amount of instances to achieve profitability	0	0
Balance			160

Buttons for 'Apply Fee' and 'Close' are at the bottom of the dialog.

#5 Create a new Network Service instance

The screenshot shows the 'NS catalogue' page in SHIELD. It displays a table of Network Services with the following data:

Network Services	Capabilities	Enrolled	actions
l23filter_rsd	Filter, L2, L3	Fri, 15 Mar 2019 10:37:38 GMT	View Enroll
lsattest_rsd	lsattest	Fri, 15 Mar 2019 10:37:44 GMT	View Enroll
wannacry3filter_rsd	WannaCry, L3, Filter	Fri, 15 Mar 2019 10:56:47 GMT	View Enroll

The 'lsattest_rsd' row is highlighted with a red box, and the 'Enroll' link is being clicked.

The screenshot shows the 'NS inventory' page in SHIELD. It displays a table of Network Service instances with the following data:

Network Service	Capabilities	Status	Instance ID	actions
lsattest_rsd	lsattest	available		view withdraw instanciate

The 'lsattest_rsd' row is highlighted with a red box, and the 'instanciate' link is being clicked. A pagination indicator shows 'Showing 1 to 1 of 1'.

Inventory

Network Service	Capabilities	Status	Instance ID	actions
I3atetest_ns	I3atetest	running	df989fa0-93c3-41fb-b6d8-4f5cflab1cdc	view terminate

Showing 1 to 1 of 1

#6 As the SecaaS Client, show the monthly summary of the billing fee as well as the details for a particular month

Billing

Year-Month	# Network Services	# NS Instances	Status	Billable Fee (€)	actions
2019-03	1	1	open	13.24	Details

Showing 1 to 1 (1 records)

Billing Detail 2019-03

Network Service	Instance ID	Status	Used from	Used to	Monthly fee (€)	Monthly usage (%)	Billable Fee (€)
I3atetest_ns	df989fa0-93c3-41fb-b6d8-4f5cflab1cdc	open	2019-03-20	2019-03-20	4.10	3.23	13.24

Total Amount (€) 13.24 €

Showing 1 to 1 (1 records)

#7-8 As the Platform administrator, show the monthly summary of the billing fee as well as the details for a particular month

The screenshot shows the SHIELD Billing dashboard. On the left is a navigation menu with options like Threats, User Management, Security Incidents, etc. The main content area is titled 'Billing' and contains a table with the following data:

Year-Month	# SecaaS Client	# Network Services	# NS Instances	# vNSFs	Status	Profit Balance (€)	actions
2019-03	1	1	1	1	open	5.17	Details

At the bottom of the dashboard, there is a footer with logos for the European Commission and Horizon 2020, and a note: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700199'.

The screenshot shows the 'Billing Detail 2019-03' page. It provides a breakdown of costs and profits. The 'Network Services' section shows a total billable fee of 13.24 €. The 'vNSFs' section shows an expense fee of 8.07 €. The final 'Total Profit Balance' is 5.17 €.

SecaaS Client	Network Service	Instance ID	Instance Status	Used from	Used to	Monthly fee (€)	Monthly usage (%)	Billable Fee (€)
shield-uw	lbattest_nsd	df989fa0-93c3-41fb-b6d8-4f5cfa11c6c	open	2019-03-20	2019-03-20	410	3.23	13.24

User	vNSF	Status	Used from	Used to	Monthly fee (€)	Monthly usage (%)	Expense Fee (€)
shield.developer	lbattest_vnrfid	active	2019-03-20	2019-03-20	250	3.23	8.07

Network Service Profit	13.24 €
vNSF Expense	8.07 €
Total Profit Balance (€)	5.17 €

The footer is identical to the previous screenshot, mentioning funding from the European Union's Horizon 2020 programme.

