Grant Agreement No.: 871808
Research and Innovation action
Call Topic: ICT-20-2019-2020: 5G Long Term Evolution

INSPIRE-5Gplus

INtelligent Security and PervasIve tRust for 5G and Beyond

D4.1: Trust mechanisms for 5G environments

Version: v1.1

Deliverable type	R (Report)
Dissemination level	PU (Public)
Due date	31/08/2021
Achieved date	10/09/2021
Lead editor	Antonio Pastor (TID)
Authors	Jean Philippe Wary (ORA), Jose Manuel Sanchez Vilchez (ORA), Morgan Chopin (ORA), Krzysztof Bocianiak (OPL), Sebastian Keller (TSG), Orestis Mavropoulos (CLS), Antonio Pastor (TID), Diego Lopez (TID), Juan Carlos Caja (TID), Vincent Lefebvre (TAGES), Gianni Santinelli (TAGES), Pol Alemany (CTTC), Ricard Vilalta (CTTC), Raul Muñoz (CTTC), Noelia Pérez Palma (UMU), Ana Hermosilla (UMU)
Reviewers	Orestis Mavropoulos (CLS), Gürkan Gür (ZHAW)
Work package, Task	WP4, T4.1
Keywords	Trust mechanisms, trust enablers

Abstract

This document is the final report related to the task 4.1 and describes the results of its activities during the project lifetime. The report details the methodology followed in T4.1 as well as the challenges identified. Those trust mechanisms investigated (trust enablers) are described and detailed as a result.

Document revision history

Version	Date	Description of change	List of contributor(s)
v0.1	23/06/21	Initial version, based on MS6	TID
v0.2	12/08/21	Aggregation of enabler owners' proposition	TID
v0.3	01/08/21	Update contributions from partners	CTTC, ORA, TAGES, THALES, TID, UMU
v0.4	16/08/21	Improvement of introduction and structure of the document, addition of Trust State of the Art in Annex.	ORA
v0.5	25/08/21	Editorial corrections	TID
v0.6	03/09/21	Reviewed version and final corrections	CTTC, ORA, TAGES, THALES, TID, UMU, CLS, ZHAW
v0.7	03/09/21	Final editorial check	EURES
v1.0	10/09/21	Submission of GA approved version	EURES
V1.1	14/02/23	Editorial corrections	TID

List of contributing partners, per section

Section number	Short name of partner organisations contributing
Section 1	ORA, TID
Section 2	ORA, THALES, TID
Section 3	ORA, TID
Section 4	CTTC, ORA, TAGES, THALES, TID, UMU
Section 5	CTTC, ORA, TAGES, THALES, TID, UMU
Section 6	THALES
Appendixes	ORA, TAGES, TID

Disclaimer

Neither the INSPIRE-5Gplus Consortium Parties nor the European Commission warrant that the information contained in the Deliverable is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using the information.

[1] http://creativecommons.org/licenses/by-nc-nd/3.0/deed.en_US

Acknowledgment

The research conducted by INSPIRE-5Gplus receives funding from the European Commission H2020 programme under Grant Agreement No 871808. The European Commission has no responsibility for the content of this document.

Executive Summary

The deliverable D4.1 describes each of the identified WP4 Trust enablers and improves the precedent internal report (MS6).

The document introduces the methodology followed in the WP4 and specifically trust mechanisms investigated in the task 4.1, a description of each enabler's functionality as well as the relevant problems and challenges to address, the state of the art in their field, the solution proposed related to the problem, the status of development and what the expected limitations are.

For each of described enablers, we identified way of extension and interaction with other INSPIRE-5GPlus enablers or potential third party's features.

In the Chapter 5 of the document, we describe how the different enablers are aligned and integrated with the overall INSPIRE-5Gplus architecture. Each trust enabler is associated with WP2 defined components in the INSPIRE-5Gplus high-level architecture.

Supplementary remarks:

For most of the project lifetime up to August 2021, WP4 partners were unable to interact in physical sessions, leading to the organisation of several virtual seminars instead. Those seminars consisted in one per week allowed to share and discuss the related state of the art, vision and concepts for trust and liability. The aim was to identify the current challenges and the interaction between those two concepts to structure the future activities of WP4.

The seminar period is distributed over eight weeks. We carried out eight seminar sessions in which we investigated, addressed, and formalized several major topics:

- *Discussion of trust and liability definitions and bibliography.*
- *Research of the duality between trust and liability concepts (will be described in D4.3).*
- *Formalization of the potential perspectives and challenges for Trusted Execution Environment (TEE) and trust.*
- *Identification of a preliminary list of WP4 challenges based on the project work description and the content of D2.1. This set of challenges may be used during the project to establish and qualify the coverage and completeness of our investigations.*

Table of Contents

List of Figures

List of Tables

Abbreviations

3GPP	3rd Generation Partnership Project
5G-PPP	5G Infrastructure Public Private Partnership
AAA	Authentication, Authorization and Accounting
AAE	Adversarial Autoencoder
AES	Advanced Encryption Standard
API	Application Programming Interface
ARM	Advanced RISC Machine
B5G	Beyond 5G
BIOS	Basic Input Output System
BSS	Business Support System
CBC	Coin-Or Branch and Cut
CNF	Containers Network Functions
CPLEX	IBM ILOG CPLEX Optimization
CSP	Cloud Service of Provider
CU	Centralized Unit
DIVE	Docker Integrity Verification Engine
DLT	Distributed Ledger Technologies
DRM	Digital Right Management
DTwC	Digital Trustworthiness Certificate
DU	Distributed Unit
ETSI	European Telecommunication Standards Institute
HLA	High Level Architecture
HMEE	Hardware Mediated Execution Enclave
HSM	Hardware Security Module
ICT	Information and Communication Technology
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IMA	Integrity Measurement Architecture
IT	Information Technology
KPI	Key Performance Indicator
MANO	Management and Orchestration
MEC	Multiaccess Edge Computing
MIP	Mixed-Integer Programming
MUD	Manufacturer Usage Descriptions
NF	Network Function
NFV	Network Function Virtualization
NFVI	NFV Infrastructure
NFVO	NFV Orchestrator

NGMN	Next Generation Mobile Networks
NDS	Network Domain Security
NST	Network Slice Template
OS	Operative System
OSM	Open Source MANO
OSS	Operations Support System
PCR	Platform Configuration Registers
PKI	Performance Key Indicators
PNF	Physical Network Function
POT	Proof of Transit
QKD	Quantum Key Distribution
RA	Remote Attestation
RAG	Risk Assessment Graphs
RCA	Root Cause Analysis
RoT	Root of Trust
RSA	Rivest–Shamir–Adleman public key cryptosystem
SDN	Software-defined networking
SEG	Security Gateway
SFC	Service Function Chaining
SGX	Intel® Software Guard Extensions (Intel® SGX). It helps to protect data in use via unique application isolation technology (protect selected code and data from modification using hardened enclaves).
SLA	Service Level Agreement
SP	Service Platform
SSLA	Security SLA
SSS	Shamir’s Secret Sharing Scheme
TBNS	Trusted Blockchain-based Network Slices
TEE	Trust Execution Environment
TOSCA	Topology and Orchestration Specification for Cloud Applications
TPM	Trusted Platform Module
TRL	Technology readiness level
TRM	Trust Reputation Manager
TXT	Intel Trusted Execution Technology
UUID	Universally Unique IDentifier
VBS	VNF Bootstrap Service
VIM	Virtualized Infrastructure Manager
VM	Virtual Machine
VNF	Virtualized Network Function
VNFD	VNF Descriptor
VSF	Virtual Security Function
V&V	Validation and Verification

1 Introduction

The document is organized as follows: This section introduces a summary of the activity done in the WP4 with emphasis on trust mechanisms (T4.1) and trust enablers investigated in Task 4.1. Section 2 provides context and information to understand the trust notion and problems to solve from the point of view of 5G networks and supportive technologies. Section 3 elaborates which main challenges should be resolved towards this goal. Section 4 details each enabler and its current status of development. Section 5 positions each enabler in the INSPIRE-5Gplus architecture and depicts an initial draft of their main interactions based on interface specifications. Finally, Section 6 sketches how the different Trust enablers may be integrated towards an added value for Trust management.

1.1 List of INSPIRE-5Gplus trust enablers

Table 1 lists the identified trust enablers in the project. It includes the assigned name, the partner/s in charge of development, the Technology Readiness Level (TRL) expected by the end of the project and a brief description of their functionality.

Enabler Name	Owner	Targeted TRL	Description
Systemic VNF Wrapper (**SYSTEMIC**)	TAGES	4-5	The VNF wrapper is a software security solution capable of modifying a binary code to deliver code confidentiality and integrity and protection against licence infringement. It processes VNF binary files and *generates a protected VNF binary file, hardening the code* against various attacks. According to the VNF execution platform, the protected VNF security module (which triggers run-time security) will be either installed on a software white box (obfuscated layer) or inside a hardware TEE when available. The deployment is therefore independent of the execution platform.
Proof Of Transit (**POT**)	TID	3-5	*The ability to guarantee that a given network packet has passed through certain nodes and in a given order* is one of the most powerful mechanisms to ensure that the services in a network are working as expected and to make them resilient against attacks. It also allows to attest the service or monitored behaviour in case of legal problems. Ordered version of Proof of Transit, will improve the solution to guarantee that the packets cross the nodes in the predefined order through the path.
Component certification tool (**CCT**)	THALES	5-7	The solution proposes a *static evaluation of the different components if possible (if they have descriptors, source code …)*. For each component, *suitable metric(s) should be defined and could be measured* automatically or manually. These metrics would be combined for defining trustworthiness properties exposed by those components.
eTRM: e-reputation management	ORA	3-4	The online reputation assessment framework (eTRM) is based on three blocks: the individual reputation block, the domain reputation block, and the self-diagnosis block. The reputation assessment framework *expresses the global*

			reputation of the domain, which is exposed to higher layers to make appropriate decisions.
Network Slice Manager for trusted Blockchain-based Network Slices (**TBNS**)	CTTC	2	This enabler aims to enforce the *public visibility for the validation results of Network Slices and its components through the use of tests*, and, once they are validated and verified, to make them public in the Blockchain. Any Network Slice Manager aiming to use such components must only check if they have been previously validated, and if so, can accept the elements and start using them in production.
Risk Analysis Graphs (**RAGs**)	ORA	3-4	The Risk Assessment Graphs (RAGs) will be extended to *a new framework* which captures the following attributes simultaneously: the topology of a system, the *vulnerabilities*, the accessibility between the components, their *external exposure, and the way all these elements may evolve over the time*.
Trust Reputation Manager (**TRM**)	UMU	2	The Trust Reputation Manager mechanism will be designed as a *Smart Contract*, which *will calculate the trust and reliability* of a cloud infrastructure, or the services deployed on it, *based on multiple parameters for both the infrastructure and the services*.

Table 1: List of foreseen trust enablers

For each of these propositions, the needed data flow and interactions are described and used to improve the first ETSIGS NFV-SEC 024[2] contribution.

The proposed WP4 enablers are positioned in the WP2 architecture and their interactions with other INSPIRE-5Gplus components are described.

[2] https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58648

2 Trust related concepts in INSPIRE-5Gplus context

The notion of trust is critical to operating multi-party end to end product lines (based on multiple and heterogeneous components chaining) environments, often composed of several stakeholders cooperating towards the same business goal. Trust concerns both the components/functions and the suppliers of these components. Trust management is essential to orchestrate security resources of an infrastructure in a cost-effective way and will be investigated in Task T4.2.

Trust concepts (see the State of the Art in Appendix A) are composed generally of technical oriented elements (where trust is measured through interactions, -direct or indirect, and/or through behaviour monitoring) and non technical oriented elements where trust is measured through other means. Non technical oriented elements include: business aspects of the truster/trustee relationship, regulatory aspects and such information such as: certifications (relevant to international or national schemes), self-assessments, audits.

The concept of trust is complementary to liability, accountability, transparency and responsibility. Each covers a different aspect related to the accomplishment of a task and the management of the underlying risks.

Responsibility corresponds to the commitment to perform a specific task while achieving specific objectives. Accountability reflects the ability to demonstrate that the task has been performed as agreed and whether the objectives have been met or not. Transparency expresses the quality of doing a task and reporting relevant information in an open way without secrets. On its side, liability is related to the obligation to perform this task, and the consequences of the achievement, partial or non-achievement of the task (rewards and penalties).

At the crossroads of these concepts, trust reflects the belief of whether **the trustee is able to perform the task, meet the objectives and report relevant meaningful information**. Ultimately, **trust measures to what extent the truster is confident to accept the risks of delegating the task to the trustee**.

2.1 Trust model approach in 3GPP

3GPP started discussing the Trust topic area regarding UMTS specifications, where two important concepts related to trust were defined:

- **NDS** or **Network Domain Security**: The security domains are networks that are managed by a single administrative authority. The same level of security and usage of security services will be applied within the security domain. A network is usually operated by a single operator, hence by constituting one security domain. However, an operator may decide to subdivide its network into separate sub-networks (TS 33.210)
- **Simple trust model** (TS 33.310): The simple trust model implies manual cross-certification. Cross-certification is done at the time the roaming agreement is made.

Throughout the last years, 3GPP dragged an implicit but important assumption. The proposed Trust models consider only one type of actor (i.e., Mobile Network Operator) which are subject to the **same homogeneous body of Regulation** (for instance EECC for Europe [46]) and the **same fraud prevention, minimal security strategy and network interoperability specification**s (see GSMA agreement for Roaming: gsma.com).

However, this context evolved with the latest generation of mobile networks, i.e., 5G. Indeed, in a 5G ecosystem - including at least 5G network operators, Verticals and others stakeholders-, heterogeneous tenants subject to different regulation corpuses will have to cohabit. Consequently, the classic trust model (i.e., existing 3GPP Trust scheme depicted in Figure 1) unfortunately does not apply any more in this heterogeneous and dynamic context. The 3GPP Implicit Trust schemes should

now be re-evaluated against eSIM[3] ecosystem as the UICC/USIM component (eSIM) is **not any more under the strict control** of an Operator[4] (see Figure 2).

- Simple Trust is based on homogeneous rules between operators. The 3GPP Trust model is not valid anymore nor applicable in heterogeneous 5G technologies and environments.
- Management of Trust is not clearly defined (at 3GPP specification level) and then essentially delegated to PKI management (this management of Trust is simplified thanks to the homogeneity of actors, imposed by GSMA common agreements and procedures for Roaming).
- In unsupervised environment, 3GPP models will have to deploy **Zero Trust approaches** between different network domains.

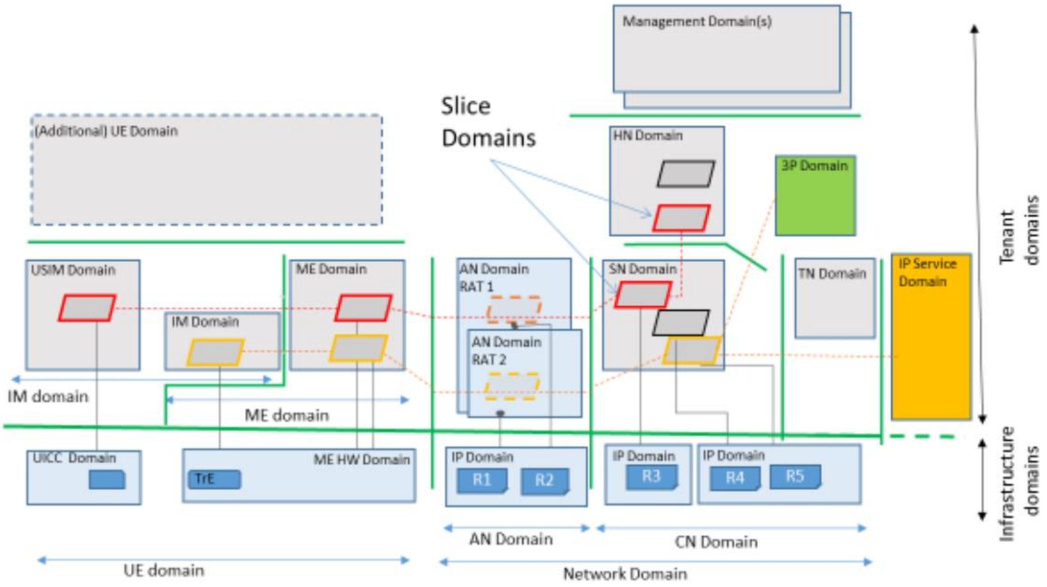


Figure 1: 5G-ENSURE/D2.4 5G Security Architecture

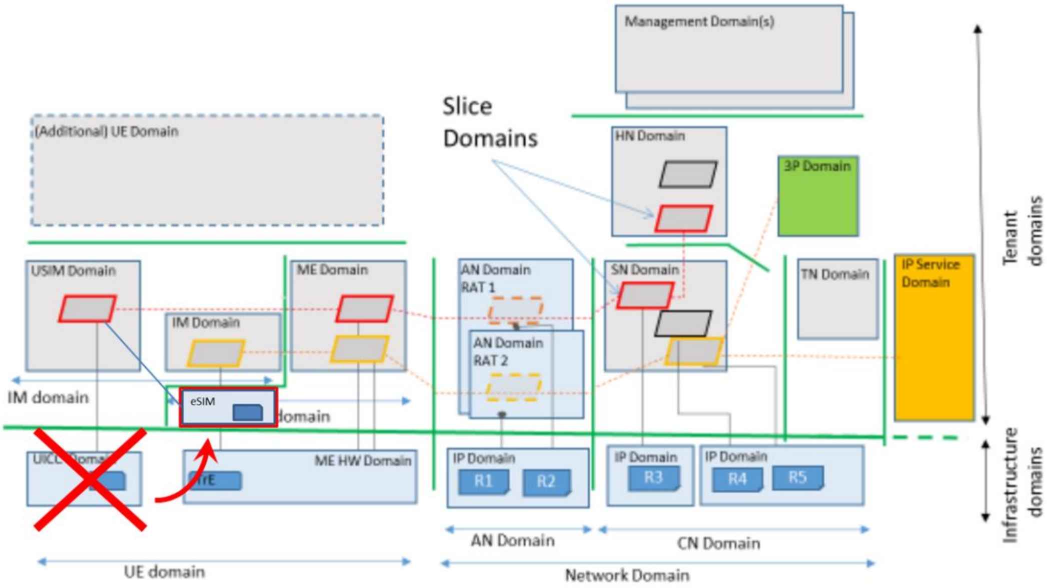


Figure 2: eSIM in 5G Security Architecture

[3] "The SIM for the next Generation of Connected Consumer Devices"; www.gsma.com

[4] https://hellofuture.orange.com/en/how-increasing-the-confidence-in-the-esim-ecosystem-is-essential-for-its-adoption/

2.2 5G challenges regarding Trust

5G is driven by (but not limited to) softwarisation principles such as Software-Defined Networking (SDN), Network Function Virtualization (NFV) and the cloud. This paves the way towards dynamicity by allowing to offer a fully customized slice tailored to Radio Access Technology (RAT) on demand.

However, this dynamicity comes with a cost, complexity, which is manifested on two factors:

- the multi-layered nature of softwarised networks, which is the underlying principle behind slicing.
- the multi-domain aspects. Indeed, 5G services are deemed to be based on enriched business cases such as "operator offer enriched by partner" or advanced consumer services based on a multi-partnership basis; what leads to a distributed management across different management entities.

In such a context, different partners will cooperate following strategic alliances to offer an end-to-end services cooperation. This leads to the blurring of responsibility because services are delivered by several stakeholders, which in turn have in turn to "rely" on each other to meet the overall service quality level. Therefore, not only cooperation suffices, but also a minimal level of information on how each partner manages its domain is necessary because each of them uses different fault management mechanisms.

The diversity in management mechanisms, employed by each partner, may lead to not respect the minimum level of fault tolerance (described in Service Level Agreement - SLA) needed to ensure the overall service quality level promised to end customers. Thus, the service quality level offered to customers becomes uncertain and even unpredictable with a fuzzy and complex responsibility assignment. Therefore, trust and liability mechanisms are of utmost interest.

Trust and liability mechanisms can benefit 5G in several ways, such as reducing the impact of misbehaving networked functions/nodes or even detecting and mitigating intentional malfunctions or attacks by immediately reducing the influence of malicious nodes. Concretely:

- Liability: To indicate the capability of each stakeholder to appropriately manage incidents in its domain in order not to impact the end-to-end service; what can allow to pinpoint the domain responsible for a service failure or attack. Trust and liability mechanisms are paramount to identify the domain(s) responsible for mismanagements leading to fault(s) and causing service failures and thus to hold those domains responsible for the damage caused to the customers.
- Trust: To increase the level of cooperation among management entities in 5G based on past collaboration evidence. This evidence can be based on the network state, which can be impacted by orchestration actions triggered by each management entity (e.g., scaling-in).
- Trust and the expected level of liability are elements to be considered by the orchestration function when it selects the network functions. This is especially the case when it comes to security functions to be chosen and deployed at an appropriate place into the network.

2.1.1 Project investigations for INSPIRE-5Gplus project

We proposed several new concepts to manage trust and liability in 5G ecosystems within WP4. We established the following aspects:

A. Under the State of the Art, trust KPIs (Key Performance Indicators) or measures are domain or subdomain specific and are difficult to be generalized over heterogeneous tenants. Is commonly agreed that inside a dedicated domain, private management of trust could be performed by the domain owner to orchestrate in a secure way for the services it may want to operate or deliver.

B. As complement to trust, liability concepts are often linked to service delivery (the QoS aspect) or Service Level Agreement (SLA) completed with specific technical KPIs related to communication. Those KPI may refer to latency, error rate, availability percentage, bandwidth, volume transfer, among other.

C. SLAs have been extensively investigated for cloud and telecom infrastructure and will need to be adapted for 5G network infrastructure and slicing. KPIs will be identified to measure security and investigate how to aggregate them so that administrators are able to optimize their orchestration choices to maximize Security SLA (SSLA) compliance. Typically, these KPIs will in some way have to illustrate the security level of an end-to-end Slice, the interface between Slice Owner and a Slice Provider or between Slice Providers and subcontracted Component Providers.

D. The three points listed above (A, B, and C) lead to the conclusion that trust and liability are **dual concepts** and will depend on which side of a domain interface we are referring to. For instance, as shown in Figure 3, box “API/Services”, located at the interface SLA interface is requesting some services and delivers a service with some KPIs. SLAs are rather related to the way command flow that are top/down (in red) in the following scheme, and trust / trust evaluation are based on local or intra-domain (depicted in blue) technical KPIs flow that are collected from the Bottom and will step by step be managed and aggregated until the Top of the Figure 3.

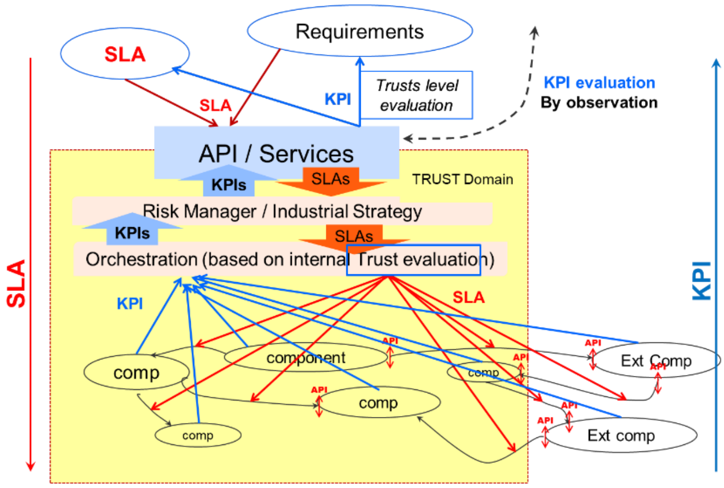


Figure 3: General presentation SLA/KPI interactions for a Domain

In the Figure 3, we assess that, for each domain, we will have to manage ‘liability’ at the external interface to commit the domain on some service quality delivery. After having committed to specific services delivery, we will have to manage and orchestrate internally the realization of required services based on the domain available resources. For this second part, we may manage an optimization process of allocation / usage of domain internal services based on precedent acquired knowledge, thanks to some internal trust metrics computed.

E. We consider the TEE concept that combines robustness (due to its hardware anchorage) and flexibility (as shielding arbitrary code) as a promising trust and liability magnifier. As far as arbitrary general-purpose software is concerned, TEE could bring total confidentiality and integrity assurances (i.e., successful side-channel attacks on TEE are exclusively targeted towards pre-known cryptographic code to extract keys). Certainty in this field opposes to the imperfect and relative trust one could bear on remotely executed software (possibly tampered during execution). As far as the network topology is concerned, TEE may also bring unambiguous identification of software instances and executing machines. TEE focus in INSPIRE-5Gplus includes local-remote code attestation, TEE-shielding of security monitoring (of other software, of the network topology, etc.), unique software-to-machine binding (and identification) and how TEE brings trust to AI/ML based automated network management. Our TEE focus embraces workflow and performance considerations as these two generally blocking factors could impede the TEE widespread usage.

The following table proposes a synthetic view on how the proposed Trust enablers address some of the previous points (A to E), as shown in Table 2.

Enabler Name	Trust and Liability concepts directly addressed
Systemic VNF Wrapper (**SYSTEMIC**)	A, E
Proof Of Transit (**POT**)	C, D
Component certification tool (**CCT**)	A, C
eTRM: e-reputation management	B, D
Network Slice Manager for trusted Blockchain-based Network Slices (**TBNS**)	B, C, D
Risk Analysis Graphs (**RAGs**)	A, B, D
Trust Reputation Manager (**TRM**)	B, C, D

Table 2: Trust enablers vs Trust & Liability concepts

3 Grand challenges

This section depicts the process and result of identification and consolidation, of trust challenges, provided as a consolidated list to be addressed by the WP4 trust enablers.

A process for detailed identification of security challenges to provide trust in 5G and Beyond 5G (B5G), and consolidation of the results were organized following a methodological process. First, the achieved results obtained in WP2 and specified in D2.1 were selected as the main source of information. This former deliverable covers in detail, relevant existing problems not resolved with the present state of the art in 5G technologies, and the new security risks still to come with the massive adoption of promising innovative technologies that will define the 5G in the long term, including the softwarisation process of mobile network communication (NFV, SDN, slicing) and supportive technologies (AI, edge computing, TEE). A second step involved a systematic collaborative identification and inventory with description of the challenges to solve, based on different visions and expertise. As a result, around 150 challenges were mapped and described.

The next stage involved a selective approach to address challenges related to WP4. Each challenge was analysed to verify what could be addressed in the field of trust and liability capacities, discarding those from other security areas. For those related to trust, an exercise of aggregation and categorization were made to provide a short list of “grand” common problems to be addressed by different technologies.

Table 3 provides the result with the list of relevant trust “grand” challenges and for each of them an approach on how to solve them through the WP4 enablers already introduced. This process includes a description the potential areas of investigation and a description of the enablers and their technological approach to the challenge described.

TRUST Grand challenge	INSPIRE-5Gplus approach to solve it
1/ Attack surface reduction for Virtualized environments (including Edge/MEC infrastructure, VM, VNF / Containers, hypervisors and internal communications in NFV environments)	• Integrity and verification of kernel-level software components during their whole life cycle (CCT) • Integrity and verification of application software components during their whole life cycle (SYSTEMIC, CCT) • Provide trustworthiness technologies for SDN, NFV over multi-domains (CCT, TBNS) • Trust in resources isolation provided by 5G security (leverage TRUST in Hypervisor technology) (TBNS)
2/ Protect SDN controller, the used applications and infrastructure from different threat vectors	• Trust that VNF software and other user-level software is unchanged from design / certification (CCT, TBNS, SYSTEMIC) • Trust that VNF software is unchanged during their execution (CCT, TBNS, SYSTEMIC) • Trust that virtual Security Functions deployed are unchanged during their execution (CCT, TBNS, SYSTEMIC) • Trust in the enforcement of rights (DRM and Licences) of the deployed application VNF or VSF software (SYSTEMIC)

	• Ensure that SDN controller and its topology information are trustable (POT, eTRM/RCA) • Provide and evaluate dynamic trust of the Domain’s topology (POT, eTRM/RCA) • Provide trustworthiness technologies for SDN, NFV over multi-domains (CCT, TBNS)
3/ Secure Verticals over 5G infrastructures	• Trust in vertical services traversing 5G Networks (TBNS, RAG) • Measure and evaluate trust level of 5G components and platforms & share trust levels with vertical(s) in a safe and trustable way (CCT, RAG, TRM, POT, TBNS)
4/ Secure AI mechanism and decisions	• Mechanism to provide trust in AI computation decisions (SYSTEMIC)
5/ 5G network resilience	• Trust in 5G Networks critical operations (RAG) • Trust in data path (POT) • Fault detection and management (eTRM/RCA)
6/ Monitor the trust of 5G networks and 5G Services due to the new complexity of these infrastructures (multi-tenant, heterogeneous, service-based architecture, multi-party)	• Define a robust trust management solution for multitenancy (CCT, TRM) • Common management of trust and liability for 5G services (CCT)

Table 3: Trust challenges and how to address them with INSPIRE-5Gplus trust enablers

Those challenges and viability of our approach will have to be demonstrated with enablers availability in the next period (management of Trust and Liability, please refer to Task 4.2 and Task 4.4). The exact process and mechanism to measure the efficiency of our trust enablers against identified challenges is specified jointly with the enabler itself. The details on these enablers are available in the following sections.

4 Trust enablers

4.1 Systemic VNF wrapper (SYSTEMIC)

4.1.1 Description of problem and challenges

4.1.1.1 Importance of software security for 5G networks and associated challenges

Network Management software, virtual Security Functions and Virtual Network Functions, are developed and run on standard platforms and operating systems and are linked to standard libraries. They altogether present a wide attack surface. If tampered, they expose unlimited risks on the network, magnified if the attack targeted software is a central element. Since the inception of the VNF networking concept by ETSI in 2013, ETSI has put a sheer-scale effort on VNF software security. The NFV-SEC GS (group specifications) and GR (group report) labelled documents reflect this prime importance at ETSI by their quantity, individual size, level of details and size of the editor lists.

These documents span over several multi-faceted security aspects of NFV authenticity verification, platform trust, VNF integrity verification, VNF confidentiality preservation, VNF license enforcement, process isolation and visibility to the platform and VNF vulnerability remediation. Appendix B of this document presents a summary of ETSI NFV security documents of reference and shows that ETSI has put special emphasis on platform and VNF attestation and integrity preservation all along VNF life cycle. Opposingly, other security items as listed above are less or not described and shall be considered as generic requirements a VNF should comply with. An important aspect of ETSI’s work is that VNF security is not only an intrinsic problem of the VNFs themselves, but it spans over their execution platform and all NFV management software dealing with them (e.g., MANO, VIM).

Our reading and analysis of the prior art (see Appendix B) has revealed specific challenges to be stressed. We provide here a non-exhaustive list of identified open challenges, specific to VNF security.

Security:

- Introspection brings ability to reverse engineer and tamper running VNFs. Any solid countermeasure will drop VNF performance (e.g., obfuscation and runtime anti tampering techniques) or lead to VNF deployment complexities (e.g., enclave on-boarding).
- Runtime integrity verification cannot be fixed by periodic certificate-based verification for both performance and workflow considerations. Monitoring sheltered agents is the cutting-edge approach but agents are still under the threat of a malicious kernel or operator.
- Remote attestation verification brings its own attack model by exposing a central point of attack: the attestation server itself. Crafted DoS attacks can be mounted on this complete network system switch. Attestation function “concentrators” can be also found inside each running platforms and share the same DoS exposition.

Workflow-security management considerations:

- Support of all different types of deployment (types of processors (AMD, ARM, Intel), workload deployment types (VM, CM, bare metal)). This is an important aspect when dealing with platform or VNF attestation.
- Keys and credentials distribution and management (including key revocation) in a multi trust domain environment and at scale is a complex organizational and NFV architectural problem. The massive ETSI specification effort reflects this complexity.
- Attestation process shall be VNF operator centric (instead of VNF vendor centric).
- The real final requirement is to check or trust the infrastructure for the correct placement of operator-validated payloads (VNFs) according to the operator-defined VNF Descriptor (VNFD).

4.1.1.2 Systemic's initial set of security functions

Systemic is offered as part of a SECaaS service offering several security functions, all applied on ready to deploy software (i.e., their binary file). The initial list of security functions given below could favourably be extended later (e.g., vulnerability remediation, right enforcement, …) and corresponds to the functions available in the course of the project:

- Authentication (aka Self-boot).
- Confidentiality preservation.
- Integrity preservation. By means of timely triggered checks on the process memory pages (during the execution of the process), the integrity of the software is validated (at the time of the measurement).

4.1.1.3 Systemic's multi-faceted security design challenges

As a reminder, in a global and large perspective, software trustworthiness shall be associated to the main properties listed below:

- Property 1. The software design and quality of coding meet the specification and user expectations
- Property 2. The software has no vulnerabilities.
- Property 3. The software is approved to execute (e.g., has been tested and validated, software is issued from a known vendor, …)
- Property 4. The software is integrated from the original form at the vendor's premise.
- Property 5. The software is kept confidential.
- Property 6. The software is used only at users with granted use right.
- Property 7. The software is executed only at designated locations.

While the two first security properties derive from the vendor ability to specify, develop and maintain its software over its entire life and can be viewed as endogenous factors of the software, the five following security properties depend on applied (external) techniques always tacking (with more or less efficiency) with inter-dependant metrics: the efficiency of the security function, the overhead induced by the protection and solution workflow and use easiness. The efficiency-to-overhead trade-off is an invariant for all academic research publications dealing with IT security in general and it applies to software security steadily.

As a software security solution, Systemic only acts on properties 3 to 7. Property 5 (i.e., confidentiality) can be viewed as a less strict requirement than Property 4 (i.e., integrity). Indeed, it may not be required by the operator or the vendor. Similarly, Properties 6 or 7 do not constitute a firm and global requirement.

We describe below Systemic design challenges according to the brought security property.

Design challenge for Property 3. Software is approved to execute

Systemic's self-boot authentication meets the security property as it checks at load time that the software originates from the author that has generated the associated signature. Checking the origin (and concurrently its integrity) of the protected variant before launching prevents changes produced by anyone intercepting the software file before and during its deployment. Processed as a self-contained software security, it does not prevent other software from being installed and to execute on the same platform, being malicious or not. An unresolved challenge is to expand the sanity check coverage to all software on a platform. As part of our preparatory work for Test Case 3 described in D5.1, we have partly solved this challenge by expanding the scope to all dependencies (i.e., shared objects, common and specific libraries, and their functions). **We are considering how this enlarged coverage can go further to all platform codes.**

Design challenge for Property 4. Software is integrated.

Software integrity is checked during Systemic authentication stage. This security property (i.e., software is integrated before launch) shall be complemented with a harder to get integrity at run time precluding introspection attacks targeting the running process memory pages. Run time integrity is offered by Systemic through timely integrity measurement made on the loaded process pages. Each integrity check is punctual (on the time scale) so that integrity attacks are possible between two measurements. The main challenge here is to reach the best trade-off between the frequency of the checks or better to trigger these checks at unpredictable timings and without causing significant overhead (as each validation check is CPU-intensive). Another challenge is to offer flexible means to deal with an integrity failure associated decision (e.g., graceful degradation, stop, keep alive) to the one that applies Systemic. Last, it is also important to keep in mind that centralized integrity verification solutions can be targeted for mounting DoS attacks. The upcoming challenge here is to **develop covert detection of code tampering combined with covert alert transmission for pertaining decision taking**. To this end, the benefits of leveraging SGX, the trusted execution environment (TEE) of Intel CPUs ubiquitous in core network servers, are the hardening of the appended Systemic routine and interesting as an enablement for the covertness sought for both detection and alert transmission.

