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

D4.3: Liability mechanisms for 5G environments

Version: v0.91

Deliverable type	R (Report)
Dissemination level	PU (Public)
Due date	31/10/2021
Submission date	29/10/2021
Lead editor	Gürkan Gür (ZHAW)
Authors	Jean Philippe Wary (ORA), Ghada Arfaoui (ORA), Jose Manuel Sanchez Vilchez (ORA), Morgan Chopin (ORA), Chrystel Gaber (ORA), Jean-Luc Grimault (ORA), Edgardo Montes de Oca, Vinh Hoa La (MI); Gürkan Gür, Wissem Soussi (ZHAW)
Reviewers	Antonio Pastor, Juan Carlos Caja Diaz (TID), Alejandro Molina, Antonio Skarmeta, Noelia Pérez Palma (UMU)
Work package, Task	WP4, T4.3
Keywords	Liability mechanisms, liability enablers, accountability, commitments

Abstract

This document is the final report related to the INSPIRE-5Gplus Task 4.3. It presents the related results of WP4 activities during the project lifetime. The report details the methodology followed in T4.3, and describes the Investigated liability mechanisms, denoted as liability enablers. Moreover, the implementation and integration of these enablers are presented.

Document revision history

Version	Date	Description of change	List of contributor(s)
v0.1	01/04/2020	Initial version	ZHAW
v0.2	26/05/2021	Updated content and structure	ZHAW
v0.5	30/09/2021	Review chapter 1 – 3	Orange, ZHAW
v0.6	30/09/2021	Enablers' descriptions/contribution	ALL
v0.7	01/10/2021	Minor updates in content	ALL
v0.8	05/10/2021	Format and errata correction	ZHAW
v0.9	16/10/2021	Reviewers' comments	TID, UMU
v0.91	21/10/2021	Final editing and GA approval	EURES

List of contributing partners, per section

Section number	Short name of partner organisations contributing
Section 1	ORA, ZHAW
Section 2	ZHAW, ORA
Section 3	ZHAW, ORA
Section 4	ORA, ZHAW, MI
Section 5	ORA, ZHAW, MI

Disclaimer

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.

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

Executive Summary

This deliverable presents the project activities and outcomes related to liability mechanisms of the Task T4.3 in the INSPIRE-5Gplus project. It details the activities done by each partner and the consortium in this task.

The report introduces the methodology followed in the WP4 and specifically liability mechanisms investigated in the T4.3. In that regard, a list of these mechanisms, named liability enablers, is provided. The report describes each enabler's functionality and details the description of the relevant problems and challenges to address together with the state of the art in their field. Subsequently, the solution proposed related to each problem, the status of development outcomes, and implementation are presented.

Finally, an alignment and integration approach with the overall INSPIRE-5Gplus architecture has been made in this task. Accordingly, each liability enabler is placed visually associated with WP2 defined components in the INSPIRE-5Gplus high-level architecture. Additionally, the activity related to interfaces definition for the integration in the INSPIRE-5Gplus architecture and among WP4 enablers has been done. The results reached at the end of the task T4.3 period are included in this deliverable where the ultimate operation of each enabler will be finalized within the integration and testing activities in WP5.

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
API	Application Programming Interface
ARM	Advanced RISC Machine
B5G	Beyond 5G
BIOS	Basic Input Output System
BSS	Business Support System
CNF	Containers Network Functions
CSP	Cloud Service of Provider
DIVE	Docker Integrity Verification Engine
DLT	Distributed Ledger Technologies
DRM	Digital Right Management
ETSI	European Telecommunication Standards Institute
HLA	High Level Architecture
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	Multi-access Edge Computing
MIP	Mixed-Integer Programming
MUD	Manufacturer Usage Descriptions
NFV	Network Function Virtualization
NFVI	NFV Infrastructure
NFVO	NFV Orchestrator
NGMN	Next Generation Mobile Networks
NST	Network Slice Template
OS	Operating System
OSM	Open Source MANO
OSS	Operations Support System
PKI	Performance Key Indicators
PNF	Physical Network Function
RA	Remote Attestation
RAG	Risk Assessment Graphs
RCA	Root Cause Analysis

SDN	Software-defined networking
SFC	Service Function Chaining
SLA	Service Level Agreement
SP	Service Platform
SSLA	Security SLA
TOSCA	Topology and Orchestration Specification for Cloud Applications
TPM	Trusted Platform Module
TRL	Technology readiness level
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

Until now, 5G development has focused on performance and functionality for fulfilling its envisaged goals and enabling advanced and ubiquitous digital services. However, since it implies a complex, inter-connected, multi-tenant and multi-domain architecture, dealing with liability is of paramount importance for its adoption. Indeed, as stated in [1], "Many security systems are really about liability rather than risk; and failure to understand this has led to many computer security systems turning out to be useless."

5G is claimed to satisfy the dramatically growing need of users and things for diverse services and massive connectivity in future networks. Indeed, 5G networks are expected to be multi-access to serve around 7 trillion heterogeneous connected things, amongst which 20 billion are human-oriented devices, with 1000x higher mobile data volume per area and 10x-to-100x higher user data rate, as stated by the 5G-PPP partnership [2]. In addition, 5G is supposed to be an extremely flexible and dynamic network to fulfil the myriad of use cases and verticals with very different requirements such as ultra-low latency or ultra-reliability. These requirements are expected to be even more challenging for Beyond 5G or 6G systems – they will offer a quantum leap over this promise and realize a revolutionary set of services with unprecedented quality and reliability levels such as Further-enhanced Mobile Broadband (FeMMB), Enhanced Ultra-Reliable Low-Latency Communication (ERLLC/eURLLC), and ultra-massive Machine Type Communication (umMTC) [3]. They are envisaged to support 1 Tbps peak data rate, 1 Tbps/m^2 area traffic capacity, and ms level latency [4]. Expected to be deployed in the 2030s, they will unlock advanced traffic demands for more flexible, data-hungry, and tactile applications [5]. In particular, as an enabler to customize the service levels expected by verticals, network slicing is proposed to deploy several logical networks on top of the same infrastructure in 5G networks. It will be instrumental in Beyond 5G networks as well. In that setting, each slice is optimized to fulfil certain objectives imposed by specific use cases [6]. Network softwarization via Software-Defined Networking (SDN) and Network Function Virtualisation (NFV) technologies is a key paradigm for implementing 5G slicing.

This technological revolution in communications and networking implies shifting to more enriched business cases or advanced consumer services where multi-party 5G networks are crucial. However, this situation requires opening up a 5G infrastructure (e.g., towers, gateways, networks) to third parties (e.g., mobile devices, IoT devices, VNF providers), which raises many questions regarding the responsibilities among partners. Moreover, due to the integration of autonomous and closed-loop control with minimal human intervention, liability becomes an obscure question since the responsibility, incident and commitment resolution is not trivial. As an example, from the slicing perspective, in the absence of any prior trust relationship, the slice provider (SP) always bears the financial or legal impact of fraud or mischief. The SP is legally bound by contracts to provide the agreed Quality of Service. Additionally, if there are accidents or frauds in a critical service using a slice (e.g., due to an insecure product or operation), the SP or any Slice Component Provider can be liable.

In this multi-party and multi-layer 5G architecture, the definition of liability and responsibilities when security breaches occur is essential to support confidence between parties and compliance with regulation, since zero-risk security cannot be achieved. As in the "Y2K Act" [7], one can assume that, legal and financial responsibility in 5G contexts will have to be distributed proportionately among any liable parties involved in a service. The appreciation of the stakeholder's liabilities is defined by 5G's worldwide deployment, multiple stakeholders and complex interconnections of hardware and software at different levels. In addition to this, many different orchestration entities appear at different layers in 5G networks, such as the VIM, VNFM and NFV, which make multiple decisions concerning the end-to-end service management with partial views of the network. This implies that responsibility regarding mismanagement and outages is blurred even within the same administrative domain.

In addition to technical aspects, liability is also a public policy issue. This is reflected in the current European discussion around a potential extension of Liability Directive to cover software and AI issues

[8], [9]. As highlighted by [54] and [55], the new features of emerging technologies such as AI, IoT, blockchain and distributed ledger share the challenge of identifying the root causes of a harmful event and attributing liabilities proportionately to the implication of stakeholders. Consequently, the expert group recommends revisiting the current liability regimes by considering the following challenges:

- The autonomy and opacity of AI systems questions the extent to which their producers can be held liable. While it is easy to understand that expert systems can be held to the same standard as a human expert, it is not the case for other algorithms such as supervised or unsupervised machine learning or algorithms which purposefully reduce or exclude human factor. Given that their behaviour depends on external sources and can evolve after their deployment. It is therefore extremely difficult to identify the exact faulty state which cause harm and whether this state is actually related to defect during manufacturing or usage.
- The openness and density of IoT systems questions the capacity of victims to prove what exactly caused the damage. Identifying which IoT device injured a person would be trivial but the root cause could be complicated to spot. Some users might -voluntarily or involuntarily- not update their IoT devices or unknowingly connect incompatible devices. Their open connectivity and potential proximity may encourage widespread propagation of malware and makes them very vulnerable to cyberattacks as described by Dorsemaine et.al. [53]. Proving that a cyberattack took place and pinpointing the origin of infection is not a trivial task and victims need to be assisted to perform it.
- The collaborative and anonymous nature of blockchain and distributed ledger technologies make it inherently difficult to attribute liabilities to individual parties.

This drive is augmented with the fact that liability is an inter-disciplinary concept going beyond security and concerning law, insurance and contractual relationships. Considering the role and importance of 5G and future Beyond 5G to realize pervasive digital and resilient services, we need to integrate liability related mechanisms such as benchmarks of attacks and defences, threat modelling frameworks, evidence logging, detection and alert mechanisms as well as remediation or recovery plans into the overall 5G security management framework to cope with forensic and litigation as advised by Anderson [1], Kingston [60] and Kumar [56]. To serve this purpose, INSPIRE-5Gplus project has been working on liability mechanisms envisaged to be a part of the holistic INSPIRE-5Gplus security architecture in WP4.

As part of a visibility, dissemination and monitoring effort about the INSPIRE-5Gplus WP4 activities, this document covers the related activities (design, development and interfacing) and the consolidated description of the proposed liability mechanisms as part of T4.3.

The document is organized as follows: This section introduces a summary of the activity done in the WP4 with emphasis on liability mechanisms (T4.3) and a summary table with the proposed liability enablers. Section 2 provides the context and information to understand the liability 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 one by one all the enablers, their implementation and development. Finally, Section 5 positions each enabler in the architecture INSPIRE-5Gplus and depicts our approach for their main interactions based on interfaces specification.