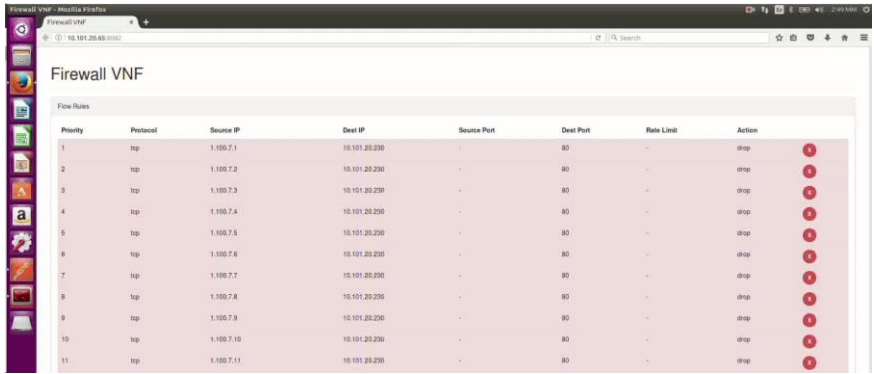
#9-10 As a Developer, show the monthly summary of billing fee as well as the details for a particular month

<p>Verdict</p>	<p>Success</p>
<p>Comments</p>	<p>The ecosystem provided by the SHIELD billing model provides both a contextual view according to the user role and an overall picture dedicated to the Platform administrator. Considering the segmentation on a monthly basis, it allows the financial department of the Platform/ISP to obtain an accurate perspective of the imputed costs, therefore facilitating not only the cost/benefit of the provided Network Services but also the most important SecaaS Clients using the SHIELD platform.</p>

11.2. Service tests

Test Case ID	TC_S01			
Description	Application of traffic rules			
Executed by	ORION, NCSR, POLITO, I2CAT, INFIL, TALAIA	Date	22/2/2019	
Purpose	To show that remediation actions that control network traffic reach the vNSFs and they in turn apply appropriate blocking or rate limiting rules to selected traffic.			
Associated Requirements	SF01, SF02, SF04, SF08, SF09			
Components involved	The SHIELD DARE is in charge of detecting anomalies and providing a remediation recipe. The SHIELD dashboard shows the remediation options to the user, who selects a specific measure to be applied. The appropriate rules are relayed to a specified SHIELD vNSF through the security orchestrator and then applied to the vNSF. In some cases, such as the IDPS vNSF, the rule may be applied directly through the vNSF's user interface (without need for the DARE to intervene); in this case, the dashboard user is promptly notified. Multiple remediation options may be possibly sent to the dashboard.			
Tools	tcpdump to create traffic mixes ingested by the DARE to trigger the remediation, traffic captures.			
Metrics	Deployment time of the vNSF. Time elapsed from ingestion of traffic to generation of remediation action in the DARE. Time elapsed from the selection of a remediation rule to the application on the vNSF.			
Pre-test conditions	Ensure that vNSFs, MANO and DARE components are running.			
Test Sequence	Step	Type	Description	Result

	1	Stimulus	Replay a copy of normal traffic and start the attack tool	A mix of normal and attack traffic is ingested by DARE's Apache Spot or Security Analytics engine.
	2	Check	The DARE component has successfully detected the anomaly	A csv file with contextual information is sent to the remediation engine
	3	Check	The remediation engine received the csv file on the anomaly	The remediation engine provides a recommended remediation recipe
	4	Check	The dashboard receives the remediation recipe	The recipe is visualised in the dashboard
	5	Stimulus	The dashboard user selects the remediation option	The option is sent to the appropriate vNSF
	6	Check	The vNSFS receive and apply the recommendation	The results are observed in the network traffic.