Design challenge for Property 5. Software is kept confidential.

Software confidentiality shows two opposing trustworthiness profiles when the code is at storage and when the code is under execution. In the former case, software can be viewed as a data file where well-known encryption primitives (e.g., AES) bring entire confidentiality assurance to the software against “static” analysis. Conversely, in the latter case, when the code pages are loaded in DRAM and the software executes, the situation is (far) more demanding. As the platform processor cannot execute encrypted code instructions, code shall be decrypted before being loaded which reduces drastically the range of possible techniques by software means. Software-based code obfuscation augments the difficulty for the attacker but does so with a direct relation between this extra load on the reverser’s desk and the extra load on the processor (aka code expansion). Hopefully, hardware-based TEE are mitigating that bottleneck with a solid security boost, offering full confidentiality with a lower overhead. Nevertheless, TEE are no panacea and code cannot be placed blindly inside TEE. As a matter of fact, so-called Trusted Computing Basis (TCB) (i.e., the software and data content of the TEE) shall be kept as minimal to prevent the unintentional inserts of possibly exploitable vulnerabilities which would then operate totally covertly. Moreover, serious overhead is still likely (as shown in SoTA analysis given above). This results in a code split between the section transferred for TEE and the section remaining in the conventional execution environment. **The main challenge of today and tomorrow is to develop smart and automated techniques magnifying the opacity of the code to the attacker while keeping the TCB as low as possible and keeping an eye on the overhead**. On this path, our project enabled the establishment of a first smart automated extraction of code sections to be processed in Intel’s SGX enclave. Further enrichments are expected to come to enlarge the scope of transferred instructions or instruction sequences.

Design challenge for Property 6 and 7. Software is used only by users with granted use rights. Software is executed only at designated locations.

Both properties are obtained with variable level of confidence, according to the “association” technique enforced and the type of association data it deals with. The difference between users and platforms is reflected by the association data while the software is common to all users and platform. In these cases, the techniques and their solidity combine security properties as stated above (confidentiality and integrity) which can be applied to the association routine itself to prevent its reverse engineering and its breaking.

Intel’s SGX brings a new path for stronger association between the software and the platform. Strong association is made by generating unitary variants of the software. Each variant executes only on the platform provisioned with a unique key. Using Intel’s SGX, compared to the separate association binding as described in the paragraph above, this stronger association creates an inner functional dependency of the software to the machine (secret). The technique exploits the secret tunneling offered by SGX to provision the secret data. SGX is also used as a shelter for executing selected instructions of the software.

4.1.1.4 Grand challenges and VNF Security by Systemic

Systemic VNF wrapper strengthens software payloads, elevating trustworthiness on these payloads, as well as the conditions of their execution (including license enforcement) wherever they are executed (in or off premises). By doing so, and because Systemic can be used on “any” software, irrespective of its functions, it contributes to the INSPIRE-5Gplus Grand challenges as stated below.

Grand challenge #1 - Attack surface reduction for virtualized environments (including MEC infrastructure, VM, VNF / Containers, hypervisors and internal communications in NFV environments)

- Systemic indirectly protects kernel level code by protecting directly payloads. Payloads cannot be tampered to access to kernel code, which is a common attack path. Systemic can be used on VNF and VSF, and confers authentication, integrity and confidentiality.

Grand challenge #2 - Protect SDN controller, the applications and infrastructure from different threat vectors

- Systemic used on VNF, VSF and application software brings trust that it is unchanged at load time and during execution. This can apply to SDN controller, too.

Grand challenge #3 Secure verticals over 5G infrastructure

- Systemic brings trust to vertical deployed software. The security properties of the software can be proven (by use of a proof convention, typically leveraging remote attestation quotes produced on Systemic metadata.)

Grand challenge #4 - Secure AI mechanism and decisions

- Systemic can be used to secure AI inferring software.

Grand challenge #5 - 5G network resilience

- Systemic can be used to create a firm binding between PoT routine and the platform, solving the actual and current loose binding.

Grand challenge #6 - Monitor the trust of 5G networks and 5G Services due to the new complexity of these infrastructures (multi-tenant, heterogeneous, service-based architecture, multi-party)

- Systemic delivers visibility and (trust-security domain) wall-piercing: By attaching the needed key to the deployed binary (self-contained binary), binaries can be authenticated at any place, inside any external trust domain without any key distribution.
- By attaching a binary to a specific machine, Systemic brings a new type of visibility. A central monitoring function can (with this new strong binding) trust with certainty in information such as: “this code instance is executed on that machine” and deliver “keep alive” or “kill process” at fine grain (instance).

4.1.2 State of the art

As Systemic is a composite security solution, aggregating several security functions, there is no competing offering or solution for such compound functional definition so that the state-of-the-art analysis should be worked out per offered function. Our survey on the integrity and confidentiality state-of-the-art is focused on future trends for 5G software security, namely taking advantage of hardware trusted execution environment. We have assembled a deep survey of academic research leveraging Intel’s SGX for network security software.

4.1.2.1 ETSI Specifications and Reports on NFV security

It shall be noticed that although ETSI NFV SEC groups have a holistic and global vision of the different security threats on NFV, the quasi-integral part of the specifications and reports deal with the essential but relatively workflow-heavy platform and VNF attestation mechanism. Indeed, as it was clearly shown in our risk analysis produced in Task 2.2 of this Project, ETSI specifications are enough to block all side channel attacks on SGX, which is for the least a very valuable security stance. The other requirements of process isolation, confidentiality, licence enforcement and vulnerability exploitation are often cited but not explicitly specified. A detailed study of ETSI NFV documents is available at Appendix B.

4.1.2.2 Academic projects leveraging TEE for Software Defined networks

We have produced a survey and analysis of three EC funded past collaborative research projects which have brought a TEE leverage as given below.
5G-City [36]

The project establishes all network nodes kernel attestation by use of Intel TXT (pre-existing to SGX) and ARM Trustzone. The project dealt with the market fragmentation, interestingly pointing out and dealing with untrusted kernel.

Our vision: Well-aligned with ETSI specifications, the project shares the same vision of deploying VNF in safe execution environment. The project's main work took place before SGX became available and testable to bring VNF or VSF confidentiality and integrity guarantees.

Dive: Docker Integrity Verification Engine from Shield project. [37]

The project is not exploring TEE per-se (as a shelter to secure code and data) but the project brought a Docker container integrity verification solution.

Our vision: The benefit of this solution shall be viewed as an extra layer to install on trusted execution environments (trusted kernel and whitelisting). As a matter of fact, Shield deployed "user level un-protected attestation agents" are exposed to local introspection attacks. As suggested by the work, the next step should be to protect the agent (by TEE) as it is exposed to tampering attacks.

5G-ENSURE project. *TruSDN*: Bootstrapping Trust on SDN. [38]

The publication and experiment *Bootstrapping Trust* Enabler (see 5G-ENSURE project) is a generic study enabler specified to deliver integrity verification of network nodes and in three different domains: to data plane (SDN) switches, controllers and VNFs. The initial road map was to develop a suite of new software components and protocols separately or forming a package delivery for orchestrator or controller. The configuration of VNF will also be under scrutiny. As result of 5G-ENSURE project, node integrity enabler emerged as a separate development focused on network switch (only) dubbed TruSDN. This project leveraged SGX: The SGX TCB content is twofold. On the one hand, the VNF are supposedly protected by a TEE protected agent, while on the other hand, virtual switches are fully integrated into TEE (SGX). From the original enabler road map as described in 5G ENSURE, CIST (i.e., the project partner in charge) has focused deeply on the virtual switch security aspect, with a full insertion of the switch code. The publication details the security threats of node impersonation, possible cuckoo attacks on SGX enclaves and the offered remediation.

Our vision: Side Channel Attacks targeting SGX were not disclosed at the time of the project. The partner in charge has however demonstrated its security expertise through its security analysis directed to the Cuckoo attack and the suggested remediation.

In addition to the EC-funded research programs, we also consider recent academic works leveraging SGX listed below.

Universal Trusted Execution Environment for securing SDN/NFV Operation [39]

The universal solution consists in leveraging either Intel SGX or AMD's SEV TEE with a common payload (i.e., VNF) through code virtualization principle and the installation of code interpreter in the corresponding targeted TEE. The solution configuration is heavy as it implies to install on either side (SEV or SGX) ad hoc code interpreters (capable to decode on the fly the VNF extracted protected code segments). The author's goal is to solve two market blockers for TEE: market segmentation and TEE technology leverage complexity.

Our vision: As stated by the authors, some short cuts on security result from this attempt to bridge two diverging technologies. In our point of view, bridging AMD's SEV and Intel's SGX will forcibly be done at high security costs. Additionally, the performance impact of code virtualization may be problematic.

Secure Software Defined Network Controller Storage using Intel SGX, [40]

The publication focuses on the SDN controller software and data security, as being viewed as the most critical element of SDN security. The authors stress the relevance and importance of protecting the data processed by the controller referred as the brain of the networks. The authors design one SGX-based controller with the main objective of protecting its data (i.e., user credentials, flow rules and configuration files.) The authors have constructed a SGX controller Proof of Concept based on the Java-based Floodlight turned to native code through the Java native interface. The baseline overhead is limited between 7 to 10%.

Our vision: The solution makes a split between the trusted part of the controller (SGX embedded native code) and the untrusted part of the controller (Java based part producing calls to the enclave code). On the security point of view, the attack should take place in the Java side as java bytecode are much easier to analyse and tamper than native code, even before native code is placed on SGX. However, the authors might have taken this security threat into account and have worked out the ad hoc division between these two parts. Nevertheless, since the Java part controls the SGX part, DDoS attack seem relatively simple to perform. Their hybrid Proof-of-Concept (PoC) shall be viewed as "toy" demonstrator developed with a pre-existing open-source controller for the convenience of the demonstration. A commercial SGX-enabled controller product would however be structured as a mono-block layout.

S-Blocks: Lightweight and Trusted Virtual Security Function with SGX [41]

S-Blocks publication presents a VSF-pronged framework for leveraging SGX security. A VSF are considered well defined and standardized functions (which can be indeed built and composed with elementary preset modules). The framework is aimed at solving two strong obstacles in using SGX: inherent relative complexity (for developers) in using SGX and inherent SGX overhead (as were enumerated in our D2.1). The authors propose a simple use S-Blocks framework, microservice-oriented architecture which decomposes VSFs into trusted and untrusted modules and provides dedicated APIs systematically. The microservices consumed are derived from FastClick, an extension of the Click modular router (a most commonly used software architecture for building modular and extensible network function). The framework simplifies the developer tasks through Click script language to compose its VSF and S-Blocks automatically selects SGX-preset variants. The presentation illustrates the benefits of the design with three critical types of virtual security functions based on the S-Blocks architecture. Finally, the performance evaluation is produced reflecting overheads in the range of 25%.

Our vision: The publication is instructive and clearly exposes the motivations of the authors (security, performance, flexibility). On the security aspect, SGX is presented with all security guaranties without considering possible Side channel attacks (SCA). Whereas S-Blocks is a well-designed prototyping tool, offering developer flexibility and easiness to compose SGX-embedded VSF, on the security point of view, using open-source building blocks certainly increase the SCA likelihood. The authors cite SCA as being out of the scope of the publication (which sounds totally correct).

LightBox [42]

As an SGX-secured middleware, LightBox motivation is to deliver unprecedented security to traffic metadata (e.g., low level headers, packet size and counts) in addition to stateful processing customization data without causing a significant penalty on the middleware-node throughput. The security motivations (steered on both types of data to protect) are well described as well as the implementation details (i.e., *etap* virtual network interface and state management module). Interestingly, the authors deviate from a naive all-inclusive SGX content inducing severe paging overhead, to a tailored and restricted usage of SGX memory map. This is done through an ad hoc content split on both sides of SGX (inside and outside), reaching elevated performance. The authors have benchmark three different middleware implementations around PRADS, LwIDS and mIDS which demonstrated that SGX security has a cost (optimization) and cannot be used without consideration on the specificities of SGX memory limitations. Conversely, they show that ad hoc code changes in the sake of improved memory use bring very acceptable performance (when they are not exceeding the original code running outside SGX).

Our vision: The authors learnt lessons showing that real-world security and performance sensitive SGX implementations is not only a matter of porting code but requires an optimization to counter-balance the specificities of SGX technology (e.g., reduced size of the Enclave Page Cache to 128 Mb). On security, the publication converges to the vision of open-source based middleware which could eventually facilitate side channel attacks which are all targeted attacks (on accessible and known code).

Trusted Click : Overcoming Security issues on NFV in the Cloud [43]

The authors advocate SGX utilization with the prospect of side channel attacks on the cloud (when SGX is not used). The challenges for SGX employment are presented as follows: Smooth integration of SGX technology in NFV generation toolchain and of course performance. NFV private data at risk are enumerated with their security sensitivity in the perspective of the Aplomb model that virtualizes all organization middleware in the cloud before reaching the Internet. The implementation is based on a Click modular router (as S-Blocks). The bandwidth delivered by the SGX layout is roughly half of the bandwidth achieved for the same library outside SGX.

Our vision: This publication is in the same vein of others (S-Blocks, S-NFV, Light Box) delivering similar messages and conclusions. SGX software integration demands optimization to maintain similar performance.

S-NFV: Securing NFV states by using SGX [44]

The authors have pioneered in the path of utilizing SGX for the benefit of network (state data) security. The motivation of securing NF internal states as containing user data as IP address, end user host details shall be efficiently protected. Such protection is viewed as essential for Content Distribution Networks (CDNs). The enumeration of NFV internal states information with a ranking of their security-sensitivity is exposed. The authors also refer to ETSI NFV security specifications, recalling the risks of datacentre NFV introspection. The publication delivers its PoC style implementation protecting Snort open-source IDS which is in fact a VSF and not a VNF and by employing OpenSGX framework for an easier SGX integration. The performance overhead is given at an average of 10x which is considerable but it should reflect the use of a OpenSGX framework first coupled with a lack of optimization (as deemed unescapable in LightBox above).

Our vision: First (and with a significant 2-year advance) to use SGX for NFV state data protection, this short-targeted PoC has been very often cited.

4.1.3 Description of the solution

4.1.3.1 General

The Systemic is a software security solution capable of modifying a binary code to deliver code authentication, integrity, and confidentiality assurances. In a further evolution of Systemic, DRM functionality or other security functions such as vulnerability remediation could be added. Systemic is offered in two-grade flavours as its deployed security routine (appended on the original binaries) can be alternatively protected by use of proprietary TAGES's Solidshield code virtualization technology or by Intel's SGX. In short, Systemic protects software running on X-86 AMD platforms (with Solidshield) and X-86 Intel platform leveraging SGX. Systemic and Systemic-SGX are the product names for these two grades.

Systemic operates on deployable software (binary files) which widely broadens its possible usage and the perimeter of the stakeholders using it as binary files can be accessed by telecoms operators, security vendors and infrastructure operators. It can be applied on any type of (compiled) software such as VNFs, VSFs or more generally on any network deployed software. Another important aspect is Systemic core feature: automation. The basic three step setup workflow is shown in Figure *4,* which depicts one SGX-enabled implementation. Some of the parameters or the selection of the protection functions are given with user assistance or can be inputs from a security server through APIs.

Systemic must tackle with its security-performance-workflow challenges disregarding the original software nature, functions, and content. In the telecom industry, there is no room for user-defined optimization, which shall be carried out by Systemic solution itself and this is done with run-time real measurements.

Moreover, binary level automatic modification confers in-depth defence of the VNF or other software in a self-contained and built-in manner, reducing dependency to external elements and easing implementations. Precisely, all required elements for authentication verification, decryption, and confidentiality preservation at runtime are inserted into the newly wrapped version of the binary as the baseline mode. As an alternative mode, the associated keys used for decryption and authentication verification can be separately provisioned to the platforms, in order to restrict the use of the code to a restricted and designated set of platforms.

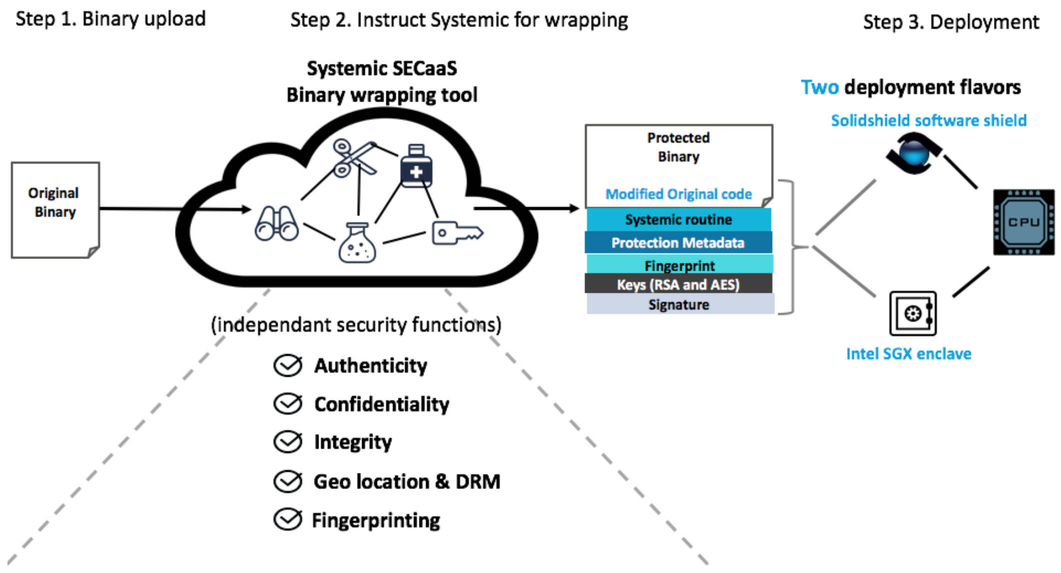


Figure 4: General presentation of Systemic VNF wrapper 3- step basic workflow

Figure 4 shows the three basic steps before the deployment of the protected version. Once uploaded, the original will be modified by the wrapper according to a protection project (parameters of the protection-selection of security functions). The wrapping tool prepares the protected version according to the type of deployment and notably if Intel's SGX can be leveraged on the targeted platforms. The figure also shows that the protection metadata which contains all protection project parameters can be appended to the protected binary. The metadata is encrypted and can be extracted (i.e., decrypted) to Systemic users (having an access to the associated key). The fingerprint follows the same data protection mechanism (as Metadata) and enables to create unitary variants in order to ease root cause analysis if a security incident is detected. Fingerprinting also enables to unambiguously control and monitor remotely each variant separately.

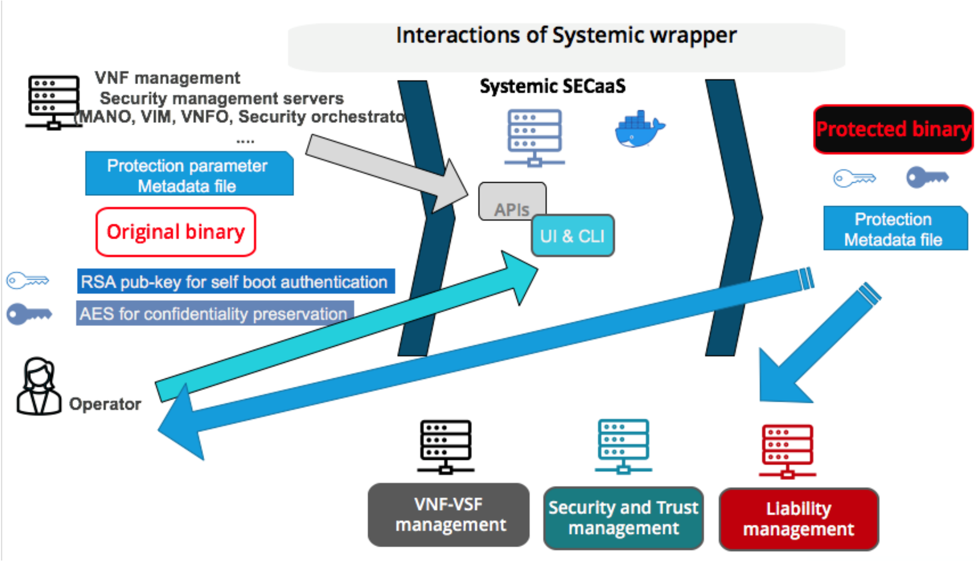

Interactions of Systemic wrapper
VNF management
Security management servers
(MANO, VIM, VNFO, Security orchestrato
....
Protection parameter Metadata file
Original binary
RSA pub-key for self boot authentication
AES for confidentiality preservation
Operator
Systemic SECaaS
APIs
UI & CLI
Protected binary
Protection Metadata file
VNF-VSF management
Security and Trust management
Liability management

Figure 5 : Interactions of Systemic wrapper

Figure 5 shows the two types of interactions of Systemic, either instructed by users through an HTML5 web interface (and/or CLI for automation) or directly by any security or deployment servers through APIs. The Systemic main exchange path relates to the binary. SECaaS wrapper consumes the original binary and delivers the protected variant back to the emitter (servers or users). Side exchange channels relate to the transmission of protection parameters, associated keys, protection file metadata and fingerprint.

When an operator interacts with the SECaaS, the **protection parameters** can be either input through the (graphical) user interface or by the command line interface to the SECaaS. Server-to-server interactions are defined and delivered by the SECaaS APIs.

The **keys** (e.g., RSA public key for authentication verification, AES key for software confidentiality preservation (by encryption)) can be brought by the operators or orchestration servers or can be directly generated by the SECaaS. On the output side, the keys are then either appended on the protected binary file (less secure) or provisioned separately to the executing platforms for security.

The **protection metadata and fingerprint** are both encrypted and located inside the binary. The decryption can be done by the servers and users provisioned with the key (symmetric encryption). The metadata file stores the parameters of the applied Systemic protections. The fingerprint is a random data appended for the purpose of unitary identification of each instance (or set of instances with the same fingerprint) of the software. Fingerprint can be used to locate a security incident which involves the software. It can also be used to strengthen the geolocation and identification of deployed instances.

4.1.4 Efficiency factors

The efficiency of the solution derives from the three different efficiency factors listed below. Our evaluation of the efficiency factor in written in blue uppercase letters:

1. **Efficiency of the authentication (**or property that the code is approved to execute**):**
   a. Systemic's baseline offer. (FULL) Its self-boot authentication is based on a standardized PKI (Public Key Infrastructure) plus hash crypto proven process. This

ensures with a very high level of confidence that any tampering attempt **on that code** will be detected before launch.
   b. SGX-enabled baseline offer. (FULL) More if the public key and digest can be provisioned on the platform by use of SGX-enabled secure tunnelling, **any process on the platfom** can be checked before launch and still with a very high level of confidence.

2. **Efficiency of the integrity assurance:** Two process stages shall be considered separately:
   a. Cold storage (on disk) or during VNF transfer: (FULL) The AES encryption prevents any "semantically meaningful" change on the encrypted code. Any blind change on the encrypted code section is possible, however but will be detected by the authentication check above.
   b. Running code: (RELATIVE OR PARTIAL). Systemic runs self-regulated integrity checks (triggered inside the control flow obfuscation technique as stated below). Between two measurements, the code can be modified by the attacker which needs to erase its footprints (i.e., restore the code to original untampered stage). This can be prevented when measurement orders cannot be intercepted. SGX-enabled implementation provides unpredictability.
   c. An integrity verification solution efficiency depends on the accuracy of the detection but as importantly on measurement unpredictability, alert transmission covertness and easiness of the decision-taking (or exhaustivity of the several software termination alternative parameters setup). For all these requirements, leveraging Intel's SGX is fully comprehensive.

3. **Efficiency of the confidentiality assurance**: Two process software life cycle stages shall be considered:
   a. Cold storage (or during software transfer): (FULL) Systemic encryption (AES 256) is sufficient to provide a very high level of assurance of code confidentiality preservation.
   a. Running code: (RELATIVE OR PARTIAL) With the security threat of a malicious operator with root access on the running machine (aka introspection), access and analysis are made on clear-text memory pages. However, our runtime obfuscation conceals the **control flow graph** (which is not resolved and shown by a debugger but can still be collected by means of execution in all of the code branches). SGX-enabled implementation significantly improves this efficiency factor by keeping (automatically) selected instructions unseen. During the course of the Project, we have first developed such a mechanism. Further progress will be made to expand the types of instruction sequences automatically executed in SGX.

4.1.5 Development outcomes

Before the start of INSPIRE-5Gplus project, we had an initial solution with main features:

- Obfuscation (code confidentiality) leverages the import table content encryption.
- Code integrity coverage is limited to the protection routines only, which is quite limiting
- No support of Intel SGX enclave
- UI based on a Windows-only native application, named Solidstudio, interfaced to Linux based protection server
- The command-line protection tool, named Sldcmd and available for Linux and Windows contains some specific OpenSSL dependencies, which required some tweaks under certain configurations
- VM-based packaging of the protection server, which featured a complex software stack making use of Apache, PHP and MYSQL and native parts.

The work that has been produced during INSPIRE-5Gplus (first 18 months period) are:

Integrity assurance reinforcement

Measurement of the complete program footprint once the code has been modified and loaded in process memory pages in the RAM

Confidentiality assurance reinforcement

Moving from our previous (external) dependency obfuscation to (internal) control flow shadowing (i.e., obfuscation) in order to cope with all types of program topology including statically compiled programs (i.e., without dependency). In association to the new control flow shadowing, we made the plugging to our run-time auto regulation mechanism. The impact of this automatic protection regulation is tangible as being inescapable when considering automated protections, acting as a permanent overhead guard.

Intel SGX support

We have made a significant step by leveraging Intel's SGX hardware-based TEE instead of software-based obfuscation in order to secure the deployed security routines and their secrets. The two solutions are offered and are functionally 1:1 equivalent (at the time of the production of this document). As noticed above, SGX leverage enables further enrichments in the domain of integrity failure remote decision-taking.

Support to dynamic run-time environments

Our study and integration work for the preparation of Test Case 3 has resulted in a new mode enlarging the scope of the authentication verification. What constitutes a Dynamic Run-Time Environment is the scenario in which the main process is agnostic of the code that will be contained in shared objects that might be conceived, compiled and delivered after the initial protection and deployment of the main program. In order to prevent the main process to arbitrarily load possibly rogue shared objects files, we introduced a new security feature to Systemic that, when enabled, enforces the signature verification of any shared object file right before being dynamically loaded into the main process.

Work on the delivery packaging

Further works have been done to refine and simplify the VNF wrapper delivery pack by:

- Solidstudio was redesigned as a WEB-based application, removing the constraints of installing a native application, which was available only for Windows
- Sldcmd, the command-line protection tool, was rewritten as a static Go binary, removing any issue related to its dependencies
- The protection server was rewritten as a micro-service, as a static Go binary, condensing all of its code to a single programming language
- Packaging of the protection server, which can serve both Solidstudio UI and the endpoint of the protection APIs, as a Docker container
- Solidock, is an alternative packaging of the protection server as a single static Go binary file, acting as a self-contained container, which does not require Docker to run

Work on APIs

The APIs of the protection end-point, which were intended for internal use with Solidstudio or Sldcmd, have been simplified and streamlined so to be exposed and used directly by other services.

4.1.6 Implementations

The current implementation packaging has been improved in the course of Project. This packaging is a progress as stated just above and with the objective of easing Systemic installation in a typical 5G infrastructure. The implementation follows the same logics of Systemic itself. The installation must be automatically worked out in any encountered infrastructure software structures.

The wrapper is either offered inside a Docker container or as a monolithic statically linked native application (aka Solidbox) in order to comply with environment without container (Figure 6). As Solidbox can be possibly packed inside a VM, the three types of deployments (e.g., VM, CM and bare metal) shall comply with all types of processing and infrastructure contexts.

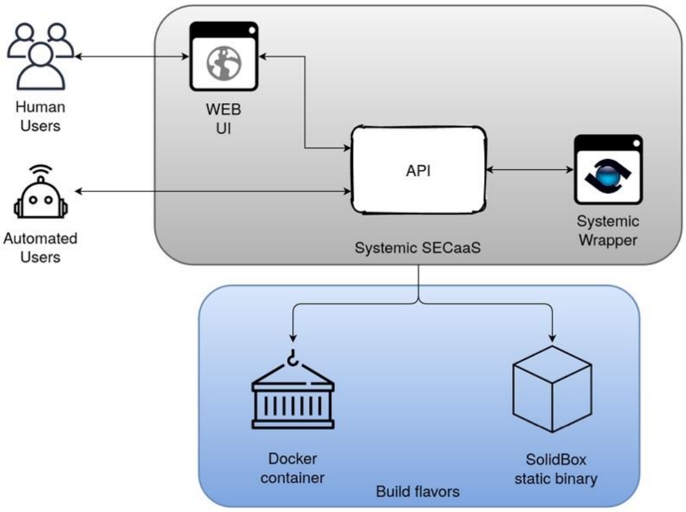


Figure 6: SECaaS implementation details

The packaging has been engineered for the SECaaS model with exposed APIs to be inserted and used by a security orchestration or any type of server.

The system requirement is minimal as the VM-based, the CM-based or the native wrapper application can be installed on any commodity server of common and standard performance. The wrapper platform does not require Intel's SGX but if this option is selected by the user, the machines which execute the protected software shall be SGX-enabled.

The user documentation is available at https://www.solidshield.com/dox-preview/systemic/. The current version shall be updated and enriched with the new features resulting from the Project (e.g., SGX enablement, control flow shadowing obfuscation, expansion of authentication coverage to the protected file dependencies, etc.). This later update is a consequence of the complete reshuffling of the SECaaS server packing which turned the Windows native UI into HTML5 web interface.

The current version of the wrapper is v21.06.01, available on TAGES company internal repository.

The solution is offered with a commercial royal bearing license on a per wrapping use (different software) or alternatively on a lump sum basis for any quantity.

4.1.7 Associated publications and patents

Two patent applications have been submitted by TAGES in the late days of 2020 (in the course of the Project) and are related to Systemic or other software security solutions. They relate respectfully to:

A method for regulating a software protection (of any kind) at run time for magnifying the protection efficiency-to-overhead ratio. In practice, run time regulation is inevitably supplied with the protected program in the hostile world. The exposure of the regulating module is the highest and requires its shielding into a hardware trusted execution environment as it is actually offered today by Systemic.

A method for creating a strong binding between a software and a platform as detailed in Chapter 4.1.3 - last paragraph. The method creates unitary variants deployed and intimately associated to one secret-provisioned platform. This method enables to create stronger binding (if not unbreakable) of software and machine, paving the way to liability-proven geolocation or hardening actual proof of transit mechanism (which shows a leak binding between the POT primitive and the hardware).

4.1.8 Residual challenges

Systemic authentication self-boot is limited to the protected binary (and recently to its dependencies). A residual challenge is to expand this authentication check coverage to any software on a platform endowing a platform white-listing capacity. The challenge is to design a practicable mechanism, advantageously leveraging existing components and minimizing the requirements of pre-installed modules on the platform.

On the **integrity security function**, efforts shall be placed on the following tasks:

- Specifying and developing the UI for defining the ad hoc type of reaction when an integrity violation is detected.
- Specifying and developing the APIs or/and interfaces to the decision taking remote position as well as the secure alert data transmission (e.g., SGX secure channel establishment after SGX remote attestation).

On confidentiality, Systemic brings a relative resistance to introspection today. It does better than what is defined by ETSI in this domain (ETSI brings no direct software protection technique) but our solution does not pretend to offer total security. There will be no conclusive and final protection but to locate the VNF code into the TEE which by itself poses workflow issues (changes on the code, processor-specific deployments, etc.). We had designed a solution that generates "semantic holes" in the protected software, rendering the global semantic extraction significantly compromised. The current implementation already generates an automated extraction of control flow instructions into the TCB (i.e., SGX content). The TCB enlargement beyond this perimeter is a residual challenge, providing higher introspection attack resilience but at the cost of higher overhead. As already offered, the overhead is regulated during the run time and this mechanism shall be maintained along with the expansion of the TCB.

Systemic modifies the original VNF binary which must be considered as a potential area of business contention between the stakeholders and namely the software vendor (if it is not the Systemic operating user) and Systemic operating user.

Without constituting a challenge per se, some efforts must be brought on:

- Reshuffling the user's documentation (which does not reflect the actual functional stage)
- Offering an automated wrapped executable testing facility

4.2 Proof of Transit (PoT)

4.2.1 Description of problem and challenges

A widespread concern about virtualized network elements is the correct forwarding of the traffic between instances that are connected through overlay networks. Any network device deployed in a production network must be capable of assessing whether a specific traffic flow passes through it and is correctly forwarded. If a node cannot guarantee this capability, it will not be accepted for production deployment. By progressively evolving from physical network functions (PNFs) to virtual network functions (VNFs) and CNF (Cloud-Native Network Functions), this task becomes harder in multitenant environment, and in case of multiple slices. It will be very common that the traffic traverse multiple intermediate nodes (possibly out of the control of the operator), that could eventually bypass a critical node within the SFC (e.g., a firewall, encryptors, etc.), based on SDN controller decisions, that could result in security breach, service degradation, data exposition, etc., without the knowledge of the client or user. Mechanisms that can generate trust in the data path are needed.

In terms of providing trust for 5G networks and their verticals, and the identified general challenges in previous section, Proof of Transit (PoT) is a technology that provides trust in the data path with the support of a Controller. PoT can be used to validate specific nodes in the network path for 5G slices confinement, or inter-domain communications.

- **Grand challenge #2:** Protect SDN controller, the applications used and infrastructure from different threat vectors
    - **Global target:** PoT will help in the grand challenge of *protect SDN controller, the applications used and infrastructure from different threat vectors.* Attacks with the aim of topology poisoning in the SDN controller can be identified. The PoT controller can keep a trusted copy of the path obtained from SDN controller when it is defined, in order to prepare the cryptomaterial to distribute to selected nodes. This way it could identify and alerts changes in the topology. Additionally, if a topology poisoning happens, and some of the nodes involved in the PoT verification are bypassed (e.g., a firewall), then the packets validation will not happen on the final node and it will be detected and discarded. This automatic protection mechanism will increase the *5G network resilience* in case of data path manipulation.

4.2.2 State of the art

PoT is based on the IETF draft [1] where the ordered version of the protocol is proposed as OPoT (Ordered Proof of Transit). PoT (without order) has been tested in one specific Linux kernel as part of extension hop by hop header on IPv6[5], not on IPv4 or as an overlay process agnostic to protocol layers. A basic implementation for OPoT was done in [2], where Quantum Key Distribution (QKD) technology is used. Associated patents and request (EP3528430, EP3289727A1) are also available.

4.2.3 Description of the solution

The ability to guarantee that a given network packet has passed through certain nodes and in a given order is one of the most powerful mechanisms to ensure that the services in a network are working as expected and to make them resilient against attacks and provide trust to users. It also allows attesting the service or monitored behaviour in case of legal problems or regulations. OPoT technology solves the lack of verification of the correct order of nodes on the path.