1.1 Definitions

Trust[2] is the most important behavioural factor in managing relationships and in overcoming risks/uncertainty. It relies either on the formalization of agreements (contracts), mutual confidence established by fruitful exchanges and acquaintance. Trust is a non-reciprocal peer-based property where the trustor forms an opinion on how good the trustee is on providing a specific service. It is the

[2] Although trust and related concepts are specifically elaborated in Tasks T4.1 and T4.2, we provide some key information here for the sake of completeness.

subjective degree of belief a trustor has on a trustee to perform a concrete task in this specific system [57]. It depends on a context and corresponds to a real number of positive collaborations between trustor and trustee [58].

Reputation reflects the collective opinion by members of a community and hence is a community-based property. It is a function of trust. In most reputation models in the state of the art [61], reputation should be very difficult to earn and very easy to lose, especially when the reputation concerns a management entity carrying out critical tasks.

Risk exposure is a combination of 1) a loss, generally the financial consequences of the risk and a peril, 2) the uncertain event provoking a loss and 3) an object of risk, the exposed resource [11].

Responsibility corresponds to a task which needs to be performed while complying with a set of objectives. In this document, we consider that a responsibility binds a provider who performs a task for a customer.

Accountability consists in setting up a governance to comply with responsibilities. It is achieved by defining roles which have the right to perform actions with specific means (capabilities) as well as the evidence required to control actions. The accountability relationship binds an *accountor* to perform a task and justify its outcomes to an *accountee* based on a framework of negotiated measures and objectives [59]. The latter is a set of 1) technical evidence which measure the gap between the actual outcome and the expected outcome, 2) design evidence which demonstrate that the effectiveness of measures, 3) evidence at policy-level demonstrating the adequacy of the measures and designs to fulfil a given objective [59]. Just as the trust relationship, accountability is non-reciprocal.

Liability corresponds to an accountability towards legislation. For example, in the case of contracts, both parties are required by the law to fulfil their end of the deal and to be able to demonstrate it.

Penalties and Incentives can be decided when the provider fails or succeeds to comply with the set of objectives agreed between both parties.

1.2 List of INSPIRE-5Gplus liability enablers

Table 1 lists the identified liability enablers in the project (their descriptions are provided in Section 4). 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[3].

Table 1 List of envisaged liability enablers

Enabler Name	Owner	Targeted TRL	Description
Remote Attestation (RAP)	ORA	3	An attestation protocol is a cryptographic protocol involving a target, an attester, an appraiser, and possibly other principals serving as trust proxies. The objective is to supply evidence that will be considered authoritative by the appraiser, while respecting the privacy goals of the target (or its owner). In the INSPIRE-5Gplus project, we focus on a specific family of these protocols, i.e., deep attestation which enables attesting the state of a system having multiple levels (e.g., hardware level, virtualization level, VM). We aim at providing a practical and provably secure deep attestation protocol.

[3] The two enablers LASM and SBO were investigated in a preliminary manner in relation to Task T4.4, and The Security By Orchestration (SBO) was not identified at the inception of the project. Therefore, those enablers will be developed and thoroughly elaborated in T4.4.

Root Cause Analysis based on ML techniques (RCA-M)	MI	5	Based on an existing enabler that is being developed in the H2020 ENACT (https://www.enact-project.eu/) project for IoT networks. It will be extended and adapted to 5G in the INSPIRE-5Gplus project. It is integrated into the monitoring framework that will detect anomalies and use the RCA-M module to determine the probable causes. The RCA-M relies on machine learning and similarity analysis. Thus, to obtain the necessary data sets for the machine learning process, different anomalies are provoked in the planned testbed that will be used for comparison.
Root Cause Analysis for VNF infrastructures (RCA-VNF)	ORA	4	An RCA scheme for performing root cause identification and analysis in virtualized infrastructure to ensure the smooth functioning of networking services relying on SDN and NFV principles.
MANIFEST	ORA	3	Liability Aware Manifests leverage existing manifests to convey responsibilities of various actors and formalize the ownership of these responsibilities. They can be used in several domains such as VNF MANO (MANagement and Orchestration) and IoT management.
GRALAF	ZHAW	2	GRALAF uses graph-theoretic analysis to identify liability based on liability formulations and responsibilities of various actors in the network.
LASM	ORA	3	A Security Orchestrator instantiation which is able to take decisions while taking into account liabilities: to provide clear repartition of liabilities for all stakeholders; to orchestrate Components (VNFs, devices, IoT) by minimizing liabilities of Slice Provider & Slice Owner and entrusting liabilities to trusted stakeholders; to calculate/compute incentives & penalties for Component Providers & Slice Provider
Security By Orchestration (SBO)	ORA + OLP	3	An orchestration scheme under affinities constraints for strong proofs of isolation when VNF of different sensitivities are operated on the same physical resource, e.g., an IoT management VNF with a VNF entailing Lawful Interception features

For each of these propositions, the needed data flow and interactions are under investigation and initial ideas were pushed as the first prospective ETSI GS NFV-SEC 024 [10] contribution with more precise and selected INSPIRE-5Gplus features. This proposition has been examined at the end of January 2021, but the work stays in progress.

Regarding the interaction with WP2 and WP3, we have positioned the proposed WP4 enablers in our high-level architecture and formulated their interactions with other INSPIRE-5Gplus components. This work will be finalized in the Task T4.4 and WP5 activities in the following period.

2 Liability related concepts in INSPIRE-5Gplus context

The notion of *liability* is crucial to operating multi-party end to end product lines (based on multiple and heterogeneous components chaining). Liability concerns both the components/functions and the suppliers of these components during the operation and post-incident situations. The liability management is essential to identify responsibilities and commitments in an infrastructure in a cost-effective way. In this section, we provide a liability-oriented description of important concepts for facilitating our treatment of challenges and rationale for our developed enablers in the following sections (please note that some definitions were already presented in Section 1.1, albeit in a general manner). These definitions are listed in Table 2.

Table 2 Liability related concepts

Concepts	Common WP4 definitions
Responsibility	A party's capability to organize itself and potential delegates in order to achieve a task as agreed and to provide evidence on its achievement.
Accountability	Gathering evidence of the execution of a task in order to "give accounts" or inform, explain why specific decisions have been taken without any consideration related to a fault.
Liability	Essentially, accountability towards legislation. But for our T4.3 work, it also refers to the financial consequence of an outage, that can increase by actions or pre-conditions which increase the probability or the impact of a potential danger [11].
Manifest for a component	The manifest formalizes the expected function delivered by the component, the resources required to deliver this function, the expected output of the component, constraints or restrictions that could be enforced by other Components of the system while keeping track of the responsibility of each entity which took part in the manifest's creation and the component's life cycle.
Manifests over chained components	Multiple components are chained (for instance, the input of some components is the output of other components) in order to deliver a complex or a higher-level function. If two components are chained/linked, the manifests of each component combine to define dependencies and interactions between both components and the compatibility of inputs/outputs should be evaluated at some point.
Root Cause Analysis (RCA)	Procedure to determine the reason and the entity causing a degradation or failure
Imputability	Judiciary term indicating the possibility to attribute the responsibility of a fact or event to a specific person or entity
Remote attestation	The purpose of an attestation protocol is to deliver evidence of a property, evidence that will be considered authoritative by the appraiser.

3 Grand challenges and our approach

This section depicts the process and result of identification and consolidation of liability challenges, as a consolidated list, which are addressed by the WP4 liability enablers.

A process of detailed identification of security challenges to provide liability in 5G and Beyond 5G (B5G), and consolidation of the result, was organized following a methodological process similar to Task T4.1 work. 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 risk still to come with the massive adoption of promising new technologies that will define the 5G in the long term, including the softwarization process of mobile network communication (NFV, SDN, slicing) and supportive technologies (AI, edge computing, TEE). The second step involved a systematic collaborative identification and inventory with description of the challenges to solve, based on different vision 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. Here, each challenge was analysed to verify what could be addressed in the field of trust and liability capacities, discarding those that were in other security areas (using Section 2 initial concepts). For those related to liability, an exercise of aggregation and categorization were made to provide a short list of “grand” common problems to be addressed by different technologies.

Next, Table 3 provides the result with the list of relevant liability “grand” challenges and, for each of them, an approach on how to solve them through the WP4 enablers already introduced in Section 1.2. This process includes a description the areas of work and a description of the enablers and their technological approach to the challenge described.

Table 3 Liability challenges and how to address them with INSPIRE-5Gplus liability enablers

LIABILITY Grand challenge	INSPIRE-5Gplus approach to solve it
Agents’ responsibilities and liabilities imputation/charging in the case of violations and security incidents in virtualized environment.	*MANIFEST (+ LASM):* They facilitate the liability-aware security management by automating management decisions to fulfil agents’ commitments and obligations. *GRALAF:* It analyses the network topology and security incidents to deduce the responsibilities in a softwarized network environment. *RAP:* It enables the security management system to detect if a VM instance has been correctly instantiated, thus helps the resolution of the responsibilities, while hosted on a given hardware infrastructure.
Attack surface reduction for virtualized environments.	*RAP:* It helps to check the “layer binding” property (see Section 4.1). *SBO:* It satisfies the need of strong proofs of isolation when VNF of different sensitivities are operated on the same physical resource, and thus reduces the attack surface.
Pervasive and granular behavioural definition and monitoring of for liability virtual services and IoT under massive connectivity.	*MANIFEST (+ LASM):* They enable behavioural definition of IoT based on standards, leading to better monitoring of liability resolution in massive connectivity regimes.
Dynamic liability and root cause analysis (integrating provability, accountability and delegation concepts).	*RCA-M:* It provides machine learning based RCA for software-defined networks for security incident detection.

	RCA-VNF: It can find the root cause dynamically concerning SDN networked topologies, where the network topology may evolve due to network reconfiguration and changes. *GRALAF:* It facilitates liability analysis using graph-theory based approaches considering failures and commitment models.
5G network resilience	*RAP:* This mechanism will increase the trust in 5G networks from the verticals perspective via attestation performed remotely, and thus improve resilience. *MANIFEST (+LASM):* It will enable the monitoring and behavioural description of massive IoT in 5G during different phases such as deployment, orchestration and operation. That will especially support the resilience of verticals in 5G and future networks. *RCA-M:* It will provide timely and accurate detection of root causes, leading to rapid resolution of security incidents. *RCA-VNF:* It will contribute to the resilience of 5G networks, as the RCA can identify those networked elements that are under failure and have to be replaced or disconnected from the network topology (quarantine situation).

