Deliverable D2.2

Use Cases and Planned Attack Scenarios

Document type	Deliverable		
Title	D2.2 – Use cases and planned attack scenarios		
Contractual due date	M06	Actual submission date	01 March 2021
Nature	Report	Dissemination Level	Public
Lead Beneficiary	ULANCS		
Responsible Author	Arash Bozorgchenani (ULANC)		
Contributions from	Antonios Gouglidis (ULANC), Qiang Ni (ULANC), Charilaos Zarakovitis (NCSRD), Wissam Mallouli (MONT), Anna Prudnikova (SECURA), Zujany Salazar (MONT), Eftychia Makri (CERTH)		

Grant Agreement No	952672	Acronym	SANCUS
Full Title	Analysis Software Scheme of Uniform Statistical Sampling, Audit and Defence Processes		
Start Date	01/09/2020	Duration	36 months
Project URL	https://www.sancus-project.eu/		

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Revision history

Version	Issue Date	Changes	Contributor(s)
0.1	18/01/2021	Initial Structure	Qiang Ni (ULANC) Antonios Gouglidis (ULANC) Arash Bozorgchenani (ULANC)
0.2	08/02/2021	PUC1 contributions	Wissam Mallouli (MONT) Zujany Salazar (MONT)
0.3	08/02/2021	PUC2 contributions	Anna Prudnikova (SECURA)
0.4	09/02/2021	PUC3 contributions	Qiang Ni (ULANC) Antonios Gouglidis (ULANC) Arash Bozorgchenani (ULANC) Charilaos Zarakovitis (NCSRD)
0.5	18/02/2021	PUC1, PUC2, PUC3 updates	MONT, SECURA, ULANC
0.6	25/02/2021	PUC2 updates	Eftychia Makri (CERTH) Konstantinos Votis (CERTH)
0.7	25/02/2021	Peer review	Antonios Lalas (CERTH)
0.8	25/02/2021	Peer review	Dimitris Klonidis (UBITECH)
1.0	01/03/2021	Review by coordinator	Hicham Khalifé (THALES)

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Table of Contents

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

List of Figures

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Glossary of terms and abbreviations used

Abbreviation / Term	Description
PUC	Pilot Use Case
FiV	Firmware Inspection Validation
CiV	Code Integrity Verification
SiD	System Intelligent Defence
MiU	Modelling of Individual Unit
GiO	Game Implicit Optimization
AcE	Attack Configuration Engine
ICT	Information and Communications Technology
3GPP	The 3rd Generation Partnership Project
ETSI	European Telecommunications Standards Institute
NFV	Network Function Virtualisation
CNM	Container Network Management
WIM	Wide area network Infrastructure Manager
MANO	Management and Orchestration
NFVO	NFV Orchestrator
OSS	Operation Support System
RAN	Radio Access Network
MEC	Multi-access Edge Computing
IoT	Internet of Things
VNF	Virtual Network Function
NF	Network Function
EIR	Equipment Identity Register
NRF	Network Repository Function
NSSF	Network Slice Selection Function
AUSF	Authentication Server Function
UDM	Unified Data Management
PCF	Policy Control Function

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

SMF	Session Management Function
UPF	User plane function
gNB	Generation NodeB
AMF	Access and Mobility Management Function
VM	Virtual Machine
CWE	Common Weakness Enumeration
ENISA	The European Union Agency for Cybersecurity
DoS	Denial of Service
DDoS	Distributed Denial of Service
SDN	Software Defined Networking
DPI	Deep Packet Inspection
MiTM	Man-in-the-Middle
CEP	Complex Event Processing
APT	Advanced Persistent Threats
IDS	Intrusion Detection Systems
QoS	Quality of Service

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Executive Summary

This report provides a list of use cases and attack scenarios that will be used to demonstrate the effectiveness of the automated detection and protection tools developed in the SANCUS project against cyber threats. This deliverable is the output of Task 2.3: Use cases definition and planning of attack scenarios, performed within WP2: Use cases, requirements and architecture.

SANCUS identifies three Pilot Use Cases (PUCs) for catching up with its performance on Firmware Layer, Virtualisation Layer and Management Layer including variations of them. This deliverable defines and elaborates the existing number of use cases in order to focus on the activities that will be undertaken during the SANCUS project.

PUC1 addresses virtualisation and management layers by focusing on representation of end-to-end services for the overall lifecycle and orchestration of 5G and IoT applications and network services. PUC2, on the other hand, targets the firmware layer and examines the performance of FiV and CiV engines and studies whether the traditional manual testing can be replaced with an automated testing. PUC3, which is the most comprehensive use case, addressing all the three layers of virtualisation, management and firmware and makes recommendations with the aim of enhancing the network and security infrastructure by performing an optimisation on security-vs-privacy-vs-reliability of the network.

The three PUCs are described following a structured way to uniformly elaborate on the description, requirements, actions and outcomes of each of the PUCs. A set of threat scenarios are also mapped to the requirements of SANCUS stakeholders and are related to conditions stemming from detecting cyber threats. These scenarios have emerged according to the actual identification of security concerns and the connectivity approaches we have studied in Task 2.1: Identification and classification of recent security concerns in the OEM supply chain and Task 2.2: Connectivity approaches in the scope of 5G cloud-native system network. Eventually, the pilot use cases will form the basis for the demonstration and validation activities and provide a focus in the project’s core activities.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

1 Introduction

In this document, the initial SANCUS architecture is introduced and the add-ons required to enable a higher level of automation amongst the components are elaborated. In order to examine the performance of the various engines in the project, three Pilot Use Cases (PUCs) are introduced and the role of the engines and the attack and threats scenarios in each of the PUCs is elaborated.

We recall that SANCUS had been initially approached as a new security suite to enable joint firmware and software validation and verification, as well as security modelling and optimisation by integrating **six main security engines**. These are:

1. FiV – Firmware Inspection Validation engine
2. CiV – Code Integrity Verification engine
3. SiD – System Intelligent Defence engine
4. MiU – Modelling of Individual Unit engine
5. GiO – Game Implicit Optimization engine
6. AcE – Attack Configuration Engine

The initial theoretical approach of the SANCUS architecture, including the connectivity of the individual engines and the cloud-native Information and Communications Technology (ICT) system network testbed prototype can be depicted at the top of **Figure 1**.

We note that after thorough investigation under the architecture design and interconnectivity activities of T2.2, the Consortium concluded that should be two important add-ons, such that the initial theoretical architecture can be integrated tangibly as a unified operational security suite. These add-ons regard the addition of a Security Orchestrator, and three Event Triggering Modules to be attached at all engines (i.e. FiV/CiV, SiD, and MiU/GiO) to enable an automated inter-operation of these engines. The upgraded configuration of the SANCUS architecture is depicted at the bottom of **Figure 1**. Details on the added modules are presented in section 5 of D2.1.

We emphasise that these add-ons are added not because the Consortium's has envisioned the initial SANCUS architecture partially or mistaken, but they are added because the Consortium decided to implement the optimal protection recommendation, while in the initial proposed system, SANCUS was only to provide optimal security recommendations. As such, we expect this effort to be considered as an important technological upgrade and valuable offering of the project to the Research Community.