[5] https://github.com/IurmanJ/kernel_ipv6_ioam

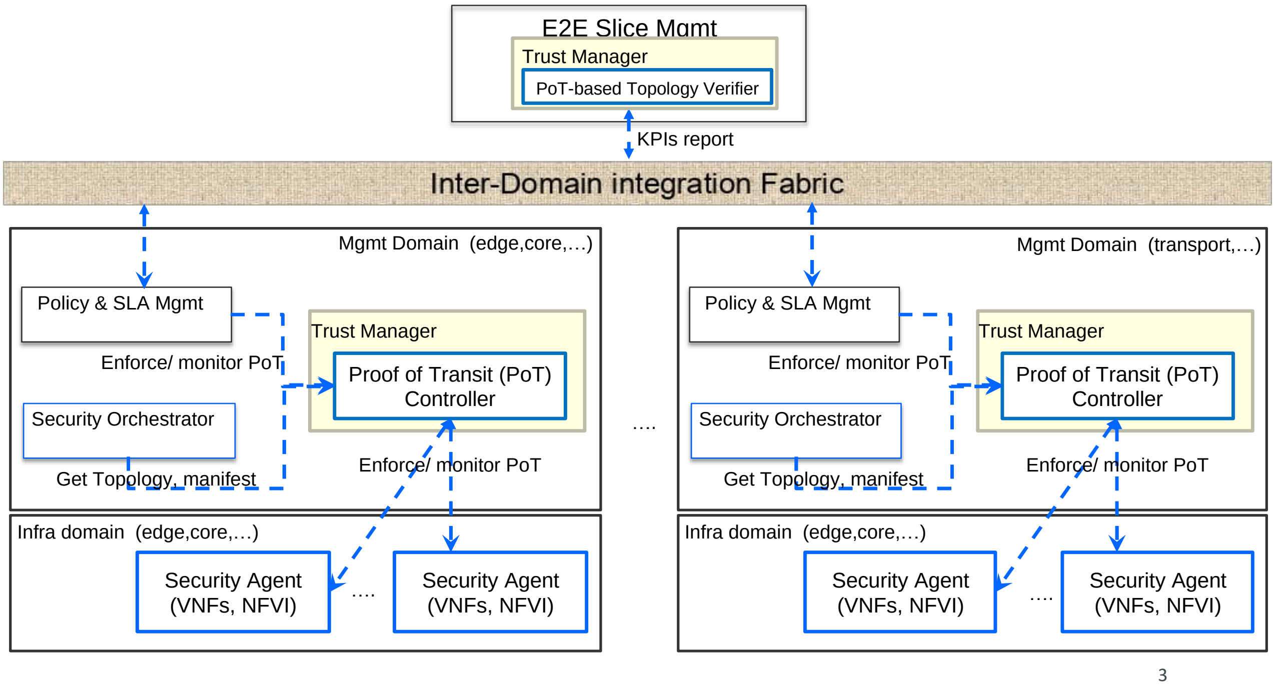

E2E Slice Mgmt
Trust Manager
PoT-based Topology Verifier
KPIs report
Inter-Domain integration Fabric
Mgmt Domain (edge,core,…)
Policy & SLA Mgmt
Enforce/ monitor PoT
Trust Manager
Proof of Transit (PoT) Controller
Security Orchestrator
Get Topology, manifest
Enforce/ monitor PoT
Infra domain (edge,core,…)
Security Agent (VNFs, NFVI)
….
Security Agent (VNFs, NFVI)
….
Mgmt Domain (transport,…)
Policy & SLA Mgmt
Enforce/ monitor PoT
Trust Manager
Proof of Transit (PoT) Controller
Security Orchestrator
Get Topology, manifest
Enforce/ monitor PoT
Infra domain (edge,core,…)
Security Agent (VNFs, NFVI)
….
Security Agent (VNFs, NFVI)
3

Figure 7: PoT components and interactions

PoT controller component is in charge to create the cryptographic schema, identify the nodes in the domain involved in the verification of the path and the order. For this process the controller collects information and identify the topology where it will apply path proof, based on the policy defined. The next step requires establishing communications with the nodes to distribute the specific material in each node's security Agent.

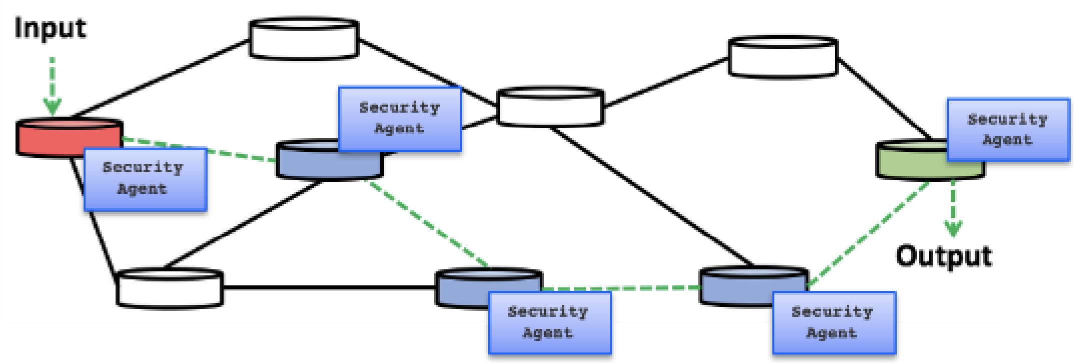


Figure 8: PoT security agent

The PoT security agent, shown in Figure 8, is in charge to enable the verification of the paths in selected nodes, and communicate with the PoT controller the status during the setup process. This information will progress as a KPI for trust and liability. These complementary areas can be achieved by monitoring or enforcing the verification.

Finally, the *PoT-based Topology* is a potential improvement that runs the role of verifier, and it could be allocated in general E2E management framework in charge to decide the domain where to apply PoT verifications and collect the verification status (see Figure 7).

4.2.4 Efficiency factors

The security policy can establish the expected level of efficiency. The trust reference value for the PoT enabler will be the event message contents related to the verification status of each node involved in the monitored topology. Additionally, if verification fails at the final node, that means that a node has been avoided or bypassed, a specific alert is generated. This process will be customized in verification frequency and number of nodes to be verified.

Additionally, it is worth to mention that the algorithms used, Shamir's Secret Sharing Scheme (SSS) [3], achieves information-theoretic security i.e., it cannot be broken by an attacker even with unlimited computing power. Nonetheless, wrong implementations or insecure channels between Controller and nodes could be exploited.

4.2.5 Development outcomes

Initial version of PoT was based on an initial proof of concept prototype used in conjunction with Quantum Key Distribution (QKD) technology referenced in the State of the art subsection to provide order verification, a.k.a. Ordered Proof of Transit (OPoT). The solution carries a dedicated protocol with specific cryptographic payload packet. This work allowed isolating the OPoT from QKD technology to increase its applicability in other areas where QKD systems are not available, and progress in the internal design of the enabler. The developments made in T4.1 have included:

- The creation of a centralized mechanism for key generation and distribution, "PoT Controller" as alternative to distributed QKD system to keep the order property and key material protection. This solution makes the oPoT compatible with quantum and classical cryptography.
- Enrich interactions between the agents and the controller to increase the visibility and status of the verifications. The added information includes the node-by-node cryptographic material and calculations.
- Include by default timestamp information, so it can be used in metrics generations (RTT, latency or hop-by-hop delays).
- Support customized ports and protocols (UDP/TCP) by configuration so it can be adapted to different environments.
- Improve metrics and log generations, so it can be used by Trust Manager in their calculations.
- A generic baseline of APIs to support PoT controller interactions with other component of the trust management of INSPIRE-5Gplus framework.

Finally, additional extensions are in progress to support deeper integration in T4.2 Trust management and to provide different KPI metrics, needed in WP5.

4.2.6 Implementation

This version of the PoT is mostly written in python 3.8 and relies on configuration files in JSON format. It is also compatible with python 3.5. There is also an option to install it using docker.

Implementation is divided into:

- Node:
  - Basic unit in which the path can be divided, receives configuration files from the controller and sends metrics.
- Controller:
  - Each time a PoT path is going to be defined, the Controller is going to be the manager of all the necessary information which later will be passed to the nodes.
- OpenAPI
  - In order to make this service accessible, the interaction with the Controller have been developed using the OpenAPI standard. These are the possible interactions with the controller through the API:
    - Create PoT path

- Get information about a path
- Delete a path

This project also supports interactions with Kafka and Prometheus in order to deliver packet metrics crossing a path.

There are two main tools that install all the necessary components. One for the **Controller** and the other for a **Node**.

The internal process are described in following Figure 9 showing the succession of events during operation.

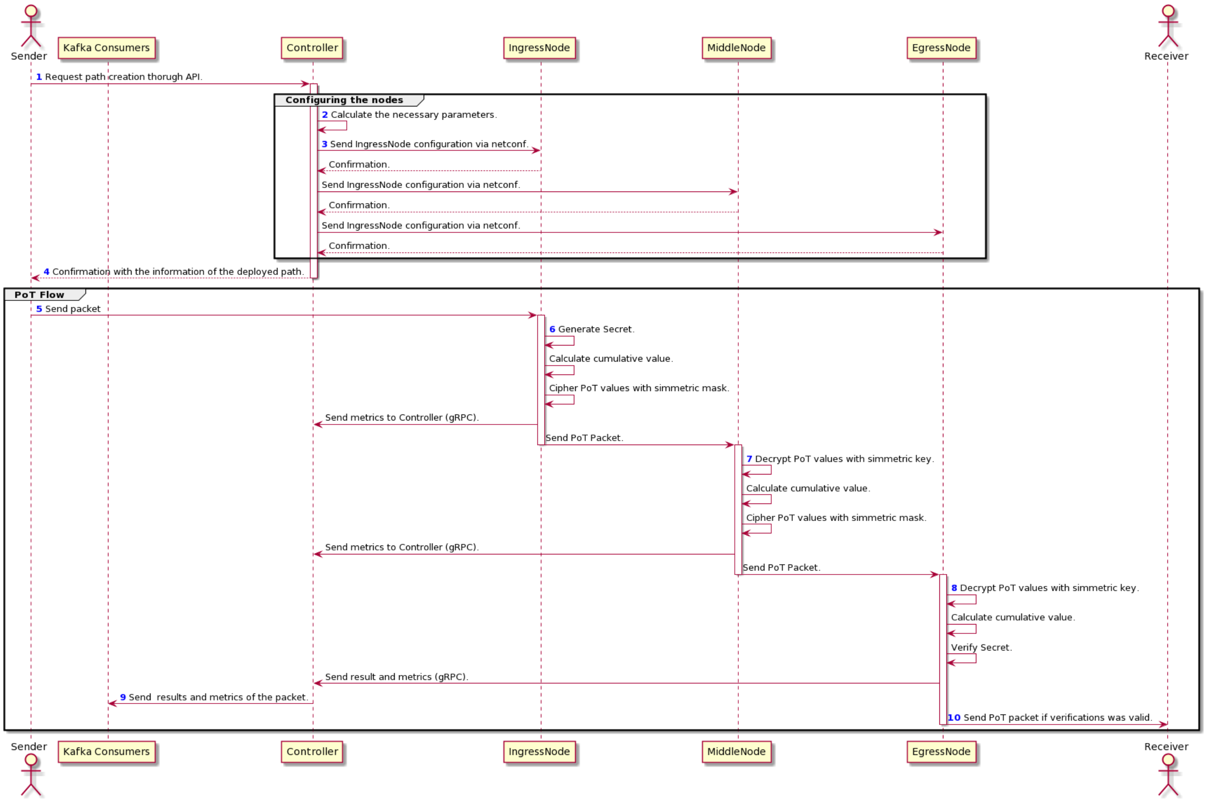
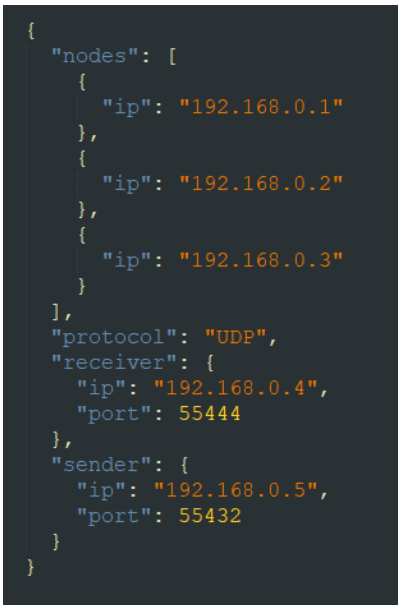

Sender
Kafka Consumers
Controller
IngressNode
MiddleNode
EgressNode
Receiver
1 Request path creation thorugh API.
Configuring the nodes
2 Calculate the necessary parameters.
3 Send IngressNode configuration via netconf.
Confirmation.
Send IngressNode configuration via netconf.
Confirmation.
Send IngressNode configuration via netconf.
Confirmation.
4 Confirmation with the information of the deployed path.
PoT Flow
5 Send packet
6 Generate Secret.
Calculate cumulative value.
Cipher PoT values with simmetric mask.
Send metrics to Controller (gRPC).
Send PoT Packet.
7 Decrypt PoT values with simmetric key.
Calculate cumulative value.
Cipher PoT values with simmetric mask.
Send metrics to Controller (gRPC).
Send PoT Packet.
8 Decrypt PoT values with simmetric key.
Calculate cumulative value.
Verify Secret.
Send result and metrics (gRPC).
9 Send results and metrics of the packet.
10 Send PoT packet if verifications was valid.
Sender
Kafka Consumers
Controller
IngressNode
MiddleNode
EgressNode
Receiver

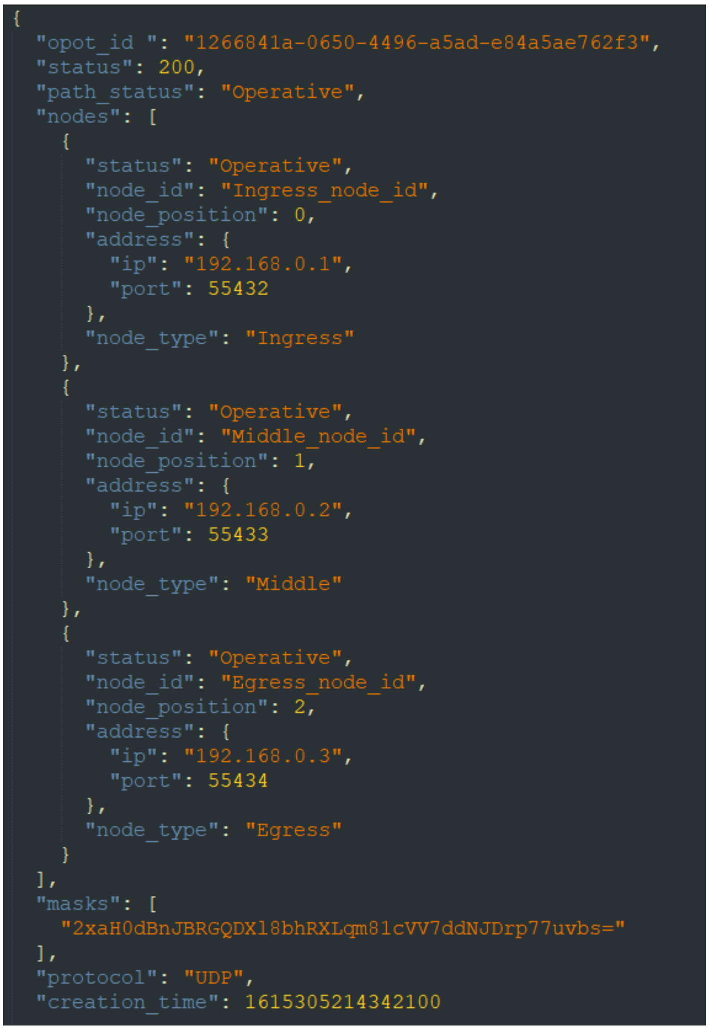
Figure 9: PoT internalinternal components workflow

1. The client sends a POST to the API as JSON formatted data containing the configuration to set up a path.



```
{
  "nodes": [
    {
      "ip": "192.168.0.1"
    },
    {
      "ip": "192.168.0.2"
    },
    {
      "ip": "192.168.0.3"
    }
  ],
  "protocol": "UDP",
  "receiver": {
    "ip": "192.168.0.4",
    "port": 55444
  },
  "sender": {
    "ip": "192.168.0.5",
    "port": 55432
  }
}
```

2. The controller calculates the Lagrange parameter for the Shamir's secret sharing scheme, which is used to identify each node, and the symmetric masks used between nodes.
3. The controller configures the first node in the via using NETCONF protocol with the structure defined in the ietf-pot-profile yang model [1].
4. The controller returns a 200 status code and the JSON which contains the status, ID, position, address and type of every node along with the masks, protocol and a timestamp (in Unix format).



```
{
  "opot_id ": "1266841a-0650-4496-a5ad-e84a5ae762f3",
  "status": 200,
  "path_status": "Operative",
  "nodes": [
    {
      "status": "Operative",
      "node_id": "Ingress_node_id",
      "node_position": 0,
      "address": {
        "ip": "192.168.0.1",
        "port": 55432
      },
      "node_type": "Ingress"
    },
    {
      "status": "Operative",
      "node_id": "Middle_node_id",
      "node_position": 1,
      "address": {
        "ip": "192.168.0.2",
        "port": 55433
      },
      "node_type": "Middle"
    },
    {
      "status": "Operative",
      "node_id": "Egress_node_id",
      "node_position": 2,
      "address": {
        "ip": "192.168.0.3",
        "port": 55434
      },
      "node_type": "Egress"
    }
  ],
  "masks": [
    "2xaH0dBnJBRGQDXl8bhRXLqm81cVV7ddNJDrp77uvbs="
  ],
  "protocol": "UDP",
  "creation_time": 1615305214342100
```

5. The client (sender) starts using the path sending the first packet (Figure 10) using the protocol previously defined.
6. The IngressNode receives the packet and creates a random number between 0 and the primer number defined in the Shamir's secret sharing scheme. Then calculates the CLM and the sequence number to generate the PoT packet.

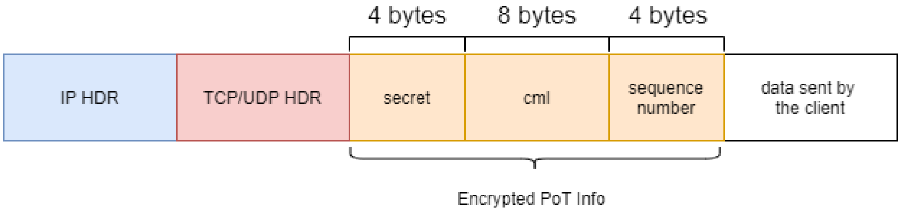
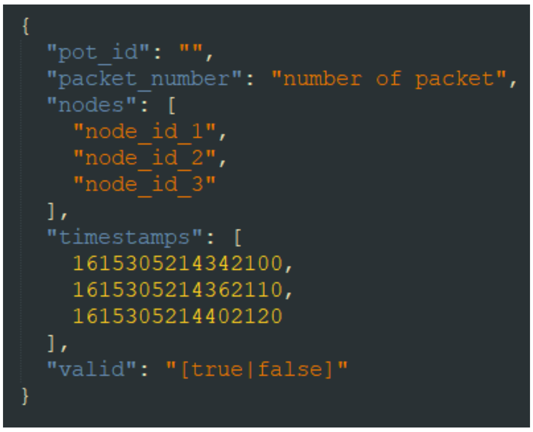


Figure 10: PoT packet

7. The Middle node receives the packet from the IngressNode and decrypts the PoT values using the symmetric mask. Then, calculates the new CLM and sends the packet to the next node.
8. The EgressNode decrypts the PoT values using the mask and calculates the CLM value. Then verifies that $CML_i\ (mod\ p) == RND$.
9. The controller receives the metrics from the EgressNode and checks if the result of the validations was incorrect and calculates where did the error occurred in case it happened. Then, the controller sends the results to the Kafka consumer.



```
{
  "pot_id": "",
  "packet_number": "number of packet",
  "nodes": [
    "node_id_1",
    "node_id_2",
    "node_id_3"
  ],
  "timestamps": [
    1615305214342100,
    1615305214362110,
    1615305214402120
  ],
  "valid": "[true|false]"
}
```

If the verification has been successful, the EgressNode sends the data from the client(sender) to the receiver.

Based on the modular design, Controller and node implementation can run in any x86 based architecture, with support for dockers or Openstack.

Documentation and code are stored in a public repository[6].

The license associated with the code is Apache 2.0.

4.2.7 Residual challenges

The main technical limitation from PoT is that it is based on the assumption that we are selecting specific nodes on the network to validate, not all nodes. Adding new nodes, such MiTM or redirection of the path between PoT nodes, will be not detected.

[6] https://github.com/HugoRP97/cne_opot_sdk_public/

4.3 eTRM: Trust and Reputation Model

4.3.1 Description of problem and challenges

5G is expected to be such a flexible and dynamic network to fulfil the myriad of use cases with extremely different requirements, such as ultra-low latency or ultra-reliability.

To meet such demanding and diverse requirements imposed by the verticals, 5G slicing is proposed to deploy several logical networks on top of the same infrastructure, in a customized way, where each slice is optimized to fulfil certain objectives imposed by specific use cases.

Software-Defined Networking (SDN) and Networks Function Virtualization (NFV) are possible candidate technologies for 5G slicing. On one hand, SDN is a networking paradigm that proposes to transition from network configurability to network programmability through network abstractions, open interfaces and the separation of control and data plane. On the other hand, NFV is a framework to virtualize network functions and deploy them in commodity hardware to lower the cost and time-to-market.

Several multi-partner business models are possible in 5G. One example is the "operator offer enriched by partner" business model coined by the NGMN alliance [4]. This model consists of an enriched connectivity offer by an operator (such as an integrated streaming service) which is enriched by a third-party application. We could imagine that this third-party application is implemented as a VNF and it is embedded in a node belonging to the third party domain. The operator acts as intermediary between its customers and the third-party to provide them with connectivity to that VNF. However, a fault in the VNF leads to an end-to-end service failure because the customers cannot receive the service. Paradoxically, although the third-party is the actual responsible party for the failure, the operator is who must compensate customers in the first place for the violation on the service quality. This example manifests the fact that responsibility in multi-partner services is blurred because the service is delivered among several parties. Indeed, each partner must trust each other regarding the decisions made by the rest of partners to contribute to meet the required overall service quality level. Therefore, not only cooperation suffices, but also a minimal level of transparency on how each partner manages their domain is essential. We need trust and reputation mechanisms to increase trust between partners in such multi-partner services.

The presented enabler can deal with several challenges (see §3 Grand Challenges) regarding network resilience and security issues very frequent in such softwarized infrastructures with logical centralization of the control plane functions. The grand challenges addressed by eTRM enabler are presented hereafter:

- **Grand challenge #2:** Protect SDN controller, the applications used and infrastructure from different threat vectors
    - **Global target:** SDN controller and its topology information are trustable.
        - The eTRM will be able to validate the input network graph and probabilistic network graph provided by the RCA (liability enabler to be developed in T4.3). However, this depends on the format of the topological information given by the specific type of SDN controller.
    - **Global target:** Provide and evaluate dynamic trust of the Domain's topology.
        - The eTRM will be able to convert and translate the probabilities given by the RCA into a reputation model.
- **Grand challenge #5:** 5G network resilience
    - **Global target:** Fault detection and management by means of RCA
        - The eTRM will be able to provide as additional indicator of the reputation of any networked element in an SDN domain, therefore resilience can be assessed based on those elements that are impacted by faults and failures.

- The RCA will be able to present a network graph and a probabilistic graph for a SDN domain that can be updated in real time depending on networking changing conditions.

4.3.2 State of the art

We focus on computational trust models [5]. Such models are generally used to assess the risk of a given interaction between two elements (a trustor and a trustee) within a system. The goal of a computational trust model is to support and automate the decision of the elements in the system with respect to engaging in a communication with other elements. The risk of engaging such an interaction is generally based on the experience perceived by the trustor from past interactions with the trustee.

Trust is generally known as the subjective degree of belief a trustor has on a trustee to perform a concrete task in this specific system.

A computational trust model is built in two phases: an evidence space and a trust space. In the evidence space, the trustor collects the evidence of the trustee node, which are first-hand evidence for direct interactions with the trustee and second-hand observations for those evidence collected through intermediary nodes directly connected to the trustee.

The trust space maps the collected evidence of the trustee to a trustworthiness value Ƭ, typically in the interval [-1,1]. The collected evidence can represent positive past interactions (depicted as p or negative past interactions (depicted as n) between trustor and trustee.

Trustworthiness concerns individual interactions between a trustee and a trustor. However, the trustor needs to collect and aggregate all evidence from all the nodes in its network to build an overall perception on the trustworthiness of the network. Reputation corresponds to this definition, being the collective perception by the network of the trustworthiness of a given trustee. Reputation calculation is very similar to the trustworthiness value but using the collected evidence p and n aggregated across all nodes in the network.

The role of a trust and reputation mechanism is to compute the trust and reputation model but also to update it with the newly collected evidence. Trust and reputation models have been applied to many different networking technologies with very different purposes. Most of those purposes are related to security.

A first example is wireless sensor networks, where the equipment is hardware-constrained and very easily compromised. A trust and reputation model can be used so to evaluate the trustworthiness of the captured information on an equipment. A second example is cognitive radio networks, where the spectrum information provided by the sensing units must be verified by means of a trust and reputation model to make sure interference between the secondary users and primary users (legitimate users have the right to transmit in those frequencies) do not occur.

Mughal et al. in [6] focus on isolating malicious SDN controllers by finding mismatches between the flows to be installed and the actually installed. The authors propose a trust model based on reputation scores of controller’s performance. Marconett et al. in [7] propose a hierarchical broker-agent system to coordinate different SDN controllers to enhance the scalability of multi-domain SDN. Each broker is located at each domain to install flows on the data plane by means of the SDN controller of that domain. The authors propose the reputation to quantify the goodness of the flows installed by each broker. Betgé-Brezetz et al. in [8] propose a trust-oriented controller proxy that intermediates between the controllers and the data plane by making sure the flows sent by different controllers are correct. There are works that focus on making an efficient use of the networking resources by the SDN applications such as the proposed by Isong et al. in [9], where trust is incorporated between SDN applications and the SDN controller.

However, we identified that these trust and reputation mechanisms for SDN are focused on security aspects where the goal is to detect malicious SDN controllers or SDN applications, but not other aspects such as performance or fault tolerance.

4.3.3 Description of the solution

This enabler brings together the notions of trust, reputation and fault management. We propose to transpose the notion of reputation, generally linked to security aspects, to score the effort made by an SDN domain to manage faults of non-intentional nature.

The goal of this section is to propose an online reputation framework for multi-domain SDN environments composed of heterogeneous domains cooperating in an end-to-end service. This framework objectively quantifies in real-time the effort made by each SDN domain to manage faults.

We propose a domain reputation metric to score each SDN domain based on the criticality of the injected faults and their probability of appearance. As a side note, this enabler depends on the RCA that will be described in D4.3.

The reputation mechanism scores in real-time the effort made by each SDN domain to maintain an acceptable end-to-end service level. This effort is scored based on the resulting probability of fault of the networking elements composing the domain, which is computed in real-time. Figure 11 shows this concept, where the reputation mechanism scores the SDN domain based on the amount of faults n occurring in a domain and their criticality for the end-to-end service.

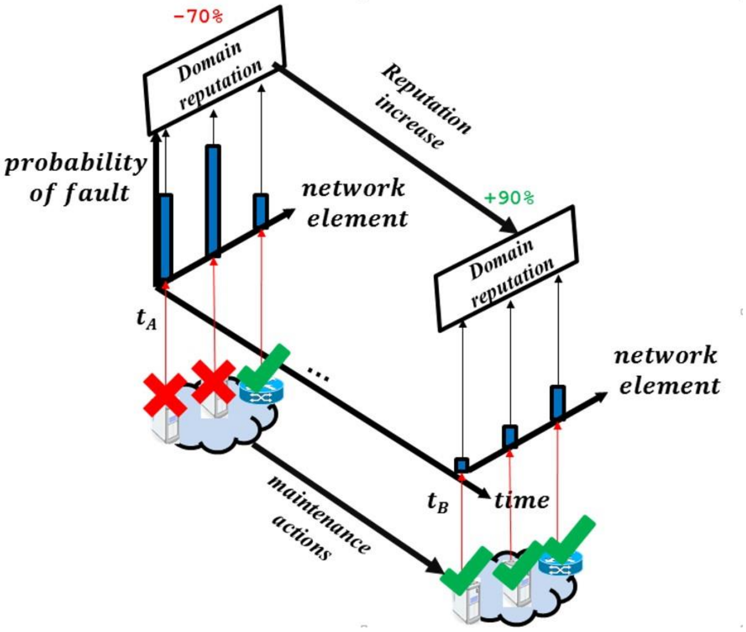


Figure 11: Domain reputation assessment based on fault tolerance in a SDN domain

In SDN, faults at the control plane tend to be more critical than faults at the data plane, so this reputation mechanism must incorporate the importance of the faulty elements in this score. For instance, there is a fault in a given domain at instant t_A impacting the SDN controller. As a result, the reputation mechanism attributes a very low reputation value because the domain has not protected its most important network element. The reputation score has a negative value to warn the rest of domains about this fact. Nevertheless, once the faulty domain undertakes the corresponding maintenance actions to mitigate those faults at instant t_B, the probability distribution of fault is therefore updated and the reputation of that domain is assessed again, increasing as a result its reputation score. The online reputation assessment framework for SDN is shown hereafter.

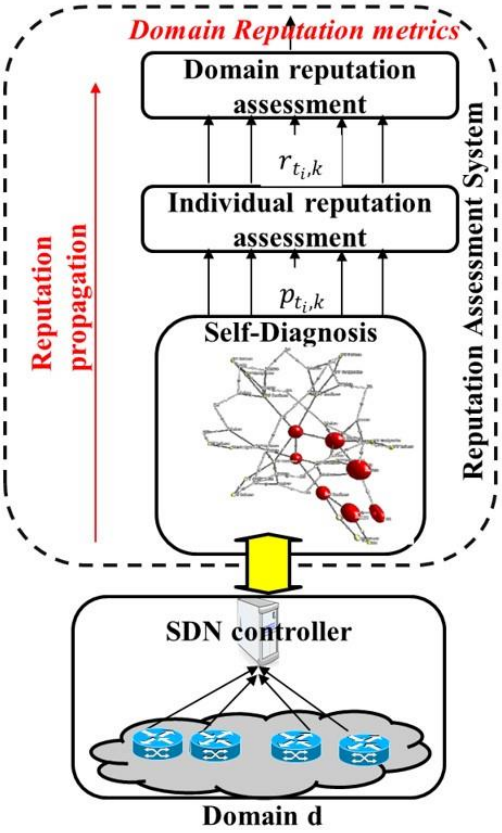

Domain Reputation metrics
Domain reputation assessment
$r_{t_i,k}$
Individual reputation assessment
$p_{t_i,k}$
Self-Diagnosis
Reputation propagation
Reputation Assessment System
SDN controller
Domain d

Figure 12: Proposed reputation assessment mechanism for SDN

This framework, shown in Figure 12, is based on three blocks, the individual reputation block, the domain reputation block, and the self-diagnosis block. The reputation assessment framework makes possible the propagation of reputation in two phases.

1. The first phase consists in the assessment of the reputation of each individual network element, where the reputation assessment block collects evidence on the individual faults in the networking resources, which are provided by the self-diagnosis block (RCA), that will be described in D4.3.

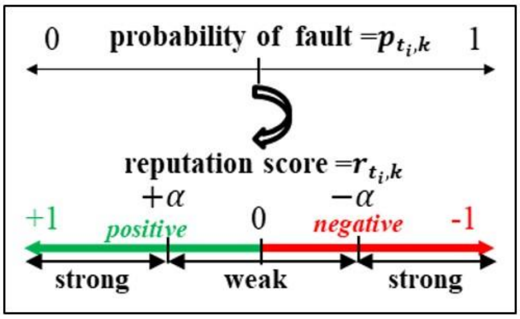

0
probability of fault $=p_{t_i,k}$
1
reputation score $=r_{t_i,k}$
$+\alpha$
$-\alpha$
+1
positive
0
negative
-1
strong
weak
strong

Figure 13: Reputation metrics based on probability of fault

The more likely the network element is faulty, the lower the reputation value gets (Figure 13).

2. The second phase consists of aggregating those n individual reputation scores corresponding to all those networked elements in the domain to compute the global reputation of the domain. These are the metrics exposed to higher layers to make appropriate decisions over that domain. At this stage, individual reputation values in the range [-1,1] will become more sparse for a domain composed of n network elements [-n,n].

4.3.4 Efficiency factors

The goal of this enabler is to reduce or increase the level of trust and reputation in a networking domain based on the probability of faults (provided by an RCA block that will be described in D4.3).

The efficiency of this enabler to compute the right values of reputation can be potentially measured by verifying that the following rules are always met:

- When the probability of fault of any networked element is neutral (around 0.5) its reputation becomes near zero as an intent to stay neutral regarding that network element.
- When probability of fault increases significantly, its reputation value should plummet to zero.
- When probability of fault drops below a certain threshold, its reputation value should increase but in a controlled way, as reputation must be very easy to lose but very difficult to earn.

Figure 14 shows how the reputation model should evolve over time. Only one sample of fault probability is needed to plummet reputation score to zero, while several positive probability of fault values (e.g., <0.3) are required to increase its reputation score. This increasing will be proportional to the number of consecutive positive samples 1, 2, and 3 provided as example.

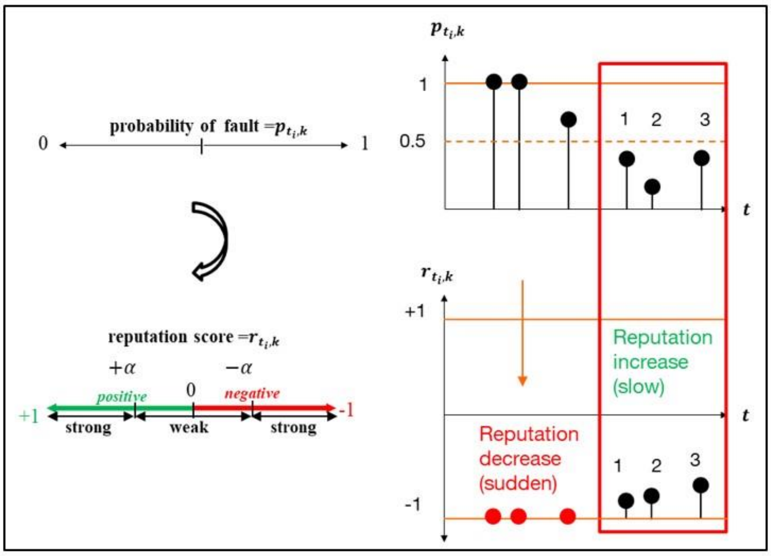

probability of fault $=p_{t_i,k}$
0
1
reputation score $=r_{t_i,k}$
$+\alpha$
$-\alpha$
positive
0
negative
+1
-1
strong
weak
strong
$p_{t_i,k}$
1
0.5
1 2 3
t
$r_{t_i,k}$
+1
Reputation increase (slow)
t
Reputation decrease (sudden)
1 2 3
-1

Figure 14: Example on how individual reputation values are updated over time

Figure 14 describes the reputation model envisioned for our enabler. In case of any network degradation leading to high probability faults, the reputation value should quickly decrease to zero.

4.3.5 Development outcomes

The recent development outcomes concern the update of the swagger API file and the publication of the enabler as open source by following a BSD license.

4.3.6 Implementation

This enabler has been implemented as an add-on on the RCA-VNF enabler.