Those challenges and viability of our approach provided the basis for enablers in the Task T4.3. The exact process and mechanism to measure the efficiency of our liability enablers against identified challenges are specified jointly with the enabler itself. The relevant details are available in the following sections.

4 Liability enablers

4.1 Remote Attestation (RAP)

4.1.1 Description of problem and challenges

The use of virtualized infrastructure became popular years ago from now: starting from services running partially or totally on cloud infrastructure up to the new mobile network generations (5G and beyond). Indeed, the softwarization of mobile network enables flexibility and dynamicity, key properties of new mobile networks, which are required to satisfy the various requirements and expectations of 5G and beyond networks. A network node is then composed of a physical infrastructure, one or many virtualized functions running on top of one or many virtual machines or containers. In this context, the software level (i.e., virtualized network functions, virtual machines or any software) can be instantiated and migrated anywhere in the network. However, in some use cases, this may arise security issues (e.g., leak of private information, exposure of sensitive software/ function, increase of attack vector, etc.). That's why some verticals may dictate a strict policy for VNFs management, namely the VNFs associated to its services must be instantiated and migrated only on top of authorized hardware infrastructure. In this context, mobile network operator, in order to honour its commitment with regards to verticals, needs mechanisms to assure that VNFs are effectively running on a designated hardware infrastructure. In other words, mechanisms that can generate trust in virtualized infrastructure are needed.

In terms of providing trust for 5G networks and their verticals, and the identified general challenges in previous section, Remote Attestation (RA) is a technology that provides trust in the virtualized infrastructure.

- **Grand challenges:** Agents' responsibilities and liabilities imputation/charging in the case of violations and security incidents in virtualized environment
    - **Global target:** RA will help in these grand challenges. RA enabler will enable to detect if a VM instance has been correctly instantiated on top of - or migrated to – a given hardware infrastructure.
- **Grand challenges:** Attack surface reduction for virtualized environments
    - **Global target:** RA will help in these grand challenges. Given a hardware infrastructure, a virtualization layer (i.e., hypervisor) and a VM, RA will help to check the so-called layer binding property, i.e., the VM is running on the designated virtualization layer running on top of a specific hardware infrastructure. This mechanism will increase the trust in 5G network from verticals side.

4.1.2 State of the art

Remote attestation protocols are a corner stone of trusted computing field. These protocols enable an attester to provide a reliable and trustworthy proof, to one or many remote parties, that some properties (e.g., Software integrity, geolocation of a network node) are ensured / guaranteed. Numerous solutions of remote attestation protocols have been proposed in literature. [13] proposes a protocol which proves that (1) a secret has been generated and stores within the secure environment ObC (Nokia Trusted Execution Environment) and (2) this secret can only be used by authorized application. In [14], Yu et al. combined TLS and software integrity checks to specify a secure architecture and communication channel enabling securely collecting integrity measures. Jacquin et al. in [15] presented a hardware-based remote attestation protocol for routers and switches enabling to decide whether a traffic has been / is being routed to the right destination.

4.1.3 Description of the solution

For the sake of simplicity, in this section, we focus on one property to attest, namely the property "layer binding". In a virtualized infrastructure, layer binding refers to the fact of proving that a virtualized infrastructure is running on a designated hardware, i.e., a VM (V1) is running on top of hypervisor (H1) which is running on a compute node (C1). The family of Remote Attestation protocols that enables "layer binding" property is called *Deep Attestation*. Our solution of deep attestation protocol takes the advantages of both single channel and multi-channel implementations while overcoming the limitations. In a 5G network, it can be used by a mobile network operator to check that a VNF has been instantiated on top of - or migrated to – a legitimated hardware infrastructure (i.e., compute node). The RA enabler is composed of three components: RA server, RA Agent T1 and RA Agent T2 as shown in Figure 1.

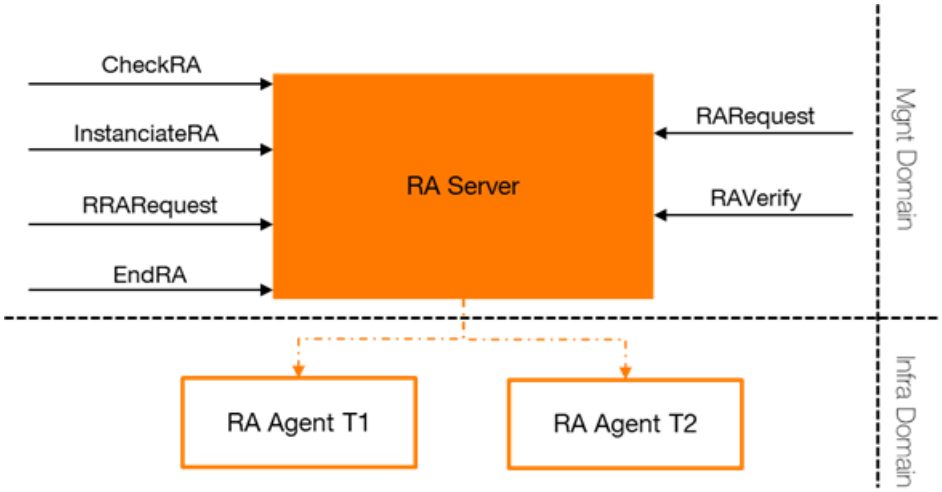

Figure 1 RA components

RA Server is the component that manages the attestation service. It can (a) check if a target is RA compliant and so can perform a RA (*CheckRA*), (b) push required software on the RA target and make the necessary setup, (c) clean up a target from RA software (*endRA*), (d) return back the active RA targets (Running RA Request, *RRARequest*) and (e) run a RA on a target and verify the result (*RARequest*, *RAVerify*). To ensure all these operations, RA server component controls a set of RA Agent T1 and RA Agent T2 components.

RA Agent T1 is a component available on the Infrastructure Domain. It is located on the virtualization layer (i.e., hypervisor) in contact with the hardware, namely, a root of trust.

RA Agent T2 is also a component available on the Infrastructure Domain. It is located on the VM layer in contact with RA Agent T1.

The RA Agents are responsible for data collection from infrastructure and the secure computation and delivery of an attestation result.

Hereafter, we give the details of our deep attestation protocol. In Figure 2, we provide an overview of our protocol. In this context, we consider TPM – and vTPM - as a Root of Trust (RoT) - and a virtualized Root of Trust (vRoT). We give further details about the implementation in the following sections.

The intuition of our protocol is as follows: The hypervisor will securely append to its attestation a list of public keys, corresponding to the VMs physically hosted on the same device. These are public attestation keys. The vTPMs will use the corresponding private keys to sign the authenticated quotes established for the VMs, thus providing linkage.

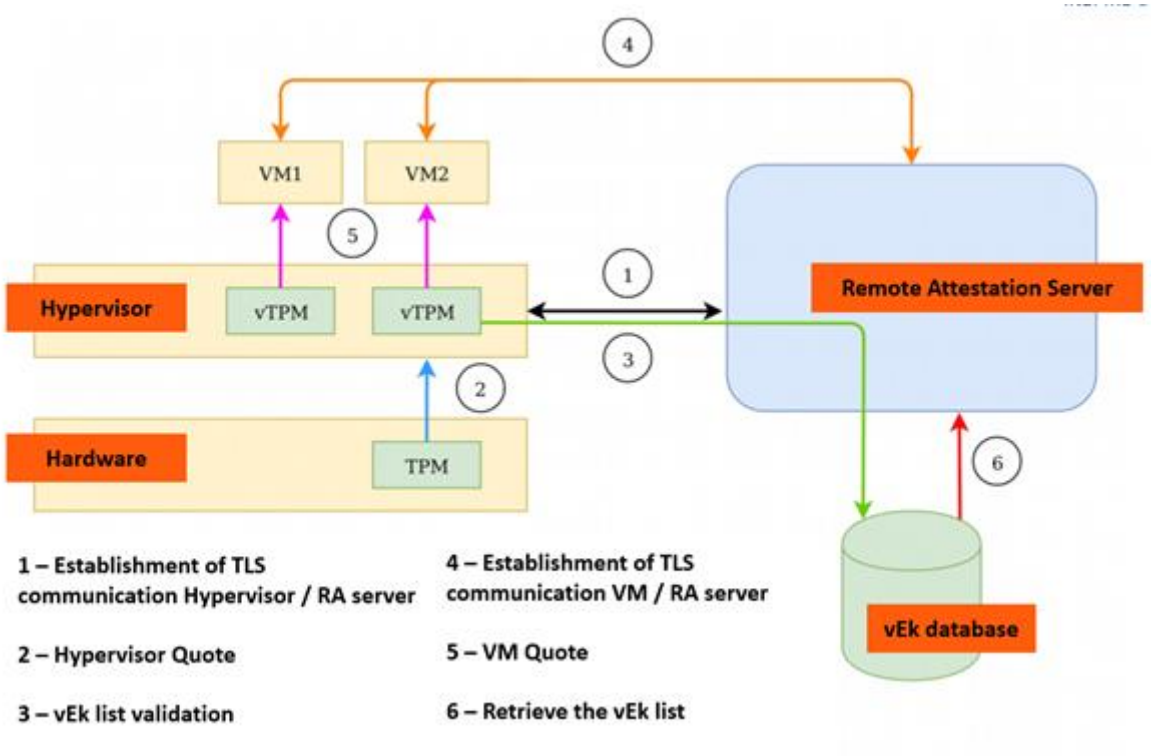

Figure 2 [RA enabler] Deep attestation protocol overview

The protocol steps are:

1. The RA Agent T1 located in the hypervisor will establish a secure TLS channel with the remote attestation server (RA Server)
2. The RA Agent T1 exchanges with the TPM to compute the attestation of the hypervisor. This operation also enables to include the list of public attestation keys of the running VMs (vEK) into the hypervisor quote.
3. If the attestation of the hypervisor is valid the RA server stores the vEK list.
4. Now, in turn, each VM establishes a secure TLS communication with the RA server.
5. Then, each VM computes its quote using its corresponding vTPM.
6. Finally, the RA server checks the validity of the VMs quotes and also the matching of the attestation keys in vEK list already stored.

We consider an infrastructure is valid if:

1. The hypervisor quote is valid
2. The VMs quotes are valid
3. The VMs attestation keys have a matching in the vEk list.

4.1.4 Efficiency factors

Our protocol consists in a cryptographic protocol using data from trust anchors (i.e., TPM (Trusted Platform Module) and vTPM (Virtual TPM)). The result of the protocol is a quote. The verification of the quote gives a Boolean (True, if the quote is valid, False, if the quote is invalid). We will prove the security of the cryptographic protocol using game-based proofs.