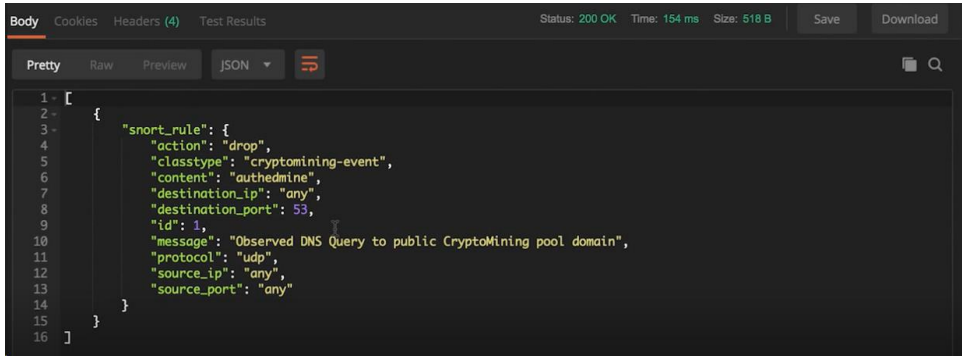
Evidence	 <p style="margin-top: 10px;">Blocking rules on NSCRD FW.</p>
-----------------	--



Classification per protocol and domain in ORION vDPI

```
<mspl-set>
  <it-resource id="vNSF2">
    <configuration xsi:type="filtering-configuration">
      <default-action>drop</default-action>
      <resolution-strategy>FMR</resolution-strategy>
      <rule>
        <priority>1</priority>
        <action>accept</action>
        <condition>
          <packet-filter-condition>
            <direction>inbound</direction>
            <source-address>91.211.1.100</source-address>
            <source-port>*</source-port>
            <destination-address>147.83.110.33</destination-address>
            <destination-port>22</destination-port>
            <protocol>TCP</protocol>
          </packet-filter-condition>
          <traffic-flow-condition>
            <rate-limit>50/s</rate-limit>
          </traffic-flow-condition>
        </condition>
      </rule>
    </configuration>
  </it-resource>
</mspl-set>
```

Recipe for rate limiting.

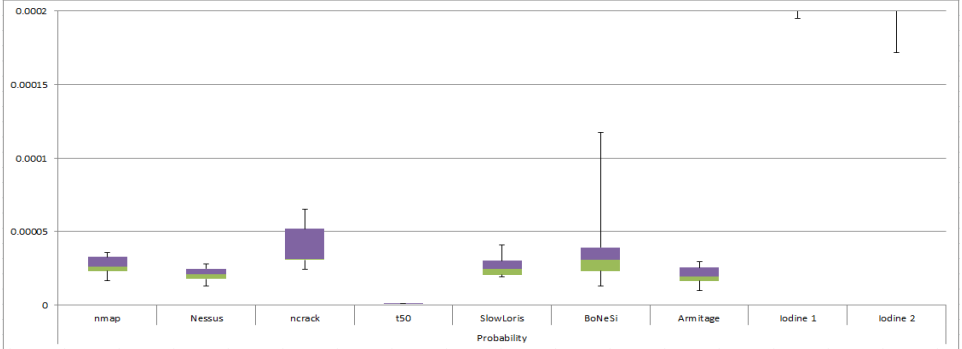
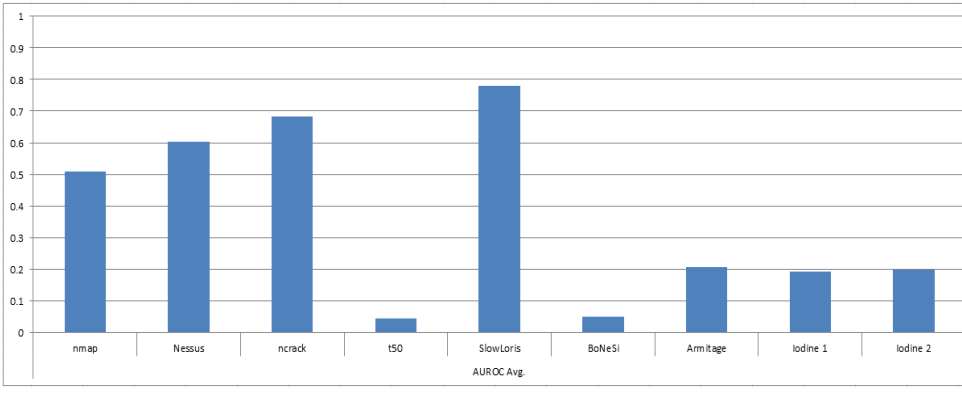
	 <pre> 1 [2 { 3 "snort_rule": { 4 "action": "drop", 5 "classtype": "cryptomining-event", 6 "content": "authedmine", 7 "destination_ip": "any", 8 "destination_port": 53, 9 "id": 1, 10 "message": "Observed DNS Query to public CryptoMining pool domain", 11 "protocol": "udp", 12 "source_ip": "any", 13 "source_port": "any" 14 } 15 } 16] </pre> <p>Application of block rule for DNS traffic in NCSRID IDPS.</p>
Verdict	The SHIELD vNSFs managed to apply the necessary rules. Traffic monitoring showed that the traffic output had the desired characteristics.
Comments	The completion of this test is verifiable since blocking and rate limiting have been shown in multiple SHIELD demonstrations.