The TRM is dependent on the RCA-VNF enabler implementation, which integrates different open source software packages. The SDN controller is based on Floodlight [84] and the SDN infrastructure is emulated with Mininet [85] The Bayesian Network algorithm is based on the Kevin Murphy's Bayesian Networks Toolbox [86], running in MATLAB [87]. We implemented the Graphical User Interface (GUI) in Python with the Qt software library [88]. The fault propagation model is visualized in 3-D with UbiGraph [89], which allows for visualizing the dynamic and interactive dependency graph encompassing the interactions among SDN resources and their components.

- System requirements: Virtual Machine on Ubuntu (64 bit), HDD 20 GB, RAM: 3512 MB
- User manuals: Download, install, execution.
- License: private license (internal process to externalize as open source code under BSD license)

4.3.7 Residual challenges

The eTRM depends on the concrete API provided by each particular SDN controller, as it is directly attached to an SDN controller. Each SDN controller propose specific API and provides the network topology in a given format (JSON based)

4.4 Component certification tool

4.4.1 Description of problem and challenges

For trusting a slice, the end-user needs to have a list of trustworthiness properties for the different parts included in the slice and to combine them in order to have a synthetic view. What are these properties? How to retrieve the information from the different parts? How to normalize these properties, how to evaluate them with metrics?

As different static properties can be evaluated using the description of the component (VNF descriptor used also for the deployment) or the source code, these properties could be evaluated by an evaluator and certified by a certification body. These properties could be exposed to any users wanting to select components following different trustworthiness constraints.

The main challenge is to define common trustworthiness properties for the different components of the 5G slice and to have metrics for each one. These metrics should be adapted for the different kinds of components.

The Component Certification Tool will provide a set of trustworthiness properties helping the entity in charge of the whole 5G architecture to monitor and to react in case of trust breach. With these properties will address the following grand challenges.

- **Grand challenge #1:** Attack surface reduction for Virtualized environments (including MEC infrastructure, VM, VNF / Containers, hypervisors and internal communications in NFV environments)
  - **Global target:** The Component certification tool (CCT) will integrate an integrity trustworthiness property giving the means to check the integrity of the different virtualized component and/or the integrity of their configuration. These values will help different tools for detecting malware insertion or configuration modifications. The reference values will be defined at design time and so the runtime specificities will not be checked. CCT will store and delivers the way for checking the integrity. The specific value to use should be delivered by component provider or by another enabler.
- **Grand challenge #2:** Protect SDN controller, the applications used and infrastructure from different threat vectors
  - **Global target:** As for the Grand challenge #1, CCT will integrate a integrity trustworthiness property giving the mean to check the integrity of the SDN Controller and the integrity of its configuration. These values will help different tools for detecting malware insertion or configuration modifications. CCT will store and deliver the way for checking the integrity. The specific value to use should be delivered by component provider or by another enabler.
- **Grand challenge #3:** Secure Verticals over 5G infrastructures
  - **Global target:** From a tenant point of view, the 5G infrastructure will be seen as a group of components and platforms. In order to help the end-user for trusting 5G, CCT will give different information regarding the design and the implementation of the different parts of the whole infrastructure. For each part, a DTwC (Digital Trustworthiness Certificate) will be delivered. CCT will give mainly information based on static analysis, but not the runtime behaviour. CCT will deliver information from VNF components and software components (mainly java component). A generic approach for delivering manually an evaluation for any kind of components will be envisaged.
- **Grand challenge #6:** Monitor the trust of 5G networks and 5G Services due to the new complexity of these infrastructures (multi tenants, heterogeneous, services based, multi-party)

- **Global target:** CCT, by delivering metrics for trustworthiness properties for each component and aggregating them will give a reference of trustworthiness at infrastructure level. This reference is based only on static properties and not based on metrics evaluated dynamically during the whole life-cycle of the infrastructure. By selecting the different components by using the values stored in the different DTwC, the 5G provider can build an infrastructure answering different trustworthiness properties. Only a part of the different kinds of component will be envisaged for the slice.

4.4.2 State of the art

Trust in ICT systems was investigated in FP7 OPTET[7] project. In this project, a conceptual model was delivered and the link with trustworthiness was established [10]. This project also delivered a set of enablers and especially a certification tool [12] evaluating the trustworthiness of an application or service developed in Java and in Opa[8]. Another output was the Digital Trustworthiness Certificate [11] describing the application (its environment, its trustworthiness problem) and the scores for different trustworthiness metrics. Based on these results, the H2020 5G ENSURE[9] project has refined the trust and trustworthiness definition applied for 5G Security architecture. This project has delivered an up-to-date trustworthiness model [13] and an enabler certifying the VNF used in 5G slices [14]. In INSPIRE-5Gplus, the goal of Certification tool is to extend this approach to other components of the slice like, for example, SDN, Cloud infrastructure, and security assets.

4.4.3 Description of the solution

The solution is to propose a static evaluation of the different components if possible (if they have descriptors, source code). For each component, adapted metric should be defined and could be measured automatically or manually. These metrics would be combined for defining trustworthiness properties exposed by the components. This approach was already followed for VNF and a Java application. The solution in INSPIRE-5Gplus would reuse an existing work and would complete it for different other kinds of component.

4.4.4 Efficiency factors

The evaluation of the trustworthiness properties for a component will give a first idea regarding its capacity to be resilient for different attacks. As the properties will be evaluated with metrics, the DTwC will give a score for each property. Depending on the SSLA requested by the 5G provider, CCT can help by selecting the most appropriate component for the requested SSLA.

4.4.5 Development outcomes

CCT developments are based on the existing version. In previous versions (see the detailed description of the enabler), it was developed for Java software (developed in the context of FP7 OPTET) and for VNF using a TOSCA descriptor (developed in the context of 5G ENSURE).

The next steps are to identify the new components to be evaluated, their trustworthiness properties and their associated metrics. The previous versions have implemented automatic evaluations but also manual evaluations (based on evidence coming from test results for example). This approach could be reused for evaluating a larger number of components.

[7] OPerational Trustworthiness Enabling Technologies | OPTET Project | FP7 | CORDIS | European Commission (europa.eu)

[8] The Opa Language: http://opalang.org/

[9] 5G ENSURE: https://www.5gensure.eu/

4.4.6 Implementation

The existing version is operational but has some limitations. It only evaluates VNF described with TOSCA and OSM and evaluates also java source code but for an outdated version of java.

This version to be modified with the following evolutions:

1. Its DtwC repository should host Certificates for different components and have an OpenAPI compliant with the need of "Trusted Blockchain-based Network Slices".
2. DtwC should be able to import data from existing enablers outputs (MANIFEST and Systemic VNF wrapper) automatically or manually.
3. The existing list of trustworthiness properties should be adapted depending on the components. This development will follow the work performed in T4.2 defining the main properties to be consolidated at domain level for TSLA.

The first one is scheduled before end of 2021 in order to be integrated with "Trusted Blockchain-based Network Slices" but for the two other ones, they will depend on the ongoing work in T4.2.

- System requirements: HW & software:

The CCT is delivered as a docker container and allow the whole process of certification. Another component is the DtwC repository which can be deployed outside the container. It is a webapp under the form of WAR and should be deployed in any Application Server (tested with TOMCAT v7).

- User manuals: Download, install, execution.

A previous version (coming from H2020 5G ENSURE project) is available (D3.4 5G-PPP Security Enablers Documentation) for the project but would need to be updated before the end of the project.

- Code repository link.

The code in a private gitlab hosted by Thales:

https://foyer.dev.theresis.org/application-security/VNFCertification

The repository of DtwC is in the same location.

- License: private or open-source (including the one selected).

Previous versions used open-source license but for the version including the different evolutions mentioned in INSPIRE-5Gplus, the choice will be more a private license, but it not set for now.

4.4.7 Residual challenges

The main constraints are the components descriptors and different possibilities are existing. For example, for VNF, the existing version was adapted for TOSCA and OSM but it is a limitation. For software, only Java code is evaluated. So the technical constraints are to define the components we want to evaluate and the technologies used for evaluating the enabler. To bypass this limitation, a manual way could be used for evaluating a component by using different reports and information provided with the component. In this case, a DTwC would be also delivered.

4.5 Trust Reputation Manager

4.5.1 Description of problem and challenges

As 5G deployment is carried out, some mechanisms are required to ensure the trust and reliability of both the underlying infrastructure and services and applications deployed on it. Therefore, it is mandatory to develop a system capable of calculating, from both historical and live data, how reliable a cloud (or a service) is. This becomes more relevant on 5G scenarios, since a client will be able to know how reliable the infrastructure on which its services are being deployed is, and a Cloud Service of Provider (CSP) will be able to know the reliability of the services that will be deployed on it. An important issue is not only the need to quantify the reliability but also offering valuable and trustable information in a non-repudiated and auditable way. For that purpose, the use of Smart Contracts is useful to perform the required operations.

Furthermore, the values obtained are envisioned to be also used by the Security Orchestrator as a trigger to deploy countermeasures if an abnormal situation occurs and needs to be mitigated.

- **Grand Challenge #3:** Secure Verticals over 5G infrastructures.
  - **Global Target:** The approach is, as said, to measure and evaluate the trust level of 5G components, platforms, etc, and share the obtained values in a safe and trustable way. In that sense, all the metrics and values obtained (metrics, historic data, SSLAs, etc.) will be used to calculate the trust value of the platform. Moreover, the process will take part in the Smart Contract to ensure the liability and trust of the obtained values.
- **Grand Challenge #6:** Monitor the trust of 5G networks and services due to the new complexity of these infrastructures (multi-tenants, heterogeneous services based, multiparty, etc.)
  - **Global Target:** The idea is being able to offer the trust values regardless of the domain or tenant. That is, when an entity needs or requires to use a different infrastructure, they will be able to know, in a liable way, the trust value of the infrastructure, without the need of further knowledge about the infrastructure itself. This will simplify relationships between domains and third party infrastructures such as other operators, which are some of the basis of 5G paradigm. For this purpose, every domain will implement a Trust Manager Service, which will be in charge of calculating and storing the trust values in data services. Therefore, they will be available when required.

4.5.2 State of the art

Trust Reputation Manager relies on two important concepts: Distributed Ledger Technologies, and the process of obtaining a "trust" score of an entity. Both terms can be used together, as we can use the benefits DLT offers (as traceability and security) to obtain trust values in a guaranteed way. There are some approaches that use them, as they are analysed in [15]. In our case, we use as DLT technology HyperLedger Fabric [16], a DLT platform that supports Smart Contracts. A Smart Contract is a piece of code (executable logic) whose results (or facts) are added to the ledger as a new transaction [17].

4.5.3 Description of the solution

To provide a solution to this problem, a Trust Manager mechanism will be implemented, designed as a Smart Contract, which will calculate the trust and reliability of a cloud infrastructure, or the services deployed on it, based on multiple values for both the infrastructure and the services. Different types of Trust Manager can be offered (with different Smart Contracts for each of them), depending on the element the trust is being calculated. The information from which trust is calculated is listed below:

- Attestation of VMs, hypervisors, and network traffic as well as the information coming from the entities dynamically deployed to enforce security policies, such as detections, decisions and reactions. This input will come from multiple monitoring services, deployed throughout the infrastructure, which will offer the information both in real-time and by storing it in a historic.

- System Model/topology of the infrastructure, as well as Manifests and VNFs software execution evidence.
- Audits from Remote Verifiers. Remote Verifiers are entities in charge of performing analysis of the services or the underlying infrastructure. The number of present Remote Verifiers is variable, as there may be several, each focused on a specific aspect of security and/or to have second opinions on certain fields. Concerning Remote Verifiers reports, a series of Smart Contracts are defined, which determine the way the attestations are performed, therefore making those attestations auditable. The remote verifiers obtained reports, using the defined smart contracts, are supported by the blockchain infrastructure, providing traceability and auditing.
- The Security policies and SSLAs defined on the environment, to ensure their compliance.

With all this information, Smart Contracts are defined, whose output includes the trust value (score obtained in a quantitative way) of VNFs and services or the VIM, as well as the SLA and SSLA compliance (if applicable) and the Security Policies verification. For trust calculation, a set of weight algorithms or conditions will be used, together with a Fuzzy Trust Evaluator, which will use weights and fuzzy logic (among others) to calculate it.

Since the trust score is calculated as a Smart Contract, the process and hence the score obtained is auditable and provides non-repudiation, as values are added to the blockchain. In this way, all the process and the events that occur are also recorded and stored in the blockchain.

4.5.4 Efficiency factors

The values obtained by the Smart Contract (the trust and reliability score) can be used as an indicator of the success of the Trust Manager. The required time to obtain the trust score can be also used as a metric of efficiency. Research of the efficiency of Smart Contracts and Hyperledger platforms is currently being performed.

4.5.5 Development outcomes

An alpha version of the Trust Reputation Manager is currently being developed. The required entries for trust calculation have been identified for those enablers identified as trust aware in the initial set of Use Cases, taking into account different entries for different entities. In addition, we are defining the way enablers should publish data to the Integration Fabric (in particular to the Kafka service embedded into it) and how the TRM will subscribe to those events to retrieve the information. The infrastructure on top of which the test cases relying on this enabler are deployed, must support this Integration Fabric, in particular the Kafka, as well as the DLT on which the Smart Contracts being defined are going to be deployed. In that regard, the testbed in Murcia has already the necessary elements as the reference deployment for the enabler.

4.5.6 Implementation

We have implemented two modules for the integration of the TRM. The first one corresponds with the publishing module, with which the enablers will be able to post the data required by the TRM. The second one is the subscription module. With the latter, the TRM will subscribe to the publications of the mentioned enablers in order to retrieve the data published by them.

The already available implementation resources are detailed next:

- System requirements: At a first glance, the main requirements involve at least 4GBs of RAM and 4 processor cores. The OS in used must be able to run the last version of Python. There is also the possibility of graphics needed according to the computational power required by ML techniques.

- TRM API for enablers' requests. GitHub Code repository link: https://github.com/INSPIRE-5Gplus/i5p-hla-api/blob/main/WP4_TrustReputationManager_UMU/TRM-api-1.0.0.yaml
- License: The development outcomes will be licenced under The MIT licence.

4.5.7 Residual challenges

The main current problem is to integrate the entries/values required to calculate the trust value from the rest of enablers. To address this challenge, the TRM will implement extensible mechanisms to be able to calculate trust values from the different INSPIRE-5Gplus enablers as well as from future security enablers. This approach will be validated in the testbed by generating multiple entries from both, INSPIRE-5Gplus security enablers and trust value generators developed for testing purposes.

Further relevant challenges are the ones posed by Hyperledger Fabric (as the calculation is performed inside a Smart Contract). There are no more technological limitations found at the time of writing.

4.6 Trusted Blockchain-based Network Slices

4.6.1 Description of problem and challenges

While most of the service security aspects during the life-cycle of a service are thought to be controlled while the service is active and being used, it is important not to forget the previous actions before the service is ready to be used.

When a user or an entity has a necessity to fulfil, the first option to solve that necessity is usually a third party with a recognised trustworthy label. The main problem of trusting third parties is the centralisation of trust around them, becoming a point of attention for attackers.

In order to avoid this centralisation of trust, an option is the use of Distributed Ledger Technologies (DLT). DLT, with its most known example Blockchain, may be a tool that can be used in order to add trust to the Network Slice components using a collaborative methodology. So, if one of the nodes involved in the Blockchain is attacked, the other nodes participating in the Blockchain are able to keep the data without being corrupted.

Based on the presented problem, this enabler aims to use Blockchain technology and the Certification Tool to face two of the six trust Grand challenges presented in Section 3:

- **Grand challenge #2:** The protection of SDN controllers, used applications and the infrastructure from different threat vectors by some approaches such as to ensure that VNF software is unchanged from design until execution or to provide trustworthiness technologies over multi-domain scenarios.
- **Grand challenge #6:** The trust monitoring of 5G networks and 5G services on the new complex infrastructures by looking to define robust trust management solutions for multi-operator scenarios.

4.6.2 State of the art

Blockchain is a Distributed Ledger Technology (DLT): a digital system that records asset transactions - i.e., money, resources, information- by saving the transactions and their detail in different places at the same moment. It might be understood as a database (DB) geographically distributed over a set of nodes that all together create a p2p network. Blockchain allows updating the information in an iterative and secure way. When a transaction is finished, its information and related metadata are saved in all nodes, making them all aware of that information and making it impossible to modify it without the others nodes knowing it. The main characteristics of Blockchain are:

- Distributed: As the data is distributed and there is no central authority, the system is robust against hacks.
- Secure: All information in the DB is encrypted using private and public keys.
- Public: The system is more transparent as there is no central authority to track and validate all the information, but all peers do it.

As there is no central authority, all the peers share the information and have the same rights to add or modify the data in the DB, which allows a stable and safe maintenance of the data. In order to do so in a fair and secure way, a consensus mechanism [18] is necessary. A consensus mechanism is a procedure that follows a set of rules publicly known by all the peers that lead to "democratic" decisions as only when there is a majority, an action is implemented. So, if one of them tries to act in a malicious way, the others will notice and block its fraudulent action. The most known examples of Blockchain are Bitcoin [19], Ethereum [20] and Hyperledger [21].

Blockchain has been already used to manage SDN/NFV networks with computing and optical network resources [22][23]. Other works have presented different possible scenarios in a multi-domain environment [24]. Other works used Blockchain to keep track of Service Level Agreements (SLA) events [25] or to quickly configure switches to be controlled by the most optimal master when their initial master goes down or becomes evil [26]. While in the previous papers and most of the networks and Blockchain literature, the focus is on the management of physical resources -e.g. optical path calculation, traffic SLA fulfilment and switches management-, but there is limited research looking into higher layer elements such as Network Services or Network Slices (NSs).

4.6.3 Description of solution

This enabler aims to enforce the security on sharing Network Slice resources before and while they are deployed. In order to do it, the objective is to use a Permissioned Distributed Ledger (PDL) based on a Blockchain network as an element to manage and ensure that only validated and verified Network Slices and their components will be accepted to be deployed and so, they can be trusted across all domains.

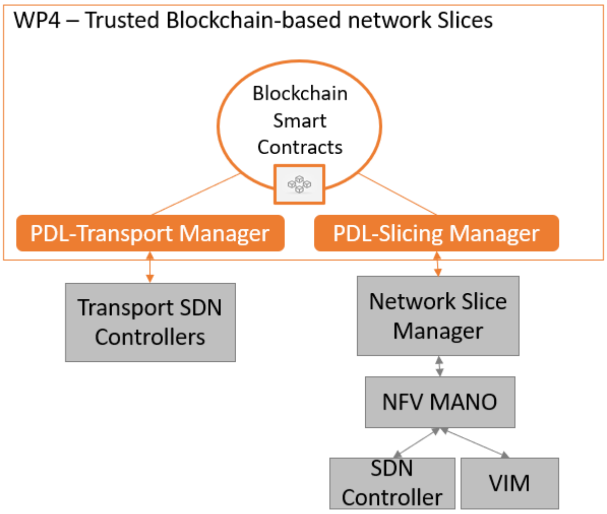

WP4 – Trusted Blockchain-based network Slices
Blockchain Smart Contracts
PDL-Transport Manager
PDL-Slicing Manager
Transport SDN Controllers
Network Slice Manager
NFV MANO
SDN Controller
VIM

Figure 15: Trusted Blockchain-based Network Slices architecture.

As Figure 15 shows, this enabler is composed of three main elements: a set of Blockchain Smart Contracts, and two different node roles, the PDL -Transport Manager and the PDL-Slicing Manager. Regarding the first of these two last elements, it is out of the scope of this project as it focuses on SDN and optical transport network aspects. The new enabler in INSPIRE5G-plus is the element called PDL-Slicing Manager which, as already described, aims to manage the Network Slicing resources from different domains using a Blockchain network.

4.6.4 Efficiency factors

The way to demonstrate the efficiency will be by proving that a Network Slice Manager may not offer a vertical service from its domain to the other multi-domain/operator Network Slice Managers, unless it has passed the appropriate tests through the certification tool. By doing so, only those descriptors properly validated will be accepted by all the other peers and will have their trust.

4.6.5 Development outcomes

Regarding the development of this enabler, the initial architecture has been designed and developed to allow a Network Slice Manager to participate in the Blockchain system. Moreover, a work on smart contracts (the current Blockchain system is based on Ethereum) has been done to allow operations between the peers. In addition, a set of tests had been planned and designed in order to start getting a set of initial results.

Currently we are finishing the tests to validate its main functionalities to manage E2E Network Slices (deploy them, terminate them and get their information). By the end of this current year, it is planned to have an initial demo integrating this enabler and the Component Certification Tool.

4.6.6 Implementation

The implementation of this enabler is done from scratch, this enabler was thought and designed originally for the INSPIRE-5GPlus project. In order to understand better how this enabler is implemented, the following characteristics are presented:

- System requirements: Hardware & software

Regarding the software requirements, this enabler need two types of associated nodes as it may have two different roles; the “PDL-transport manager” needs an SDN Controller to manage transport SDN networks (out of the scope of this project), while, on the other side, the “PDL-Slicing manager” role (main focus in this project) needs an NFV Orchestrator in order to manage the E2E Network Slices. In our case, we selected the SONATA NFV software as in this single piece of software, there are already an NFV Orchestrator and a Network Slice Manager. Finally, this enabler has been developed using Python language, the 3.9 version.

In relation with the hardware requirements, this enabler has no high-demanding requirements. For example, four of instances of this enabler have been deployed in the same physical machine in order to develop and verify its functionalities using an emulated Blockchain network. In fact, if there is any hardware requirement, those are requested by the use of the Blockchain system. As in our testbed we are currently using machines with 16GB of RAM memory, currently this amount seems to be good enough.

- User manuals: Download, install, execution.

Currently there are no user manuals for installation or usage (API) as its main functionalities are still being developed, but the objective is to have these information files by the end of the INSPIRE-5Gplus project, allowing any interested person to use this enabler. Despite there is no user manual, the idea is to have an installation as simple as possible. For that end, among the files composing the software of this enabler, there is a configuration fill-in with a set of parameters to configure, such as the NFV Orchestrator or the SDN Controller IP addresses and ports to associate the enabler with them and allow the exchange of information using a REST API.

In order to show how users can use this enabler, these are the current steps to install this enabler:

- Download the enable’s code from GitHub:

  ```
  $ git clone https://github.com/INSPIRE-5Gplus/i5p-netslice-mgr
  ```

- Open the configuration file called config_env.env and write the IP and port values where your enabler will offer its services together with the IP and port values of either the NFV orchestrator or the SDN Controller.
- Once done, use the following command to start the enabler:

  ```
  $ python3 main.py
  ```

- Code repository link.

This enabler’s code can be found in the INSPIRE-5Gplus GitHub repository:

https://github.com/INSPIRE-5Gplus/i5p-netslice-mgr

- License: private or open-source (including the one selected).

This enabler is being developed as an open-source project in order to make it available to the community once the INSPIRE-5Gplus project finishes.

4.6.7 Associated publications and patents

During this last year, we have described a set of initial results generated by this enabler through the presentation of three conference articles that present the evolution of this enabler during this last months. In the first two articles [27][28] the main focus was on the interaction between the Blockchain peers and the management of Network Slice resources, in the last article [29] we did one step further and added the use of Transport networks to interconnect with specific deployed virtual links the slice-subnets deployed in each domain to compose the E2E Network Slice.

4.6.8 Residual challenges

Currently no limitations appeared, but based on the work done up until now, if any limitations should appear, they would come from the selected Blockchain system and the use of smart contracts to deploy in it. For example, one of the Blockchain weaknesses is the necessity of a large number of nodes. The testbed to be used during the experimental and validation phase have a reasonable number of machines to be used as peers but, the testbed is shared with other projects and so, their time plans and use of machines could affect the correct test case evaluation.

4.7 The Risk Assessment Graphs

4.7.1 Description of problem and challenges

Assessing the risk in 5G networks is a rather a challenging task due to several factors e.g., large number of equipment, non-linear interactions evolving over time, virtualization. Indeed, the main problem here is to provide a risk assessment model that takes into account both the inherent dynamicity of 5G networks (due, for instance, to the slicing technology) together with the heterogeneity of the connected equipment. So, the model needs to be adaptable to the network change over time and capable of modelling any type of equipment. The main challenges that we address are the followings:

- **Two Grand challenges: #3/** Secure Verticals over 5G infrastructures and **#5/** 5G network resilience
    - **Two Global targets:** Trust in 5G Networks vertical services host / Trust in 5G Networks critical operations: The RAG delivered in T4.1 will be able to evaluate the security level of the provided 5G infrastructure to the Verticals. And it will be able to propose specific counter-measures to achieve the desired security level. In the 2021 version (T4.1), the RAG infrastructure will only model a VNF management infrastructure, i.e., we will be able to demonstrate the chaining of several Vertical's VNF over these infrastructures (and compute the optimized security counter measures placement to achieve the targeted security level). Extension in complexity will be investigated in T4.2 (Trust Management).

4.7.2 State of the art

The current state of the art includes multiple contributions to evaluate and quantify risks in ICT systems. Among the models that exist in the literature, one can mention the attack graphs [30][31] and dependency graphs [32][33]. The former relies on graph theory to describe how existing exploits may be chained to get root access to a system (also called an attack path). This kind of mathematical model offers several advantages such as providing a compact way to express the different possible attack scenarios on a system. Furthermore, the use of a graph offers a rather intuitive support for justifying the provided counter-measures or assessment measures to non-experts.

Regarding the dependency graphs, they are also based on graph theory and aim at modelling the inter-dependencies of the different components of a system. These types of graphs are mostly used to decide what would be the best answer against ongoing attacks, while attack graphs are used to give a risk assessment measure of the system.

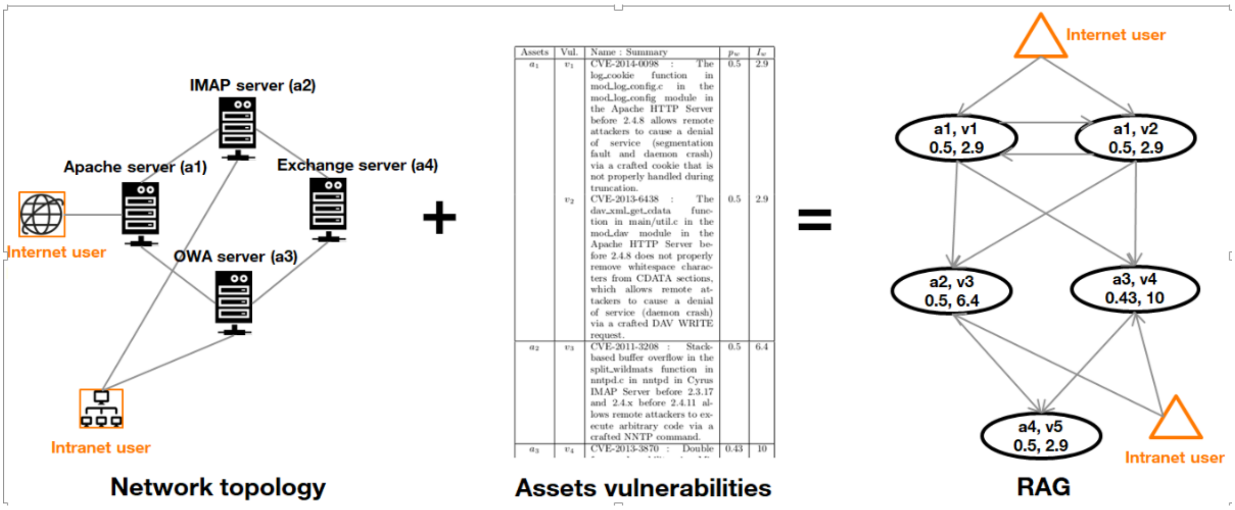
Most of these approaches consist of static models built during the design phase and do not consider threats that could occur during the lifecycle of a system. In order to cope with this issue, a new risk assessment framework to supervise the state of complex ICT systems has been proposed in [34]. The concept of the Risk Assessment Graphs (RAGs) and a quantitative risk evaluation approach have been developed.

4.7.3 Description of the solution

The concept of RAGs provides a new framework that captures simultaneously the topology of a system, the vulnerabilities, the accessibility between the components, their external exposure, and the way all these elements may evolve over the time. Thus, RAGs provide a framework for fine qualitative and quantitative risk assessment approaches to assess the impact of the exploitation of the vulnerabilities and their exposition surface throughout the nodes of the graph; to compute risk indicator metrics; and to observe their evolution over several time periods.

More precisely, the system is represented as a directional graph in which a node can be either be an asset-vulnerability pair or an access point. An arc in the RAG means that the exploitation of a vulnerability of the source node exposes the target node to the exploitation of its vulnerability. A path corresponds to a potential violation of a node. A potentiality function and an accessibility function are also introduced in the model. The former evaluates the likelihood of each attack at each time slot. On the other hand, the accessibility function gives the ratio of time the system assets are accessible from each other at each time slot. The accessibility and potentiality functions are used to evaluate, respectively, the nodes and the arcs at each time slot (see Figure 16).

RAGs could be used as an input to determine the best strategies to secure a system. Given a set of available countermeasures associated with the vulnerabilities (ranging from firmware updates or patches to VNF deployments), several optimization models have been developed to solve security-issue optimization problems [35], e.g., where to place countermeasures a priori to mitigate the risk of a chain of exploits.



Assets	Vul.	Name : Summary	p_w	I_w
a_1	v_1	CVE-2014-0098 : The log_cookie function in mod_log_config.c in the mod_log_config module in the Apache HTTP Server before 2.4.8 allows remote attackers to cause a denial of service (segmentation fault and daemon crash) via a crafted cookie that is not properly handled during truncation.	0.5	2.9
	v_2	CVE-2013-6438 : The dav_xml_get_cdata function in main/util.c in the mod_dav module in the Apache HTTP Server before 2.4.8 does not properly remove whitespace characters from CDATA sections, which allows remote attackers to cause a denial of service (daemon crash) via a crafted DAV WRITE request.	0.5	2.9
a_2	v_3	CVE-2011-3208 : Stack-based buffer overflow in the split_wildmats function in nntpd.c in nntpd in Cyrus IMAP Server before 2.3.17 and 2.4.x before 2.4.11 allows remote attackers to execute arbitrary code via a crafted NNTP command.	0.5	6.4
a_3	v_4	CVE-2013-3870 : Double	0.43	10

Figure 16: RAG solution

4.7.4 Efficiency factors

We propose to implement the RAG model and the associated optimization algorithms and test them to validate our approach in a use case involving the placement of VNFs in the most secure possible way. The efficiency of our method can be measured by the fact that it can substantially reduce, a priori, the probability that an attacker succeeds in reaching its target. In other words, the average number of succeeded attacks (e.g., DoS attacks) should be lowered.

4.7.5 Development outcomes

Currently, the RAG model, and risk evaluation algorithms and the optimization algorithm for solving the counter-measure placement problem are now re-implemented in Python C++17 17 (name of the C++ standard approved in 2017) in order to improve their running time. This enables the algorithms to scale much better than the previous Python implementation. as well as the optimization algorithm for solving the counter-measure placement problem using the MIP solver CPLEX. The REST API is currently in development and relies on Flask (https://flask.palletsprojects.com) and a Python binding of the C++ code.

4.7.6 Implementation

The implementation is done in C++ 17 with a binding in Python. It requires the use of an integer programming solver such as CPLEX which may require a licence.

System requirements: Ubuntu 20.04 (64bits), High-end CPU, 32GB

User manuals: README

Licence: private (Orange Source Charter)

4.7.7 Residual challenges

The optimization algorithms will depend on NetworkX (python package) and a MIP solver such as CBC or CPLEX which are rather CPU-time consuming. Therefore, the platform where the algorithms will be deployed must be powerful enough to run those solvers.

5 Trust enablers integration into INSPIRE-5Gplus architecture

The High-Level Architecture (HLA) of INSPIRE-5Gplus shown in Figure 17 serves a framework to integrate the different trust enablers described in Section 4 into the common architecture. The mechanism will be based on interactions based on interfaces between enablers and common components. Next subsections describe the position of each trust enablers and the identified interactions.

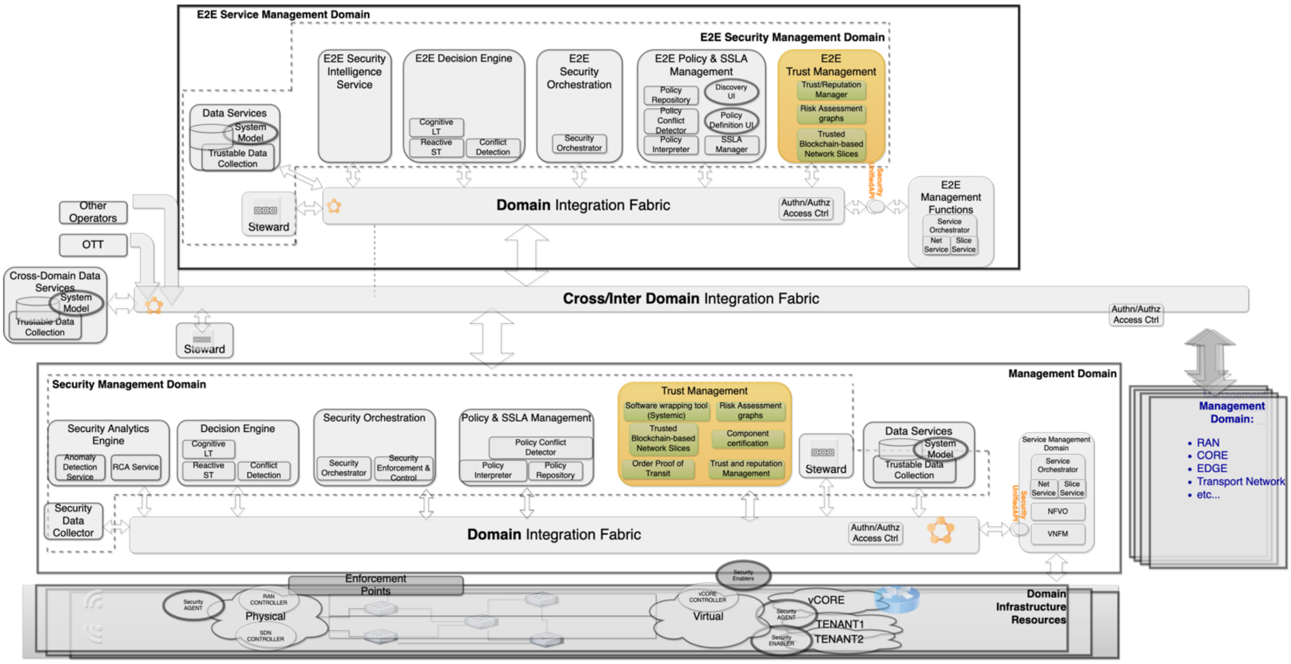


Figure 17: Trust in INSPIRE-5Gplus High Level Architecture (HLA)

5.1 Systemic VNF Wrapper

5.1.1 Enabler placement and interactions

5.1.1.1 Potential interactions with HLA components

Figure 18 below shows the most relevant placement of Systemic VNF wrapper inside the project's high level architecture. For clarity, the figure is a partial view of the complete HLA. The light blue areas represent the main areas of interactions of Systemic: security orchestration, trust management and Service management. As part of the Trust management area itself, Systemic may be interfaced with the other components inside the block but we have not set the linking arrows inside the block for clarity. The dark blue components are potential recipients or emitters of requests and data transfers to Systemic. The relevance and reality of each links are not yet defined at the time of writing of this deliverable and will be defined before the Project termination.