4.1.5 Development outcomes

The work that has been done during INSPIRE5G+ project is:

- The definition and development of the external APIs
- Finalizing the implementation and proceeding to some tests
- Submitting a scientific paper to an international conference.
- Discussion with partners in order to create new use cases and collaborations

4.1.6 Implementation

Our implementation consists of three parts: an agent for the hypervisor, an agent for the Virtual Machines, and an attestation server written in Python 3. Any computer equipped with a TPM 2.0 (which can also be emulated) and which has virtualization capacities suffices for the purposes of our implementation. We provide our code as well as a detailed tutorial to how to install and configure both the infrastructure we consider and our implementation here:

(`https://github.com/AnonymousDeepAttestation/deep-attestation`)

We will describe that infrastructure and our implementation in the following part. We summarize our architecture in Figure 3. For the sake of clarity, we represented only two VMs. Our hypervisor is a laptop running Ubuntu 20.04.1 (kernel version 5.4.0-58) with an Intel i5-10210U CPU, 8GB RAM and a Nuvoton TPM NPCT75X.We used KVM to turn this laptop into a hypervisor. All virtual machines are QEMU virtual machines (version 4.2.1) with 2 cores and 4G RAM running Ubuntu 20.04.1. In order to achieve a high attestation performance, we used a full virtual TPM implementation, using QEMU with `libtpms` version 0.7 and `swtpm` version 0.5.

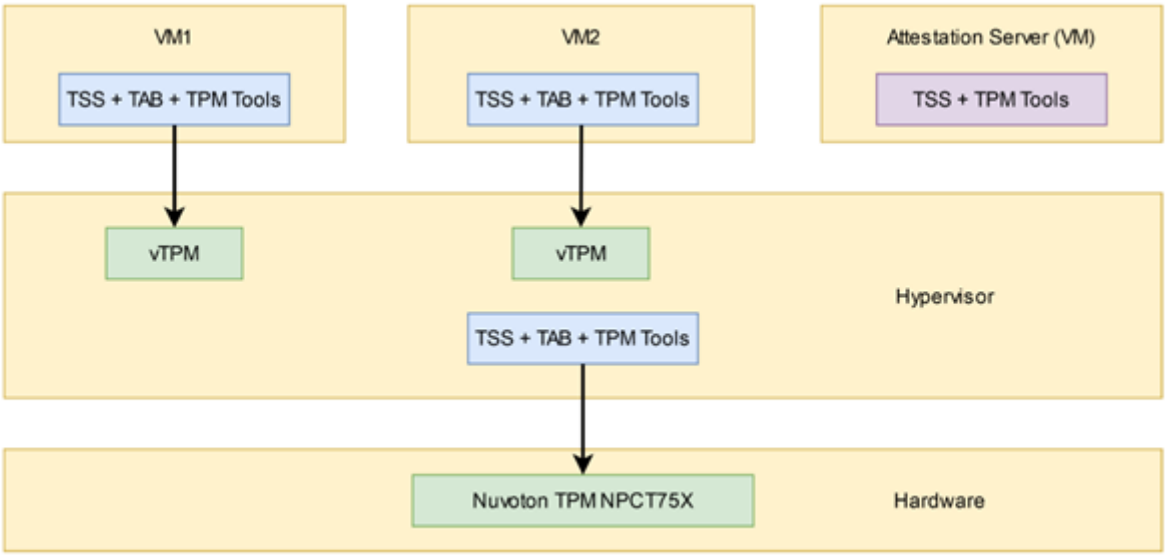

Figure 3 [RA enabler] Deep Attestation protocol - Implemented architecture

The hypervisor, server, and VMs communicate through a virtual network. A virtual bridge on the hypervisor redirects packets on the appropriate virtual interface. To communicate with the TPM we used tpm2-tss, tpm2-abrmd and tpm2-tools from the tpm2-software. Note that the tpm2-tss project implements the TPM software stack (TSS), which is an API specified by the Trusted Computing Group to interact with a TPM.

The tpm2-abrmd implements the access broker and resources to manage concurrent access to the TPM and manage memory of the TPM by swapping in and out of the memory as needed (hardware TPM have limited memory). tpm2-tools are a set of command-line tools based on TSS, which are used to send commands to the TPM. We used Python to wrap tpm2-tools commands.

The attestation server is also a virtual machine, with the same characteristics as those above. This allows to test our implementation on a single machine. We establish a secure connection between the client and the server by using Python's SSL library and then sending protocol messages as encoded json strings directly into TLS socket.

4.1.7 Associated publications and patents

We have a paper under submission in ACNS conference. The paper is entitled "A cryptographic view of deep-attestation, or how to do provably-secure layer-linking,". The authors of this paper are: Ghada ARFAOUI (ORANGE), Pierre-Alain FOUQUE (IRISA), Thibaut JACQUES (IRISA), Pascal LAFOURCADE (LIMOS), Adina NEDELCU (ORANGE), Cristina ONETE (XLIM) and Léo ROBERT (LIMOS). https://tinyurl.com/8whrttrc.

We also have a related patent titled "Method for Providing Certificates Implemented by a Virtualised Computing Platform"[4] about this topic. The patent proposes a deep attestation protocol but uses another approach.

4.1.8 Residual challenges

For the RA enabler, we implement a deep attestation protocol. Our solution is hardware dependent, i.e., it is based on a specific type and implementation of the so-called root of trust - TPM. Network nodes without TPM or with another type of root of trust (e.g., TrustZone) cannot run our protocol.

[4] https://worldwide.espacenet.com/publicationDetails/biblio?CC=WO&NR=2020212501A1&KC=A1&FT=D&locale=fr_EP#

4.2 Root Cause Analysis (RCA-M)

4.2.1 Description of problem and challenges

The RCA-M enabler is a systematic process for identifying "root causes" of problems or events and an approach for responding to them. RCA is based on the idea that effective management requires more than merely "putting out fires" when problems are detected, but also finding ways to prevent them. 5G and B5G mobile networks have introduced a much more dynamic infrastructure based on virtualization (e.g., NFV) and programmability (e.g., SDN). The fact that networks will need to manage a huge number of devices and different stakeholders (e.g., service providers and tenants) need to interact, make mobile networks much more complex and vulnerable. Thus, diagnosing the problems related to performance, but also security, is a major challenge to achieve the necessary automation to manage them efficiently. Network operators require self-healing networks that have a fully automated troubleshooting management process, and the monitoring system should be capable of detecting and diagnosing the issues, respectively done by a security analysis and the RCA-M, as a prerequisite of triggering the adequate recovery actions.

- **Grand challenges:** Dynamic liability and root cause analysis (integrating provability, accountability and delegation concepts).
    - **Global target:** RCA-M will help address this grand challenge by facilitating the Identification of the root cause of security breaches in the NFV/SDN-based virtualized and dynamic environments.
- **Grand challenges:** 5G network resilience
    - **Global target:** RCA-M will help address this grand challenge by providing the security orchestrator with the information that is needed to better react and mitigate detected attacks.

4.2.2 State of the art

The current state of the art research is concerned with RCA of quality and performance issues. There is no commercial solution that specifically addresses RCA of 5G security breaches.

The literature has various works on RCA and its application. In [16], Mfula et al. proposes an adaptive RCA for self-healing in 5G networks. Currently, the actual detection and diagnosis of faults or potential faults is still a manual and slow process often carried out by network experts who manually analyse and correlate various pieces of network data such as, alarms, call traces, configuration management (CM) and key performance indicator (KPI) data in order to come up with the most probable root cause of a given network fault. In this paper, the authors propose an automated fault detection and diagnosis solution called adaptive root cause analysis (ARCA). The solution uses measurements and other network data together with Bayesian network theory to perform automated evidence-based RCA. As it uses a probabilistic Bayesian classifier, it can work with incomplete data and it can handle large datasets with complex probability combinations. Experimental results from stratified synthesized data affirmatively validate the feasibility of using such a solution as a key part of self-healing (SH) especially in emerging self-organizing network (SON) based solutions in LTE Advanced (LTE-A) and 5G.

In [17], Munoz et al. work on RCA for Self-Organizing 5G networks. In the field of self-healing, the heterogeneous character of fifth generation (5G) radio access networks (RANs) will provide a diversity of measurements and metrics that can be translated into valuable knowledge to support detection and diagnosis activities. The more complete the information gathered, the better the SON mechanisms will be able to effectively analyse and solve radio problems. However, temporal dependence between metrics has not been previously addressed in the literature. In this paper, a self-healing method based on network data analysis is proposed to diagnose problems in future RANs. The proposed system analyses the temporal evolution of a plurality of metrics and searches for potential interdependence under the presence of faults. Performance is evaluated with real data from a mature Long-Term Evolution (LTE) network. Results show that the proposed method exploits the available data in the context of heterogeneous scenarios, reducing the diagnosis error rate.

In [18] and [19], Mdini et al. elaborate an approach and tool that analyses labelled connection logs to identify the major contributors to the network inefficiency (e.g., a faulty core device) as well as the incompatibilities between different elements (e.g., make and model of a phone not being able to access a service). The unsupervised machine learning approach uses Call Data Records (CDRs) recording information on regular phone calls and Session Data Records (SDRs) that track every Internet connection in cellular networks. The success/fail labels are used to identify major contributors corresponding to feature value pairs (or their combination) corresponding to elements causing a significant decrease of the network efficiency. The signatures responsible for the network inefficiency are then classified into equivalence classes, which are groups of signatures corresponding to the same problem. Then a graph is generated outlining the dependencies between issues occurring in the network. This graph needs to be pruned to remove unnecessary nodes denoting false problems (i.e., elements appearing as inefficient because they share a part of their logs with malfunctioning ones). The tool ARCD was developed and evaluated to show that it was able to identify the major contributors and the most widespread incompatibilities. The precision (detection accuracy) and the recall (detection rate) were found to be higher than 90%.

4.2.3 Description of the solution

In the context of INSPIRE-5Gplus, the RCA enabler relies on machine learning algorithms to identify the most probable cause(s) of detected anomalies based on the knowledge of similar observed ones. Figure 4 shows the high-level architecture of the implemented enabler.