With regards to our architecture, we adopt Demokritos NCSRD's combined Edge computing and 5G telecom infrastructure, which is fully compliant to the latest 3GPP and European Telecommunications Standards Institute (ETSI) Network Function Virtualisation (NFV) specifications and integrates the NOKIA Container Network Management (CNM) system and at Wide area network Infrastructure Manager (WIM). We also adopt the Open Source Management and Orchestration (MANO) NFV Orchestrator (NFVO), and OpenStack runtime. Furthermore, we exploit the NCSRD network Operation Support System (OSS/BSS), and the vertical application orchestrator, so as, the control framework will allow dynamic orchestration, deployment and monitoring of vertical applications and network services. The testbed Radio Access

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

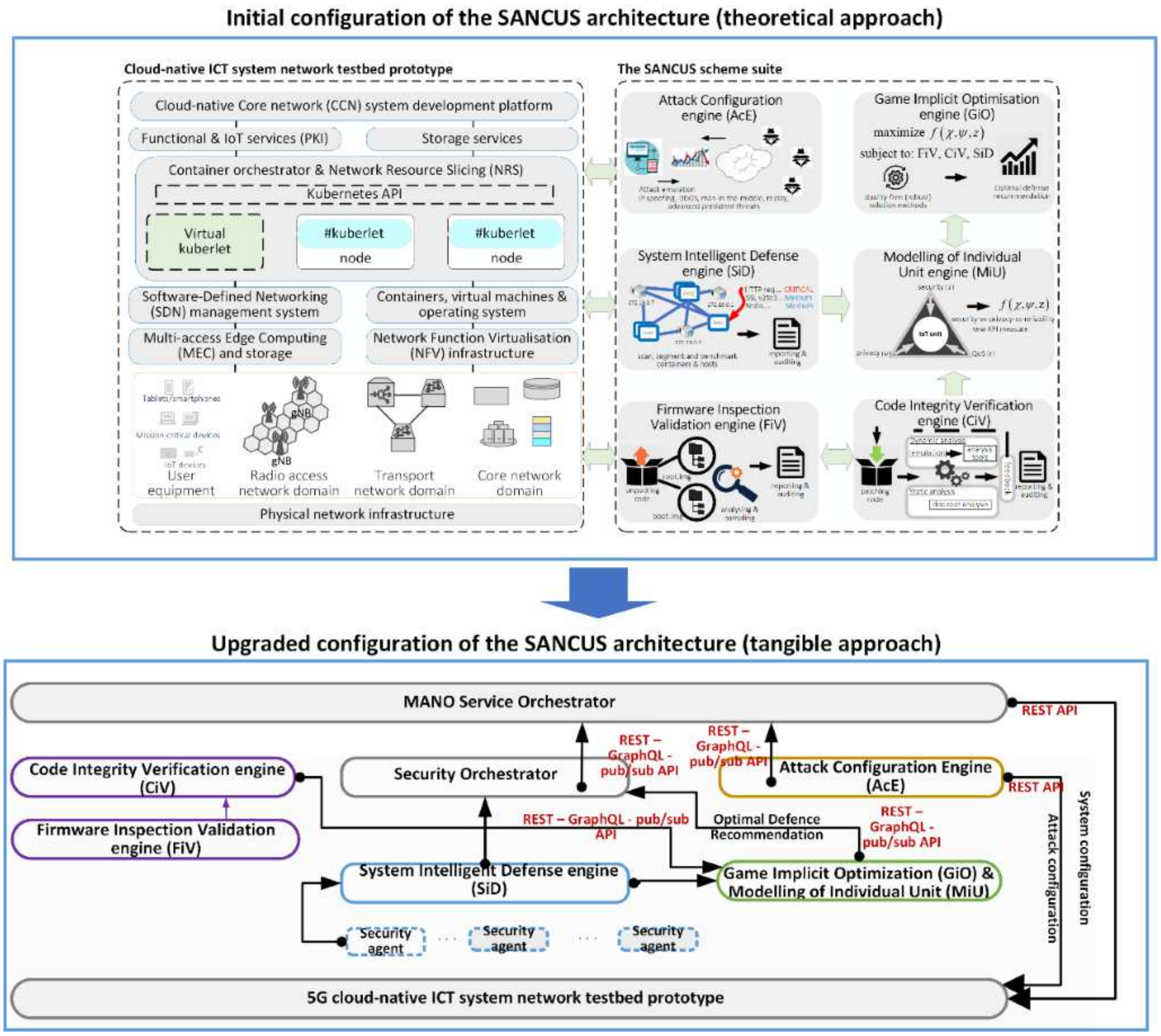
Initial configuration of the SANCUS architecture (theoretical approach)
Cloud-native ICT system network testbed prototype
Cloud-native Core network (CCN) system development platform
Functional & IoT services (PKI)
Storage services
Container orchestrator & Network Resource Slicing (NRS)
Kubernetes API
Virtual kuberlet
#kuberlet node
#kuberlet node
Software-Defined Networking (SDN) management system
Containers, virtual machines & operating system
Multi-access Edge Computing (MEC) and storage
Network Function Virtualisation (NFV) infrastructure
User equipment
Radio access network domain
Transport network domain
Core network domain
Physical network infrastructure
The SANCUS scheme suite
Attack Configuration engine (AcE)
Game Implicit Optimisation engine (GiO)
maximize $f(\chi,\psi,z)$
subject to: FiV, CiV, SiD
System Intelligent Defense engine (SiD)
Modelling of Individual Unit engine (MiU)
Firmware Inspection Validation engine (FiV)
Code Integrity Verification engine (CiV)
Upgraded configuration of the SANCUS architecture (tangible approach)
MANO Service Orchestrator
REST API
Code Integrity Verification engine (CiV)
Firmware Inspection Validation engine (FiV)
Security Orchestrator
REST – GraphQL - pub/sub API
REST – GraphQL - pub/sub API
Attack Configuration Engine (AcE)
REST API
REST – GraphQL - pub/sub API
Optimal Defence Recommendation
REST – GraphQL - pub/sub API
System Intelligent Defense engine (SiD)
Game Implicit Optimization (GiO) & Modelling of Individual Unit (MiU)
Security agent
Security agent
Security agent
System configuration
Attack configuration
5G cloud-native ICT system network testbed prototype

Figure 1: The SANCUS architecture. Top: initial approach for only extracting the optimal defence recommendation. Bottom: upgraded approach for extracting-and-applying the optimal defence recommendation.

Network (RAN) infrastructure is composed of high-performance equipment and latest generation programmable radio. The user will be able to execute the attacks at various levels and towards different targets. For example, it will be possible to attack Multi-access Edge Computing (MEC) applications for Internet of Things (IoT) systems from/to network connected devices, or to attack container/Virtual Network Function (VNF) networks services, or even single control building blocks (e.g. OpenStack, OSM, Docker, etc.). It is noted that the testbed is suitable to integrate the SANCUS scheme suite to test any type of attack scenarios requested by the platform user. Furthermore, it is remarked that the testbed provides three interfaces for configuration, monitoring and emulation of attacker entities (i.e. the IoT abstraction,

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

cloud-based and open software (configuration) interfaces) to interact with the SANCUS platform and the remote network platforms of our Partners THALES and NOKIA.