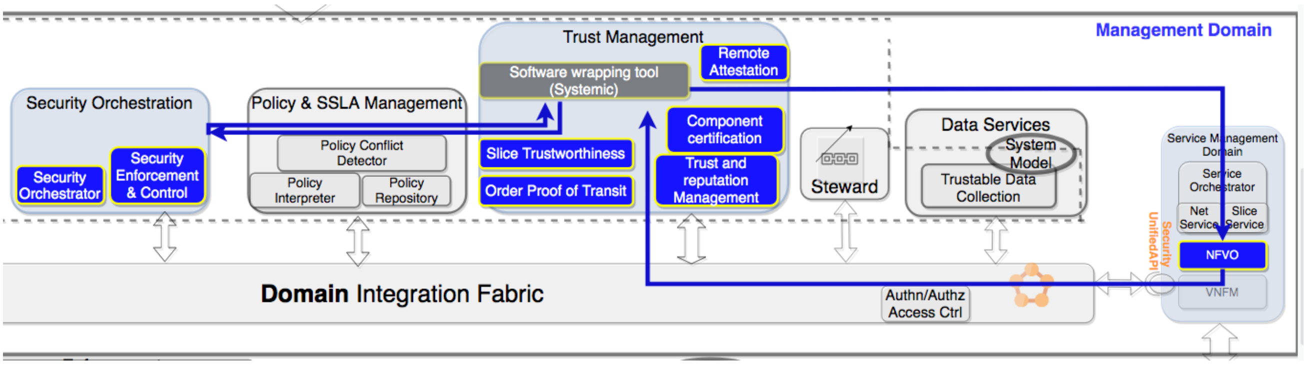


Figure 18: Systemic VNF wrapper placement in INSPIRE-5Gplus HLA

As opposed to other enablers of the Project, Systemic does not directly depend on such inter-enabler interactions. In fact, Systemic SECaaS can be offered as a stand-alone and separate transversal security enabler. There are not firm and specified interfaces for Systemic.

Nevertheless, its integration and interfacing with other enablers bring automation, a very valuable, if not unavoidable, feature of 5G and beyond network management.

Interfacing Systemic also enlarges the scope of trust and liability management as all Systemic-provided software security properties can integrate the scope of metrics accountable for trustworthiness and liability management. Hence, expanding beyond the today's classical and current circle of trustworthiness metrics is a noticeable plus for INSPIRE-5Gplus.

Interfacing Systemic is therefore an enriching and solidifying path which strengthens Systemic's position for 5G networks and beyond.

As a consequence, a significant and high-value part of Tages work in INSPIRE-5Gplus is the design of comprehensive and easy APIs aiming at its easy use by other components of the network security, trust, liability and management.

A synthetic view of Systemic potential and non-exhaustive interfaces is given here, highlighting the main objective and link data of the interface. This list can be extended, and some links may appear to be misplaced or subject to revisions.

Link to enabler	Link objective	Link data description
Security orchestrator (SO)	SO instructs Systemic to apply security properties (on a designated software to deploy)	○ Protection project parameters ○ ID of the software ○ keys
Security enforcement and control (SEC)	SEC instructs Systemic to apply security properties Systemic reports the applied security properties (on a designated software to deploy)	○ Protection project parameters ○ ID of the software ○ Keys ○ Metadata-fingerprint
Slice trustworthiness	Systemic reports the applied security properties (on a designated software to deploy	○ Metadata partial content (security properties parameters)
Order proof of transit (PoT)	PoT instructs Systemic (to bind node-deployed OPoT primitives to designated platform)	○ Key ○ PoT node primitive
Remote attestation (RA)	RA is used to construct quotes on Systemic metadata (for liability management)	○ Metadata partial content (security properties parameters)
Component certification (CC)	CC delivers Digital Trustworthiness certificates, which include Systemic's security properties (among other metrics)	○ Metadata partial content (security properties parameters)

Trust and Reputation Manager (TRM)	TRM elaborates and reputation labels based on the Systemic security properties	○ Metadata partial content (security properties parameters)
NFVO	NFVO transmits unprotected software and collects protected ones to and from Systemic	○ Original software ○ Protected software ○ Protection project parameters, ○ Keys (to and from Systemic)

5.1.1.2 Use case and Test Case interactions

At the time of writing this document, Systemic is integrated in:

- Use case 13 Network attacks over encrypted traffic in SBA
- Test case 3-4 (the two test cases are merged into a single test case).

In fact, both use and test case solicit the same relationship between Systemic and Montimage MMT Virtual Security Function. The relationship is expressed by: Systemic protects MMT. Therefore, MMT is the beneficiary of Systemic security but cannot be viewed as a structural interface, tied between two architectural components as depicted in Section 5.1.1.1. Indeed, none of these architectural interfaces is either depicted in the current use cases or test cases. The main objective of our work in the first half of the project was to offer an automated solution leveraging Intel 's SGX to secure network code. The deep integration of Systemic inside the HLA is the next step to take in the course of the Project.

In test case 3-4, the protection of MMT by Systemic is depicted here:

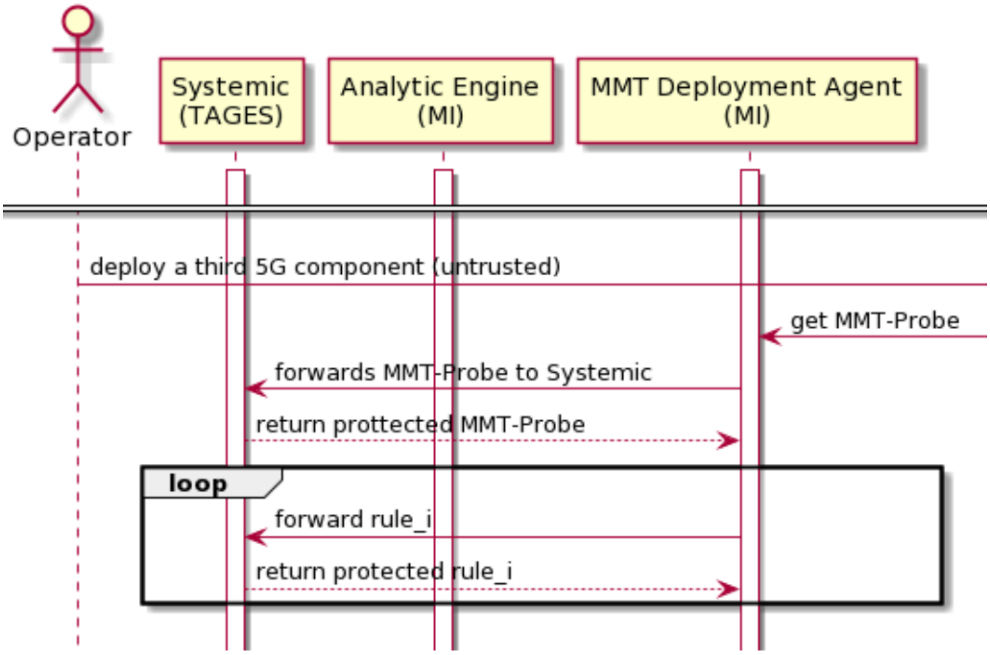

Operator
Systemic (TAGES)
Analytic Engine (MI)
MMT Deployment Agent (MI)
deploy a third 5G component (untrusted)
get MMT-Probe
forwards MMT-Probe to Systemic
return prottected MMT-Probe
loop
forward rule_i
return protected rule_i

Figure 19: Workflow of Systemic protection

Figure 19 depicts the simple input-output conversion by Systemic, consuming non-protected software and delivering protected version. However, this workflow does not show the deep interactions with HLA components. A deeper integration of Systemic in the HLA with structural enablers and components (service management, security, trust and liability management) will derive into several chains of events and associated workflow charts.

5.1.2 Interfaces specification

In Table 4 we show an initial draft version for Systemic VNF Wrapper controller interfaces.

Method	URL	Required Data Objects	Returned Data Object
POST	/systemic/analyse/	input binary file	Binary Analysis JSON
POST	/systemic/protect/	JSON protection project input binary file	Zip Compressed Archive
PUT	/systemic/patch-metadata/	JSON metadata descriptor Protected binary file	Zip Compressed Archive
POST	/systemic/get -metadata/	JSON metadata descriptor Protected binary file	Decrypted Metadata Content

Table 4: Systemic VNF Wrapper API

The version returns a plain JSON object which contains the following fields:

Name	Description	Sample value
success	Request success (boolean)	true | false
version	Protection server version (string)	21.07.01
build_time	Protection server build date time (string)	2021-07-21 11:40:05

Table 5: Systemic JSON file content

For standard Systemic protections, it is sufficient to perform a POST request, with “multipart/form-data” content-type, to the protect URL, containing the input-binary file and an optional protection settings JSON file.

The stub of a JSON protection project contains the following fields:

Name	Description	Sample value
systemic	Enables Systemic (boolean)	true | false
sgx	Leverage Intel SGX enclave (boolean)	true | false
encryptionsymmetrickey	The AES key used to encrypt the binary (string) AES-128 for Systemic SGX AES-256 for Systemic	8596d827dba716[...]
signingprivate	The RSA key used to encrypt the binary (string)	-----BEGIN RSA PRIVATE KEY[...]

	RSA-3092 for Systemic SGX RSA-4096 for Systemic	
metadatakey	The AES key used to encrypt metadata (string) AES-128 for Systemic SGX AES-256 for Systemic	b0a2e848fca09cf8b[...]
metadata	The metadata content (string)	Test 123

Table 6: Protection project stub

The request returns a Zip Compressed Archive that contains the output Protected Binary File together with a JSON formatted report of the protection.

The report JSON file includes the following fields:

Name	Description	Sample value
success	Protection success (boolean)	true | false
project	The supplied protection project (JSON object). If no Systemic keys where supplied, the ones generated at random by the protection servers will be shown here.	Refer to Table 4 Protection project stub
version	The protection server version (string)	21.07.01

Table 7: Systemic JSON file content

When launching the Systemic VNF Wrapper container, an environment variable can be set to enable the container to serve a complementary Web application that helps the user to define the input protection JSON.

The Binary Analysis JSON file, returned from the analyse request contains detailed information about the input binary and is mostly useful to define custom protection settings.

5.2 Proof of Transit

5.2.1 Enabler placement and interactions

The Policy & SSLA management can enforce the execution of the PoT validations on the network (e.g., regulations, SLA, etc.) towards the trust management component of PoT controller. Also, the PoT controller can obtain a vision of the nodes involved from the Security orchestrator (alternatively, from the service Management Domain). The enforcement policy will mandate monitoring the packets. The PoT controller will enforce the specific configuration, using the IETF draft specification [1] against the security agents in the nodes assigned. Once the policy is active, the alerts will be collected by the PoT controller deliver to each Trust Management Domain so that it can evaluate this information in terms of trust or reputation and integrate with the E2E Trust management.

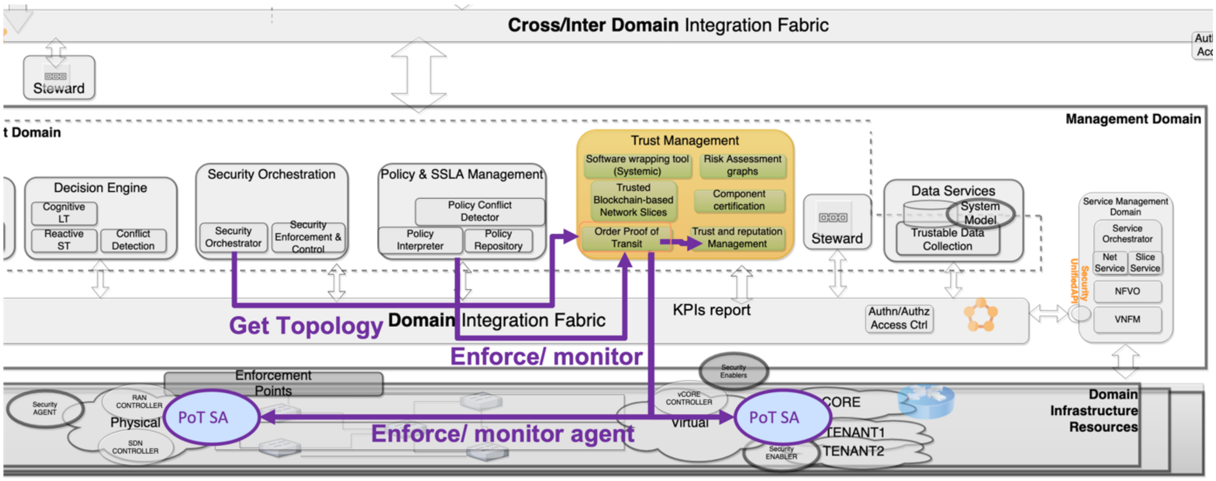


Figure 20: PoT placement in INSPIRE-5Gplus HLA

The PoT component integration in INSPIRE-5Gplus (Figure 20) is defined within the Use Case 13 and it will be integrated in the Test Case 4: E2E Encryption TEE secured SECaaS.

The role of the enabler is to generate and deliver metrics about the path of the traffic in the slide. The Security orchestrator will require the enforcement of the PoT verification to increase the trust level in the connectivity, to guarantee that the traffic is crossing over an IPsec tunnel.

5.2.2 Interfaces specification

The external PoT controller API interfaces is shown in Table 8.

Method	URL	Required Data Objects	Returned Data Object
GET	/pot/controller/path/{uuid}	UUID	Path JSON
POST	/pot/controller/path	---	Path JSON
DELETE	/pot/controller/path/{uuid}	UUID	---

Table 8: Proof of Transit API

They are designed to create, monitor and destroy the PoT. Returned objects contains detailed information of the created path and their status. One example is shown in Figure 21.

```
{
 "path_info": {
  "opot_id ": "1266841a-0650-4496-a5ad-e84a5ae762f3",
  "nodes": [
   {
    "status": "Operative",
    "node_id": 0,
    "address": {
     "ip": "192.168.0.1",
     "port": 55432
    },
    "node_type": "Ingress"
   },
   {
    "status": "Operative",
    "node_id": 0,
    "address": {
     "ip": "192.168.0.1",
     "port": 55432
    },
    "node_type": "Ingress"
   }
  ],
  "masks": [
   "2xaH0dBnJBRGQDXl8bhRXLqm81cVV7ddNJDrp77uvbs="
  ],
  "protocol": "UDP",
  "creation_time": 1615305214342100
 }
}
```

Figure 21: Example of returned object

Additional information is generated in JSON (Figure 22) based on Kafka protocol to publish the metrics into the integration fabric.

```
{
 "pot_id": "",
 "packet_number": "number of packet",
 "nodes": [
  "node_id_1",
  "node_id_2",
  "node_id_3"
 ],
 "timestamps": [
  1615305214342100,
  1615305214362110,
  1615305214402120
 ],
 "valid": "[true|false]"
}
```

Figure 22: PoT metrics JSON format

Internal interfaces, between PoT controller and PoT agents will be based in IETF standard YANG model defined in [1] over NETCONF interface.

As additional information, the concrete YAML file swagger for the REST API interface is provided in Appendix C.2.

5.3 eTRM : Trust and Reputation Manager

5.3.1 Enabler placement and interactions

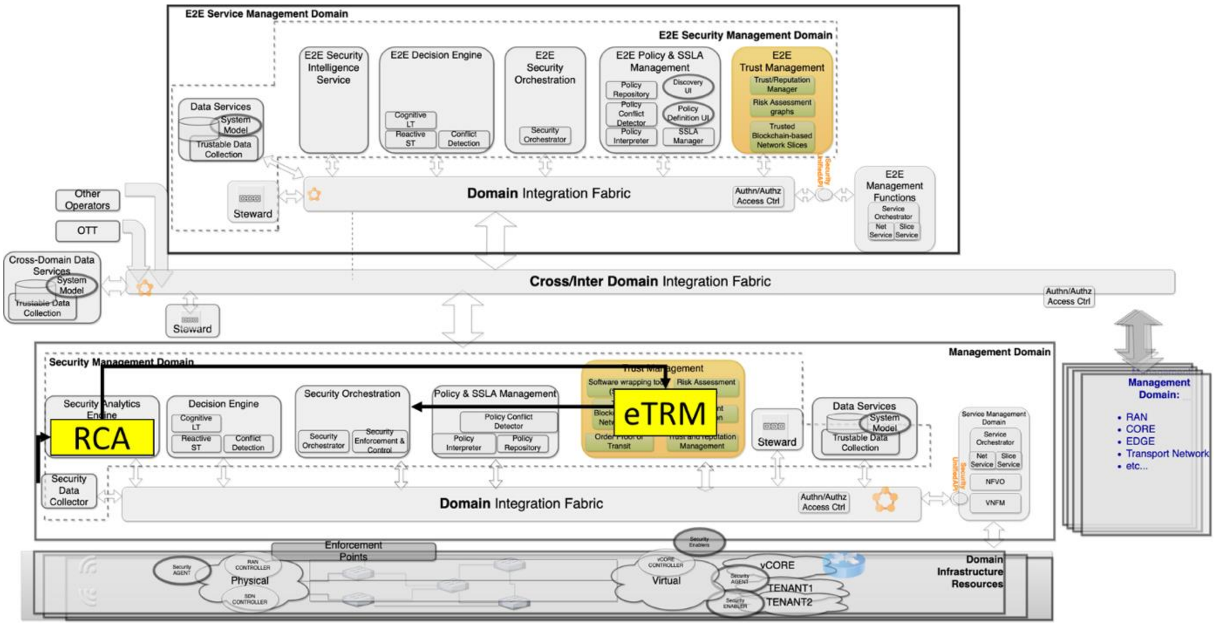


Figure 23: eTRM placement in INSPIRE-5Gplus HLA

The eTRM is located inside the Trust Management module and inside the Security Management Domain (Figure 23). The eTRM interacts with the rest of the elements and modules inside the INSPIRE-5Gplus architecture as follows:

- TRM gets the probabilities provided in real-time by the RCA after a fault failure has been diagnosed and updates the reputation model,
- TRM provides the reputation metrics as additional indicator to the security orchestrator to warn about risky networked elements, including the SDN controller on the domain,
- The RCA gets the network topology from the SDN controller which is validated. This network topology in the shape of graph will be also received by the eTRM to map reputation values on it.

5.3.2 Interfaces specification

The eTRM is in charge of assessing the reputation values of a given SDN domain, including its own SDN controller and interconnected nodes (switches and hosts).

The key parameters related to our enabler are the following:

- timeslot: contains the time of the day (int64) given as input to build the graph
- nodeId: the ID of the node. It can be an SDN controller, a host or a switch.
- sourceNodeId and targetNodeId: Identifiers of links, those IDs are referring to existing nodeIDs

Three types of links are considered in eTRM:

- switch-switch link
- host-switch link
- controller-switch link (control link)

eTRM can provide several JSON objects (see Figure 24 and Figure 25) as response to several GET requests, all schematized in the following Table 9.

Method	URL	Required Data Objects	Returned Data Object
POST	/trust/trm/reputation/{timeSlot}/pgm	timeSlot (int64)	--
GET	/trust/trm/reputation/{timeSlot}/domain reputation	timeSlot (int64)	Reputation graph (JSON)
GET	/trust/trm/reputation/{timeSlot}/node/{nodeId}	timeSlot (int64) nodeId (int64)	Reputation value for that given nodeId (JSON)
GET	/trust/trm/reputation/{timeSlot}/link/{sourceNodeId}/{targetNodeId}	timeSlot (int64) sourceNodeId (int64) targetNodeId (int64)	Reputation value for that connection/link(JSON)

Table 9: eTRM: Trust and Reputation Manager API

```
nodes: [{
            "type": "controller",
            "reputation": -1,
            "id": 0
          },
          {
             "type": "switch",
             "reputation": -1,
            "id": 1
          },
          {
             "type": "host",
             "reputation": -1,
            "id": 2
          },
          {
             "type": "host",
             "reputation": -1,
            "id": 3
          },
          ]
```

Figure 24: JSON with the returned reputation graph object containing nodes (simplified)

```
        links: [ {
            "type": "control link",
            "reputation": -1,
            "source": 0,
            "target": 1
          },
          {
            "type": "host-switch link",
            "reputation": -1,
            "source": 2,
            "target": 3
          },
          {
            "type": "host-switch link",
            "reputation": -1,
            "source": 2,
            "target": 1
          }
          ]
```

Figure 25: JSON with the returned reputation graph object containing links (simplified)

As additional information, the concrete YAML file swagger is provided in Appendix C.1.

5.4 Component certification tool

CCT offers for each component a way to be certified at design time. Based on the description of the component or based on its source code, an automatic evaluation is performed for building its trustworthiness metrics. At run time, the DTwC generated at design time could be used for selecting the most suitable component for a slice (for example, depending on the requested SSLA) and for building a whole trustworthiness view of the slice itself. This second aspect is used in Figure 26.

In INSPIRE-5Gplus, two different integrations are planned with other enablers:

- Integration with “Trusted Blockchain-based Network Slices” enabler: This integration will be in the context of the Test Case 1 (Anticipated Cooperative Collision Avoidance) by giving the certification information for a slice-subnet to the enabler. In this case, the public API of CCT will be used (see below).
- Integration with “MANIFEST” and “Systemic VNF” wrapper will use private and dedicated interfaces for importing and transforming some data. These interfaces are not described here.

5.4.1 Enabler placement and interactions

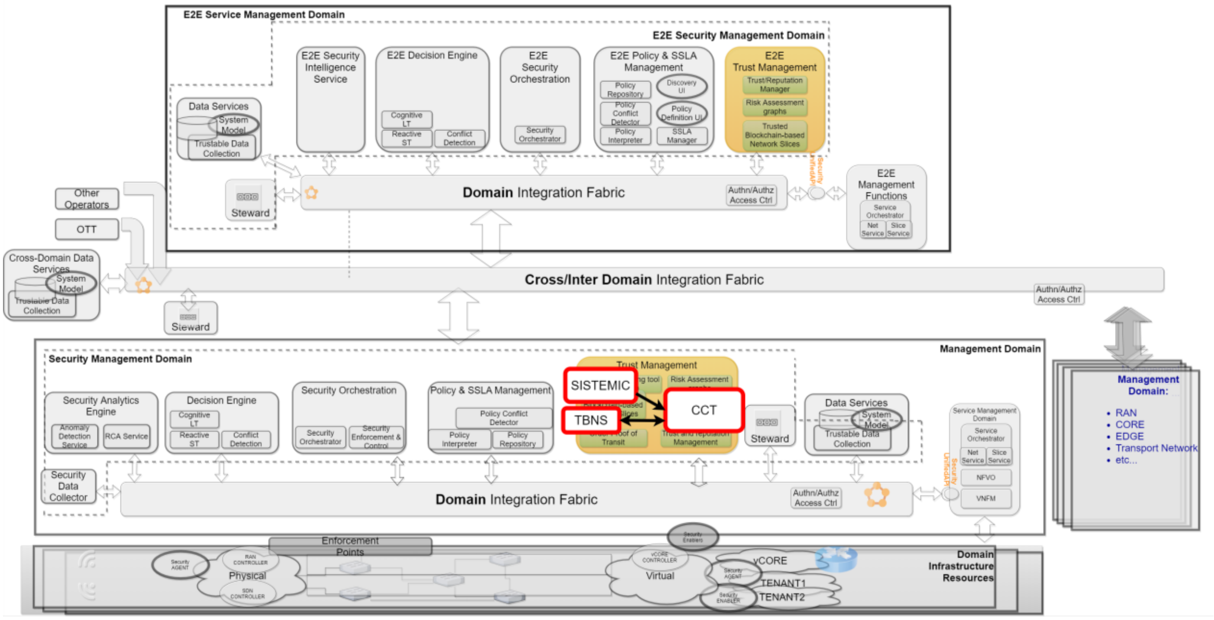


Figure 26: CCT placement in INSPIRE-5Gplus HLA

5.4.2 Interfaces specification

At design time, CCT is used as a standalone tool for evaluating any type of components. It is used as a tool and no API is provided. At run time, the interface of CCT is used for giving the list of DTwC with the different trust metrics for the different components. It is possible to retrieve only one DTwC based on its UUID or it is possible to retrieve all DTwC. It could be used at E2E service management domain level.

Method	URL	Required Data Objects	Returned Data Object
GET	/rest/files/type/{componentID}	UUID	text (DTwC)
GET	/rest/files/searchHashs/{type}	type (string)	JSON object
GET	/rest/files/downloadDTwCByHash/{hash}	hash (string)	text (DTwC)

POST	/CertificationRepository/rest/files/uploadDTwCs?VNFType=type"	file (DTwC)	--

Table 10: Component certification tool API

By this way, CCT will give different trustworthiness properties for the components involved in the slice.

5.5 Trust Reputation Manager

5.5.1 Enabler placement and interactions

The TRM enabler will interact, through the Integration Fabric mostly via pub/sub services, with the entities from which it has to obtain data (to calculate the trust value), such as SSLAs and Security Data Collector as well as with E2E Trust Manager and with the Blockchain (Steward). The latter is required to calculate the requested trust value, as it is obtained from the execution of a Smart Contract. The enabler's obtained values, as well as the computed trust scores will be stored in a database for further historical post-processing.

To get the needed parameters to compute the trust, the process defined below comprises all the involved parties. As an example, we will describe the interaction (through the Integration Fabric) with the PoT (Proof of Transit) enabler, but the same procedure must be carried out by any other enabler when providing their specific information to the TRM.

Interaction between TRM and PoT (valid for any enabler), this process is depicted also in Figure 27:

1. PoT Controller will periodically publish parameters to the Integration Fabric (Kafka).
2. TRM will retrieve the enabler parameters from a subscription to Kafka.
3. TRM will compute a given entity's Trust Scores through Smart Contracts by combining the parameters retrieved from subscriptions to the Integration Fabric
4. The Smart Contracts referenced data will be stored in Data Services (as Trustable Data Collection becoming part of the defined Trustable Data Services defined in INSPIRE-5Gplus High-Level Architecture).
5. Enablers will eventually request a given entity's Trust Score, through the API described in Table 11.

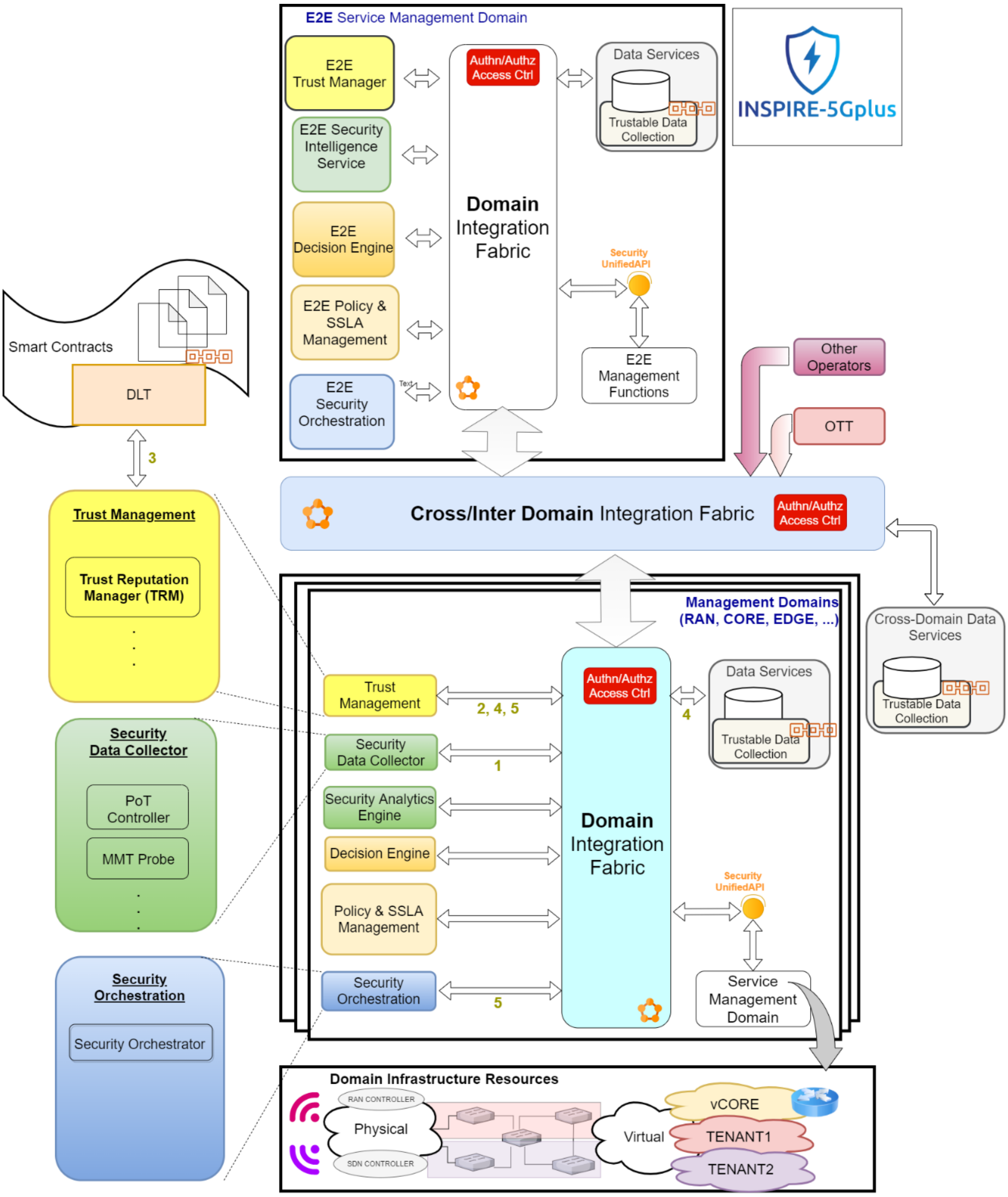


Figure 27: TRM placement in INSPIRE-5Gplus HLA

<u>End to end workflow for the TCs in which the TRM is integrated</u>

TRM is present in Test Case 3-4 and in Test Case 6.

In TC3-4 the main objective is to deploy security mechanisms like IPsec tunnelling and to detect/mitigate attacks such as VNF creation and manipulation. In this context, the TRM will maintain the trustability of each component updated in the virtualized infrastructure, including the trust calculation of the behaviour of IPsec tunnel, the Malicious deployed VNF, and the Monitoring System (MMT Probe/ STA Agents).

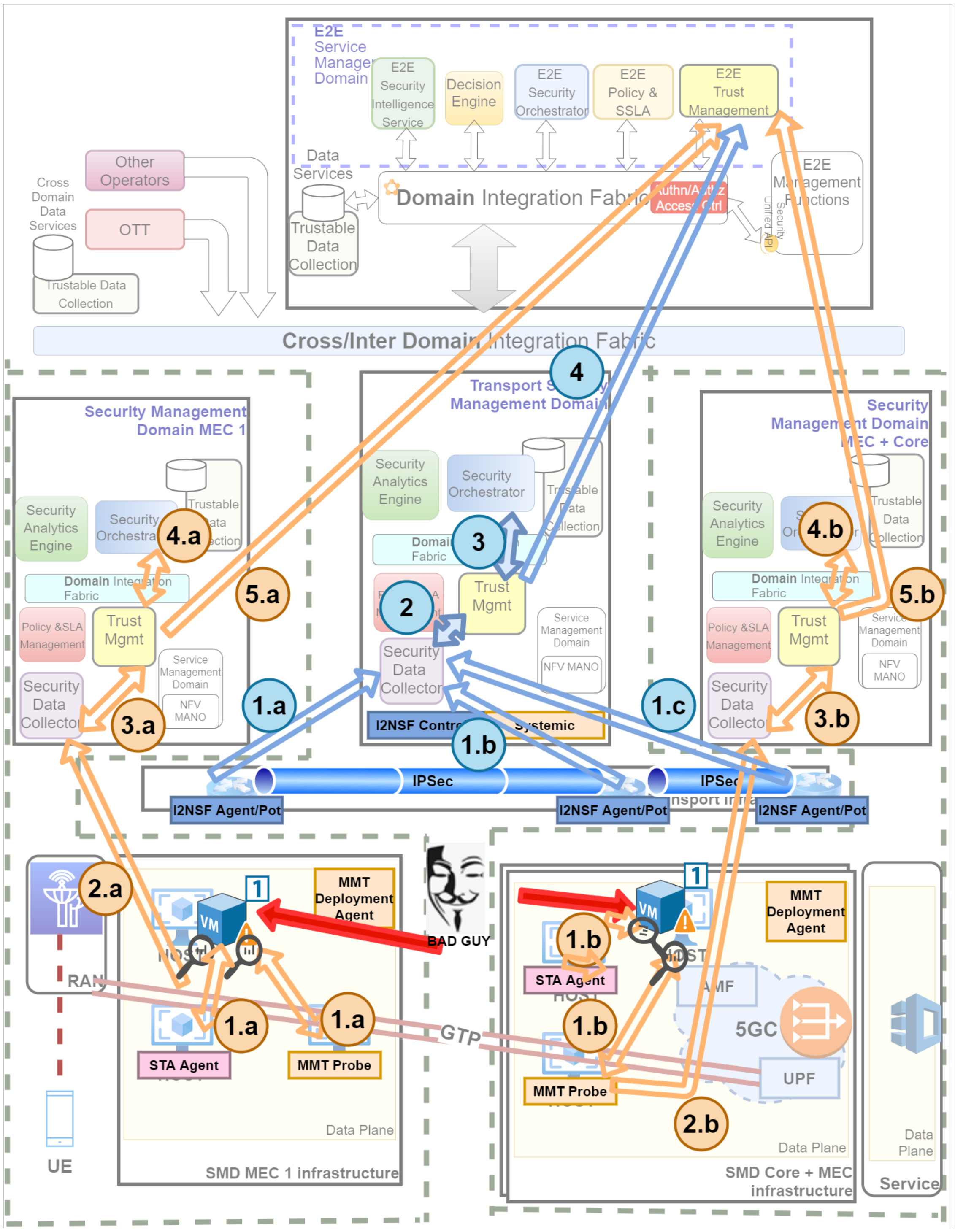


Figure 28: TC3-4 Workflow

In Figure 28, in blue, we can observe once the IPsec tunnelling has been deployed how the TRM updates the trust calculation based on Security Data Collector information. Also in brown, we represent how the TRM interacts in the monitoring and attack mitigation procedure. In both flows, TRM send its calculations to the E2E TRM so the Decision Engine will be capable of selecting best fitted solution for each involved domain.