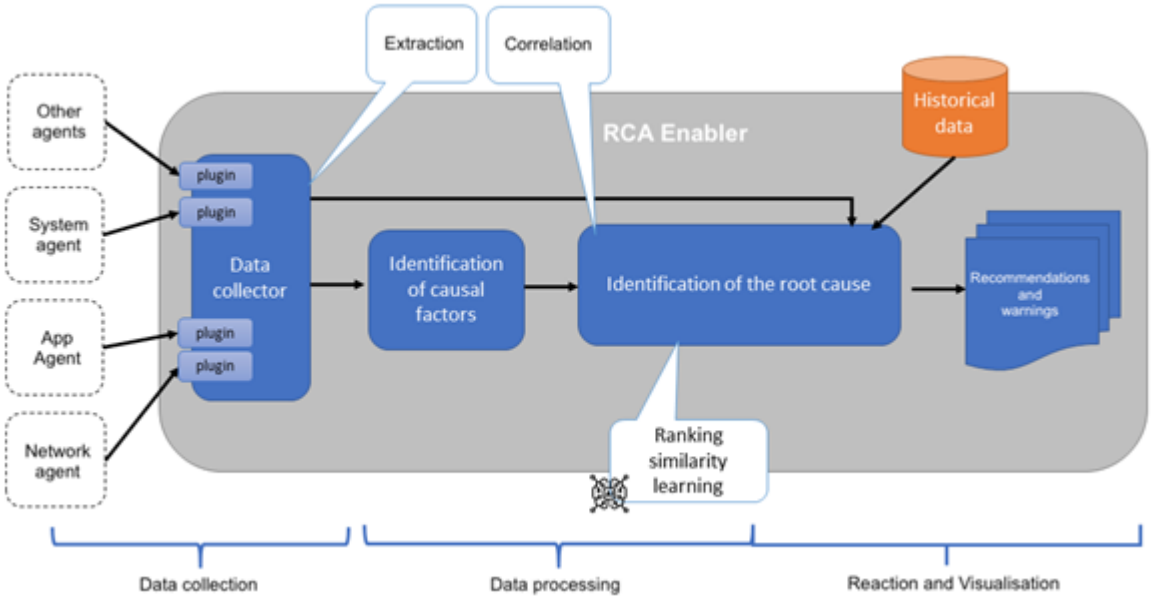

Figure 4 MMT-RCA high level architecture

The **data collector** allows gathering information from different sources (e.g., network, application, system, hardware) by relying on dedicated monitoring agents. It has a plug-in architecture that enhances its extension to new data formats. Parsing such data allows extracting various attribute values relevant to the origin of any detected incident. We automatically select the most relevant attributes by using several feature selection algorithms. These attributes increase the analysis accuracy and reduce the data dimensions as well as the computation resources needed.

The **historical data** is a set of data used for learning purposes. It consists of labelled records collected over time. These records describe the original cause of several incidents (e.g., a sensor is no longer permitted to send data to the central gateway) and the relative attribute values (e.g., downstream data bit-rate measured in the central gateway decreased).

The **historical data** is constructed by two means:

- Active learning: By actively performing different tests including the injection of known failures and attacks. In this case the collected data can be easily labelled since we deal with a controlled system.
- Passive learning: Once an incident is detected without knowing its origin, thanks to the aid of the system experts, classical RCA is performed by debugging different logs and correlating various events to determine the corresponding root causes. The result of this task can be stored in the database with its relevant attributes values.

The historical data are derived from these two sources. The idea is to determine when the system reaches a known undesirable state with a known cause. It involves using the concept of Similarity Learning [20], i.e., Ranking Similarity Learning. The RCA tool calculates the similarity of the new state with the known ones. It presents the most similar states in the relative similarity order. The final goal is to recognize the incident's root origin by using historical data. In this way, the tool can recommend to the operator which countermeasures to perform based on known mitigation strategies.

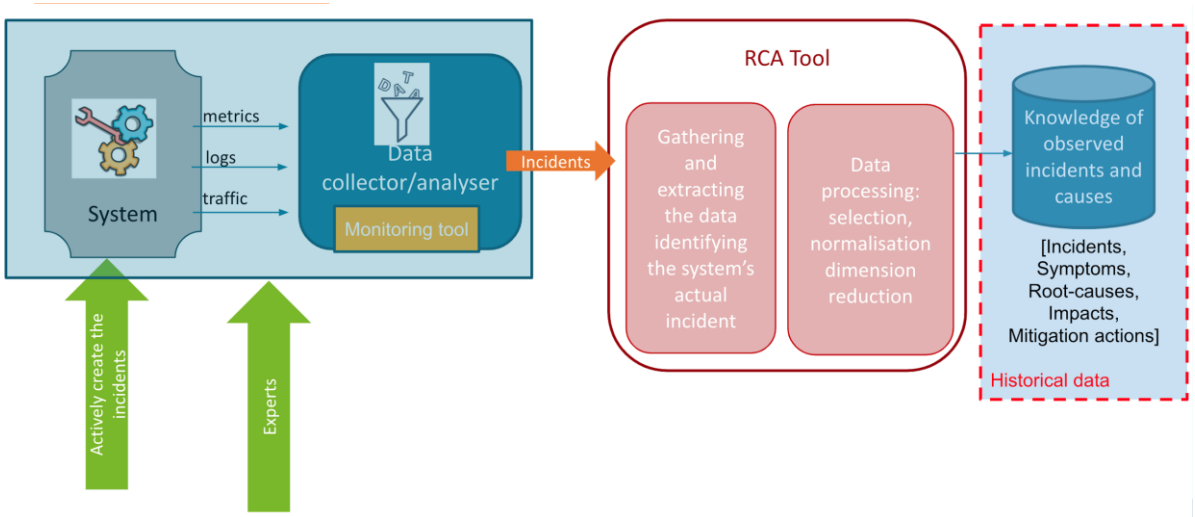

Figure 5 RCA - M Knowledge acquisition phase

The RCA Enabler works following two phases: the knowledge acquisition phase (Figure 5) and the monitoring phase (Figure 6). The former is for building a historical database of known problems and incidents. The latter consists of monitoring the system in real-time, analysing the newly-coming incident by querying the historical data, and suggesting possible root causes. It is worth noting that passive learning in the knowledge acquisition phase can be continuously run during the monitoring phase.

Analysing a system requires different statistics and data, i.e., the logs, metrics, network traffic, and any data that could identify the system state. A **data collector** is necessary and can be provided by the system, or an enabler can be deployed to collect different types of data, namely:

- Capturing network traffic: For example, MMT-Probe (TCP/IP networks) and MMT-IoT (IoT-6LoWPAN) are able to sniff and record the network traffic.
- Reading and extracting logs: The current version of the RCA enabler supports by default reading the data input in the form of JSON and CSV files. Other formats can be rapidly taken into account thanks to the extensibility of MMT-Probe (e.g., creating new plug-ins).

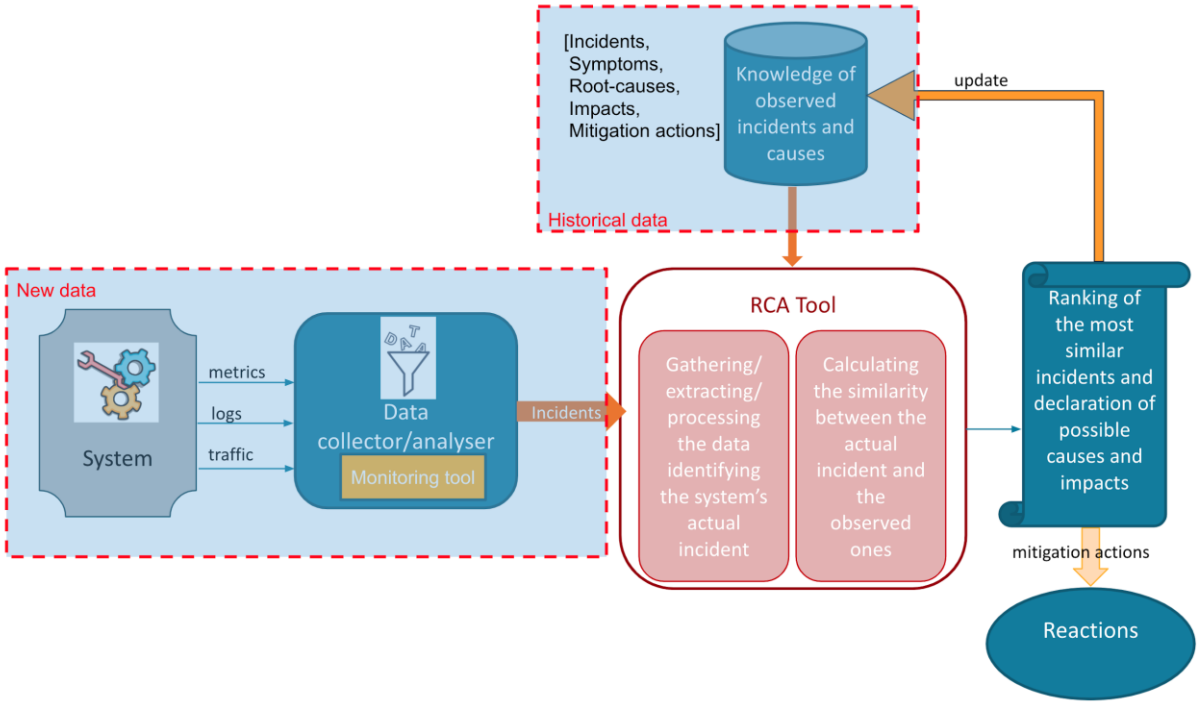

Figure 6 RCA - M Monitoring phase

In the knowledge acquisition phase, the data can be collected in two ways:

- Actively injecting or reproducing known failures and attacks, then collecting labelled corresponding data.
- Passively monitoring the system, debugging different logs and traces, correlating various events and particularly by consulting the system experts to determine the corresponding root causes as well as the relevant data.

In the monitoring phase, the data is collected and transmitted to the RCA enabler in real-time. In theory, there is no restriction in the type of data to be gathered. On the contrary, a maximum of data for identifying the system functionalities is desirable. Even though some data could be redundant, data processing steps are performed to extract the most pertinent data.

4.2.4 Efficiency factors

First and foremost, as a Machine Learning-based solution, **the statistics and monitoring data** that can be collected from the system impact tremendously the efficiency of the tool. It consists of the learning dataset during the off-line knowledge acquisition phase and the data recuperated in real-time during the monitoring phase. RCA requires enough relevant monitoring data attributes and significant domain/system knowledge that can reflect the changes in the under-monitored system. For example, a system can exhibit similar symptoms of two unrelated failures. We need, thus, a higher level of granularity in the monitoring indicators and a deeper analysis to distinguish two unrelated failures. Besides, in some specific systems, the data may not be and have the same frequency as other information RCA uses for the diagnosis (e.g., IoT battery-powered devices may have a low-activity mode to extend their operation autonomy. In this mode, sensed data may not be synchronized). Therefore, RCA must be able to deal with out-of-order data.