In order to validate the performance of different engines and examine their automation process, we define the following three PUCs.

- PUC1 targets the virtualisation and management layers. It focuses on representing the end-to-end services for the overall lifecycle and orchestration of 5G and IoT applications and network services.
- PUC2 addresses attacks and threats in the firmware layer. It will mostly focus on examining the performance of FiV and CiV engines; specifically, how effectively and efficiently the firmware validation and verification processes can be performed, and most importantly, if the traditional manual testing can be replaced by automated testing.
- PUC3 is a more comprehensive use case compared to the previous two PUCs by targeting all three layers considered in the SANCUS project, i.e. virtualisation, management and firmware layers. It will focus on the SANCUS efficacy in assessing the network environment heterogeneity and making specific recommendations to enhance the network and security infrastructure by optimising the security-vs-privacy-vs-reliability performance of the network.

In order to provide information in a uniform way for all three use cases, a structured way of describing them has been followed. For each of the PUCs a brief description, engaging modules, pre- and post-conditions, a basic flow or sequence of the actions of the use case and example threat scenarios are described. The modules in each PUC indicate the main engines that are involved in the processes of each PUC. Following, a list of pre-conditions, i.e. enablers of the PUCs are provided and characterized by any inputs and requirements for the engines. Subsequently, a basic flow (sequence of actions) of the use case is described, which demonstrates dependencies amongst engines or modules. The flow diagrams depict the way engines may operate (e.g. sequentially, concurrently) with respect to each other. A list of post-conditions is defined as the output of each PUC when the performance of one or more participating engines is evaluated, and furthermore, validated. Post-conditions can be the result of the task(s) associated to each PUC. Finally, a list of attack scenarios of variable threat level (i.e. easy, medium and high) are associated to the participating engines in each PUC.

The structure of the report is as follows. Section 2 describes PUC1, including the pre-conditions, post-conditions, the involved modules and platforms, sequence of actions and some example threat scenarios. The same description structure about PUC2 is described in Section 3. Section 4 is dedicated to PUC3 with a similar structure as the previous PUCs. The report concludes in Section 5.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

2 Pilot Use Case 1

2.1 Brief Description

PUC1 focuses on representing the end-to-end services of 5G and IoT applications and network services. The goal is to perform a continuous risk assessment process in the network, end-user's equipment and virtualised infrastructure.

PUC1's AcE performs as a network traffic and threat generator, modelling complex cyber-attacks over the functions and interfaces of a 5G service-based architecture, using the 5G virtualised core provided by NOKIA. Alongside, the SiD performs runtime monitoring over the system, in order to detect threats and vulnerabilities, and trigger the MiU and GiO engines, that allow PUC3 to perform mitigation actions. PUC1 aims to emulate and detect security incidents in the user equipment.

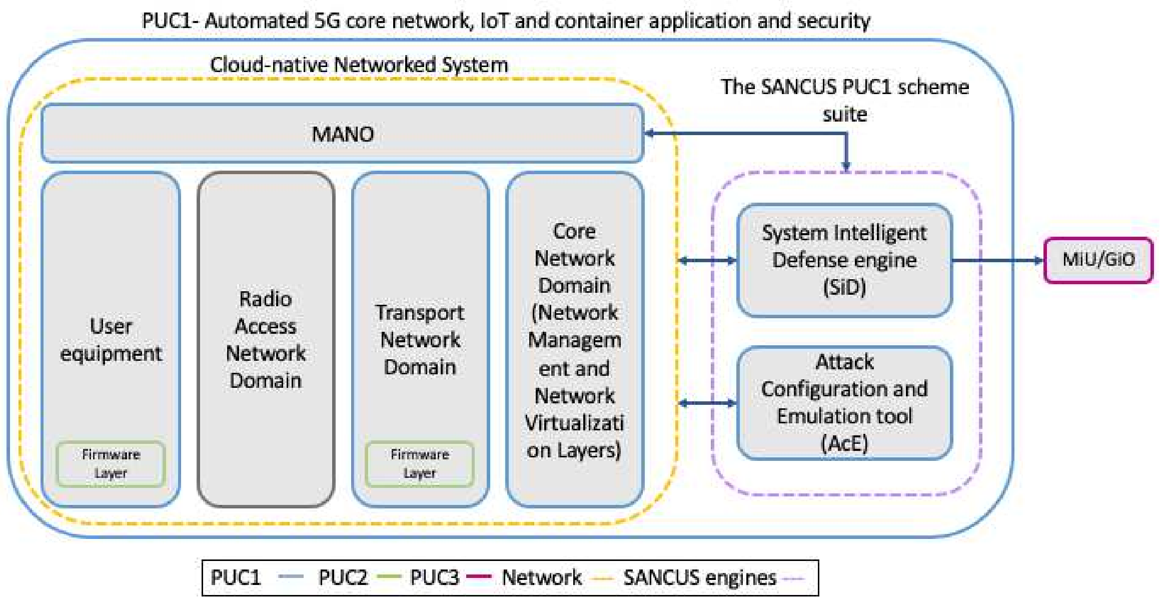
PUC1- Automated 5G core network, IoT and container application and security
Cloud-native Networked System
The SANCUS PUC1 scheme suite
MANO
User equipment
Firmware Layer
Radio Access Network Domain
Transport Network Domain
Firmware Layer
Core Network Domain (Network Management and Network Virtualization Layers)
System Intelligent Defense engine (SiD)
Attack Configuration and Emulation tool (AcE)
MiU/GiO
PUC1
PUC2
PUC3
Network
SANCUS engines

Figure 2: PUC1- Automated 5G core network, IoT and container application and security

2.2 Modules

Figure 2 depicts how the PUC1 will target specific threats at the user equipment, and transport and core networks domain. PUC1 is composed of the following engines:

Attack Configuration Engine (AcE): it is used for configuring and emulating a cloud-native 5G network composed of a set container services, applications and security mechanisms, as well as, acting as network traffic and threat generator that models complex cyber-attacks. It is composed of the following modules:

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

- System configurator
- Traffic generator
- Attack injector
- ML attack modeling and emulations

System Intelligent Defense (SiD) : The SiD engine provides automated validation and verification of the security of open-source systems and it dynamically assess the risks of running containers against threats, attacks, ransomware, viruses, breakouts and other suspicious activities. It is composed of the following modules:

- Asset identification and classification
- Vulnerability identification
- Rule/ML-based threat identification
- Root cause analysis
- Continuous risk value computation and forecasting
- Alert reporting
- MiU/GiO connector
- Security agents
- Security administrator dashboard

Cloud-native Networked System: it comprises all the basic Network Function (NF) elements that are needed for common 5G applications. These elements are deployed as cloud instances of the relevant network functions through either open stack or containers virtualisation approach.

2.3 Pre-conditions

One key pre-condition is that PUC1 refers to a cloud-native 5G network topology (as is the case in all state of the art 5G systems considered today), thus applying the SBA in a containerised approach.