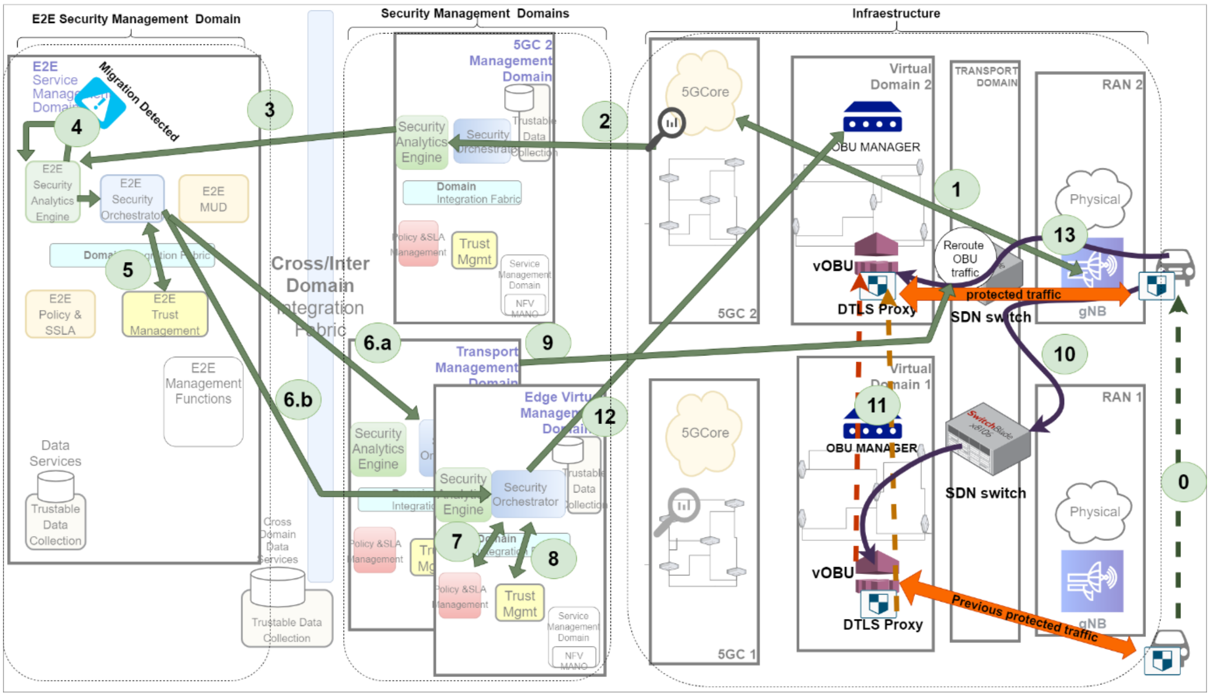


Figure 29: TC6 Workflow

In Figure 29, TC6 workflow is shown, where the TRM is in charge of calculating the reputation of the Virtual Domain where the vOBU is going to be deployed (5) and updating its value once the migration is performed whether it was successful or failed, and the vOBU no longer pertains to that Domain. Additionally, the TRM of the selected Edge Virtual SMD calculates the reputation score of the specific vOBU that is going to be deployed (8), updating its score once the vOBU has been released or in each successful/fail migration process.

5.5.2 Interfaces specification

As mentioned before, the TRM service, inside the Trust Management block, is in charge of assigning trust and reputation values to the corresponding monitored 5G entities, for instance, VNF like Access and Mobility Management Functions (AMFs), User Plane Functions (UPFs) or Authentication Server Functions (AUSFs), nodes, infrastructure, etc., using historical data and monitoring data. To provide the resulting scores to security management entities and end users in 5G virtualized networks, **enablers should periodically publish their current status/information in Kafka**. Then, TRM will collect data from Kafka and compute the Trust Score of the given entity.

For enablers to publish relevant information for the computation of the Trust Score, a set of potential parameters has been defined, a priori, that complies with the ETSI recommendations [42]:

- Geographical location
- Jurisdiction/regulatory location (public/private)
- Logical (network) location
- Hardware capabilities
- Software capabilities
- Execution instance history
- Chain of trust
- Time elapsed since last trust audit/check
- Date
- Time of day

- Security of network
- Appropriate use of encryption techniques
- Extent to which the software was initially hardened
- Measures in place to maintain the integrity of the software
- Physical security of the various locations over which the NFV components are deployed

From this list of initial parameters, enabler owners must look into which ones could be retrieved from their enabler to later compose a JSON file which will contain them and be sent through Kafka to the TRM. After agreeing with the different enablers' owners the more suitable set of parameters that will be provided, a general JSON template will be distributed to them so we can count on a standard file that will be filled and sent through Kafka. In this way, simplifying the data post-processing in the TRM.

A first tentative example that we have already defined with the Proof of Transit enabler (PoT) is depicted next.

Logs generated by the PoT Controller contain data collected from each packet at each PoT node in the following format:

- path_id: deployed PoT ID.
- node_id: ID of the corresponding node sending data.
- packet_number: identifies the packet traversing the PoT.
- valid: true/false. Reports whether the validation of the PoT was correct.
- timestamp: instant of packet reception at the node.

Even though the data is generated in CSV as original Format for this specific case, parameters will have to be converted to JSON file format to follow the established standard and to publish it in Kafka, for instance in the following manner (*Figure 30*):

```
{
  "pot_id": "pot_id",
  "packet_number" : "packet_number",
  "nodes": [
    "node_id1",
    "node_id2",
    "node_id3"
  ],
  "timestamps": [
    "timestamp1",
    "timestamp2",
    "timestamp3"
  ],
  "valid": "true | false"
}
```

Figure 30: JSON file with updated information published by an enabler (PoT in this example)

For interested entities to request the TRM to **provide** the value of trust **of a given entity**, it will expose the following API (as initial draft).

Method	URL	Required Data Objects	Returned Data Object
GET	/trust/vnf/{vnfID}	UUID	Trust Value

POST	/trust/vnf/{vnfID}	UUID, Trust Value	–
GET	/trust/infrastructure/{infrastructureID}	UUID	Trust Value
POST	/trust/infrastructure/{infrastructureID}	UUID, Trust Value	–
GET	/trust/domain/{domainID}	UUID	Trust Value
POST	/trust/domain/{domainID}	UUID, Trust Value	–

Table 11: Trust Reputation Manager API

5.6 Trusted Blockchain-based Network Slices

5.6.1 Enabler placement and interactions

The TBNS enabler aims to enable the deployment of End-to-End Network Slices in multi-domain and multi-operator scenarios using Blockchain as the key element to provide trust among domain users/operators in the process. The TBNS will interact with the Network Slice Manager and SDN Controllers placed in each domain and will only share the information labeled as "certified" by the Component Certification Tool (see Figure 31).

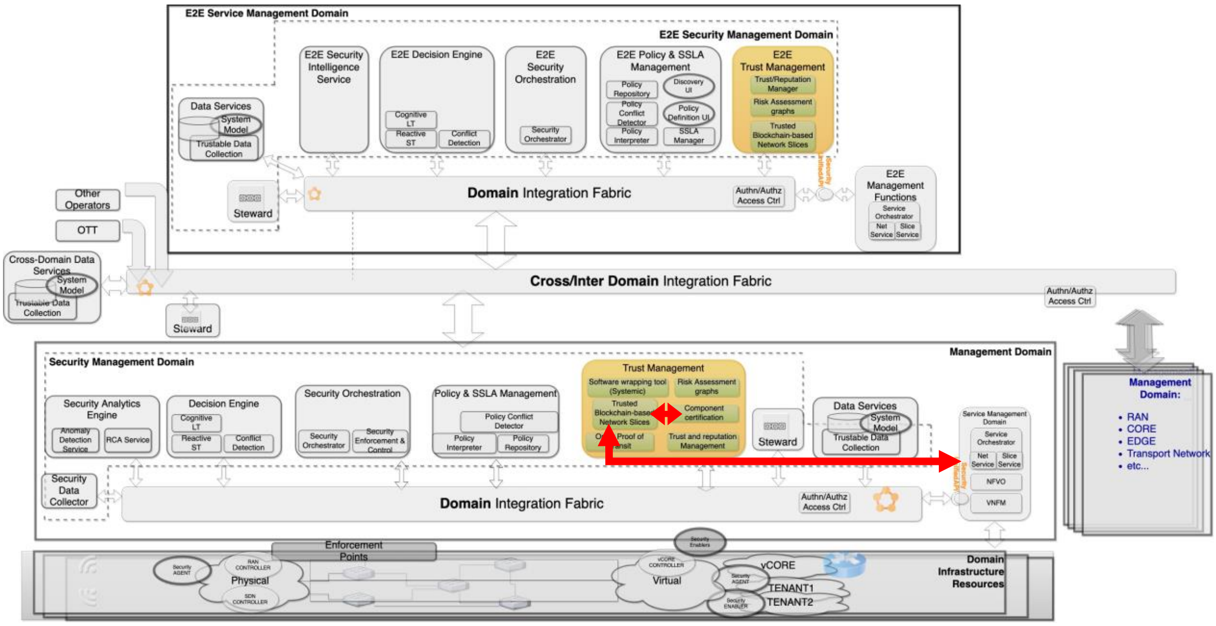


Figure 31: TBNS placement in INSPIRE-5Gplus HLA

This enabler is planned to be used in the TC1 scenario 2 as this TC aims to present two different scenarios based on the Illustrative Use Case 1 (IUC1) called Secured and Sliced ACCA (Anticipated Cooperative Collision Avoidance) described in the INSPIRE-5Gplus deliverable D2.2. Based on this IUC1, a Test Case (TC) has been designed to demonstrate the functionality of this enabler together with the integration of this enabler with the Component Certification Tool. The TC related to this enabler is the TC5 as presented in D5.1. As presented in Figure 32, in this TC, the idea is to have a set of multiple operator domains (both NFV and SDN domains) and deploy in a collaborative way (through the use of a Blockchain network) the End-to-End (E2E) Network Slices using resources placed in the different domains without the need of a central manager/orchestrator on top of the infrastructure.

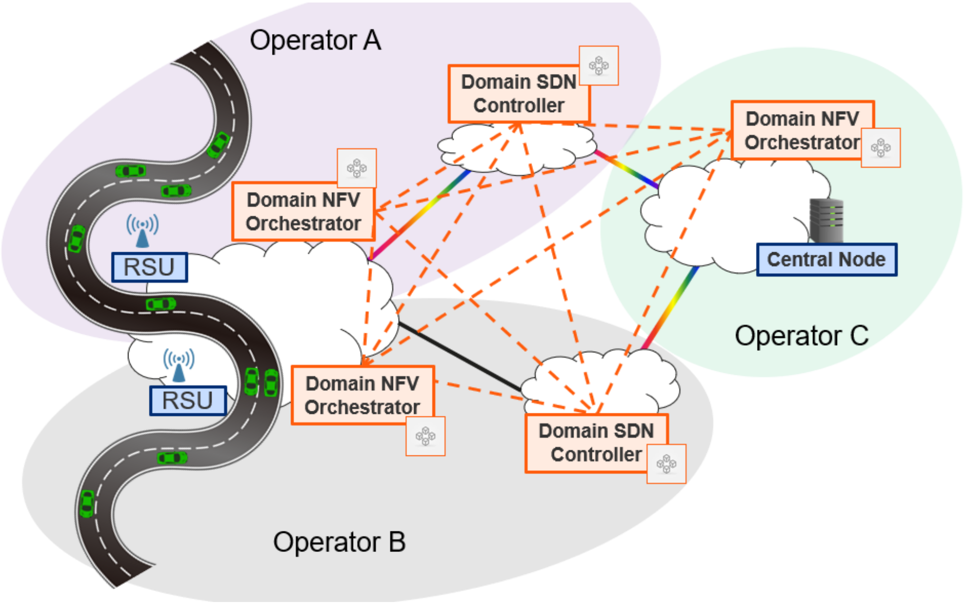


Figure 32 - TC5 architecture.

In order to understand better this TC, the interaction between the TBNS and the CCT and the E2E Network Slice deployment work-flows are presented in the following figures.

Figure 33 presents the interaction of the TBNS and the CCT enablers to distribute the information only those Network Slice Templates (NSTs) that were previously certified by the CCT enabler. Once distributed their information reference, any other NFV domain may request the deployment of those NSTs in the domain where they belong. This deployment procedure is then presented in Figure 34 and Figure 35, in which an E2E Network Slice is requested and deployed.

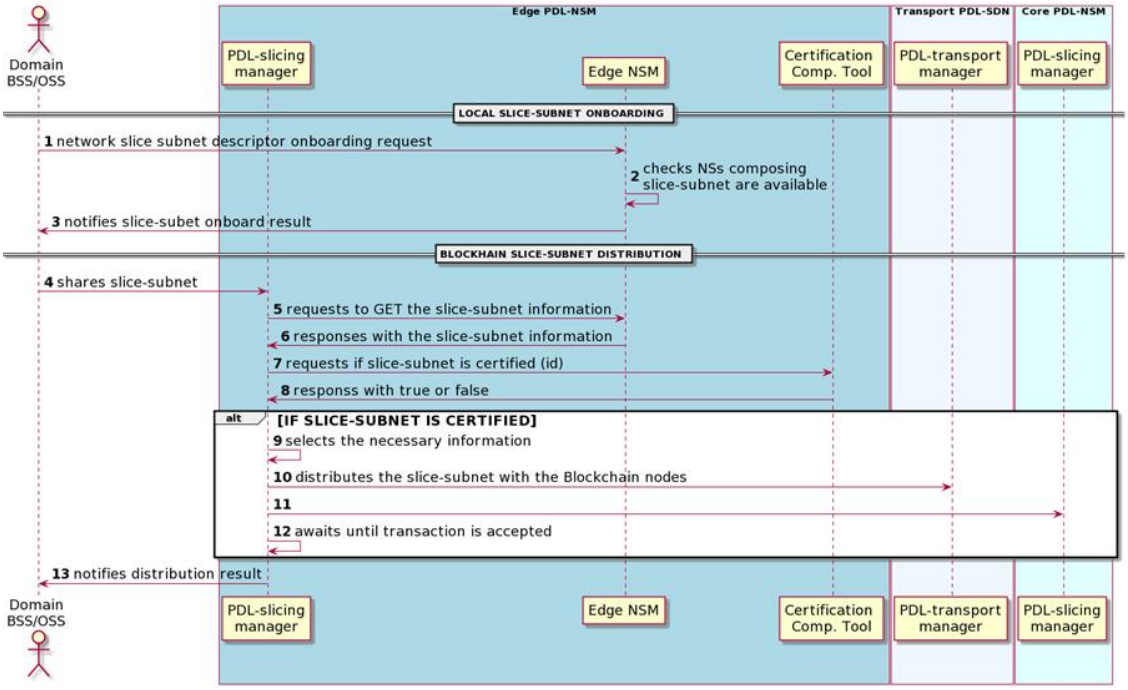

Edge PDL-NSM
Transport PDL-SDN
Core PDL-NSM
Domain BSS/OSS
PDL-slicing manager
Edge NSM
Certification Comp. Tool
PDL-transport manager
PDL-slicing manager
LOCAL SLICE-SUBNET ONBOARDING
1 network slice subnet descriptor onboarding request
2 checks NSs composing slice-subnet are available
3 notifies slice-subet onboard result
BLOCKCHAIN SLICE-SUBNET DISTRIBUTION
4 shares slice-subnet
5 requests to GET the slice-subnet information
6 responses with the slice-subnet information
7 requests if slice-subnet is certified (id)
8 responss with true or false
alt
[IF SLICE-SUBNET IS CERTIFIED]
9 selects the necessary information
10 distributes the slice-subnet with the Blockchain nodes
11
12 awaits until transaction is accepted
13 notifies distribution result
Domain BSS/OSS
PDL-slicing manager
Edge NSM
Certification Comp. Tool
PDL-transport manager
PDL-slicing manager

Figure 33 - Distribution of certified NSTs.

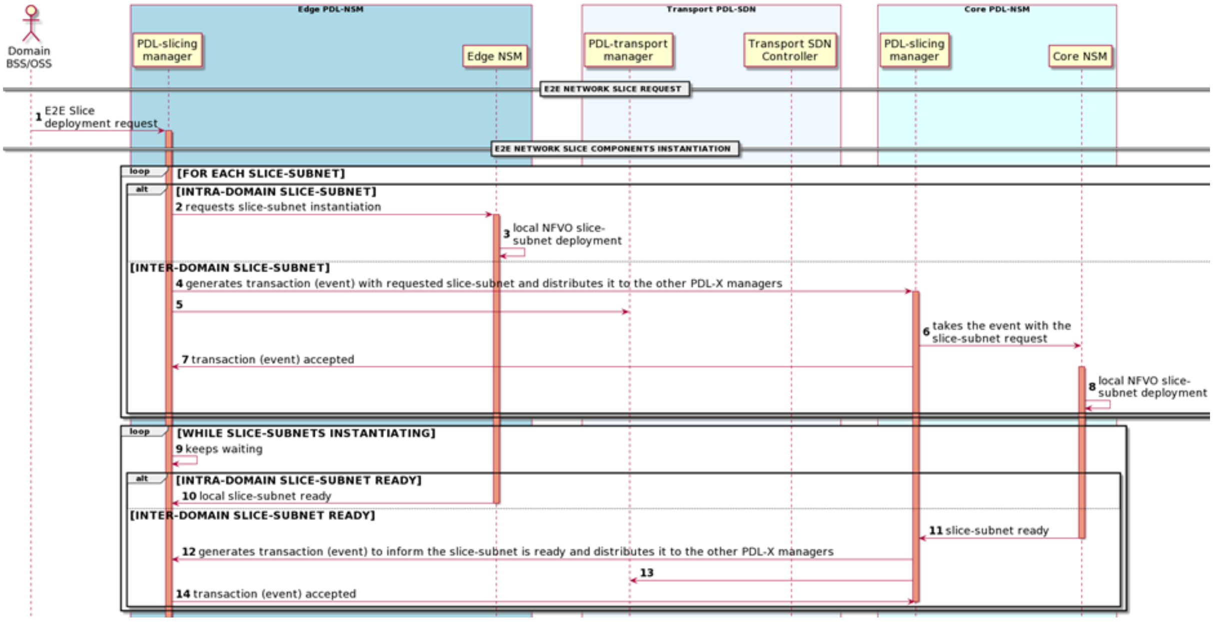

Domain BSS/OSS
Edge PDL-NSM
PDL-slicing manager
Edge NSM
Transport PDL-SDN
PDL-transport manager
Transport SDN Controller
Core PDL-NSM
PDL-slicing manager
Core NSM
E2E NETWORK SLICE REQUEST
1 E2E Slice deployment request
E2E NETWORK SLICE COMPONENTS INSTANTIATION
loop [FOR EACH SLICE-SUBNET]
alt [INTRA-DOMAIN SLICE-SUBNET]
2 requests slice-subnet instantiation
3 local NFVO slice-subnet deployment
[INTER-DOMAIN SLICE-SUBNET]
4 generates transaction (event) with requested slice-subnet and distributes it to the other PDL-X managers
5
6 takes the event with the slice-subnet request
7 transaction (event) accepted
8 local NFVO slice-subnet deployment
loop [WHILE SLICE-SUBNETS INSTANTIATING]
9 keeps waiting
alt [INTRA-DOMAIN SLICE-SUBNET READY]
10 local slice-subnet ready
[INTER-DOMAIN SLICE-SUBNET READY]
11 slice-subnet ready
12 generates transaction (event) to inform the slice-subnet is ready and distributes it to the other PDL-X managers
13
14 transaction (event) accepted

Figure 34 - E2E Network Slice deployment work-flow (part 1).

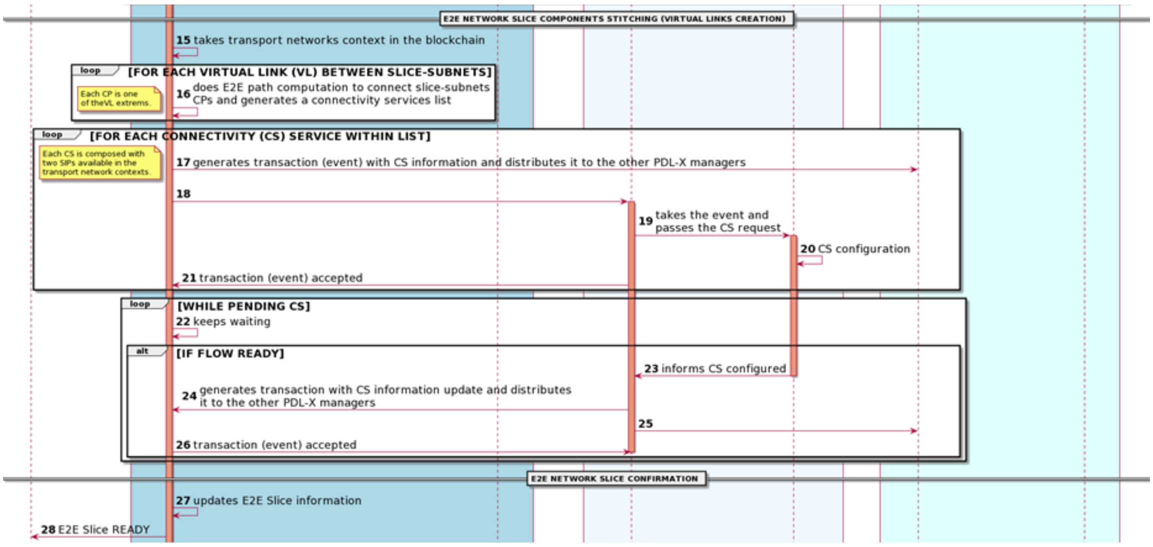

E2E NETWORK SLICE COMPONENTS STITCHING (VIRTUAL LINKS CREATION)
15 takes transport networks context in the blockchain
loop [FOR EACH VIRTUAL LINK (VL) BETWEEN SLICE-SUBNETS]
Each CP is one of theVL extrems.
16 does E2E path computation to connect slice-subnets CPs and generates a connectivity services list
loop [FOR EACH CONNECTIVITY (CS) SERVICE WITHIN LIST]
Each CS is composed with two SIPs available in the transport network contexts.
17 generates transaction (event) with CS information and distributes it to the other PDL-X managers
18
19 takes the event and passes the CS request
20 CS configuration
21 transaction (event) accepted
loop [WHILE PENDING CS]
22 keeps waiting
alt [IF FLOW READY]
23 informs CS configured
24 generates transaction with CS information update and distributes it to the other PDL-X managers
25
26 transaction (event) accepted
E2E NETWORK SLICE CONFIRMATION
27 updates E2E Slice information
28 E2E Slice READY

Figure 35 - E2E Network Slice deployment work-flow (part 2).

5.6.2 Interfaces specification

The current status of this enabler’s API is presented in Table 12 with functionalities such as different GET operations to retrieve the NSTs (slice-subnets) from the local domain, those distributed in the Blockchain or the sum of both “types”. We consider that there is not POST action to add the NSTs as they should be added using the local Network Slice Manager (or NFV Orchestrator). Regarding the management of E2E Network Slices, two POST actions are available: one to deploy and one to terminate once the service is no more necessary. Finally, in addition to the previous two POST actions, there is one last GET action to retrieve the information belonging to those E2E Network Slices instantiations (either they are active or terminated).

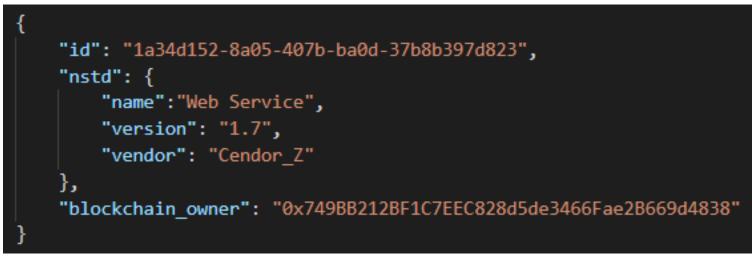
Method	URL	Data Objects	Response Codes & Data Object
GET	'/pdl-slice/slice-subnets'	—	200, slice-subnets list
POST	'/pdl-slice/slice-subnets/<subnet_ID>'	UUID	200, -

GET	'/pdl-slice/slice-subnets/local'	—	200, slice-subnets list
GET	'/pdl-slice/slice-subnets/local/<subnet_id>'	UUID	200, slice-subnet item
GET	‘/pdl-slice/slice-subnets/blockchain'	—	200, slice-subnet list
GET	'/pdl-slice/slice-subnets/blockchain/<subnet_ID>'	UUID	200, slice-subnet item
POST	'/pdl-slice/e2e-slice'	JSON	200, -
GET	'/pdl-slice/e2e-slice'	—	200, E2E Slice JSON list
GET	‘/pdl-slice/e2e-slice/<e2eslice_ID>’	—	200, E2E Slice JSON
POST	'/pdl-slice/e2e-slice/terminate/<e2eslice_ID>'	UUID	200, -

Table 12: Trusted Blockchain-based Network Slices API

To understand better with the type of data objects that this API deal with, the following lines present the structure of the different JSON objects used or retrieved in the previous API calls.

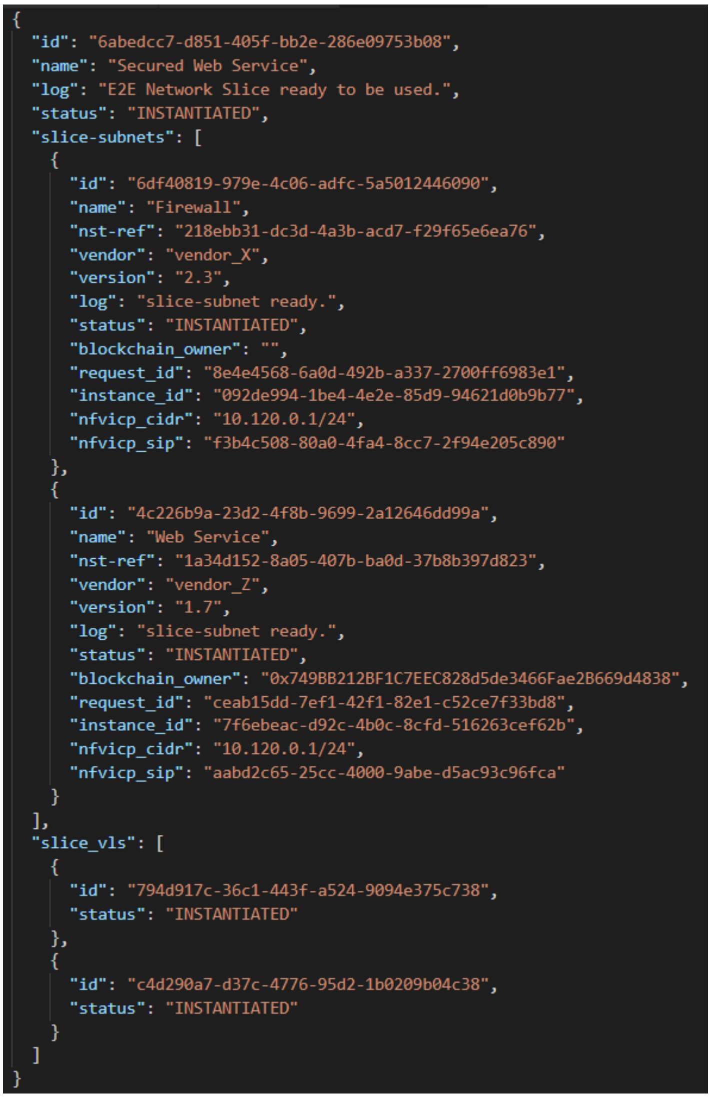
The first JSON data object is the slice-subnet. Figure 36 presents the JSON structure with the slice-subnets information. It contains the essential information as it presents the service it contains, its ID (defined by its owner before it is distributed in the Blockchain) and the Blockchain owner address to know to add it in the Blockchain transaction and so, only the owner takes care of its deployment.



```
{
    "id": "1a34d152-8a05-407b-ba0d-37b8b397d823",
    "nstd": {
        "name":"Web Service",
        "version": "1.7",
        "vendor": "Cendor_Z"
    },
    "blockchain_owner": "0x749BB212BF1C7EEC828d5de3466Fae2B669d4838"
}
```

Figure 36 - slice-subnet JSON.

The second JSON data object is the E2E Slice. As presented in Figure 37, it is structured in three main parts, the basic information of the E2E Slice (id, name, status and log), the list of slice-subnets composing it (with the blockchain owner address if they belong to another domain) and the list of virtual links to interconnect the slice-subnets.



```json
{
  "id": "6abedcc7-d851-405f-bb2e-286e09753b08",
  "name": "Secured Web Service",
  "log": "E2E Network Slice ready to be used.",
  "status": "INSTANTIATED",
  "slice-subnets": [
    {
      "id": "6df40819-979e-4c06-adfc-5a5012446090",
      "name": "Firewall",
      "nst-ref": "218ebb31-dc3d-4a3b-acd7-f29f65e6ea76",
      "vendor": "vendor_X",
      "version": "2.3",
      "log": "slice-subnet ready.",
      "status": "INSTANTIATED",
      "blockchain_owner": "",
      "request_id": "8e4e4568-6a0d-492b-a337-2700ff6983e1",
      "instance_id": "092de994-1be4-4e2e-85d9-94621d0b9b77",
      "nfvicp_cidr": "10.120.0.1/24",
      "nfvicp_sip": "f3b4c508-80a0-4fa4-8cc7-2f94e205c890"
    },
    {
      "id": "4c226b9a-23d2-4f8b-9699-2a12646dd99a",
      "name": "Web Service",
      "nst-ref": "1a34d152-8a05-407b-ba0d-37b8b397d823",
      "vendor": "vendor_Z",
      "version": "1.7",
      "log": "slice-subnet ready.",
      "status": "INSTANTIATED",
      "blockchain_owner": "0x749BB212BF1C7EEC828d5de3466Fae2B669d4838",
      "request_id": "ceab15dd-7ef1-42f1-82e1-c52ce7f33bd8",
      "instance_id": "7f6ebeac-d92c-4b0c-8cfd-516263cef62b",
      "nfvicp_cidr": "10.120.0.1/24",
      "nfvicp_sip": "aabd2c65-25cc-4000-9abe-d5ac93c96fca"
    }
  ],
  "slice_vls": [
    {
      "id": "794d917c-36c1-443f-a524-9094e375c738",
      "status": "INSTANTIATED"
    },
    {
      "id": "c4d290a7-d37c-4776-95d2-1b0209b04c38",
      "status": "INSTANTIATED"
    }
  ]
}
```

Figure 37 - E2E network Slice JSON

5.7 The Risk Assessment Graphs

5.7.1 Enabler placement and interactions

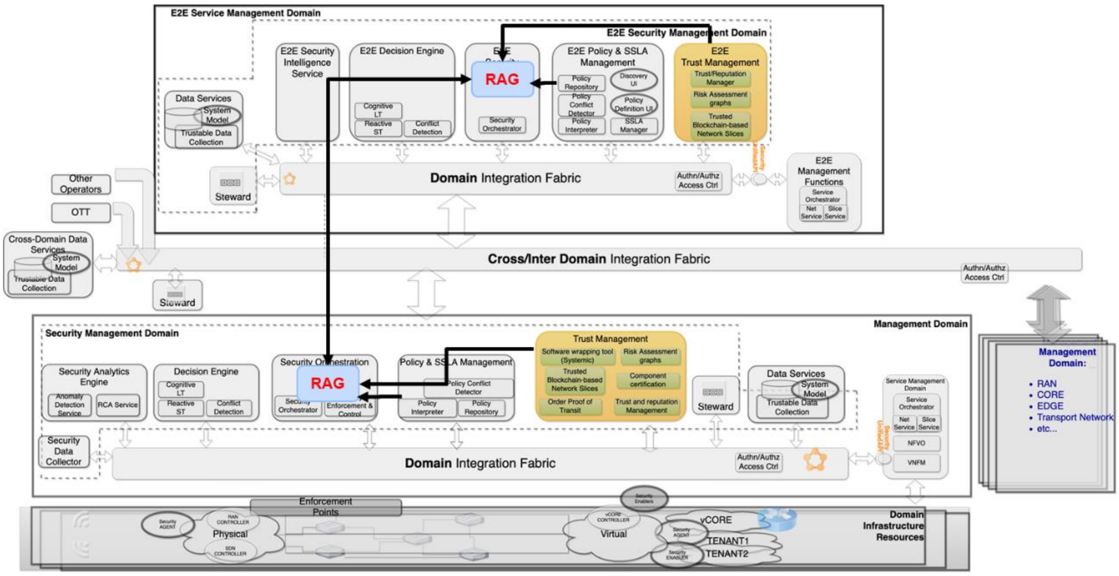


Figure 38: RAG placement in INSPIRE-5Gplus HLA

Arrows description depicted in Figure 38:

- RAG – RAG: hierarchical interaction between several vision of topologies
- RAG – Trust Management: collect of targeted security levels per sub-domain
- RAG – Security Orchestrator: optimized placement strategy (for Vertical's VNF and counter measures) with respect to Policy, SSLA and trust management constraints
- RAG – Policy and SSLA management: topology of connectivity between components and available countermeasures at this level of topology

5.7.2 Interfaces specification

Method	URL	Required Data Objects	Returned Data Object
GET	'/security/rag/create/{ragName}'	--	JSON
POST	'/security/rag/cms'	JSON	JSON
POST	'/security/rag/risk/{timeSlot}'	JSON	JSON
POST	'/security/rag/risk/{timeSlot}/node/{nodeId}'	JSON	JSON
POST	'/security/rag/risk/{timeSlot}/link/{sourceNodeId}/{targetNodeId}'	JSON	JSON
POST	'/security/rag/mitigateRisk/{timeSlot}'	JSON	JSON

Table 13: Risk Assessment Graphs API

See the Appendix C.3 for a more complete description of the RAG API.

6 Trust management approach

The Task 4.1 has defined a set of enablers provided by the different partners of INSPIRE-5Gplus. Each of these enablers is detailed and contributes to different trust mechanisms. Special attention has been paid to integrating these different enablers in order to provide a higher level of trust information.

In Task 4.2, the goal is to manage trust in multi-tenant/multi-party/multi-domain. For achieving it, the idea is to provide, at each domain level, a dashboard of trust. These dashboards could be combined after providing a whole trust dashboard at multi-domain level. The approach for building this dashboard is the following:

- Definition of TSLA (Trust Service Level Agreement) and the associated trust properties,
- Identification of trustworthiness properties of the multi-party infrastructure in a domain,
    - Evaluated at design time
    - Evaluated at runtime

The dashboard will reflect the TSLA and each item will be computed by using the different trustworthiness properties, each one evaluated with a metric. For defining and identifying the TSLA and the trustworthiness properties (and the associated metrics), a top-down/bottom-up approach will be followed. Top-down approach will start with what we want to display for TSLA based on existing knowledge of the partners and on existing stands. The bottom-up approach will start with the existing enablers defined in this deliverable delivering different trust properties.

The result of this approach will be a set of trust properties designed and delivered (according to the project time limitations) INSPIRE-5Gplus enablers with their metric. A part of them will exist and be delivered by existing enablers but another part would need to provide by future works. In the context of Task 4.2, a framework will be chosen for combining these trust properties and for computing the trust dashboard reflecting the TSLA. Specific attention will be paid to having a convergence between these TSLA metrics and the existing KPIs defined in WP5.

This effort’s main contribution will be to contribute to the close loop defined in D2.2 (Section 4.5) by providing some alerts regarding the TSLA metrics. These alerts will be managed by the Security Orchestration.

All these elements, approach, and definitions will be detailed in INSPIRE-5Gplus Deliverable D4.2.

References

[1] F. Brockners, et al. “Proof-of-Transit.” IETF internet draft, draft-ietf-sfc-proof-of-transit

[2] A. Aguado et al., "Quantum technologies in support for 5G services: Ordered proof-of-transit." 45th European Conference on Optical Communication (ECOC 2019), Dublin, Ireland, 2019, pp. 1-3.

[3] A. Shamir, “How to share a secret.” Communications of the ACM, 22(11), 612-613, 1979.

[4] N. Alliance, “5G white paper.” Next generation mobile networks, white paper, vol. 1, 2015.

[5] H. Yu, Z. Shen, C. Miao, C. Leung, and D. Niyato. “A survey of trust and reputation management systems in wireless communications.” Proceedings of the IEEE, 98(10):1755–1772, Oct 2010.