Second, **attribute selection** (also known as feature selection [21]) is one of the core concepts in Machine Learning that tremendously impacts the model performance. For complex systems, it is common that the data collected is too complicated or redundant. In other words, there might be some irrelevant or less important attributes (i.e., noises) contributing less to the target variable. Removing

the noises helps not only to improve the accuracy but also to reduce the training time. It is the first and most essential step that should be performed automatically based on the feature selection techniques or manually by system experts.

Third, the data used as the input of RCA is normally heterogeneous. A step of **data normalization** is needed for eliminating measurement units and making the attributes comparable despite different value ranges. The way RCA normalizes data input is also an important efficiency factor.

Finally, our RCA approach relies on similarity learning to identify the most probable cause(s) of detected anomalies based on the knowledge of similar observed ones. The accuracy of the results depends on the algorithm used for calculating the **similarity score**. Thus, computing the similarity score based on more than one similarity and distance measure will help improve the confidence and precision of the results. In the training phase, we determine the measures. Therefore, when we compare a known state and its repetition, we have a similarity score as high as possible. Besides, to avoid false positives, the similarity between a common proper state and each known malicious state should be as low as possible.

4.2.5 Development outcomes

The enabler consists of the Python libraries running in the background for receiving and analysing the monitoring data and the NodeJS / Node-RED GUIs for visualizing the collected statistics as well as the analysis results. The videos listed below will demonstrate how the dashboards look.

Videos:

- *Playlist (will be kept updated with new videos):*
  `https://www.youtube.com/playlist?list=PL8Qwr001wPnY4lsCBzuBv0h-xilmTfg8z`
- *RCA-Principal idea (September-2020):*
  `https://youtu.be/0Hixis8QXo0`
- *RCA Dashboard (Beta version- October 2020):*
  `https://youtu.be/Ri4EU66qb4A`
- *RCA ITS (March 2021):*
  `https://youtu.be/5eU2oSNIArU`

In addition, in the following table we provide details about the API exposed by RCA that enables the integration of RCA with other enablers. A set of high-level commands are exposed by RCA in the form of a REST API.

Table 4 RCA-M API

Method	Resource	Content-type	Description
GET	/rca/getall	Response: application/json	To retrieve all the records (known incidents) stored in the historical database
GET	/rca/getone	Response: application/json	To retrieve a record (a known incident) stored in the historical database identified by its ID
POST	/rca/insertone	Response: application/json Parameter: application/json	To report a newly detected incident with all the related traces so that it could be inserted to the historical database
GET	/rca/logs	Response: text/plain	To retrieve all the logs of the tool

GET	/rca/status	Response: application/json	To retrieve the current status of the monitored system and how it looks similar (in percentage) to the known incidents
GET	/rca/help	Response: application/json	To retrieve the API documentation
POST	/rca/update	Response: application/json Parameter: application/json	To push an update regarding a record (a known incident) in the historical database
POST	/rca/deleteone	Response: application/json Parameter: application/json	To delete a record (a known incident) in the historical database
POST	/rca/deletemany	Response: application/json Parameter: application/json	To delete multiple records (known incidents) in the historical database

4.2.6 Implementation

Dependencies: `https://bitbucket.org/montimage/mmt-rca/src/master/root_cause_analysis/package.json`
Installation: `https://bitbucket.org/montimage/mmt-rca/src/master/root_cause_analysis/README.md`

4.2.7 Associated publications and patents

A paper is under redaction. It will assess the utilization of P4 for programming the network switches and for generating more useful metrics/statistics to improve the performance of RCA Enabler.

4.2.8 Residual challenges

The most evident open challenge is the scalability of RCA. In many practical cases, the number of components/indicators to be taken into the analysis could be enormous. This can lead to a big volume of data processed. Similarly, the historical database of learned events can become remarkably more voluminous and sophisticated. It raises a natural concern about the response time of the tool.

Besides, it is also important to note that failures usually propagate in complex systems through causal chains and produce evolving fingerprints of noisy symptoms. One of the first tasks to accomplish for an automated tool helping humans troubleshooting a system is to group events that are causally connected and keep unrelated events separated. Achieving this is often not straightforward since components of a system can exhibit similar symptoms of two unrelated failures. The failure can also present symptoms very similar to a normal activity. For example, when we no longer receive sensed data from a sensor, it is difficult to determine whether the sensor has moved out of the range (normal activity); or the communication has failed (failure). To deal with this problem, we need a higher level of granularity in the monitoring indicators and a deeper analysis to distinguish the different cases.

4.3 MANIFEST

4.3.1 Description of problem and challenges

Liability-Aware Orchestrator-Manager (Task 4.4 enabler: LASM) needs to have some input data on which to reason and take decisions related to liability. These data will highlight the levels of responsibility of each actor and their commitments made through a network component. However, existing manifests do not explicitly render responsibilities. We propose liability-aware MANIFESTs to fill this gap for liability and security integration. The challenge will be to define several levels of responsibilities and ensure ownership of the defined responsibilities [22].

The main challenge is related to liability language definition and level of commitments and is always under investigation. Another issue is the dynamicity of component context and chaining of different stakeholders and stakeholder's responsibilities that will have to be carefully investigated in Task 4.4.

WP4 grand Challenges covered by MANIFEST approach:

- Agent's responsibilities and liabilities imputation/charging in the case of violations and security incidents in virtualized environment.
- Pervasive and granular behavioural definition and monitoring for liability virtual services and IoT under massive connectivity
- 5G network resilience

The aim of MANIFEST is to describe and dynamically allocate or describe responsibility between stakeholder or SSLA/commitments by each party of production chain. It will have to be populated and managed by a Liability Management system like LASM, and in particular interconnected with the different RCA systems of the project. The MANIFEST alone could not deliver or comply with this grand challenge, but as a dynamic and contextual Identity Card of a component, MANIFEST become a critical enabler to achieve those Challenges.

4.3.2 State of the art

In the 5G ecosystem, several profiles and manifests coexist implicitly. A VNF provider can describe a VNF in terms of deployment requirements (resources, connectivity and interface) and operational behaviour through a VNF descriptor (VNFD) [23]. The Network Service Descriptors (NSD) [24] is a deployment template that describes a network service's deployment and operational requirement. It is used by the Network designer to create a network service.

The SUIT manifest provides a unified update format to enable unified IoT firmware update management systems and define behaviour of IoT device performing updates to ensure predictable results [25].

Recommendations for control on a Component's usage can also be expressed as done by the Manufacturer Usage Definition [26] (MUD) profiles for IoT devices. MUD profiles do not describe properties but recommendations. This implies that the Manufacturer does not take any responsibility for implementing controls on the Components, but it is up to the component user (IM) to enforce them.

Neither of these manifests formalizes a commitment. They contain descriptive information to help the initialization and the life cycle management of the component. The manifest we propose is a single document which describes the commitments and responsibilities of the stakeholders involved throughout its lifecycle.

4.3.3 Description of the solution

The proposed manifest is modular and follows the 5G infrastructure component throughout its life cycle, as depicted in Figure 7.

During the manufacturing phase, the Component Provider builds the component by using the building blocks provided by software editors, hardware manufacturers or Service Providers. The Component Provider provides a first version of the manifest based on the description of features and preliminary usage recommendations. Then, the Validator tests the component, evaluates risks and compliance to applicable requirements. Based on its observations, it can add properties or describe controls or requirements, called usage constraints that need to be enforced by the Slice Provider (SP) to guarantee normal functioning or avoid exploitation of a known vulnerability. At the end of these steps, the manifest contains the description of a class of Component.

Afterwards, the SP lists the Component in its Catalog and may perform additional tests. It identifies operation constraints, similar to usage constraints, except that they express conditions to comply with specific infrastructure requirements, company policy or local regulation. An enhanced Component Class Description is produced.

Finally, The SP uses the manifest to decide how, where and under which conditions a Component should be deployed. After putting the Component in service, the SP uses the Manifest to decide whether and how to observe and manage the Component. It can also be used as a baseline to define expected behaviour for monitoring. At this stage, the Component Class Description is amended with a Component Instance Description which provide instantiation details as well as observed KPIs.

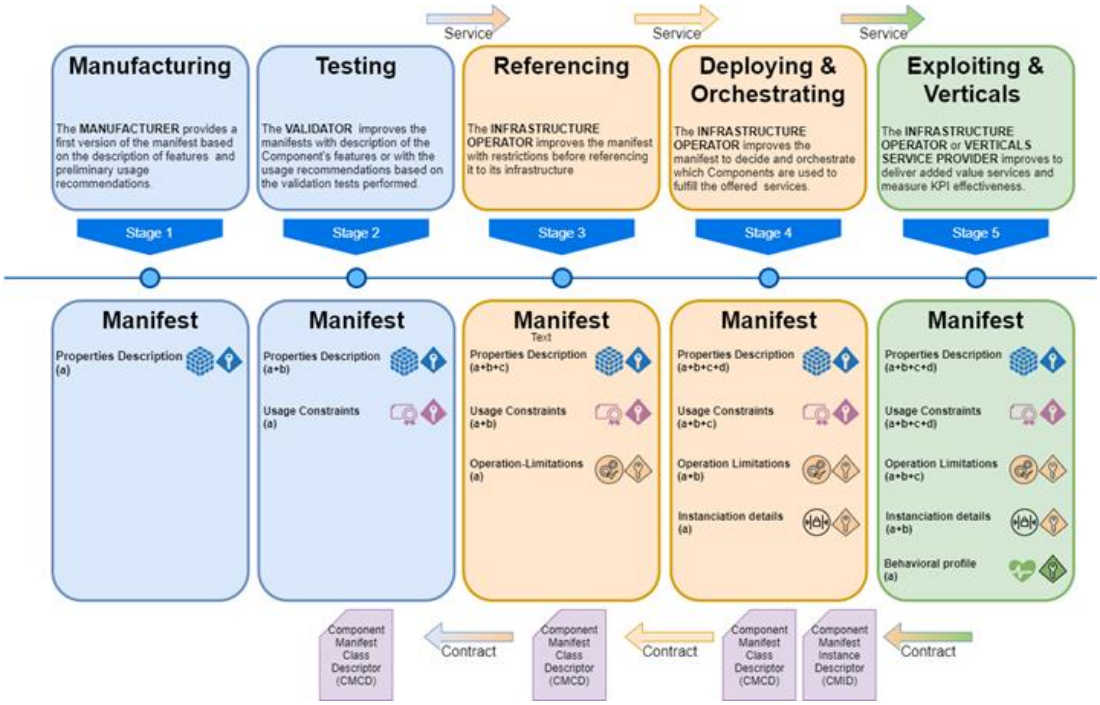

Figure 7 Different stages and Manifests

Thus, each stakeholder is able to express its commitments and expectations to/from other parties. In turn, this will help SPs to formalize their risks and take decisions in order to manage the level of risk of their infrastructures. In the future, we will propose organizational and technical solutions to allow the SP to publish and share manifests that it enhanced as part of LASM system.