The second pre-condition considers that security agents are pre-deployed in the network during the implementation and verification of the scenarios. As part of SiD, the security agents are distributed probes over the system, with the objective of measuring, collecting, analyzing and reporting events on the network. It is noted that the deployment phase of functions and agents is not under the scope of the project since the focus is on the identification of vulnerabilities.

With respect to the AcE engine, there are also two main inputs that are required, namely, a collection of publicly available datasets of network vulnerabilities, and network topologies, and a collection of scripts to automate the deployment of attacks and topologies.

2.4 Sequence of actions

System Intelligent Defence (SiD): The sequence of actions for the SiD engine is as follows. The security agents are first set through the security administrator dashboard (web-based application) over a pre-

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

deployed system. Then, the Asset identification and classification module identifies and classifies (i) the location, (ii) associated business processes, (iii) traffic at data elements, and (iv) threats and risks associated to each data element, in the network, end-users' equipment and the virtualised infrastructure. Security agents monitor the network and send reports with network-based events to the Continuous risk value computation and forecasting module. The Rule/ML-based threat identification module identifies existing and unknown threats, through the security agents deployed on the network. It includes (i) the identification of system attacks and security events, as well as (ii) the identification of current and future security issues. The vulnerability identification module identifies threats and vulnerabilities on the system, relaying on the Common Weakness Enumeration (CWE) open database of vulnerabilities. The Continuous risk value computation and forecasting module quantifies the risk of the detected threats, as a function of the likelihood, the magnitude of their impact, and the capability of applying mitigating controls at network level. Additionally, it predicts the impact of future risks in terms of joint security and cost. Relevant dashboards are displayed to security operators. Then, the root cause analysis module performs a systematic analysis for identifying "root causes" of problems or events. Finally, the alert reporting module produces reports of threats and vulnerabilities in raw format. The MiU/GiO connector module formats the reports in matrix form and triggers the MiU/GiO engines in PUC3.

Attack Configuration Engine (AcE): The sequence of actions for the SiD engine is as follows. The System configurator deploys the targeted network topology, based on the collection of network topologies. Then, the traffic generator generates legitimate 5G network traffic. Once, the traffic is generated, the Attack injector generates malicious traffic, playing as an attacker based on the collection of network vulnerabilities and scripts to generate the attack message flow. Finally, the ML attack modeling and emulations module combines the legitimate and illegitimate traffic and inject it into the networks, emulating attacker/defender patterns.

2.5 Post-conditions

The post-conditions for the PUC1 include (but not necessarily limited to)

- list of attacks on emulated traffic (AcE output),
- list of detected attacks
- root cause of detected attacks,
- list of alerts to be sent to the MiU and GiO,
- forecast of detected risks impact in joint terms of security and cost,
- network status and visualization of risk level of the running ICT system on the dashboard.

2.6 Example threat Scenarios

PUC1- Scenario A (PUC1.A)
Threat severity: Low
Detection: Signature-based

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

This section will target generic threats that European Union Agency for Cybersecurity (ENISA) defines around the 5G mobile network [1]:

1. **Denial of Service (DoS):** DoS is a threat in which a bad actor aims to make a network resource unreachable by provisionally or indeterminately interfering or disrupting the network service. The attack encompasses making an immense number of requests or traffic flow in a way that the network suits partly or completely unavailable for regular users. Numerous types of threats may lead to a DoS such as flooding, amplification, signaling storm and saturation attacks. An attack combining multiple vectors may lead to a distributed DoS (DDoS) attack. 5G edge and core components, are suitable to this type of threat, for example by flooding attacks of core network components, or authentication traffic spikes.
2. **Data breach, leak, theft destruction and manipulation of information:** this threat is against integrity, and confidentiality of information, related to subscribers, business, configuration, services or networking. This comprises the theft, unauthorized access and potential publication of subscribers' information, company confidential information (intellectual property, commercial and financial data) or government/state-related information (classified information). Moreover, user credentials, encryption keys, network security logs, software configuration, etc. may be stolen and benefit malicious actor conducting different types of attacks and crimes, such as fraud.
3. **Eavesdropping/Interception/Hijacking**: this threat involves the tampering of the application and communication layers from the 5G network elements (Software Defined Networking (SDN) controller, network function, edge node, virtualisation orchestrator). It includes the eavesdropping on subscriber' data, confidential information, system time, subscriber location, electronic messages, signal of data relayed over the network. This threat classification comprises man in the middle and Session hijacking attacks, as well as traffic sniffing.
4. **Abuse of authentication:** This threat comprises the abuse the 5G authentication systems by threat actors, using diverse techniques such as the theft of user credentials, brute force of user accounts, password cracking, user identity masking and impairment of an IoT grouping authentication. It targets the entry points of the network: user equipment (mobile devices and IoT), operation and management interfaces, roaming and vertical services.
5. **Identity theft or spoofing:** In this attack, the attacker spoofs the identity of a legitimate network actor and interacts with the network functions controlled by the legitimate controller (i.e., elements of the data plane) to trigger several other types malicious activity (instigate network flows, divert traffic, etc.). It includes the use of social engineering, brute force user account/password cracking as techniques to spoof or steal user credentials.

PUC1- Scenario B (PUC1.B)
Threat severity: Medium
Detection: ML/AI based

This section will target encrypted traffic. It focuses on threats that can be performed by authorized but malicious users (internal attacks), by relying in encryption techniques to avoid the mechanisms for

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

detecting their unauthorized activity, occult interaction with the command servers of malicious programs, steal data inside encrypted traffic, manage command and control communications, etc.

Deep Packet Inspection (DPI) techniques have been used to identify malicious activity in encrypted traffic. It requires a significant computational effort because the amount of encrypted traffic normally is quite large. The analysis of statistical features extracted from the traffic has proven to provide a high classification accuracy [2]. Subsequently, machine learning techniques are widely used in order to detect anomalous activities that denote possible cyber-attacks, which also carries questions about the privacy of the end-users.

Afterwards, the main challenge in this PUC1 scenario is detecting malicious activity in encrypted traffic in a fashion that balances the requirements of privacy, utility and performance. Several attacks such as DoS, Man-in-the-Middle (MiTM) and impersonation attack will be detected within ML/AI techniques, as well as attempts to deliver malware or steal sensitive information.

PUC1- Scenario C (PUC1.C)
Threat severity: High
Detection: Complex Event Processing (CEP)

This detection will target Advanced Persistent Threats (APT) specified in WP5. The fundamental difference between APT and convectional attacks is that in an APT a malicious actor aims to gain control of a specific (network of) system(s), while continuing undetected for an extensive period of time. The adversary frequently relies on zero-day exploits, to ensure maintaining continued control over a recently compromised system.

Conventional Intrusion Detection Systems (IDS) are not well-suited to APTs. Signature-based detectors are inadequate to attacks that exploit new vulnerabilities. Anomaly-based systems have concerns to model long-term behaviour patterns, and they are subject to evasion techniques, since they normally examine only short sequences of system calls and events. Moreover, there are systems that attempt to capture long-term program behaviours, nevertheless they have been limited to avoid high computational and memory overheads [3].