Test Case ID	TC_S02		
Description	DoS protection		
Executed by	ORION, NCSRID, TALAIA, INFILI, POLITO	Date	20/2/2019
Purpose	The SHIELD systems detect and mitigate different types of DoS/DDoS attacks.		
Associated Requirements	SF04, SF05, SF08		
Components involved	SHIELD vNSFs, SHIELD DARE (SA/CA engines and remediation engine)		
Tools	<p>The following tools were used to simulate a variety of (D)DoS attacks, ranging from flood-based to protocol-based attacks.</p> <ul style="list-style-type: none"> • BoNeSi (to launch TCP & UDP floods) • HULK (HTTP flood) • Slowloris (Partial POST requests) 		

	Captures of normal, non-offending traffic and tcpreplay are used to simulate normal conditions in the network prior to and during the attack.			
Metrics	Response time Detection rate ROC/AUROC curves			
Pre-test conditions	Ensure that vNSFs, MANO and DARE components are running.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Replay a copy of normal traffic and start the attack tool	A mix of normal and attack traffic is ingested by DARE's Apache Spot or Security Analytics engine.
	2	Check	The DARE component has successfully detected the anomaly	A csv file with contextual information is sent to the remediation engine
	3	Check	The remediation engine received the csv file on the anomaly	The remediation engine provides a recommended remediation recipe
	4	Check	The dashboard receives the remediation recipe	The recipe is visualised in the dashboard
	5	Stimulus	The dashboard user selects the remediation option	The option is sent to the appropriate vNSF
	6	Check	The vNSFS receive and apply the recommendation.	The results are observed in the network traffic.
Evidence	SHIELD Y1/Y2 demos. Results are documented in the paper: C. M. Mathas, O. Segou, G. Xylouris, D. Christinakis, M. A. Kourtis, C. Vassilakis and A. Kourtis. 2018. In SIG Proceedings of CyberTIM Workshop, ARES conference, Hamburg, Germany, August 2018 (CyberTIM Workshop, ARES Conference 2018), 9 pages.			


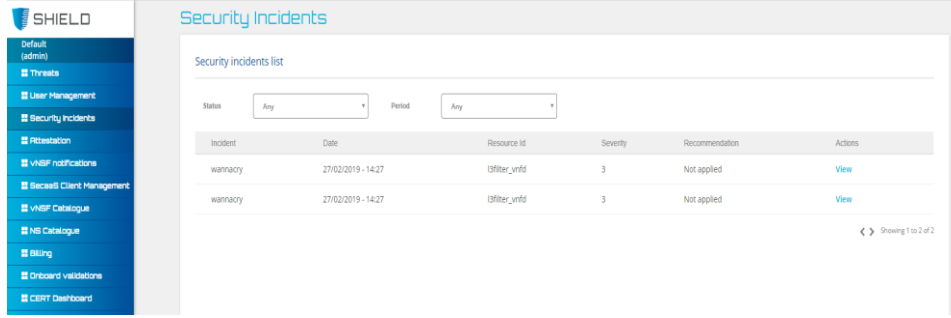
Verdict	(D)DoS attacks are detected and mitigated in a timely manner with sufficient detection rate. False positive detection occurs more commonly than false negatives.
Comments	This is verifiable through the DDoS Y1 demo and Slowloris Y2 demo.