As depicted above, the manifest is enriched at each step with new information which describe the guarantees of the service offered by the component at this stage. At each stage, the manifest serves as a support to establish a contract.

The manifest highlights the responsibilities expected from each step's provider and consumer. The current version focuses on:

- **Features Properties:** describe the properties of the network component in terms of use and security. The stakeholders are responsible for the exactness and completeness of the descriptions. For example, if the manufacturer states that its IoT device has a Bluetooth interface, then the product should have a Bluetooth interface and not a Wi-Fi interface. Similarly, if the manufacturer does not declare a Wi-Fi interface, then the product that he delivers must not contain a Wi-Fi interface. Such an interface can be added by an integrator at a later stage under his own responsibility.
    - For this, the integrator can either add a new feature on top of the ones committed by the manufacturer or take ownership of the commitments taken by the manufacturer. In the first case, the manifest contains two feature-description objects (the first signed by the manufacturer and the second signed by the integrator). In the second case, the manifest contains only one feature-description object with all the features and is signed only by the integrator.
- **Usage constraints**: Advices that the stakeholders in the supply provide on how to use the network component. There is no commitment on these elements, and it is up to the user/consumer of the network component to comply with these advices while using it. However, the usage constraints described here can be used by actors in the supply chain to demonstrate they have duly informed users of their products/services. Applying the controls and set ups described by these usage constraints are up to the users. The responsibility of the supply chain stakeholders is limited to providing advices that effectively address the limitations and vulnerabilities.
    - Similar to features, usage constraints can be written by multiple actors.
    - We chose to include usage conditions rather than vulnerabilities description to avoid providing too much information to potential attackers.
- **Operational limitations:** restrictions imposed by the infrastructure administrator before referencing the component in the operator's catalogue. These restrictions are based on internal security policies and can concern any component of a 5G ecosystem. Its author is responsible in case these restrictions lead to a limited functioning of the component. For example, a Validator identify a vulnerability in an IoT device and add a patch recommendation in the usage constraints of the manifest, the administrator can refer to this recommendation and imposes the correction of the vulnerability before deploying the IoT object.
- **Header:** the metadata of the manifest. This information is present for descriptive purposes. It concerns the manifest and the component being described. The LASM can use this information to choose the right component for each one available in the catalogue, for example, in order to add traffic control, it could select a security-traffic-control type component.
- **Authors:** lists the authors involved in the creation of the manifest and the certificates of the keys used to authenticate their endorsement of the claims and recommendations. Authors are responsible for the correctness of their statements. For example, if an IoT contains a Wi-Fi interface that was not declared by the Component Manufacturer, this may have an impact of the reputation of this actor.
- **SignedFiles:** contain information on the signing process. This process is crucial because it will allow to materialize the commitment of each actor. The authentication, integrity and non-repudiation properties will be satisfied, so each author will not be able to deny having committed to the descriptions made. A PKI infrastructure is required for the generation and storage of keys and also for the verification and authentication of the validity of the different actors involved in the signature process.

4.3.4 Efficiency factors

Our goal is to provide a manifest which:

- Identifies the actors which take a responsibility in the lifecycle of the Component
- Expresses the properties that can be expected from the Component
- Expresses the constraints that the user is expected to apply in order to use the Component in a way that avoids security or performance issues,
- Expresses the operational constraints that a user may choose to apply to the Component in order to take into account a specific context linked to the infrastructure where the Component is integrated
- Demonstrates the authenticity of the claims made by authors. In particular, it is necessary to ensure the integrity of the assertion and to authenticate its author in order to materialize its commitment to the claim.
- Can be used generically for any component of the 5G ecosystem
- Is modular to evolve at each step of its lifecycle and allow multiple stakeholders to take commitments independently.

Another way to evaluate efficiency of MANIFEST will be potential and demonstrable interaction between MANIFEST structures and other enablers of the project, in particular RCAs and RAG enablers for WP4.

Table 5 Comparison of MANIFEST with other manifests

Criteria	VNFD	NSD	MUD profile	MANIFEST
Identify actors	▢	▢	▢	■
Express properties	▢	▢	▢	■
Express usage constraints	■	■	—	■
Express operational limitations	—	—	—	■
Authenticity of claims	—	—	—	■
Genericity	—	—	—	■
Modularity	■	■	■	■

In Table 5, the black square represents a feature supported by the manifest, the empty square represents a partially supported feature and — means it is not covered. As can be seen in our comparison, only our MANIFEST meets all the above criteria.

4.3.5 Development outcomes

We released a format of manifest which describe a class of Components and several examples to illustrate its use. We also developed services in the Liability-Aware Security Manager of T4.4 which use this manifest.

4.3.6 Implementation

The manifest format is expressed in JSON. It partially uses the YANG-based MUD formalism.

The manifest is consumed by two LASM services developed in T4.4. The first one manages the slice provider's catalogue of network components. A referencing policy decides whether a Component is rejected or accepted and whether operational constraints have to be added to the manifest. The

second service manages the KPIs calculated for a given instance of the Component and adds them in the Manifest Component Instance Description.

4.3.7 Associated publications and patents

The scope of the research made for the manifest led to an article with the title “Energy-aware Service Level Agreements in 5G NFV architecture” that has been written and published. The main goal of this article is to allow the entity in charge of referencing and placement of the virtual network functions (VNFs) for network service, to take into account energy consumption in addition to other pre-existing aspects.

4.3.8 Residual challenges

T4.4 will concentrate on the development of the Component Instance Description part of the Manifest. Current version of manifest is mostly expressed in proprietary format and YANG. The next challenge is to extend TOSCA specification with liability and accountability properties proposed in the manifest. As a result, the manifest would be easily interpreted by orchestrators which widely use TOSCA to describe service deployment as well as routers or network devices which commonly use YANG to describe network configurations. The operational limitations need to be enriched beyond the MUD Threat profile, in order to express conditions and expected countermeasures.

4.4 Root Cause Analysis – VNF (RCA - VNF)

4.4.1 Description of problem and challenges and State of the art

Model-based self-diagnosis relies on a dependency graph which indicates how faults propagate through the network and eventually lead to service failures. In model-based self-diagnosis approaches, an RCA algorithm exploits this dependency graph to calculate the root cause. However, in the state of the art, much more attention is paid to the RCA algorithms in use, to the detriment of the generation of the dependency graph, which in most cases, is manually built from operational team's knowledge. This manual generation is valid for static network topologies and for statically predefined services, but not for dynamic and elastic networks such as those expected with SDN and NFV, where the dependency graph must be continuously updated. This mechanism is coined by Honkonnou et. al. in [27] as self-modelling.

We propose to classify the model-based self-diagnosis approaches into topology-aware and service-aware, which we define hereafter.

Topology-aware approaches: Topology-aware approaches build the dependency graph from the network topology and this graph only considers how faults in network resources could impact other network resources. The impact of faults in network resources on services or clients is not then considered. As examples, Steinder and Sethi [28] propose a self-diagnosis approach for end-to-end services over bridged networks, based on a self-modelling approach to build the diagnosis model from the network topology. However, the authors do not address the impact of the network faults on the service layer. Bennacer et al. in [29], propose a self-diagnosis approach for VPN-based services and utilize a self-modelling approach that computes the dependencies among the physical symptoms of each network node through statistical tests. They proposed a hybrid RCA module based on the combination of Bayesian Networks (BN) with Case-Based reasoning to reduce the high diagnosis time and increase the low accuracy provided by the BN algorithm. The impact of faulty nodes on the VPN service was not addressed. In [30], we proposed a self-diagnosis framework that generated on-the-fly the dependency graph from the network topology and the logical resources running on top of nodes. The impact of faulty physical and logical nodes on services was not studied. As a result, the diagnosis is focused on the entire network topology and logical resources so the uncertainty on the root cause is higher when the diagnosed network topology becomes large.

Service-aware approaches: We define service-aware approaches as an extension of topology-aware approaches in order to take into account the impact of faulty network re-sources on services as well as the clients using them. The diagnosis can then focus on faults leading to service failures and impacting on client experience. As examples, Bahl et. al. proposes a self-diagnosis algorithm for IT infrastructures [31], based on a self-modelling approach that computes the service dependencies in a given network topology when clients access to some database in the IT infrastructure. The diagnosis focuses on faults impacting the clients of the IT infrastructure. Hounkonnou et al. propose a self-diagnosis approach for IMS networks [27] based on a self-modelling approach that utilizes a multi-layered model based on the four IMS layers. First, the self-modelling approach builds the dependency graph from the affected service and its underlying resources over a fixed network topology, and then extends this graph with those clients where service observations reduce the uncertainty on the root cause.

WP4 grand Challenges covered by RCA-VNF approach:

- Dynamic liability and root cause analysis
- 5G network resilience

The aim of RCA-VNF is to find the root cause dynamically concerning SDN networked topologies, where the network topology may evolve due to network reconfiguration and changes. It can detect new network elements when connected to the network topology and instantiate their inner dependencies and connect those to the network dependency graph by regenerating it every time there is a change. Overall, it can contribute to the resilience of 5G networks, as the RCA can identify those networked

elements that are under failure and have to be replaced or disconnected from the network topology (quarantine).

The RCA is key for liability purposes, where there is need to know the identifier of those network elements whose performance is far from normal, due to operational misbehaviour, of intentional ones.