APTs, are considered the most challenging to detect and defend against, especially in 5G cloud-native environments due to their highly distributed architecture. APTs can hugely impact on NFV by affecting untargeted services and tenants hosted in the same physical host.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

3 Pilot Use Case 2

3.1 Brief Description

PUC2 focuses on performing the security compliance checks, as well as risk assessment on the IoT devices connected to SANCUS testbed on the firmware level. This PUC starts when firmware images from the end-user IoT devices are extracted. The firmware will be automatically analysed, and any actual security risks will be reported to the risk management platform based on the combination of outcome produced by the FiV and CiV engines.

The extraction of the firmware is performed by FiV. During the extraction process, sanitisation techniques are used to prepare any potentially corrupted files of firmware to prepare them for further code integrity verification performed by CiV. The end result of the use case includes the risk report produced by a combination of FiV and CiV engines working together. An overview of PUC2 is depicted in **Figure 3**.

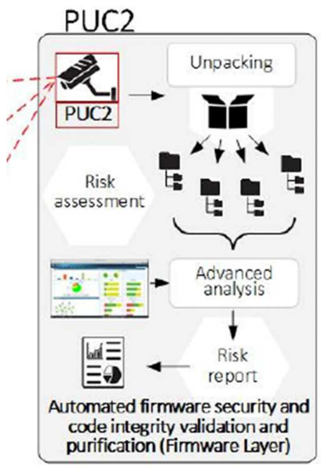
PUC2
PUC2
Unpacking
Risk assessment
Advanced analysis
Risk report
Automated firmware security and code integrity validation and purification (Firmware Layer)

Figure 3: Overview of PUC2

The goal of PUC2 is to examine the performance of FiV and CiV engines in how effectively and efficiently they can perform the firmware validation and verification process. No mitigation measures of any identified vulnerabilities will be applied within PUC2. The end result of the FiV and CiV validation process is the report containing risk assessment results. Moreover, the output of FiV and CiV will be used to verify whether MiU and GiO in combination can make decisions on risk acceptance / rejection or any other action to be implemented for identified risks. On top of that, the output of FiV and CiV might be provided to AcE as an input to perform validation of whether those vulnerabilities are exploitable on network level.

It is proposed to run CiV for 8 hours and automatically stop the process after this period of time (as an example taken from requirements from Common Criteria). Otherwise, the process might be infinite. The use case ends when the report with identified findings is prepared to be sent as an input for PUC3. for the other step for PUC2 might include sending the list of vulnerabilities to AcE engine as an input for PUC1.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Main indicative examples of PUC2 validations include:

- How effectively SANCUS can perform firmware unpacking and find the current risks
- Fine-grain and improve accuracy of the results regarding firmware risks that arise from the devices with zero impact in the network(s)
- Produce the same results as if in a manual way
- Quantify the security risk and confidence level for:
    - Both known and unknown vulnerabilities
    - Outdated components
    - Code and unsafe functions with changing quality over time
    - Memory corruptions per file
    - Undocumented and hard-coded credentials and cryptographic material
    - Weak authentication
    - Debuggers and compilers
    - Weakly encrypted connections
    - Vulnerable components and interfaces
    - Potentially vulnerable binaries and scripts, among others.

3.2 Modules

The following modules/engines are involved in PUC2:

- FiV: Automated vulnerability inspection management engine which combines static (symbolic) and dynamic analysers to maximize the surface of vulnerability discovery, where multiple different unpackers and samplers will allow for searching through binary firmware images and recompose fragmented code portions for smoother and faster verification
- CiV: A new method for enabling precise and scalable taint analysis taken into consideration that taint analysis is a demand driven problem, which enables lazy computation of vulnerable information flows, instead of computing a complete data-flow solution, which is the reason for the traditional contrast between scalability and precision.

3.3 Pre-conditions

To perform PUC2, the following pre-conditions should be satisfied:

- The firmware has to be supported by the existing FiV’s firmware unpackers and CiV's emulator (currently ARM64 with a potential expansion to MIPS)
- Location of engines: the server with FiV and CiV might be located on Secura's premises

3.4 Sequence of actions

The flow of the PUC is shown in **Figures 4 and 5,** and is performed in the following steps:

- The firmware image is manually extracted from router

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

- The firmware image is uploaded on the server where FiV and CiV reside
- The firmware image is processed by FiV, the result is a thousand of unpacked files prepared for further analysis
- The extracted firmware files are automatically transferred to CiV after the unpacking process is over
- The extracted firmware files are processed by CiV, whereas the results are a list of identified vulnerabilities with scores (in the form of a report)
- The report is manually transferred to MiU+GiO to perform PUC3
- Additional step includes manual transfer of the report to AcE as an input to try and exploit identified vulnerabilities.

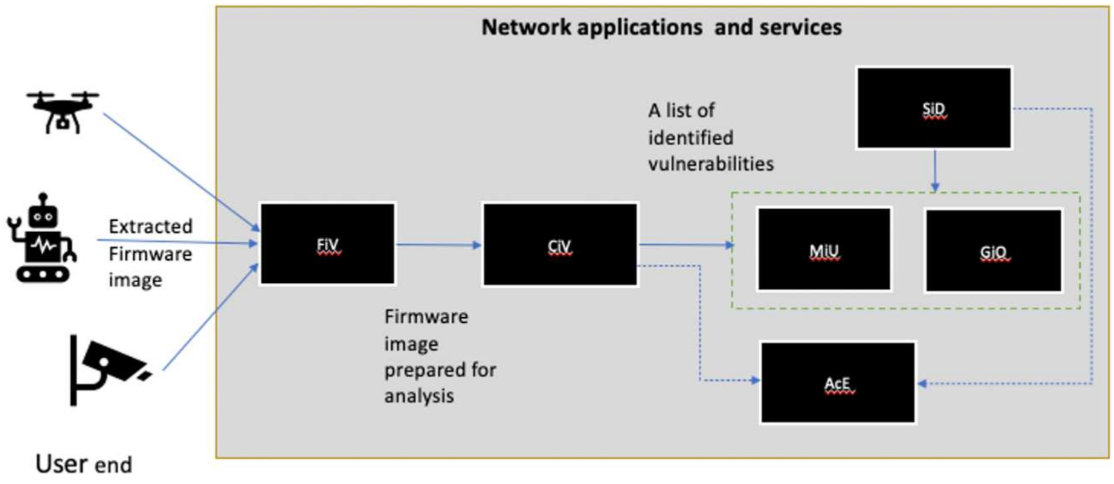
Network applications and services
A list of identified vulnerabilities
SiD
Extracted Firmware image
FiV
CiV
MiU
GiO
Firmware image prepared for analysis
AcE
User end