Test Case ID	TC_S03			
Description	Protection from vulnerability exploitations			
Executed by	NCSR, ORION	Date	10/2/2019	
Purpose	To test Apache Spot's detection capabilities against popular tools that scan for vulnerabilities or exploit frameworks.			
Associated Requirements	SF03, SF05			
Components involved	A clean, baseline version of Apache Spot, attack tools etc. set up in a VM, a VM replaying normal traffic within the network. Further analysis using Python, Excel and Matlab.			
Tools	The following tools were used to simulate scanning and vulnerability exploitations: <ul style="list-style-type: none"> • Nmap • Nexus • Ncrack • T50 • Armitage • Metasploit framework 			
Metrics	Response time, Detection rate, ROC/AUROC curves, Apache Spot threat index (probability)			
Pre-test conditions	A testing framework to launch scanning attacks etc. mixed with normal traffic, a functioning "clean" version of Spot.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Replay normal traffic and start the attack simulation tools	A mix of normal and attack traffic is ingested in Spot

	2	Check	Spot threat index results	Export to timestamped csv files for further analysis
Evidence	<p>Threat Index (Probability)</p>  <p>Average AUROC value</p> 			
Verdict	Partial – detection characteristics can be improved, boxplots indicate highly asymmetrical PDFs.			
Comments	None			

Test Case ID	TC_S04		
Description	Malware protection		
Executed by	I2CAT, TID	Date	22/2/2019
Purpose	Verify and assess SHIELD capabilities to identify and mitigate malware activity		
Associated Requirements	SF06		

Components involved	vDPI vNSF, DARE, Dashboard			
Tools	<p>The following tools have been used:</p> <ol style="list-style-type: none"> 1. TCPReplay 2. vDPI 3. OSM 4. Anomaly detection: Autoencoder 5. Classification: Random forest 6. Recipe generation 7. Dashboard 			
Metrics	<p>Accuracy</p> <p>Precision</p> <p>Recall</p> <p>Response time</p>			
Pre-test conditions	Ensure that the whole infrastructure and components are up and running			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Collect network traffic	Netflow ingested
	2	Stimulus	Run anomaly detection and classifies	Threats classified
	3	Check	No malware detected	true
	4	Stimulus	Reproduce and collect malware in the network	Netflow ingested
	5	Stimulus	Run anomaly detection and classifies	Threats classified
	3	Check	Malware detected	true
	Evidence	<p>Evidence of working is the year two demo and the pilots (year 2.5 demo)</p> <p>Also the following screenshots show the process. Firstly the launching of the anomaly detection + classification + recipe generation</p>		

	 <p>Secondly, the recipes shown in the dashboard</p> 
<p>Verdict</p>	<p>Success. The 5 IPs corresponding to wannacy are detected and no other IP is classified as wannacy</p>
<p>Comments</p>	<p>-</p>

<p>Test Case ID</p>	<p>TC_S05</p>		
<p>Description</p>	<p>Protection from data exfiltration</p>		
<p>Executed by</p>	<p>ORION, NCSR, TALAIA, INFIL, POLITO</p>	<p>Date</p>	<p>15/1/2019</p>
<p>Purpose</p>	<p>The SHIELD systems detect and mitigate a DNS tunneling attack, which is a common data exfiltration method.</p>		
<p>Associated Requirements</p>	<p>SF02, SF04, SF05</p>		
<p>Components involved</p>	<p>SHIELD DARE (CA engine and remediation engine)</p>		