[6] B. K. Mughal, S. Hameed, and G. M. Shaikh, “A centralized reputation management Scheme for isolating malicious montroller(s) in distributed Software-Defined Networks.” ArXiv e-prints, November 2017.

[7] D. Marconett and S. J. Yoo. “Flowbroker: A software-defined network controller architecture for multi-domain brokering and reputation.” J. Netw. Syst. Manage., 23(2):328–359, April 2015.

[8] S. Betg-Brezetz, G. B. Kamga, and M. Tazi. “Trust support for SDN controllers and virtualized network applications.” In Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), pages 1–5, April 2015.

[9] B. Isong, T. Kgogo, F. Lugayizi, and B. Kankuzi. “Trust establishment framework between sdn controller and applications.” In 2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pages 101–107, June 2017.

[10]OPTET Consortium, “Socio-economic requirements for trust and trustworthiness.” Project Deliverable D2.1, 2013, available on http://www.optet.eu

[11] OPTET Consortium, “New trustworthiness certification process.” Project Deliverable D4.1.1, 2013, available on http://www.optet.eu

[12]OPTET Consortium, “Trustworthy factory specification.” Project Deliverable D4.2.3, 2015, available on http://www.optet.eu

[13]5G ENSURE Consortium, “Trust model.” Project Deliverable D2.2, 2016, available on https://www.5gensure.eu/

[14]5G ENSURE Consortium, “5G-PPP security enablers technical roadmap.” Project Deliverable D3.9 2017, available on https://www.5gensure.eu/

[15]E. Bellini, Y. Iraqi and E. Damiani, “Blockchain-Based Distributed Trust and Reputation Management Systems: A Survey.” in IEEE Access, vol. 8, pp. 21127-21151, 2020, doi: 10.1109/ACCESS.2020.2969820.

[16]Hyperledger Fabric, https://www.hyperledger.org/use/fabric

[17]Smart Contracts, https://hyperledger-fabric.readthedocs.io/en/latest/smartcontract/smartcontract.html

[18]S.S. Shetty, C.A.Kamhoua and L.L.Njilla, “Blockchain for distributed systems security.” Wiley-IEEE Computer Society, 2019.

[19]S. Nakamoto,“Bitcoin: A peer-to-peer electronic cash system.” Cryptography Mailing list at https://metzdowd.com, Mar. 2009.

[20]G. Wood, “Ethereum: A secure decentralised generalised transaction ledger.” Ethereum project yellow paper, vol. 151, no. 2014, pp. 1–32, 2014.

[21]The Linux Foundation. (2020). Hyperledger official website, [Online]. Available: https://www.hyperledger.org/, accessed: 08/26/2020

[22]S. Kou, H. Yang, H. Zheng, W. Bai, J. Zhang, and Y.Wu, “Blockchain mechanism based on enhancing consensus for trusted optical networks.” in 2017 Asia Communications and Photonics Conference (ACP), 2017,pp. 1–3.

[23]H. Yang, H. Zheng, J. Zhang, Y. Wu, Y. Lee, and Y. Ji, “Blockchain-based trusted authentication in cloud radio over fiber network for 5G,” in 2017 16th International Conference on Optical Communications and Networks (ICOCN), 2017, pp. 1–3.

[24]R. V. Rosa and C. E. Rothenberg, “Blockchain-based decentralized applications for multiple administrative domain networking,” IEEE Communications Standards Magazine, vol. 2, no. 3, pp. 29–37, 2018.

[25]S. Fichera, A. Sgambelluri, A. Giorgetti, F. Cugini, and F. Paolucci, “Blockchain-anchored failure responsibility management in disaggregated optical networks,” in 2020 Optical Fiber Communications Conference and Exhibition (OFC), 2020, pp. 1–3.

[26]Y. Liang, H. Yang, Q. Yao, S. Guo, A. Yu, and J. Zhang, “Blockchain-based efficient recovery for secure distributed control in software defined optical networks,” in 2019 Optical Fiber Communications Conference and Exhibition (OFC), 2019, pp. 1–3

[27]P. Alemany, R. Vilalta, R. Muñoz, R. Casellas and R. Maríinez, "Peer-to-Peer Blockchain-based NFV Service Platform for End-to-End Network Slice Orchestration Across Multiple NFVI Domains," 2020 IEEE 3rd 5G World Forum (5GWF), 2020, pp. 151-156, doi: 10.1109/5GWF49715.2020.9221311.

[28]P. Alemany, R. Vilalta, R. Muñoz, R. Martínez, R. Casellas, "Managing network slicing resources using blockchain in a multi-domain software defined optical network scenario," Optical Communications (ECOC) 2020 European Conference on, pp. 1-4, 2020.

[29]P. Alemany, R. Vilalta, R. Muñoz, R. Casellas and R. Martínez, "End-to-end network slice stitching using blockchain-based peer-to-peer network slice managers and transport SDN controllers," 2021 Optical Fiber Communications Conference and Exhibition (OFC), 2021, pp. 1-3.

[30]Steven Noel, Lingyu Wang, Anoop Singhal, and Sushil Jajodia, “Measuring security risk of networks using attack graphs,” IJNGC, 1(1), 2010.

[31]Steven J. Templeton and Karl Levitt, “A requires/provides model for computer attacks.” Proceedings of the 2000 workshop on New security paradigms, NSPW’00, New York, NY, USA, 2000. ACM.

[32]Steven Noel, Sushil Jajodia, Brian O’Berry, and Michael Jacobs. “Efficient minimum-cost network hardening via exploit dependency graphs.”, Proceedings of the 19th Annual Computer Security Applications Conference, ACSAC’03, Washington, DC, USA, 2003. IEEE Computer Society.

[33]Kheir, N.; Debar, H.; Cuppens-Boulahia, N.; Cuppens, F.; Viinikka, J., "Cost evaluation for intrusion response using dependency graphs." Network and Service Security, 2009. N2S '09. International Conference on, vol., no., pp.1,6, 24-26 June 2009

[34]Kheir, N., Mahjoub, A. R., Naghmouchi, M. Y., Perrot, N., Wary, J. P: “Assessing the risk of complex ICT systems.” Annals of Telecommunications, 1-15 (2017)

[35]A. Ridha Mahjoub, M. Naghmouchi, and N. Perrot, “A bilevel programming model for proactive countermeasure selection in complex ICT systems.” Electronic Notes in Discrete Mathematics, vol. 64, pp.295–304, 02 2018.

[36] 5G CITY, https://www.5gcity.eu

[37] DIVE SHIELD, https://www.shield-h2020.eu

[38] ENSURE, https://www.5gensure.eu

[39] Lefebvre et al. “Universal trusted / Universal trusted execution environments for securing SDN/NFV Operations.” ARES 2018.

[40] Youssef et al. “Secure Software-Defined Networks controller storage using Intel Software Guard Extensions,” International journal of Advanced Computer Science and Applications. Nov 2020.

[41] Wand et al. “S-Blocks: Lightweight and Trusted Virtual Security Function with SGX.” IEEE Transactions on Cloud Computing. April 2020**.**

[42] Duan et al. « LightBox: Full-stack Protected Stateful Middlebox at Lightning Speed.” CSS 2019. London

[43] Coughlin et al. “TRUSTED CLICK. Trusted Click: Overcoming Security issues of NFV in the Cloud.” SDN-NFV Sec’17, March 22-24 2017, Scottsdale, AZ, USA

[44] Shih et al. **“**S-NFV: Securing NFV states by using SGX.” ACM International Workshop 2016. S NFV

[45] ETSI Trust Recommendations: https://www.etsi.org/deliver/etsi_gs/nfv-sec/001_099/003/01.01.01_60/gs_nfv-sec003v010101p.pdf

[46] European Electronic Communications Code Updating EU Telecom Rules : https://digital-strategy.ec.europa.eu/en/library/european-electronic-communications-code-updating-eu-telecom-rules

[47] L. Condamin; J.-P. Louisot; P. Naïm “Risk quantification - management diagnosis and hedging.” John Wiley and Son, 2007.

[48] Olaitan Olaleye; Talmai Oliveira; Alexandru Darie, “Managing emerging risks and liabilities in data-enabled solutions.” 2017 IEEE 6th International Congress on Big Data, 2017.

[49] Mark Vinkovits; Sankt Augustin “Towards requirements for trust management; 2012 Tenth Annual International Conference on Privacy, Security and Trust.” IEEE Xplore, 2012.

[50]Scot Tucker “Engineering trust: a graph-based algorithm for modeling, validating, and evaluating.*” Trust*. IEEE International Conference On Big Data Science And Engineering, 2018.

[51]Olaitan Olaleye; Talmai Oliveira; Alexandru Darie, “Managing emerging risks and liabilities in data-enabled solutions.” 2017 IEEE 6th International Congress on Big Data, 2017.

[52]Nima Dokoohaki; Mihhail Matskin, “Effective design of trust ontologies for improvement in the structure of socio-semantic trust networks.” International Journal On Advances in Intelligent Systems, vol 1 no 1, year 2008 on: http://www.iariajournals.org/intelligent_systems/intsys_v1_n1_2008_paged.pdf

[53]Christopher Leturc, "Raisonner sur la manipulation dans les systèmes multi-agents: une approche fondée sur les logiques modales." Intelligence artificielle [cs.Al]. Normandie Université, 2019

[54]T.W.A. Grandison, “Trust Management for Internet Applications.” PhD thesis, Imperial College of Science, Technology and Medicine, University of London, Department of Computing, 2001.

[55]Diego de Siqueira Braga; Marco Niemann; Bernd Hell, “Survey on Computational Trust and Reputation Models.” ACM Comput. Surv. 51, 5, Article 101 (November 2018)

[56]Mohammed Laeequddin; B. S. Sahay and all, “Measuring trust in supply chain partners ’relationships.” Measur. Bus. Excell. 14, 3 (2010) pages 53-69, 2010.

[57]Obed Jules; Abdelhakim Hafid; Mohamed Adel Serhan, “Bayesian network, and probabilistic ontology driven trust model for sla management of cloud services.” 2014 IEEE 3rd International Conference on Cloud Networking (CloudNet), 2014.

[58]Roger C. Mayer; James H. Davis; F. David Schoorman, “An integrative model of organizational trust.” The Academy of Management Review Vol. 20, No. 3 (Jul., 1995), pp. 709-734 (26 pages) on: https://www.jstor.org/stable/258792?seq=1#metadata_info_tab_contents

[59]Hisham Salah; Mohamad Eltoweissy, “Towards collaborative trust management.” 2017 IEEE 3rd

International Conference on Collaboration and Internet Computing, 2017.

[60]Teddy Edu-yaw; Eric Kuada, “Service level agreement negotiation and monitoring system in cloud computing.” ieeexplore.ieee.org on: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8507131

[61]Farhana Jabeen; Zara Hamid; Adnan Akhunzadan; Wadood Abdul; Sanaa Ghouzali, “Trust and reputation management in healthcare systems: taxonomy, requirements and open issues.” IEEEAccess, 2018.

[62]Jouni Hiltunen; Jarkko Kuusijärvi, “Trust metrics based on a trusted network element.” 2015 IEEE Trustcom/BigDataSE/ISPA, 2015.

[63]Elaine M. Sedenberg; James X. Dempsey, “Cybersecurity information sharing governance structures: an ecosystem of diversity, trust, and tradeoffs.” 2018. on: https://arxiv.org/abs/1805.12266

[64]Chrystel Gaber; Claire Loiseaux; Mohamad Hajj; Jean-Luc Grimault; Laurent Coureau; Jean-Philippe Wary, “How increasing the confidence in the eSIM ecosystem is essential for its adoption.” Orange on: https://hellofuture.orange.com/en/how-increasing-the-confidence-in-the-esim-ecosystem-is-essential-for-its-adoption/

[65]Shweta Kaushik; Charu Gandhi, “Multi-level trust agreement in cloud environment.” IEEE Xplore, 2020.

[66]Kamran Ahmad Awan; Ikram Ud Din and all, “Robusttrust - a pro-privacy robust distributed trust management mechanism for internet of things.” IEEEAccess, 2019.

[67]Chung-Wei Hang; Zhe Zhang; Munindar P. Singh, *Shin: “*Generalized Trust Propagation with Limited Evidence.” IEEE Computer Society, 2013.

[68]Nelson Kibichii Bore; Ravi Kiran Raman and all, “Promoting distributed trust in machine learning and computational simulation.” IEEE Xplore

[69]Adil Maarouf; Abderrahim Marzouk; Abdelkrim Haqiq, “Towards a trusted third party based on multi-agent systems for automatic control of the quality of service contract in the cloud computing.” 1st International Conference on Electrical and Information Technologies ICEIT’2015

[70]Xiaohui Cheng; Yang Luo; Qiong Gui, “Research on trust management model of wireless sensor networks.” 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC 2018)

[71]Thirukkumaran. R; Muthu kannan. P, Survey “Security and trust management in internet of things.” 2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN), 2018.

[72]Hanxu Wang; Jiulei Jiang; Weimin Li, “A dynamic trust model based on time decay factor”. 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovations, 2018.

[73]Bin Mu; Shijin Yuan, “A method for evaluating initial trust value of direct trust and recommender trust.” 2010 International Conference on Computer Design And Applications (ICCDA 2010)

[74]Abdullah Al-Noman Patwary; Anmin Fu and all, “Authentication, access control, privacy, threats and trust management towards securing fog computing environments: a review.” 2020. on: https://arxiv.org/abs/2003.00395

[75]Xiaolan Xie; Yang Li, “Trust management model of cloud computing based on multi-agent.” 2015 International Conference on Network and Information Systems for Computers, 2015.

[76]Yubiao Wang; Junhao Wen and all, “A novel dynamic cloud service trust evaluation model in cloud computing.” 2018 17th IEEE International Conference on Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And

Engineering, 2018.

[77]E. S. Shamila; V. Ramachandran, “A new trust evaluating model for web services access.” IEEE Xplore, 2010.

[78]Markus Jäger; Stefan Nadschläger; Josef Küng, “Concepts for trust propagation in knowledge processing systems - a brief introduction and overview.” 2018 17th IEEE International Conference on Trust, Security and Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering, 2018.

[79]Artem Vorobiev; Nargiza Bekmamedova, “An ontological approach applied to information security and trust.” 18th Australasian Conference on Information Systems Information Security and Trust via Ontologies 5-7 Dec 2007

[80]Ping Wang; Jing Qiu, “Trust and probability analysis in P2P network.” 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 2011.

[81]ETSI Industry Specification Group (ISG) Zero touch network and Service Management (ZSM). 2019. Zero-touch network and Service Management (ZSM); Reference Architecture. ETSI GS ZSM 002 V1.1.1. 2019. on: https://www.etsi.org/deliver/etsi_gs/ZSM/001_099/002/01.01.01_60/gs_ZSM002v010101p.pdf

[82]Jordi Ortiz; Ramon Sanchez-Iborra; Jorge Bernal Bernabe; Antonio Skarmeta; Chafika Benzaid; Tarik Taleb; Pol Alemany; Raul Muñoz; Ricard Vilalta; Chrystel Gaber; Jean-Philippe Wary; Dhouha Ayed; Pascal Bisson; Maria Christopoulou; George Xilouris; Edgardo Montes de Oca; Gürkan Gür; Gianni Santinelli; Vincent Lefebvre; Antonio Pastor; Diego Lopez, “INSPIRE-5Gplus: intelligent security and pervasive trust for 5G and beyond networks.” ARES ’20: Proceedings of the 15th International Conference on Availability, Reliability and Security. ACM Digital Library, 2021. on: https://dl.acm.org/doi/abs/10.1145/3407023.3409219

[83]Israr Ahmad; Kok-Lim Alvin Yau; Mee Hong Ling; Sys Loong Keoh, “Trust and reputation management for securing collaboration in 5G access networks: the road ahead.” IEEE Access, 2020. on: https://ieeexplore.ieee.org/abstract/document/9050780

[84]“Floodlight OpenFlow Controller.” [Online]. Available: http://www.projectfloodlight.org/floodlight/

[85]“Mininet: An Instant Virtual Network on your Laptop (or other PC)” [Online]. Available: http://mininet.org

[86]Kevin Murphy’s Bayesian Networks Toolbox, MIT AI lab,200 Technology Square, Cambridge. Available: http://www.ai.mit.edu/˜murphyk/Software/BNT/bnt.html

[87] MATLAB Release 2013A, The MathWorks, Inc., Natick, Massachusetts, United States.

[88]“Qt software package for Python.” [Online]. Available:http://www.qt.io/download/

[89]“UbiGraph 3-D graph representation tool.” [Online]. Available:http://www.ubietylab.net/ubigraph/

Appendix A

A.1 Introduction to Trust concepts in 5G environment

5G is designed to be "open" and software driven, letting operators the possibility to use modular software components from different vendors. Open standards and modular software help operators to get visibility into 5G system, for example through monitoring techniques. The cloud native design of 5G networks facilitates this transparency. In addition, NFV and SDN techniques allow dynamic monitoring implementation.

The fact that various functions (VNF) and equipment in 5G slices are potentially provided by third parties (which sometimes have no trust relationship with the service provider, e.g., the operator-), leads us to consider the notions of trust and liability.

In addition, as infrastructures grow and become more complex, it is no longer possible to manage manually trust and accountability. The need to consider trust and liability stems also from the fact that some future end-to-end architectures will be designed to span over multiple domains. This is the case, for example, of the ETSI's ZSM framework [81] which architecture is designed to empower full automated network and service management in multi-domains environments including operations across legal operational boundaries [82].

The openness and visibility in 5G can ease trust management. In addition, such visibility can permit to adapt the security resources to combat the threat of hackers (in triggering countermeasures to mitigate ongoing attacks or threats in the 5G network).

Functions and components deployed in the infrastructure are all the more sensitive when they deal with virtual security functions. Such security functions can be for example: vFirewall, vChannel Protection, virtual Intrusion Detection System (vIDS), virtual Authentication, Authorization and Accounting (vAAA) and vProxy [82].

Since the operator has an obligation to provide a certain quality of service, (contractualized with its customers), it is necessary for the operator to have the ability to choose the functions in charge of securing the infrastructure or the service.

In this respect, the notion of *trust* is crucial for choosing at best the components. Trust concerns both the components/functions and the suppliers of these components.

Operators need tools to manage trust and orchestrate security resources in their infrastructure.

As a matter of fact, trust helps to orchestrate resources because this is a crucial parameter to be considered when selecting a security function or when deploying such function at an adequate place into the network.

On the other hand, orchestrating resources can help to increase the trust. For example, this is the case when a security function is to be placed at an adequate location to compensate the risk arising from untrustworthy functions in the vicinity. Such kind of security functions may be for example monitoring functions which observe the behavior of the network (or service) in order to help to ascertain the level of trust of some components already in place.

In addition, managing the trust helps to identify the *level of liability* which will be a key concept to take into account in the new networks. On this regard, the European commission, has noted that «legally ascertaining the allocation of liability is pivotal for the successful development and roll-out of the internet of things (European Commission, 2015), implying that improper delineation of liability concerns will hamper technology adoption.»[51]

A.2 Components of trust

A.2.1 Trust definition

According to [52], there is a variety of trust definitions and there is no agreement on a generic definition. Researchers mostly have defined trust depending on the context and orientation of their work. In other words: «trust is seen as having a purpose or a context.»[52]

In a general scope [53] describes trust as: «the willingness of an agent to rely on another agent in relation to certain acts or words with respect to a certain property such as, for example, sincerity, reliability, willingness to act, ability to perform a given task.» As per [54]: «trust is the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context». However, this notion of "firm belief" seems reductive, as trust can be measured by a continuous value and not by a simple "0/1" quotation. In other words, trust representing this belief or perception of an entity is: «in the extent to which future actions and reactions are appropriate in a collaborative relationship with entity.»[83]

The paper [55] states that a «common concept is visualizing trusting as the relationship between a so-called Truster and a Trustee. As depicted in Figure 39, the truster is the party that is in need of some service and thus places his/her trust into the trustee, a second entity who is supposed to provide the required service [56].»

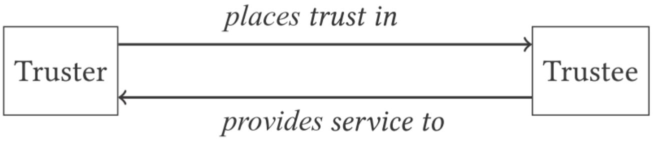


Figure 39 : The relationship between trustee and truster.

In our context of 5G composite infrastructures, we can see trust as when an entity P (Provider) is to provide a Service (S) to an entity C (Customer); in this context, trust is relevant to the probability that the entity P satisfies the request from entity C for the Service S. [57]

Another aspect of trust is the fact that «trust comprises the intention to accept vulnerability»[50]. According to [50], «trust requires taking risks and accepting some amount of vulnerability to accomplish a specific goal […]. The decision to trust needs to be weighed against the level of risk in the trust relationship.»

In addition, many scholars have asserted that trust is a gradual process which evolves over time [58]. As a matter of fact, the dynamic dimension of trust has to be taken into account, especially in the context of a network infrastructure that can evolve over time.

In order to approach a certain completeness in definition of trust in our 5G composite infrastructures context, we propose to model a set of concepts dealing with trust. The proposed model (in UML) is depicted in Figure 40 and is commented below. As we can see, trust is composed generally of technical oriented elements (where trust is measured through interactions -direct or indirect ones and/or through behavior monitoring) and no technical oriented elements where trust measurement is measured through other means. No technical oriented elements include business aspects of the truster/trustee relationship, regulatory aspects and such information such as certifications (relevant to international or national schemes), self-assessments, audits, evaluation tests and information from a "Security Panel" (concept mentioned in [62] "sharing information in the ecosystem").

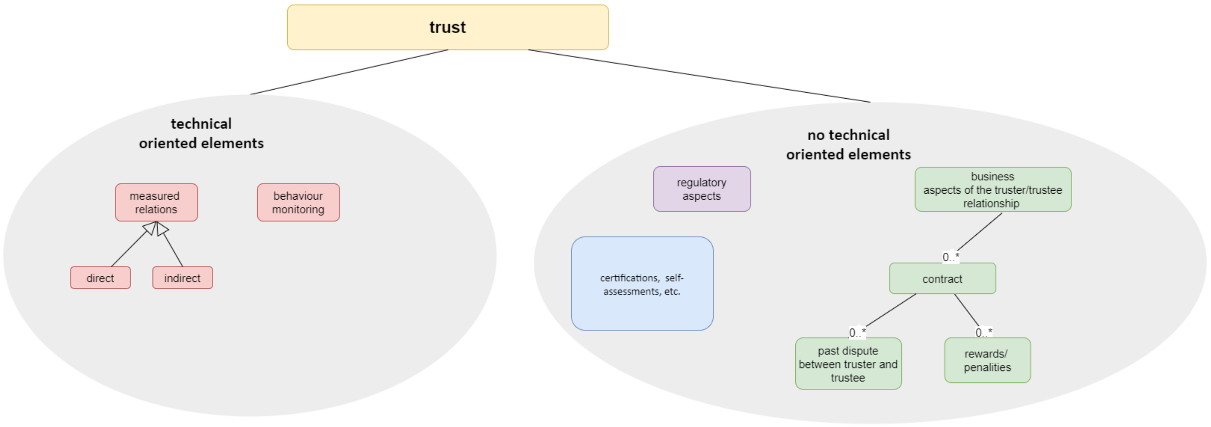

trust
technical oriented elements
measured relations
behaviour monitoring
direct
indirect
no technical oriented elements
regulatory aspects
business aspects of the truster/trustee relationship
certifications, self-assessments, etc.
0..*
contract
0..*
0..*
past dispute between truster and trustee
rewards/ penalities

Figure 40 : The composition of trust

We will deal with technical oriented elements further on. We detail below the non-technical oriented elements:

Business elements

Business elements entail question of whether there is a relationship between the truster and the trustee. The effective establishment of a contract is very generally favoured by a trust already placed in the other contracting party. And in the other hand, it may occur that the level of trust is higher when the parties have contracted with each other (than when there is no business relation between them). Contracts can be declined in obligations of means and/or obligations of results. Concerning past or current contracts, they may be linked with possible disputes between the truster and the trustee which may influence the level of trust.

Regulatory aspects

A truster may place more trust in a trustee than in another, depending on the regulatory framework that prevails for both entities. For example, if the truster and the trustee fall under the same jurisdiction, the trust may be greater.

A.2.2 Targets of trust in 5G composite infrastructures context

The targets of trust in the context of 5G composite infrastructures can be summed up as in Figure 41.

1) Trust in the "builder or owner of the component".
2) Trust in the component itself.
3) Trust in a function of the component.
4) Trust in the environment where takes place the truster/trustee relationship.

When we deal with "Trust in a function of the component (3)", it could be noticed that the "Trust in the environment (4)" can be bypassed, by:

- first implementing a cryptographic security channel between the truster and the trustee function
- and second: relying on the "Trust in the component itself (2)" e.g., through a protection of the function itself inside the component.

When such measures are not possible, other techniques, such as monitoring, must be implemented to alleviate the risks of an untrustworthy environment.

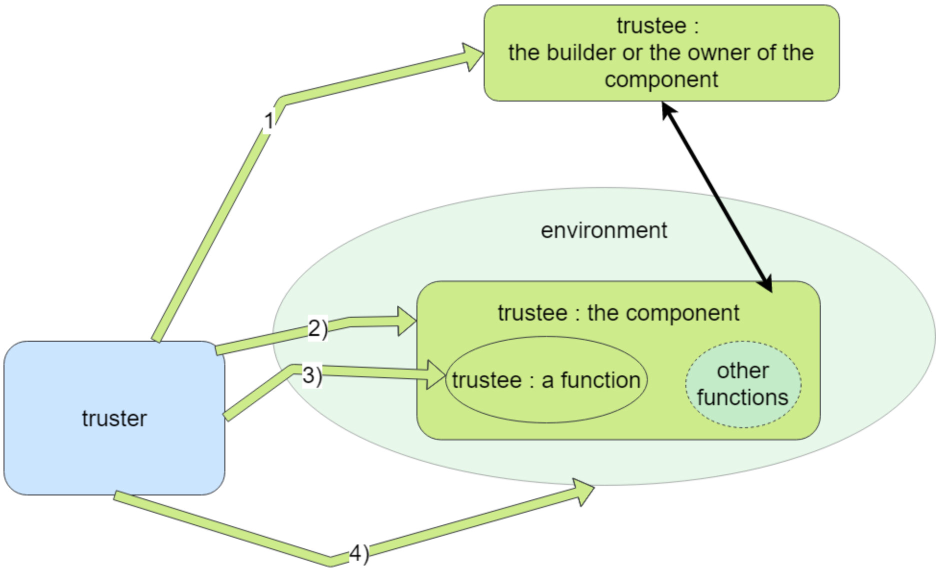


Figure 41 : Targets of trust in 5G composite architectures

A.2.3 Trust environment

First, the **scale** in which the trust is measured is very important. 5G composite infrastructures (made of different functions –VNF-, components, etc...), are not at the same scale (in terms of the number of actors) as for example social networks «where massive numbers of consumers, providers and brokers are largely autonomous […].»[59] Nevertheless, some trust-awareness techniques adopted for other domains such as P2P (e.g., reputation-based trust-measurement techniques) may find application in 5G services composite architecture context.

Asymmetry between actors has also to be taken into account. This is the case, for example, in the context of Cloud computing, where there is sometimes «information asymmetry when two parties are involved in a transaction, and that one party tends to operate in their own interest at the expense of the other party».[60] In such context, a «dilemma exists due to information asymmetry where the CSP ("Cloud Service Provider, ndlr) has more information than the CSU (Cloud Service User, ndlr) such that the CSU cannot directly ensure that the CSP is acting in their (CSU) interest.» [60]

In addition, as quoted by [61] : «If an entity e1 trusts another entity e2, this does not mean e2 will trust e1. Trust may or may not be unidirectional and asymmetric […]. Trust is mutually independent between the two sides.». Such asymmetry may be encountered in 5G composite infrastructures.

There is also the question of **subjectivity**, especially when humans intervene in the trust assessment, for example when trusted metric information is presented in a user-friendly form to involve humans in the decision. [62] As indicated in [61] in the domain of interpersonal relations: «Trust is subject to the expectations one person has of another.».

To assess the trust level, **sharing information in the ecosystem** is central. Source [63] establishes a "Taxonomy of Information Sharing Models", among which we can find categories such as government-centric, government-prompted industry-centric, corporate-initiated peer-based, small highly vetted individual-based groups, open-source sharing platforms [63]. For example, in the eSIM (embedded SIM) domain, [64] proposes to set up a new organisational role called “Security Panel”, involving stakeholders as co-actors and co-responsible for the global security. This Security Panel (instantiated onto different regulation contexts) would create a security knowledge-based on the eSIMs security levels and would build a framework for authorized actors to share information related to security exposure and risk evaluation. Such Security Panel would be coupled with certification schemes.

Some sources indicate that a "**third party**", a "broker" or a "trust evaluator" is indispensable in the context of implementing trust assessment and monitoring through SLAs (Service Level Agreement). For example, in the Cloud domain, [65] proposes a "Cloud Trust Evaluator (CTE)" as the «one of the most important entity in cloud environment which is responsible for evaluating the trust value of different entities. […] A user contacts with CTE to find the trust value for a particular service. CTE will respond back with details of trust value of service provider […] User will select the CSP whose trust value is either equal or higher than the threshold value.» [65]. But mandating a third party is only a point of view; other sources propose other architecture as decentralized ones. We will come back on this later.

The contractualization dimension is also an important aspect when considering trust. A distinction must be made between cases where a contract is established and cases where there is no contract. It is also necessary to distinguish the presence of intermediaries in the contractualization. In particular, the presence of a broker may complicate the trust model.

A.3 Trust management

A.3.1 Architectures

The topology of the network affects the methods of measurement and calculation of the trust. Architecture for trust management can be classified as: centralized and distributed (also called "decentralized"[83]).

In centralized architecture, one server is used to calculate the trust value for all the components of the architecture.

In distributed architecture, «the trust dissemination is distributed which helps nodes10 to act independently and make them able to evaluate the trust locally. The major significance of distributed trust is that the nodes do not have to rely on any centralized authority.»[66]

Centralized and distributed architectures have their own advantages and disadvantages. For example: «The centralized approach requires network-wide information which may not be able to cater for real-time response; while the distributed approach requires local information only which may not be optimal for making network-wide decision.»[83]

In some decentralized approach as that of [67] (in the scope of IoT), a truster can «estimate the amount of trust to place in a trustee according to estimates from a small number of agents in proximity to the trustee.» In addition, a «node can distribute the evaluated experience of a particular node with other nodes.»[66]

Distributed architectures for trust assessment can be implemented by implementing "agents" techniques. [68] «Agent systems are self-contained software programs that embody domain knowledge and behave with a particular degree of independence to achieve specified goals.»[69]

Between a centralized and a distributed architecture, there are intermediate types such as the "cluster network topology" as described in [70]. This is a somewhat hierarchical architecture, where "cluster head nodes" are responsible for collecting the direct trust values uploaded by each wireless sensor nodes under them.

Distributed architecture for trust assessment in 5G composite architectures context can find an application in MEC based architecture.

A.3.2 Evaluation of trust

In this chapter, we will focus on the evaluation of trust for the targets "Trust in the builder or owner of the component", "Trust in the component itself" and "Trust in a function of the component", as illustrated in Figure 41.

[10] Ndlr : When we quote citations using the term "node", it should be considered in this document more generally as a hardware or software component or function.

The composition of trust (meaning the elements which are used to evaluate the trust) can be various. As per [66], trust can be divided into three major components: knowledge, reputation and experience. The article [65] indicates that the «trust evaluation among different entities is based upon multiple functions as: feedback evaluation, risk monitoring, data availability, reward/punishment selection and time factor analysis.» The paper [50] states that the truster must have some knowledge of the trustee's behaviour and motivations before they act on trust. One can notice that this applies in numerous configurations in the 5G composite network context.
Concerning trust management (and measurement) based on a penalties and rewards, components or actors accumulating penalties made visible to other entities may see decreasing the trust other entities place on them. On the contrary, a component or actor accumulating visible rewards will see its level of trust increasing. This method can influence the behaviour of components of the system toward gaining trust.

Trust measures can be divided into two broad categories: measures to determine a "direct trust" and measures to determine an "indirect trust".

Direct trust [71]

Direct trust is based on experiences or observations, direct interactions between the two entities that are trustor node and the trustee node. Direct trust is event-driven, which means that a node only evaluates trust when an event occurs between two nodes.[66] In addition, trust value based on experiences or observations may take into account a temporal decay factor: when trustor and trustee do not interact for a certain time, the trust value may decrease over time.[72]

Indirect trust [71]

In Indirect trust, there are no direct interactions or experiences between the truster and the trustee. An Indirect trust can be established when the truster cannot directly observe the communication behaviours of the trustee.[71]

Indirect trust is based on indirect experiences. This can be seen also as a kind of transitivity; for example: when Alice trusts Bob and Bob trusts Cherry, Alice will trust Cherry. [52]

Trust by reputation is a particular declination of indirect trust. It is also called "Recommended Trust"[71] or "Trust in recommendation"[52]. The difference between Direct trust and Recommended Trust (called recommender trust is illustrated in Figure 42.

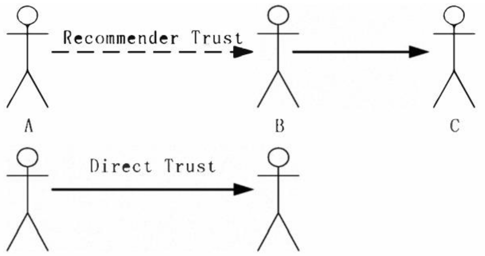


Figure 42 : Direct trust and recommender trust

The notion of reputation involves always more or less the notion of «network-wide belief or perception of the trustworthiness of an entity»[83], implying a collective belief or aggregated opinion of a group of entities in another entity in a network community [83].

In current life, there is reputation-based trust for example, when «Cherry trusts Bob to recommend a good dentist».[52] In network or services domains, trust is based on the recommendations and opinions of the other nodes. «The reputation-based trust model is broadly applied in peer-to-peer (P2P), e-commerce services, social media, and user reviews.» [74] but also in the Cloud domain (e.g. [65],[57]). Comparing the ratings by the truster and by the trustee of acquaintances that truster and trustee have in common [67] is also a kind of Trust by recommendation.

Trust measurement encompasses both direct trust and indirect trust measures. This in the case at least in the following domains: Cloud (e.g. [65] [75] [76]), Web Services Access (e.g. [77]), and IoT (e.g. [71]).

A.3.3 Trust modelling and quantification

A.3.3.1 Trust models

There are many domains where trust models and reputations systems have been studied, for example: Cloud computing, web browsing in general (and E-commerce in particular), social networks, healthcare, among other sectors, and more generally the question of how the knowledge processing systems can work with trust values [78].

Trust models are often based on graphs. «Trust modelling with graphs can express complicated system nuances and yet be clearly understood.» [50]. In [67], trust relationships «naturally form a trust network, weighted directed graph in which vertices represent agents, and edges represent directed trust relationships weighted by trust level. The outcomes of prior interactions affect each edge weight, and a truster can evaluate these assessments to decide whether or not to interact with a prospective trustee.»