Figure 4: Sequence of the actions among the engines

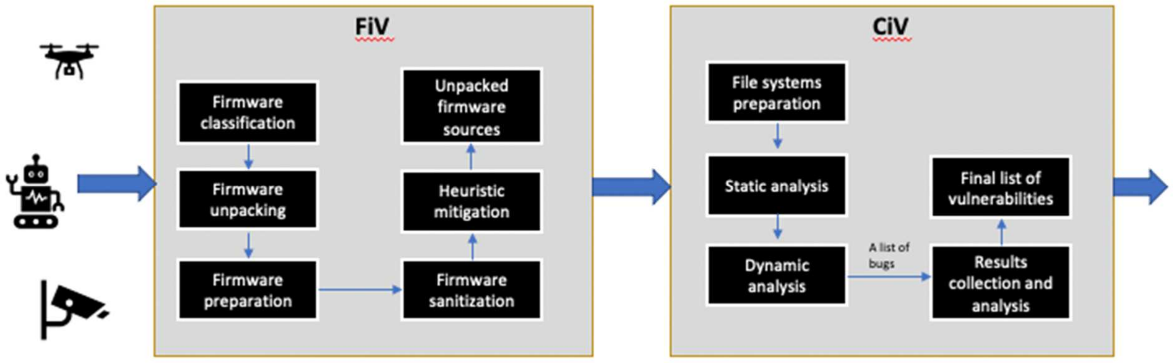

Figure 5: Illustration of the automated firmware validation and verification process driven by FiV and CiV engines

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

3.5 Post-conditions

In the general case, a list of vulnerabilities will be provided to the 'user/operator' of the system. This data can be used by the AcE and the MiU/GiO engines. The output includes (and is not limited to):

- List and description of vulnerabilities in the firmware
- Initial score for each vulnerability (without trying to exploit) based on CVSS
- A final score including results of exploitability for each vulnerability (a base is CVSS) based on the results of exploitability from AcE
- Information about vulnerabilities to form a decision on what needs to be done next based on a decision-making system.

3.6 Example Threat Scenarios

The main threat that will be examined in PUC2 is the presence of vulnerabilities in the firmware of IoT devices. Based on the type of vulnerability and another factors, each vulnerability will be scored to form a risk assessment report. Moreover, the threat of exploiting identified vulnerabilities by a malicious actor will be examined. Based on different types of potentially identified vulnerabilities, the following threats will be examined [4]:

- Use after free, threat severity is Low;
- Nullpointer dereference, threat severity is Low.
- Integer Overflow, threat severity is Medium
- Buffer overflows, threat severity is High;
- Format string attacks, threat severity is High

The description for each potential threat on a firmware level for IoT device is presented below.

PUC2- Scenario A (PUC2.A)
Threat severity: Low

Exploitation of use after free vulnerability: The use of previously freed memory can have any number of adverse consequences, ranging from the corruption of valid data to the execution of arbitrary code, depending on the instantiation and timing of the flaw. The simplest way data corruption may occur involves the system's reuse of the freed memory. Use-after-free errors have two common and sometimes overlapping causes:

- Error conditions and other exceptional circumstances.
- Confusion over which part of the program is responsible for freeing the memory.

In this scenario, the memory in question is allocated to another pointer validly at some point after it has been freed. The original pointer to the freed memory is used again and points to somewhere within the new allocation. As the data is changed, it corrupts the validly used memory; this induces undefined behaviour in the process [5]. It can impact:

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

- **Integrity**: the use of previously freed memory may corrupt valid data, if the memory area in question has been allocated and used properly elsewhere.
- **Availability**: if chunk consolidation occurs after the use of previously freed data, the process may crash when invalid data is used as chunk information.
- **Access Control (instruction processing)**: if malicious data is entered before chunk consolidation can take place, it may be possible to take advantage of a write-what-where primitive to execute arbitrary code.

PUC2- Scenario B (PUC2.B)
Threat severity: Low

Exploitation of Null-pointer dereference vulnerability: The program can potentially dereference a null-pointer, thereby raising a Null-Pointer Exception. Null-pointer errors are usually the result of one or more programmer assumptions being violated. Most null-pointer issues result in general software reliability problems, but if an attacker can intentionally trigger a null-pointer dereference, the attacker might be able to use the resulting exception to bypass security logic or to cause the firmware to reveal debugging information that will be valuable in planning subsequent attacks. A null-pointer dereference takes place when a pointer with a value of NULL is used as though it pointed to a valid memory area [6]. It has impact on availability: null-pointer dereferences invariably result in the failure of the process.

PUC2- Scenario C (PUC2.C)
Threat severity: Medium

Exploitation of Integer Overflow vulnerability: An integer overflow is possible to happen, when there is an attempt to store inside an integer variable a value that is bigger than the maximum value the variable can hold [7]. For example, when the calculation 3,454,680,369 + 1 is performed and there is an attempt to store the result that is greater than the maximum value for the integer type, the result of the storing, depends completely on the language and the compiler. However, most languages and compilers raise no error at all and perform a wraparound, modulo operation, or truncation, or they have other Undefined Behaviour (UB). In this situation, the result is most often 0. This means, that if an integer overflow happens during financial calculations, it may, for example, result in the customer receiving credit instead of paying for a purchase or may cause a negative account balance to become positive. More specifically, it might reverse the flow of money from a victim's account into the attacker's. A very big positive number in a bank transfer could be cast as a signed integer by a back-end system. Furthermore, an integer overflow vulnerability can lead to a buffer overflow. It can impact [8]:

- **Availability**: In terms of technical impact it is possible to provoke DoS like: crash, exit, restart, Resource Consumption (CPU), Resource Consumption (Memory) or Instability. In such case, this weakness generally leads to undefined behaviour, causing crashes or a very high chance of experiencing infinite loops is being created. The risk of overflow increases by using loop variables.
- **Integrity**: In terms of technical impact it is possible to cause memory modification. For example, if the value in question is important to data, simple data corruption has occurred. Additionally, if the

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

wrap around results in other conditions such as buffer overflows, further memory corruption may occur.

- **Access Control, Confidentiality**: In terms of technical impact, it is possible to cause unauthorized execution of code or commands and the protected mechanism could be bypassed. The vulnerability can lead to buffer overflows, as mentioned above, which, in this case, is possible to allow the adversary to execute arbitrary code.

<u>**PUC2- Scenario D (PUC2.D)**</u>
Threat severity: High

Exploitation of Buffer overflows vulnerability: A memory buffer bound threat concern, arises from the improper restriction of operations within the bounds of a memory buffer. Application performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer. The threat applies particularly to certain programming languages that allow direct addressing of memory locations and do not automatically ensure that these locations are valid for the memory buffer that is being referenced. This can cause read or write operations to be performed on memory locations that may be associated with other variables, data structures, or internal program data. If the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow. This has impact on all three confidentiality, integrity and availability [9].