Tools	<p>The following tools were used to simulate scanning and vulnerability exploitations</p> <ul style="list-style-type: none"> • Iodine was used to create the tunnel to the client • iPerf was used to generate “malicious” traffic between the server and the endpoint. • A traffic generator that loads webpages has been used to create “normal” DNS traffic. • Tshark was used to capture all the traffic in .pcap files in order to have them ingested by the DARE. 			
Metrics	<p>Response time</p> <p>Detection rate</p> <p>ROC/AUROC curves</p>			
Pre-test conditions	<p>Ensure that the DARE components are running.</p>			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	Iodine sets a tunnel between the compromised client and a malicious authoritative DNS server.	The tunnel is set.
	2	Stimulus	iPerf generates traffic between the server and the client representing malicious commands.	The traffic is successfully received and decrypted by the malicious server.
	3	Stimulus	Benign traffic is being simulated by a traffic generator that loads webpages.	The traffic is successfully received and decrypted by the malicious server.
	4	Stimulus	Tshark is used to capture all the traffic packets (.pcap) and sent to the DARE for ingestion.	Traffic is successfully ingested in the DARE Hadoop filesystem (HDFS).
	5	Stimulus	The anomaly detection module of the DARE is performing a suspicious connects analysis to detect	The DARE successfully detects the attack.

			the attack and provide a visual representation.	
	6	Check	The DARE component has successfully detected the anomaly	A csv file with contextual information is sent to the Dashboard and to the Remediation engine.
	7	Check	The Dashboard is used to present the detection results.	A list of the malicious packets is shown in the Dashboard.
	8	Stimulus	The Remediation Engine receives the analysis' results to create a mitigation policy. that will block the connection between the server and the endpoint.	The HSPL rule is translated down to a set of machine-readable policies (MSPL) and is forwarded to actuating vNSFs.
	9	Check	The Remediation Engine has created a mitigation policy for the threat.	An MSPL rule has been successfully created and can be used as input from an actuating vNSF.
Evidence	<p>SHIELD Y1 demo.</p> <p>Results are documented in the paper:</p> <p>C. M. Mathas, O. Segou, G. Xylouris, D. Christinakis, M. A. Kourtis, C. Vassilakis and A. Kourtis. 2018. In SIG Proceedings of CyberTIM Workshop, ARES conference, Hamburg, Germany, August 2018 (CyberTIM Workshop, ARES Conference 2018), 9 pages.</p>			
Verdict	<p>The results were successful in the case of DNS tunnelling as the attack was detected by the DARE. It should be noted that the engine produced a significant number of false-positives, which can be partially attributed to the fact that the utilised machine learning algorithm (based on Apache Spot) required a large amount of normal traffic to be trained and create a reference model for it, which could not be simulated in lab conditions. As normal traffic increases, true positive rates increase as well.</p>			
Comments	<p>This is verifiable through the DNS Tunnelling Attack Y1 demo.</p>			

Test Case ID	TC_S06			
Description	Protection from malicious web scripts			
Executed by	NCSRD, ORION, UBI	Date	10/1/2019	
Purpose	Blocking of malicious scripts should be done on network level, before they reach the user, even without intervention from the SHIELD DARE.			
Associated Requirements	SF06, SF07, SF09			
Components involved	The IDPS vNSF, the SHIELD dashboard.			
Tools	A page infected with coinhive cryptominer.			
Metrics	Rapid deployment of IDPS service, rapid deployment of IDPS rules.			
Pre-test conditions	A functioning instance of the IDPS service and dashboard.			
Test Sequence	Step	Type	Description	Result
	1	Stimulus	A user visits the infected website	The malicious script starts mining
	2	Check	Application of the rule to the IDPS service	The malicious script stops working
Evidence	Shown in Y2 Cryptojacking demonstration.			
Verdict	Successful. This test shows the capacity of a service to protect against many types of malicious scripts, without the need to configure individual devices with protection measures (e.g. browser extensions, antivirus software etc.)			
Comments	None.			