Using ontologies can complete the use of graphs. This is the case in [79], where system components communicate with each other by sharing a common vocabulary in order to mitigate attacks through a trust-based collaboration. Article [52] in that regard describes trust relations and their subcomponents using ontologies in order to efficiently design and engineer trust networks in semantic web-enabled social systems. Possibly more adaptable to 5G composite infrastructures, [57] proposes an ontology-based framework for allocating resources taking into account the level of trust. The framework measures the reputation of service providers and updates it continuously if SLA violations are detected. The system also notifies the providers in this case.

A.3.3.2 Trust quantification

Trust relation has a trust metric, which can be quantitative and/or qualitative, characterizing the degree to which the truster trusts the trustee. This quality or quantity represents the intensity and level of trust [52].

Some sources consider that «trust is a binary decision and that trying to assign a level to trust is technically misleading. Trust is the state comprising the intention to accept risk to accomplish a goal.»[50] In such approach, the easiest ways to represent trust values is the usage of a binary trust model: 0 = don’t trust and 1 = fully trust. [78]. However, of course, it is possible to represent trust level values within a continuous span (e.g., from 0 to 1) or as a kind of percentage view. [78]

A.3.3.3 Trust computing method

The methods to compute the trust level are numerous: weighted average method, probability theory, fuzzy logic, game theory and machine learning concepts [71]. Generally, these methods measure trust through learning of exchanges, events and behaviours. The question then arises as to the initial value of the trust to be considered into the computation algorithm. This initial value can be null or can be established by different methods (such as for example that of [73] based on probabilistic techniques). Probabilistic trust calculation models are often quoted in the literature. They are used to determine the likelihood that the trustee will behave in the manner intended or agreed upon. In P2P Network context, probability techniques combine feedbacks for deriving trust ratings [80]. Trust levels can be evaluated through Bayesian networks (probability of observing states given an history), using metrics on equipment availability, response time and MTBF (mean time between failures).[57]

Positive and negative experiences are often used for calculating trust. For example: «Alice’s trust in Bob is modeled as the pair ⟨r,s⟩, where r represents Alice’s positive experiences with Bob and s represents her negative ones.» The probability of the next experience being positive is calculated with r and s with a probabilistic method.

A challenge in trust measurement is the measurement of possible disparate elements, such as for example: indicators of the availability of security functions, presence of potentially dangerous unexpected communications and the fact that a contract exists or not for a particular component. In this respect, the combination of ontology-based techniques dealing with qualitative aspects and other techniques dealing with quantitative aspects may be an interesting path to explore.

A.3.4 Requirements for trust management systems

A "trust management system" must be able to implement 4 types of processes (not all types are necessarily implemented in all use cases):

1) Assessing trust a priori before implementing (or continuing) a relationship.
2) Using trust assessment to feed information to an orchestrator in charge of choosing (or confirming in its role or removing) a component in the system.
3) During running the relationships involving the component in question, checking that the level of trust observed is consistent with what was expected, acceptable or agreed (if contracted).
4) Use the information gleaned from process 3 (checking) to refine process 1.

For identifying possible requirements in terms of trust management on distributed systems, I.Ahmad et al. [83] conducted a study focused on human users (designers and developers) whose opinions have been gathered in focus group workshops. Among the notable requirements that emerge from this study, some of which can certainly be taken up in the 5G composite infrastructures context; we can note in particular:

- The need for trust to be calculated transparently and continuously. This question of transparency applies rather in a decentralized infrastructure where each component may have a different owner.
- The notion of "Pre-trusted entities" which may give some first indications of trust before the interactions.
- Rather taking into account in the calculation of trust recommendations coming from recommenders which preferences are similar to preferences of the truster.
- The possibility to choose between different options sorted by trust level.

Appendix B ETSI studied documents

2014. ETSI GS NFV-SEC001; NETWORK FUNCTIONS VIRTUALISATION (NFV). NFV SECURITY; PROBLEM STATEMENT. This specification is the VNF security initial point, enumerating the potential vulnerabilities of NFV of several kinds. Regarding remote attestation, the authors raised the “outstanding” operational problem of secure key distribution over sheer scale, site-distributed and highly dynamic VNF instantiation. Another discussed topic is the resource isolation and secure crash. This problem statement can be viewed as the question being answered by the two following specifications which on one side discuss on the VNF on boarding management and on the other side delivers security and trust guidance.

2014 ETSI GS NFV-MAN 001 V1.1.1 (2014-12). NETWORK FUNCTIONS VIRTUALISATION (NFV); MANAGEMENT AND ORCHESTRATION This specification delivers a general VNF Life cycle management by MANO. This includes the on-boarding of VNFs, software images provided in the VNF Package for the different VNF components are catalogued in one or more NFVI-PoPs, using the support of VIM.

2014 ETSI GS NFV-SEC003 SECURITY AND TRUST GUIDANCE. This specification gathers all information useful to keep the VNF components in a secure state all over the different life cycle stages including at VNF crashing. The VNF Package and VNF descriptor and the associated certificate-based security is stated, as well as the secure boot (leveraging Intel TXT technology). This specification can be viewed as ETSI foundation and initial referential for VNF Security.

2016 ETSI GS NFV-IFA 011 V2.1.1 (2016-10) , NETWORK FUNCTIONS VIRTUALISATION (NFV); MANAGEMENT AND ORCHESTRATION; VNF PACKAGING SPECIFICATION The specification details the packaging of VNFs to be delivered to service providers, focusing on the holistic end-to-end view of the VNF Package lifecycle, from design to runtime, capturing development as well as operational views. VNF Package lifecycle management end-to-end use cases as well as NFV Architectural Framework functional blocks are detailed.

2017 ETSI GS NFV SEC 012 V311 (11-2017). SPECIFICATIONS ON SYSTEM ARCHITECTURE FOR EXECUTION OF SENSITIVE NFV COMPONENTS. This specification makes an exhaustive list of all must-have (and a few may-have) for the execution platform of deemed more sensitive NFV or NFV components. The definition of the sensitiveness or exposure to risk is arbitrary and the list of such sensitive software will grow with time and the emergence of new attack vectors. This specification is therefore more to be viewed as a wish list for the host platform with all possible security enablement possibly leveraged by the different stakeholders (VNF vendor, VNF operator, cloud service provider) during the life cycle of future sensitive VNF. One shall notice that both Hardware-Mediated Execution Enclave (aka TEE) and Hardware-Based Root of Trust (to be associated as a TPM) are both specified, thus expected to be present on the platforms. In fact, HMEE is supposed to shelter a security agent that brings security functions to externally placed VNFs while the TPM is needed for the RA. To be noted that the HMEE is itself remotely attested and the agent is supposedly dealing with run time integrity verification. The TPM is still needed to cover the need of load time workload and platform attestation through the RoT-CoT platform code verification. TEE and TPM shall be viewed as offering complementary security assurance. TEE is not replacing TPM but are shielding local agents offering runtime integrity checks. One could consider that TPM could eventually do run time integrity checks too. However, TEE-shielded local agents do it can produce at a much lower computational costs than certificate-based attestations by the TPM. Security agents can also execute other security functions (as being arbitrary code) such as and not limited to behavioural monitoring, which are out of reach for rigidly defined and crypto-centric TPM. A security challenge is stipulated regarding security agent. Introspection attack (with an access on the host OS) can simply remove the agent from the executed process.

2017 ETSI GS NFV SEC 013 V311 (02-2017). SPECIFICATION ON NETWORK FUNCTIONS VIRTUALISATION (NFV) RELEASE 3; SECURITY; SECURITY MANAGEMENT AND MONITORING SPECIFICATION. The specification document details the functional and security requirements for automated, dynamic security policy management and security function lifecycle management, and Security Monitoring of NFV system. NFV security monitoring is explored with different use cases in order to establish security requirements. The monitoring functions and other security NFV shall be first secure bootstrapped through a specific protocol associating a VNF Bootstrap Service and a VBS agent located inside the VNF. The secure bootstrapping ensures that the VNF has been securely bootstrapped on its NFVI, according to all security and policy configuration information defined by the VNF security controller. Both VBS and NFVI shall be HMEE equipped for secure boot process.

2017 ETSI GR_NFV-SEC007 (2017-10). REPORT ON ATTESTATION TECHNOLOGIES AND PRACTICES FOR SECURE DEPLOYMENT. This report explores how attestation (certificate-based authentication and integrity) can be leveraged for the attestation of VM (with a vTPM-based RoT), network nodes and finally abstract-level network services. The report initiates discussion on how HMEE (in ETSI terminology) can be used for solving runtime integrity verification.

2017 ETSI GR NFV-SEC 009 V1.2.1 (2017-10) ; NETWORK FUNCTIONS VIRTUALISATION (NFV); NFV SECURITY; REPORT ON USE CASES AND TECHNICAL APPROACHES FOR MULTI-LAYER HOST ADMINISTRATION This report highlights and stresses the need for multi-layer administration for NFV. This special technique is aimed at preventing malicious access for view or for change on a VNF memory by a root access user. This need is referred as introspection. This document introduces and defines ETSI Hardware Mediated Execution Enclave HMEE which by definition is made of several security assurances that are all met by SGX Intel (ability to shelter arbitrary-defined software components, ability to inject data, integrity and confidentiality of this code and data against any external codes, whatever their privileges on the machine, …). This document also details the attacks on memory by introspection (from the OS to the application and from the application to the OS).

2018 ETSI GS NFV-SOL 004 V2.5.1 (2018-09) ; NETWORK FUNCTIONS VIRTUALISATION (NFV) RELEASE 2; PROTOCOLS AND DATA MODELS; VNF PACKAGE SPECIFICATION This specification defines the VNF package and especially the certificate file that enables the authenticity and integrity verification of the VNF.

2019 ETSI GR NFV-SEC005 REPORT ON CERTIFICATE MANAGEMENT. This report details the benefits and implications of PKI infrastructure for NFV deployment. Several alternative for key pair generation are described (NFVI, HMEE, HSM). This report describes the different types of certificates according to the managing entity (OSS/BSS or MANO) and the relationship between the NFVI and the MANO essentially.

2019 ETSI GR NFV SEC 018 V1.1.1 REPORT ON NFV REMOTE ATTESTATION ARCHITECTURE This report from brings the vision of the stakeholders, operational implications over different competing RA architecture (MANO space or tenant space). It also shows that behind this generic term (i.e., remote attestation), several level of assurance, a.k.a. LoA (spanning from no RA to VNF software full package RA at run time), can be enforced. The document drafts the operational implications of the key and measurements exchanges between the Cloud Service Provider and its client (the operator). It also alerts on the need for accessing the measures at the NFVI and organize their exchanges with the MANO). Globally, the operations of key storage, measurement and measurement storage, report generation and processing can be produced at different sites by different organisations and the targeted software (for measurement) be of a very different nature, origin and life cycle time constant. It comes out that RA for VNF is not a unique process precisely specified. Instead, RA is a principle which can be implemented differently according to the VNF deployment configurations (architecture, firmware, type of VNF deployment) and the targeted LoA. Higher LoAs require RA for both hypervisor and VM content (at load time or at run time). The document also stresses technical need and challenge of binding a VM to an attested uniquely defined hypervisor. The discontinuity in the memory space management caused by virtualization and so called the “semantic gap” effect is explained and is derived from two distinct roots of trust, respectively hardware-based and virtual for the two domains. In practical terms, this discontinuity is only to be known and as both VM and hypervisor cannot be solidly bound, the good practice is to leverage a bottom-up approach. RA on VM content is only meaningful and relevant if and when the underneath hypervisor has been (separately) attested. This statement is good common sense and a valuable reminder. Two technical challenges can be extracted between the lines of this document. The DLT based layout (final section of the document) relies on the well-known distributed security and storage of the measurements-reports. This opposes to a centralized RA server, highly attracting for the adversary. DoS attack can be simply designed when targeting this server. Concentration of RA management bears its own security threat. Secondly, the smart engineering of the symmetric key based protocol for proof of attestation is motivated by digital certificate’s high computational cost (and related overhead). RA is a costly operation. If this cost can be considered as fully acceptable at workload load time, it differs when RA is used during execution. The frequency of the measurement has to be carefully defined.

2020 ETSI GS_NFV-SEC 021: NETWORK FUNCTIONS VIRTUALISATION (NFV) RELEASE 2; SECURITY; VNF PACKAGE SECURITY SPECIFICATION In this specification, security is applied to VNF Package. The VNF operator needs to select into a fixed set of validated VNFs (against VNF vendor certificates) one of these VNFs. It is important to add the VNFD and a functional validation of the VNF at the operator into the catalogue. VNFD can then take part of the frozen and measured data, while the operator, by delegation can ask the VNF orchestrator to sign a validated package. As a consequence, the VNF Orchestrator catalogue is expanded with new stored images which correspond to operator tested VNF and VNFD deployment flavours. As far as the VNF orchestrator is the one that processes the VNFD data, it can do it only if it relates to a VNF which VNF Package includes such VNFD and is inside the stored VNF packages. Secondly, the expansion specification integrates VNF encryption too. According to the operator VNF policy, the VNF Package shall be stored encrypted in the NFVO registry. This expansion specification defines the signature processing done the NFVO, which shall therefore receive the operator signature for that.

2021 ETSI GS NFV-SEC 024 V0.0.5 (2020-11) NETWORK FUNCTIONS VIRTUALISATION (NFV); SECURITY; SECURITY MANAGEMENT (DEC. 2020 -DRAFT VERSION, WORK IN PROGRESS). This specification replaces the NFV-SEC 013 as given above. It establishes the need for “cross layer security” which in practice reinforces the need to remediate to introspection. The notion of trust domain is specified and a multi trust domain security management, spanning from application to hardware level is detailed. HMEE shielded Security agents are encouraged.

Appendix C Implementation details

C.1 Trust and Reputation Manager (Swagger API)

```
openapi: "3.0.0"
info:
  description: "REST API for the Trust and Reputation Manager (TRM)-Orange "
  version: "0.2"
  title: "TRM API"
  contact:
    email: "jose2.sanchez@orange.com"
paths:

/security/trm/reputation/{timeSlot}/reputationdomain:
    get:
      operationId: "api.security.rca.reputationdomain"
      summary: "returns a reputation domain graph"
      description: ""
      parameters:
      - name: "timeSlot"
        in: "path"
        description: "The time slot "
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      responses:
        "200":
          description: "Success"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/reputationdomain"
        "400":
          description: "Invalid status value"
      tags:
        - "e-trm"

/security/trm/reputation/{timeSlot}/node/{nodeId}:
    get:
      operationId: "computeNodeProbability"
      summary: "Compute the probabliity of node {nodeId} at time slot {timeSlot} "
      description: ""
      parameters:
      - name: "timeSlot"
        in: "path"
        description: "The time slot "
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      - name: "nodeId"
        in: "path"
        description: "The time slot "
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      responses:
        "200":
          description: "Success"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/NodeReputationResponse"
        "400":
          description: "Invalid status value"
      tags:
        - "e-trm"

/security/trm/reputation/{timeSlot}/link/{sourceNodeId}/{targetNodeId}:
    get:
      operationId: "computeLinkReputation"
      summary: "Compute the reputation of link ({sourceNodeId},{targetNodeId}) at time slot {timeSlot} "
      description: ""
      parameters:
      - name: "timeSlot"
        in: "path"
        description: "The time slot "
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      - name: "sourceNodeId"
        in: "path"
        description: "Source of the link"
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      - name: "targetNodeId"
        in: "path"
        description: "Target of the link"
        required: true
        schema:
          type: "integer"
          format: "int64"
          minimum: 0
      responses:
        "200":
          description: "Success"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/LinkReputationResponse"
        "400":
          description: "Invalid status value"
      tags:
        - "e-trm"

components:
  schemas:
    reputationdomain:
      type: "object"
      required:
      - "name"
      - "nodes"
      - "links"
      properties:
        name:
          type: "string"
        nodes:
          type: "array"
          items:
            $ref: "#/components/schemas/Node"
        links:
          type: "array"
          items:
            $ref: "#/components/schemas/Link"
      example:
        name: "SDN reputation graph"
        nodes: [
          {
            "type": "controller",
            "reputation": -1,
            "id": 0
          },
          {
            "type": "switch",
            "reputation": -1,
            "id": 1
          },
          {
            "type": "switch",
            "reputation": -1,
            "id": 2
          },
          {
            "type": "switch",
            "reputation": -1,
            "id": 3
          },
          {
            "type": "host",
            "reputation": -1,
            "id": 4
          },
          {
            "type": "host",
            "reputation": -1,
            "id": 5
          },
          {
            "type": "host",
            "reputation": -1,
            "id": 6
          }
          ]
        links: [
          {
            "type": "control link",
            "reputation": -1,
            "source": 0,
            "target": 1
          },
          {
            "type": "control link",
            "reputation": -1,
            "source": 0,
            "target": 2
          },
          {
            "type": "control link",
            "reputation": -1,
            "source": 0,
            "target": 3
          },
          {
            "type": "switch-switch link",
            "reputation": -1,
            "source": 1,
            "target": 2
          },
          {
            "type": "switch-switch link",
            "reputation": -1,
            "source": 2,
            "target": 3
          },
          {
            "type": "host-switch link",
            "reputation": -1,
            "source": 1,
            "target": 4
          },
          {
            "type": "host-switch link",
            "reputation": -1,
            "source": 2,
            "target": 5
          },
          {
            "type": "host-switch link",
            "reputation": -1,
            "source": 3,
            "target": 6
          }
          ]
    NodeReputationResponse:
      type: "object"
      properties:
        timeSlot:
          type: "integer"
          format: "int32"
        nodeId:
          type: "integer"
          format: "int32"
        reputation:
          type: "number"
          format: "float"
      example:
        timeSlot : 3
        nodeId : 1
        reputation : 1
    LinkReputationResponse:
      type: "object"
      properties:
        timeSlot:
          type: "integer"
          format: "int32"
        nodeId:
          type: "integer"
          format: "int32"
        reputation:
          type: "number"
          format: "float"
      example:
        timeSlot : 3
        sourceNodeId : 3
        targetNodeId : 1
        reputation : -1
    Node:
      type: "object"
      properties:
        id:
          type: "integer"
          format: "int64"
        asset:
          type: "string"
        reputation:
          type: "array"
          items:
            type: "number"
            format: "float"
    Link:
      type: "object"
      properties:
        id:
          type: "integer"
          format: "int64"
        source:
          type: "integer"
          format: "int64"
        target:
          type: "integer"
          format: "int64"
          items:
            type: "number"
            format: "float"
      required:
      - "name"
      - "nodes"
      - "links"
      example:
        name: "SDN reputation domain"
        nodes: [ ]
```

C.2 Proof of Transit (Swagger API)

```
openapi: 3.0.0
info:
  title: OPoT Open API
  version: 0.0.1
  description: API for managing OPoT Paths.
  contact: {}
servers:
  - url: /api/v2/
    description: Base path of the API
paths:
  '/pot/controller/path/{uuid}':
    parameters:
      - schema:
          type: string
          format: uuid
        name: uuid
        in: path
        required: true
        description: uuid of the existing path
    get:
      summary: Get information about a existing OPoT Path
      responses:
        '200':
          $ref: '#/components/responses/path_information'
        '404':
          $ref: '#/components/responses/error'
        '500':
          $ref: '#/components/responses/error'
      operationId: opot_sdk.api.get_path_info
      description: Retrieve the status and information of a existing path
      tags:
        - opot
    delete:
      summary: Destroy OPoT Path
      operationId: opot_sdk.api.destroy_path
      responses:
        '200':
          $ref: '#/components/responses/default'
        '404':
          $ref: '#/components/responses/error'
      description: Destroying a OPoT Path
      tags:
        - opot
  /pot/controller/path:
    post:
      summary: Create OPoT Path
      operationId: opot_sdk.api.create_path
      responses:
        '200':
          $ref: '#/components/responses/path_information'
        '500':
          $ref: '#/components/responses/error'
      requestBody:
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/PathDescriptor'
            examples:
              example_ports:
                value:
                  protocol: UDP
                  nodes:
                    - ip: 192.168.0.1
                    - ip: 192.168.0.2
                      port: 55002
                  receiver:
                    ip: 192.168.0.3
                    port: 55003
                  sender:
                    ip: 192.168.0.4
                    port: 55004
              example_no_ports:
                value:
                  protocol: UDP
                  nodes:
                    - ip: 192.168.0.1
                    - ip: 192.168.0.2
                  receiver:
                    ip: 192.168.0.3
                    port: 55444
                  sender:
                    ip: 192.168.0.4
                    port: 55432
              example_local_test:
                value:
                  protocol: UDP
                  nodes:
                    - ip: 127.0.0.1
                      port: 55555
                    - ip: 127.0.0.1
                      port: 55556
                  receiver:
                    ip: 127.0.0.1
                    port: 55432
                  sender:
                    ip: 127.0.0.1
                    port: 55433
      description: Create a OPoT Path
      tags:
        - opot
      parameters: []
components:
  schemas:
    PathInfo:
      title: PathInfo
      type: object
      x-examples:
        PathInfo:
          creation_time: 1616488134568011
          masks:
            - 1JpbYd5bYEJsXBMlkcXADet3CjVLC+303ZYQpnA7QE4=
          nodes:
            - address:
                ip: 192.168.0.200
                port: 55432
              node_id: 0
              node_type: Ingress
              status: Operative
            - address:
                ip: 192.168.0.201
                port: 49158
              node_id: 1
              node_type: Egress
              status: Operative
          opot_id: cdf2b942-8bb1-11eb-a94e-eb152c183a2f
          protocol: UDP
          status: Operative
      description: Information about a deployed path
      properties:
        'opot_id ':
          type: string
          format: uuid
        nodes:
          type: array
          minItems: 2
          items:
            type: object
            properties:
              status:
                type: string
              node_id:
                type: integer
              address:
                $ref: '#/components/schemas/Address'
              node_type:
                type: string
                example: Ingress
            required:
              - status
              - node_id
              - address
              - node_type
        masks:
          type: array
          minItems: 1
          items:
            type: string
            minLength: 44
            maxLength: 44
            example: 2xaH0dBnJBRGQDXl8bhRXLqm81cVV7ddNJDrp77uvbs=
        protocol:
          type: string
          pattern: UDP|TCP
          example: UDP
        creation_time:
          type: integer
          format: time
          example: 1615305214342100
      required:
        - 'opot_id '
        - nodes
        - masks
        - protocol
        - creation_time
    PathDescriptor:
      title: PathDescriptor
      type: object
      description: 'Descriptor of the path that will be created '
      x-examples:
        PathDescriptor_port:
          protocol: TCP
          nodes:
            - ip: 192.168.0.1
            - ip: 192.168.0.2
              port: 55002
          receiver:
            ip: 192.168.0.3
            port: 55003
          sender:
            ip: 192.168.0.4
            port: 55004
        PathDescriptor_no_port:
          nodes:
            - ip: 192.168.0.200
            - ip: 192.168.0.201
            - ip: 192.168.0.202
          protocol: UDP
          receiver:
            ip: 192.168.0.155
            port: 33333
          sender:
            ip: 192.168.0.150
            port: 33334
        PathDescriptor_test:
          protocol: UDP
          nodes:
            - ip: 127.0.0.1
              port: 55000
            - ip: 127.0.0.1
              port: 55001
          receiver:
            ip: 127.0.0.1
            port: 55432
          sender:
            ip: 127.0.0.1
            port: 55432
      properties:
        protocol:
          type: string
          pattern: UDP|TCP
          example: UDP
          description: Protocol that is going to be used for the path. UDP or TCP
        nodes:
          type: array
          minItems: 2
          uniqueItems: true
          items:
            $ref: '#/components/schemas/Address'
        receiver:
          $ref: '#/components/schemas/Address'
        sender:
          $ref: '#/components/schemas/Address'
      required:
        - protocol
        - nodes
        - receiver
        - sender
    Address:
      title: Address
      type: object
      properties:
        ip:
          type: string
          format: ipv4
        port:
          type: integer
          maximum: 65353
          minimum: 0
          example: 55432
      required:
        - ip
      x-examples:
        Address_port:
          ip: 192.168.0.1
          port: 55432
        Address_no_port:
          ip: 192.168.0.1
      description: IPv4 Address with the port
  securitySchemes: {}
  responses:
    path_information:
      description: Example response
      content:
        application/json:
          schema:
            type: object
            properties:
              path_info:
                $ref: '#/components/schemas/PathInfo'
          examples:
            example-1:
              value:
                path_info:
                  'opot_id ': 1266841a-0650-4496-a5ad-e84a5ae762f3
                  nodes:
                    - status: Operative
                      node_id: 0
                      address:
                        ip: 192.168.0.1
                        port: 55432
                      node_type: Ingress
                    - status: Operative
                      node_id: 0
                      address:
                        ip: 192.168.0.1
                        port: 55432
                      node_type: Ingress
                  masks:
                    - 2xaH0dBnJBRGQDXl8bhRXLqm81cVV7ddNJDrp77uvbs=
                  protocol: UDP
                  creation_time: 1615305214342100
    error:
      description: Example response
      content:
        application/json:
          schema:
            description: ''
            type: object
            properties:
              detail:
                type: string
                minLength: 1
              status:
                type: number
              title:
                type: string
                minLength: 1
              type:
                type: string
                minLength: 1
            required:
              - detail
              - status
              - title
              - type
          examples:
            example-1:
              value:
                detail: Details of the error
                status: 500
                title: Title of the error
                type: string
    default:
      description: Default response
      content:
        application/json:
          schema:
            type: object
            properties:
              message:
                type: string
              status:
                type: number
  examples: {}
tags:
  - name: opot
```

C.3 Risk Assessment Graph (Swagger API)

```
openapi: "3.0.0"
info:
  description: "REST API for the Risk Assessment Graph"
  version: "0.1"
  title: "RAG API"
  contact:
    email: "morgan.chopin@orange.com"
paths:
  /security/rag/create/{ragName}:
    get:
      operationId: "api.security.rag.generate"
      summary: "Generate a RAG"
      description: ""
      parameters:
        - in: "path"
          name: "ragName"
          schema:
```

```
      type: "string"
     required: true
     description: "The RAG name"
   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
        $ref: "#/components/schemas/Rag"
   tags:
    - "rag"
 /security/rag/cms:
  post:
   operationId: "api.security.rag.getCms"
   summary: "Returns the set of cournter-measures
available for a RAG"
   description: ""
   requestBody:
    content:
     application/json:
      schema:
       $ref: "#/components/schemas/Rag"
   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
         type: "array"
         items:
          $ref: "#/components/schemas/Cm"
   tags:
    - "rag"
 /security/rag/risk/{timeSlot}:
  post:
   operationId: "api.security.rag.global_risk"
   summary: "Compute the global risk at time slot
{timeSlot} for a given RAG"
   description: ""
   requestBody:
    content:
     application/json:
      schema:
       $ref: "#/components/schemas/Rag"
    required: true
   parameters:
    - in: "path"
     name: "timeSlot"
     schema:
      type: "integer"
     required: true
     description: "The time slot"

   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
        $ref:
"#/components/schemas/GlobalRiskResponse"
   tags:
    - "rag"
 /security/rag/risk/{timeSlot}/node/{nodeId}:
  post:
   operationId: "computeNodeRisk"
   summary: "Compute the risk of node {nodeId} at
time slot {timeSlot} for a given RAG"
   description: ""
   requestBody:
    content:
     application/json:
      schema:
       $ref: "#/components/schemas/Rag"
   parameters:
   - name: "timeSlot"
    in: "path"
    description: "The time slot "
    required: true
    schema:
     type: "integer"
     format: "int64"
     minimum: 0
   - name: "nodeId"
    in: "path"
    description: "The time slot "
    required: true
    schema:
     type: "integer"
     format: "int64"
     minimum: 0
   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
        $ref:
"#/components/schemas/NodeRiskResponse"
    "400":
     description: "Invalid status value"
   tags:
    - "rag"

/security/rag/risk/{timeSlot}/link/{sourceNodeId}/{t
argetNodeId}:
  post:
   operationId: "computePropagatedRisk"
   summary: "Compute the propagated risk of link
({sourceNodeId},{targetNodeId}) at time slot
{timeSlot} for a given RAG"
   description: ""
   requestBody:
    content:
     application/json:
      schema:
       $ref: "#/components/schemas/Rag"
   parameters:
   - name: "timeSlot"
    in: "path"
    description: "The time slot "
    required: true
    schema:
     type: "integer"
     format: "int64"
     minimum: 0
   - name: "sourceNodeId"
    in: "path"
    description: "Source of the link"
    required: true
    schema:
     type: "integer"
     format: "int64"
     minimum: 0
   - name: "targetNodeId"
    in: "path"
    description: "Target of the link"
    required: true
    schema:
     type: "integer"
     format: "int64"
     minimum: 0
   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
        $ref:
"#/components/schemas/PropagatedRiskResponse
"
    "400":
     description: "Invalid status value"
   tags:
    - "rag"
 /security/rag/mitigateRisk/{timeSlot}:
  post:
   operationId: "solvePCSP"
   summary: "Compute the best counter-measures
placement strategy at time slot {timeSlot} for a given
RAG"
   description: ""
   requestBody:
    content:
     application/json:
      schema:
       $ref:
"#/components/schemas/MitigateRiskBody"
    required: true
   parameters:
    - in: "path"
     name: "timeSlot"
     schema:
      type: "integer"
     required: true
     description: "The time slot"

   responses:
    "200":
     description: "Success"
     content:
      application/json:
       schema:
        $ref:
"#/components/schemas/MitigateRiskResponse"
   tags:
    - "rag"
components:
 schemas:
  MitigateRiskBody:
   properties:
    rag:
     $ref: "#/components/schemas/Rag"
    securityConstraints:
     $ref:
"#/components/schemas/SecurityConstraints"
    cms:
   type: "array"
   items:
    $ref: "#/components/schemas/Cm"
SecurityConstraints:
  type: "array"
  items:
   type: "object"
   properties:
    accessId:
     type: "integer"
    nodeId:
     type: "integer"
    difficultyThreshold:
     type: "number"
     format: "float"
  example:
   [
    {
     accessId : 5,
     nodeId : 2,
     difficultyThreshold : 7
    },
    {
     accessId : 6,
     nodeId : 1,
     difficultyThreshold : 8
    }
   ]
MitigateRiskResponse:
 type: "object"
 properties:
  timeSlot:
   type: "integer"
   format: "int32"
  minimumCostValue:
   type: "number"
   format: "float"
  placementStrategy:
   type: "array"
   items:
    type: "object"
    properties:
     nodeId:
      type: "integer"
     counterMeasure:
      type: "string"
GlobalRiskResponse:
 type: "object"
 properties:
  timeSlot:
   type: "integer"
   format: "int32"
  globalRisk:
   type: "number"
   format: "float"
 example:
  timeSlot : 4
  globalRisk : 29.2
NodeRiskResponse:
 type: "object"
 properties:
  timeSlot:
   type: "integer"
   format: "int32"
  nodeId:
   type: "integer"
   format: "int32"
  risk:
   type: "number"
   format: "float"
 example:
  timeSlot : 3
  nodeId : 1
  risk : 4.9
PropagatedRiskResponse:
 type: "object"
 properties:
  timeSlot:
   type: "integer"
   format: "int32"
  sourceNodeId:
   type: "integer"
   format: "int32"
  targetNodeId:
   type: "integer"
   format: "int32"
  propagatedRisk:
   type: "number"
   format: "float"
 example:
  timeSlot : 3
  sourceNodeId : 3
  targetNodeId : 1
  propagatedRisk : 4.9
Node:
 type: "object"
 properties:
  id:
   type: "integer"
```

```
          format: "int64"
        asset:
          type: "string"
        vulnerability:
          type: "string"
        impact:
          type: "number"
          format: "float"
        entry_point:
          type: "boolean"
        probability:
          type: "array"
          items:
            type: "number"
            format: "float"
    Link:
      type: "object"
      properties:
        id:
          type: "integer"
          format: "int64"
        source:
          type: "integer"
          format: "int64"
        target:
          type: "integer"
          format: "int64"
        accessibility:
          type: "array"
          items:
            type: "number"
            format: "float"
    Cm:
      type: "object"
      properties:
        id:
          type: "integer"
          format: "int64"
        name:
          type: "string"
        effect:
          type: "integer"
          format: "int64"
        cost:
          type: "integer"
          format: "int64"
        nodes:
          type: "array"
          items:
            type: "integer"
      example:
        id: 0
        name: "CM1"
        effect: 11
        cost: 6
        nodes: [1, 2]
    Rag:
      type: "object"
      required:
      - "name"
      - "nodes"
      - "links"
      - "cms"
      properties:
        name:
          type: "string"
        nodes:
          type: "array"
          items:
            $ref: "#/components/schemas/Node"
        links:
          type: "array"
          items:
            $ref: "#/components/schemas/Link"
      example:
        name: "SDN Risk Assessment Graph"
        nodes: [
          {
            "asset": "cisco_2106_wireless_lan_controller",
            "vulnerability": "CVE-2012-0368",
            "impact": 6.9,
            "entry_point": false,
            "probability": [0.5],
            "id": 0
          },
          {
            "asset": "cisco_2106_wireless_lan_controller",
            "vulnerability": "CVE-2013-1235",
            "impact": 2.9,
            "entry_point": false,
            "probability": [0.5],
            "id": 1
          },
          {
            "asset": "cisco_nexus_5548up",
            "vulnerability": "CVE-2013-5556",
            "impact": 10,
            "entry_point": false,
            "probability": [0.155],
            "id": 2
          },
          {
            "asset": "cisco_nexus_5548up",
            "vulnerability": "CVE-2013-5556",
            "impact": 10,
            "entry_point": false,
            "probability": [0.155],
            "id": 3
          },
          {
            "asset": "cisco_nexus_5548up",
            "vulnerability": "CVE-2013-5556",
            "impact": 10,
            "entry_point": false,
            "probability": [0.155],
            "id": 4
          },
          {
            "entry_point": true,
            "id": 5
          },
          {
            "entry_point": true,
            "id": 6
          }
          ]
        links: [
          {
            "accessibility": [1.0],
            "source": 0,
            "target": 1
          },
          {
            "accessibility": [1.0],
            "source": 1,
            "target": 0
          },
          {
            "accessibility": [0.2],
            "source": 0,
            "target": 2,
          },
          {
            "accessibility": [0.2],
            "source": 0,
            "target": 3,
          },
          {
            "accessibility": [0.2],
            "source": 0,
            "target": 4,
          },
          {
            "accessibility": [0.2],
            "source": 2,
            "target": 0,
          },
          {
            "accessibility": [0.2],
            "source": 3,
            "target": 0,
          },
          {
            "accessibility": [0.2],
            "source": 4,
            "target": 0,
          },
          {
            "accessibility": [0.2],
            "source": 1,
            "target": 2,
          },
          {
            "accessibility": [0.2],
            "source": 1,
            "target": 3,
          },
          {
            "accessibility": [0.2],
            "source": 1,
            "target": 4,
          },
          {
            "accessibility": [0.2],
            "source": 2,
            "target": 1,
          },
          {
            "accessibility": [0.2],
            "source": 3,
            "target": 1,
          },
          {
            "accessibility": [0.2],
            "source": 4,
            "target": 1,
          },
          {
            "accessibility": [0.2],
            "source": 2,
            "target": 3
          },
          {
            "accessibility": [0.2],
            "source": 3,
            "target": 2
          },
          {
            "accessibility": [0.2],
            "source": 4,
            "target": 3
          },
          {
            "accessibility": [0.2],
            "source": 3,
            "target": 4
          },
          {
            "accessibility": [1.0],
            "source": 5,
            "target": 2
          },
          {
            "accessibility": [1.0],
            "source": 6,
            "target": 4
          }
          ]
```