<u>**PUC2- Scenario E (PUC2.E)**</u>
Threat severity: High

Exploitation of Format String vulnerability: The format string argument carries information on how the parameters passed to the output stream are to be formatted. According to this information, the format string is evaluated and for each format specifier, a value from the execution program stack is pushed and converted into an acceptable form, in order to be passed to the output stream [10]. For example, when a snippet is occurred, the format string "%s" is accessing the argument 'str' from the program execution stack, which would have been popped into the stack by the caller function. After that, the argument appears on the screen. The name of a format string vulnerability is a 'channeling problem'. The 'channeling problem' is defined as a problem which occurs when two different types of information channels are merged into one and the active channel is distinguished by using special escape characters or sequences. The term 'data channel' is used to describe most of the channels, which are not parsed actively but just copied, while the term 'controlling channel' is used to describe the second channel. This malfunction can become a severe security threat if the input that is used in one channel is supplied. To sum up, channeling problems are no security holes itself, but they make bugs exploitable [11]. They can impact:

- **Confidentiality:** In technical terms, the "Read Memory" error is possible to appear. This can lead to allowance of information disclosure which can make exploitation of the program more simple and easy.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

- **Integrity, Availability:** In technical terms, format strings problems can lead to modification of memory and execution of unauthorized code or commands. Furthermore, like integer overflow, these problems can result in the execution of arbitrary code.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

4 Pilot Use Case 3

4.1 Brief Description

Keeping firewalls, routers, switches and other devices configured to minimise risks and to comply with security policies is time-consuming and costly, especially for large networks. Effective security policy management also requires device-level analysis, network-level compliance analysis and risk assessment of ongoing changes, e.g. network, firmware. The design, deployment and support of such solutions for ensuring that the network can dynamically run in optimum way and that organisations can benefit from latest policy updates and solution offerings. This PUC will be used to validate the whole SANCUS scheme and its ability to support cybersecurity assessments at all examined layers (i.e. virtualisation, management, firmware) and evaluate its capacity for optimising the security-vs-privacy-vs-reliability performance of the network in a dynamic and automated manner.

4.2 Modules

PUC3 involves all the SANCUS engines and focuses in particular on the MiU and GiO engines with the goal to highlight their offered multi-dimensional modelling and optimisation benefits. In this section, these two engines are briefly explained. The outcomes of FiV, CiV and SiD will be fused to the MiU engine, where they will be expressed in the form of mathematical optimisation criteria. The methodology will rely on recent efforts of our Partners NCSRD and ULANCS, as shown in [12] [13]. Each criterion includes the risk factors that impact on the security and privacy of network devices. Hence, each criterion can be rated according to its degree of relative importance to another criterion. Subsequently, data risks related to each IoT device can be grouped. To this aim, the MiU may exploit some theories and techniques to weight the risks and to dimension the weighted risks within an expected utility fitness function unified with the minimum throughput QoS requirements of each IoT device.

The GiO engine, on the other hand, is responsible for optimising the outcome of the MiU engine. The outcome of the MiU is forwarded for processing by the GiO engine, where the results will be optimised. The GiO engine captures the heterogeneity among all the created expected utility fitness functions of the IoT devices. This can be done by integration of these functions as optimisation rules into some game problems, enabling the devices autonomously compete for maximizing their utilities.

4.3 Pre-conditions

The main pre-conditions (either directly or indirectly) to PUC3 are stemming from the FiV, CiV, SiD and AcE engines. After unpacking a given set of firmware images by FiV and performing pre-processing sanitisation, the unpacked firmware sources are fused into the CiV engine. The CiV engine later performs static and dynamic analysis for detecting both firmware and application code vulnerabilities where it is likely to locate exploitable bugs in a rather reliable way. CiV also creates full threat model on the 5G firmware and automated threat validation. Simultaneously, the SiD engine uses innovative continuous risk assessment mechanism to perform security validation of the Docker runtime deployment environment. It then provides a list inventory of countermeasures and risk control options to be fused into the MiU engine. In other words, the outcomes of FiV, CiV and SiD are processed by MiU engine.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

Finally, the AcE engine will emulate the defined PUCs by introducing a set of network modelling and emulation module, traffic and attack generators, abstraction module for IoT devices, and Big Data analytics and visualization for automated labelling and processing of huge amount of gathered data. Along with the other engines, the original system state in terms of the value of the different state variables is also a pre-condition to PUC3. The above populates the following data:

- Existing security controls
- Likelihood and severity of vulnerabilities
- Units at risk and impact
- Identified threats and vulnerabilities

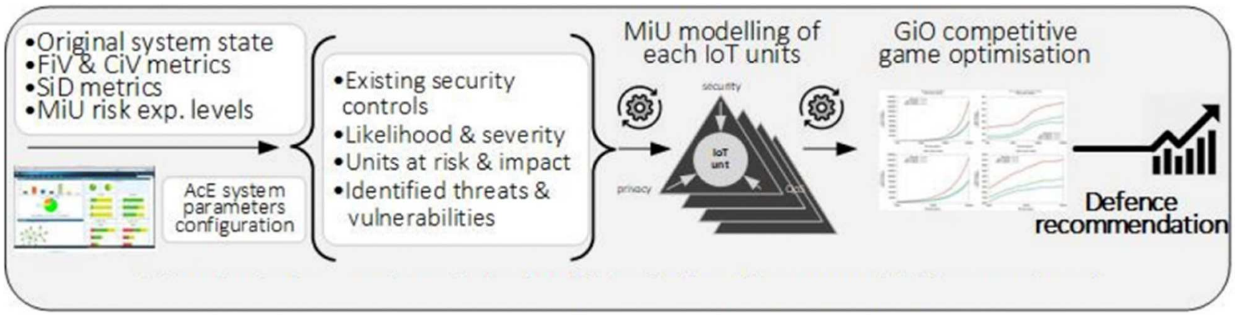
•Original system state
•FiV & CiV metrics
•SiD metrics
•MiU risk exp. levels
AcE system parameters configuration
•Existing security controls
•Likelihood & severity
•Units at risk & impact
•Identified threats & vulnerabilities
MiU modelling of each IoT units
security
privacy
IoT unit
GiO competitive game optimisation
Defence recommendation

Figure 6: Automated cybersecurity optimisation (virtualisation, management and firmware layers)

4.4 Sequence of actions

As depicted in **Figure 6**, the pre-conditions populate data in PUC3 considering the following information flow. Initially, the original state of the system is identified in terms of the value of different state variables, for instance, connected platforms, number of devices and end-users. Then the SiD engine triggers the software continuous assessment process to obtain the cyber-risk metrics, e.g. successful attack rate, software patches, new or modified scripts. Moreover, the cyber-risk metrics (for instance, unauthenticated devices, outdated network components) from the FiV/CiV and SiD engines are obtained. Having received this information, the MIU & GiO module triggers.

Considering the input data and the configuration, the MiU engine models the system to identify its risk exposure levels. This includes the quantification of security parameters of network devices with respect to the communication protocols, data traffic, the type of encryption used for their applications and users' data, etc. The MiU makes estimation of the likelihood of the cyber-attacks associated to each device in the network. Then the GiO engine makes optimal recommendations regarding the cyber-security control. The recommendation is quantified by the security-vs-privacy-vs-reliability maximisation of the GiO engine. It should be noted that various cyber-attack scenarios, network traffic loads, topologies and respective cybersecurity optimisation cases will be considered and examined. Finally, the MiU and GiO modules forward the output to the security orchestration module in order to make a decision on where the security agents should be placed.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

4.5 Post-conditions

Since PUC3 has two modules, in this section we briefly explain the post-conditions of both modules and the PUC3. The MiU aims to express its feature modelling by means of final formulas, and shapes expected utility objectives for arranging the fitness functions of all IoT units. This function represents the probability-weighted fitness functions of all possible final states of the security-vs-privacy-vs-reliability performance of the overall network. The post-condition for the GiO engine is designing the optimal defence recommendation. This is done, by maximizing the performance of the security-vs-privacy-vs-reliability efficiency subject to the probability that IoT devices, applications, end-users can co-exist without any risk of being attacked. PUC3 validations will expand from the NCSRD's to THALES's networks to examine how SANCUS approach can enterprise its capacity for protecting networks remotely. All in all, there will be a validation for the SANCUS decision-making scheme for its potential to recommend realistic, affordable baseline measures that maximize security, privacy and QoS reliability jointly, along with improving the ICT system resilience and sustainability.