REFERENCES

- [1] L. Law, Susan Sabett and J. Solinas., “How to Make a Mint: The Cryptography of Anonymous Electronic Cash,” *American University Law Review*, vol. 46, no. 6, 11 January 1997.
- [2] “Bitcoin main website,” [Online]. Available: <https://bitcoin.org/en/>. [Accessed February 2019].
- [3] “Ethereum main website,” [Online]. Available: <https://www.ethereum.org/>. [Accessed February 2019].
- [4] “Monero main website,” [Online]. Available: <https://ww.getmonero.org/>. [Accessed February 2019].
- [5] A. Narayanan, J. Bonneau, E. Felten, A. Miller and S. Goldfeder, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*, Princeton: Princeton University Press, 2016.
- [6] CheckPoint Software Technologies Ltd, “December's Most Wanted Malware: Crypto-Miners Affect 55% of Businesses WorldWide,” 2017.
- [7] “Coinhive main website,” [Online]. Available: <https://coinhive.com/>. [Accessed February 2019].
- [8] H. Partz, “Coinhive Code Found on 300+ Websites Worldwide in Recent Cryptojacking Campaign,” [Online]. Available: <https://cointelegraph.com/news/coinhive-code-found-on-300-websites-worldwide-in-recent-cryptojacking-campaign>. [Accessed February 2019].
- [9] D. Goodin, “Now even YouTube serves ads with CPU-draining cryptocurrency miners,” *ArsTechnica*, 26 January 2018. [Online]. Available: <https://arstechnica.com/information-technology/2018/01/now-even-youtube-serves-ads-with-cpu-draining-cryptocurrency-miners/>. [Accessed February 2019].
- [10] Kaspersky Lab, “Loapi - This Trojan is hot!,” 18 December 2017. [Online]. Available: <https://www.kaspersky.com/blog/loapi-trojan/20510/>. [Accessed February 2019].
- [11] “Stratum mining protocol,” [Online]. Available: https://en.bitcoinwiki.org/wiki/Stratum_mining_protocol. [Accessed February 2019].
- [12] Joint Research Center, “JRC guide to technological roadmapping,” [Online]. Available: http://forlearn.jrc.ec.europa.eu/guide/4_methodology/meth_roadmapping.htm. [Accessed February 2019].
- [13] ETSI ENI ISG, <https://www.etsi.org/technologies/experiential-networked-intelligence>
- [14] ENI ongoing PoCs, [Online]. Available: https://eniwiki.etsi.org/index.php?title=Ongoing_PoCs
- [15] Apache Ambari, [Online]. Available: <https://ambari.apache.org/>
- [16] Splunk Enterprise analytics, [Online]. Available: <https://www.splunk.com/>
- [17] Elastic stack, [Online]. Available: <https://www.elastic.co/>

- [18] IHS Market, “New Regulations impact EMEA cybersecurity market,” [Online]. Available: <https://news.ihsmarket.com/press-release/technology/new-regulations-impact-emea-cybersecurity-market-2016-ihs-says> . [Accessed February 2019].
- [19] Ovum, “Defining the next-gen managed security services provider,” 2017.
- [20] Forrester, “Forrester Vendor Landscape: Global Managed Security Services,” 2017.
- [21] ENISA, “Exploring the opportunities and limitations of current Threat Intelligence platforms,” 2018.

LIST OF ACRONYMS

Acronym	Description
BoNeSi	BotNet Simulator
DDoS	Distributed Denial of Service
DD4BC	DDoS for Bitcoin
DoS	Denial of Service
DNS	Domain Name System
GUI	Graphical User Interface
HSPL	High-level Security Policy Language
IMA	Integrity Measurement Architecture
ISP	Internet Service Provider
MANO	Management and Orchestration
MSPL	Medium-level Security Policy Language
NFV	Network Function Virtualisation
NFVI	Network Function Virtualisation Infrastructure
NS	Network Service
KVM	Kernel-based Virtual Machine
OSM	Open Source MANO
SDN	Software Defined Networking
SecaaS	Security-as-a-Service
PCR	Platform Configuration Register
TC	Trusted Computing
TCP	Transmission Control Protocol
TPM	Trusted Platform Module
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VNSF	Virtual Network Security Function
VNSFO	Virtual Network Security Function Orchestrator
VIM	Virtual Infrastructure Manager
VM	Virtual Machine