4.6 Example Threat Scenarios

Three threat levels have been considered characterizing the severity of the threats. Once FiV/CiV and/or SiD/AcE engines detect a vulnerability within the firmware or code or traffic, a ranking will be made and fed to the PUC3 engines for decision making. These outputs include list and description of vulnerabilities in firmware, or attacks in emulated traffic, initial and final score for each vulnerability, list of detected attacks and their root cause, list of alerts and forecast of the detected risks impacts on cost. In the following, potential threat scenarios are listed that may be considered under PUC3:

PUC3- Scenario A (PUC3.A)
Threat severity: Low

In PUC3, a threat scenario of low severity is considered when the severity of the threats in FiV/CiV and/or SiD/AcE engines' managed threats are also of low level, e.g. exploitation of the freed memory threat, null pointer dereference threat, DoS, eavesdropping and abuse of authentication. Specifically, a combination of engines is considered in order to understand how a change in the firmware of a device and/or the software will be detected by the FiV/CiV and SiD engines, respectively. In the assumption that a null-pointer dereference vulnerability is detected by the FiV/CiV, the MiU/GiO may recommend to perform sanity checks on pointers. Similarly, if SiD detects a DoS attack the MiU/GiO may recommend the instantiation of a security control (e.g. firewall) in the infrastructure. Depending on the nature of threats the MiU/GiO will recommend an optimal defence countermeasure for manual application.

PUC3- Scenario B (PUC3.B)
Threat severity: Medium

A threat scenario of medium severity is considered when the severity of the threats in the SiD/AcE engines are of medium level. However, the MiU/GiO engines' recommendations are forwarded to the security orchestrator for enforcement of the recommendations by the security and MANO orchestrator. For

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

instance, in the case of a man-in-the-middle attack detected by the SiD/AcE engines the MiU/GiO will propose an optimal strategy for protecting the privacy of data. The recommended strategy will be forwarded to the security and MANO orchestrator for the automated enforcement of the strategy.

PUC3- Scenario C (PUC3.C)
Threat severity: High

A threat scenario of high severity may be considered when the MiU/GiO engines will receive inputs from multiple engines and for an optimal recommendation to be taken and enforced in a (semi-)automated manner by the security and MANO orchestrator. The scenarios in this case may have to cope with high severity threats detected by the individual engines. Similarly to the low severity scenarios, the MiU/GiO engines will recommend an optimal decision ensuring the security, privacy and availability of the SANCUS architecture. The optimal recommendations will be enforced in an automated manner, when the recommendations can be applied on the virtualised infrastructure via the instantiation of security controls and/or re-organisation of existing virtualised services. Manual intervention may be required for the enforcement of recommendations that may not be supported by the security and MANO orchestrator, e.g. firmware patching.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

5 Conclusion

This report detailed the deliverable D2.2 "Use Cases and Planned Attack Scenarios", which is the main output of "Task 2.3: Use cases definition planning of attack scenarios", performed within Work Package 2 "WP2 - Use Cases, Requirements and Architecture". The deliverable first introduces the initial and upgraded configuration of SANCUS architecture. The add-ons enable a higher level of automation among the system components.

The main findings that this report provides are a list of use cases, and attack scenarios that will be used to demonstrate the effectiveness of the automated detection and protection tools developed in the project against cyber threats. To this aim, three Pilot Use Cases (PUCs) have been introduced in three sections, where for each PUC, a brief description is given. This is accompanied with pre-conditions and post-conditions which are the input and output requirements/modules of each PUC. Moreover, main modules/engines involved in each PUC are described and the sequence of the actions performed in each PUC is explained. Finally, example threat scenarios have been provided for each PUC where they have been categorized into three threat levels of easy, medium and high.

The PUCs will set the basis of our demonstration and validation activities in the SANCUS project (WP6) and will provide a focus in the project's core activities in WPs 3, 4 and 5. These scenarios will be also mapped to the requirements of SANCUS stakeholders and will be related to conditions stemming from detecting cyber threats.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672

6 References

[1] ENISA, "Threat Landscape and Good Practice Guide for Software Defined Networks/5G," E. U. A. f. C. ENISA, [Online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-for-5g-networks [Accessed 2021]., 14 December, 2020.

[2] H.-R. J. N. S. N. R. N. M. Baldini G, "Mitigation of Privacy Threats due to Encrypted Traffic Analysis through a Policy-Based Framework and MUD Profiles Symmetry," 2020; 12(9):1576. https://doi.org/10.3390/sym12091576.

[3] X. P. T. B. A. M. J. &. S. M. Han, "Unicorn: Runtime Provenance-Based Detector for Advanced Persistent Threats," Proceedings 2020 Network and Distributed System Security Symposium, doi:10.14722/ndss.2020.24046, 2020.

[4] Aaron Guzman and Aditya Gupta, "Firmware Security – Preventing memory corruption and injection attacks", March 17, 2020 [Online]. Available: https://www.embedded.com/firmware-security-preventing-memory-corruption-and-injection-attacks/#:~:text=Common%20memory%2Dcorruption%20vulnerabilities%20such,the%20stack%20or%20the%20heap.&text=Buffer%20overflows%20in%20Embedded%20Linux,operating%20system.

[5] CWE-416: Use After Free [Online]. Available: https://cwe.mitre.org/data/definitions/416.html .

[6] CWE-476: NULL Pointer Dereference [Online]. Available: https://cwe.mitre.org/data/definitions/476.html .

[7] E. a. O. M. Perla, " A Taxonomy of Kernel Vulnerabilities," in *A guide to kernel exploitation: attacking the core*, Elsevier, 2010.

[8] "Common Weakness Enumeration," [Online]. Available: https://cwe.mitre.org/data/definitions/190.html. [Accessed 2021].

[9] CWE-122: Heap-based Buffer Overflow [Online]. Available: https://cwe.mitre.org/data/definitions/122.html .

[10] T. Newsham, *Format string attacks,* 2000.

[11] S. Sinnadurai, *A comparison of techniques to prevent Format String Attacks,* Citeseer.

[12] Q. N. a. J. S. C. C. Zarakovitis, "New Energy Efficiency Metric With Imperfect Channel Considerations for OFDMA Systems," *IEEE Wireless Communications Letters,* vol. 3, no. 5, p. 473-476, 2014.

[13] C. C. Z. a. Q. Ni, "Maximizing Energy Efficiency in Multiuser Multicarrier Broadband Wireless Systems: Convex Relaxation and Global Optimization Techniques," *IEEE Transactions on Vehicular Technology,* vol. 65, no. 7, p. 5275-5286, 2015.

The research leading to these results has received funding from the European Union Horizon 2020 Program (2014-2020) under grant agreement n° 952672