4.4.2 Description of the solution

The self-modelling algorithm depends on the network resources to be modelled. Table 4 shows some examples of the resources to be modelled in softwarized infrastructure. This self-modelling algorithm receives as input the network descriptor coming from the SDN controller. It creates the network dependency based on each network element (nodes and links) found in the network descriptor based on the following steps:

1. Identifies the type of network element (node or link)
2. Instantiates its corresponding template according to the type of element (GN for nodes or GL for links)
3. Instantiates the dependency sub-graph of that network element
4. Appends the instantiated dependency graphs to the network dependency graph following the topological connections interpreted from the SDN controller

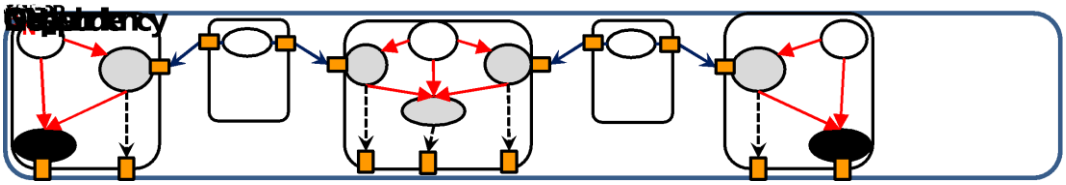

Figure 8 Example of network dependency graph (Q=3 nodes and P=2 links)

Figure 8 shows a very brief example on how the Network Dependency Graph is built for a network topology composed of two hosts connected through a OVS open switch.

Table 6 Types of resources considered per layer

Layer	*Resources*
physical	links, switches, hosts, controllers, ports, NICs, CPU
logical	Flows, controller application, OpenFlow application, VNFIs
virtual	virtual links, VNFs
service	VPN, NAT, firewall, streaming, etc.

Firstly, the network dependency graph in the shape of DAG is generated by the self-modelling algorithm and is then filled with the observations gathered from the following network components as shown in Figure 9:

- CPU load on network nodes
- state of switches' ports,
- state of SDN controller's ports,
- state of hosts' network cards,
- state of the SDN controller application,
- state of the OpenFlow client applications running on switches, of VNFs on hosts if any
- state of the video streaming application running on the clients and on server

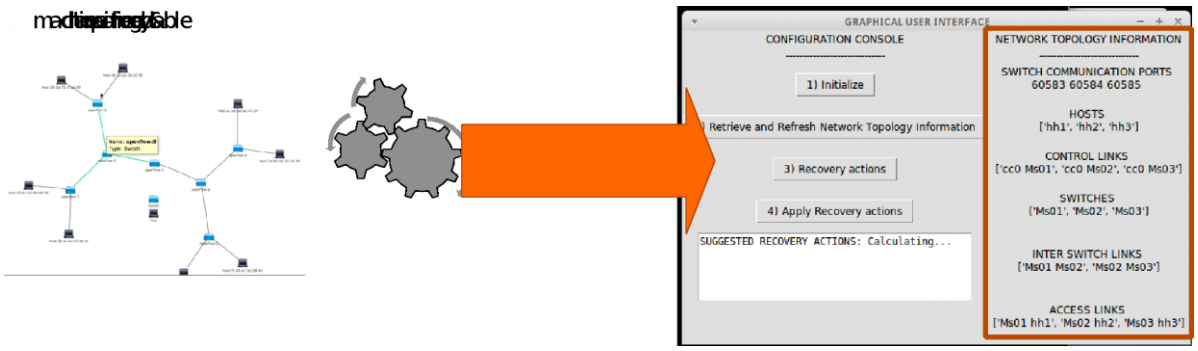

Figure 9 Transformation of the network topology into a machine-readable format

The scenario of the demonstration is shown in Figure 10. A new client demands the video content to the streaming server (1) [52], which starts sending it. However, for this content to reach the client, the SDN controller must install the necessary flows on the switches (2). The GUI monitors the current network topology provided by the SDN controller in a periodic basis and it classifies the network elements (3) into different nodes (hosts, switches, and controllers) and links.

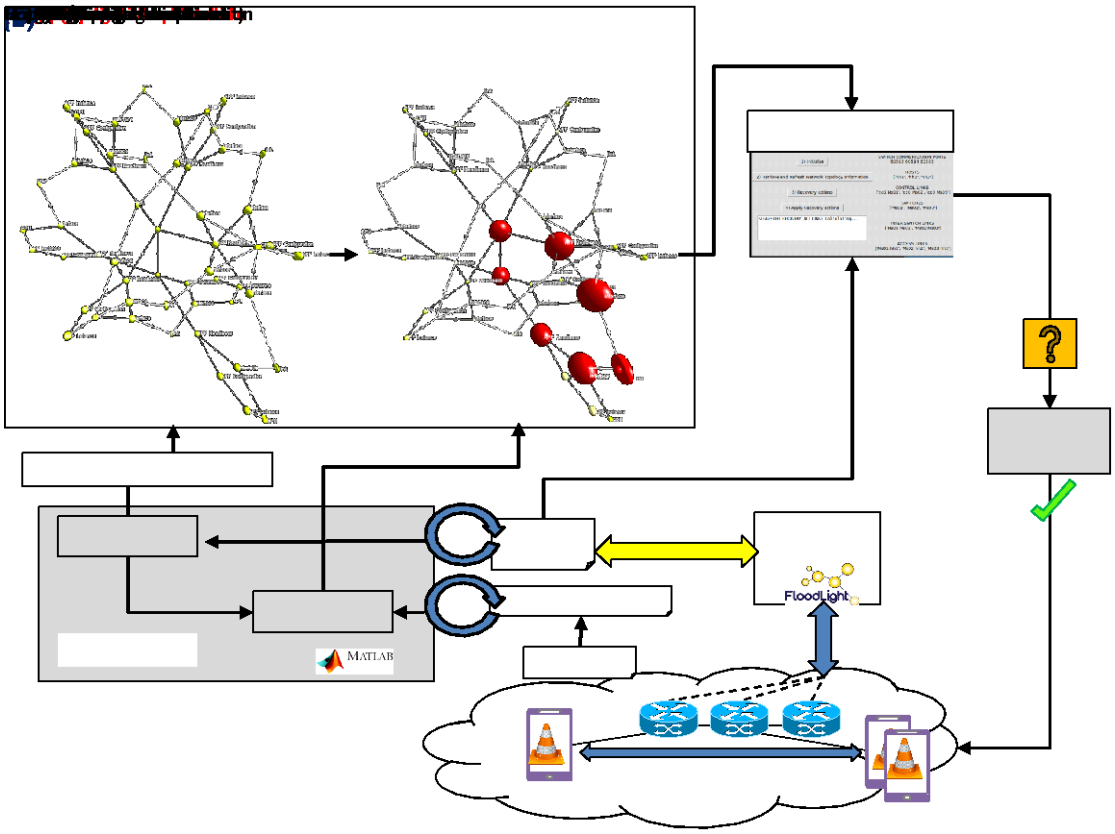

Figure 10 Implementation of the self-healing framework

The workflow of the testbed for a streaming oriented scenario shown in Figure 9 is as follows: A new client arrives and demands the content to the streaming server **(1)**, which starts sending the content. The SDN controller installs the necessary flows on the intermediate OpenFlow switches **(2)** to allow the streaming packets coming from the content server to be sent to the client and vice versa. Then, the Graphical User Interface (GUI) retrieves the network topology from the SDN controller and classifies the network elements in hosts, switches, links and logical ports, which are shown in the GUI **(3)**. The self-modelling algorithm generates the fault propagation model **(4)** from this classified list of network elements by instantiating their corresponding templates and assembling their respective dependency graphs, as explained in the previous chapter. If the network topology changes, the self-modelling block regenerates the fault propagation model by incorporating the newly added elements.

Once the model is generated, the root cause analysis module will be triggered by the alarms indicating a malfunction in the streaming service and it will update the fault propagation model **(5)** with the root cause(s). It indicates as root cause a network node but also its internal component (CPU, port, card, application, etc.). Once the root cause is identified, the root cause analysis block suggests a recovery action **(6)** that will be validated by a human administrator **(7)** once is proved it re-establishes the streaming service.

4.4.3 Efficiency factors

The challenges covered by our enabler are mainly two-fold, as specified below:

- Ability of the RCA to grasp the network structure (model representation) ever evolving grasp network infrastructure
- Devise the most relevant learning and diagnostic methods-approaches with a special focus on Deep learning

In the following part, we will elaborate on how those threats or capabilities are tackled by our enabler.

Ability of the RCA to grasp the network structure (model representation) ever evolving grasp network infrastructure: We propose a self-modelling algorithm that can grasp the network structure in the shape of dependency graph (DAG).

Figure 11 can show the efficiency, expressed in this case by the time taken by the RCA algorithm to generate the DAG corresponding to two different network topologies (linear and tree) both with an increasing number of networked elements.

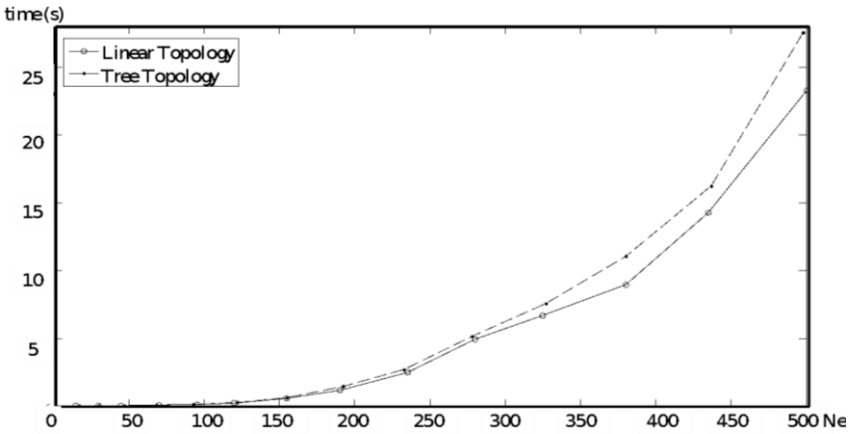

Figure 11 Speed as a function of the number of elements

Concerning the number of vertices generated on the DAG, we can see how those grow exponentially as a function of the number of hosts in both network topologies, tree and linear, in a similar way in Table 7.

Table 7 Number of vertices (V) as a function of the number of hosts (N_H)

Topology/ N_H	**4**	**8**	**16**	**32**	**64**	**128**	**256**
Tree	62	130	266	538	1082	2170	4346
Linear	72	140	276	548	1036	2180	4356

Devise the most relevant learning and diagnostic methods-approaches with a special focus on Deep learning: The RCA block is in charge of identifying the root cause with the Bayesian networks approach to calculate the probability of faulty elements in the current network topology.

The root cause is calculated by propagating those network observations onto the DAG. As it can be seen in Figure 12, the RCA provides with the same DAG but the vertices are coloured and resized according to their "a posteriori" probability calculated by the RCA for each network component. For instance, the redder a vertex and the bigger is, the higher its root cause probability is, contrarily, the greener a vertex and the smaller is the lower its a posteriori probability is. Figure 11 shows how the root cause focuses on a subset of the network components, depicted in red and bigger sizes, with respect to the rest of network components, which become smaller and greener.

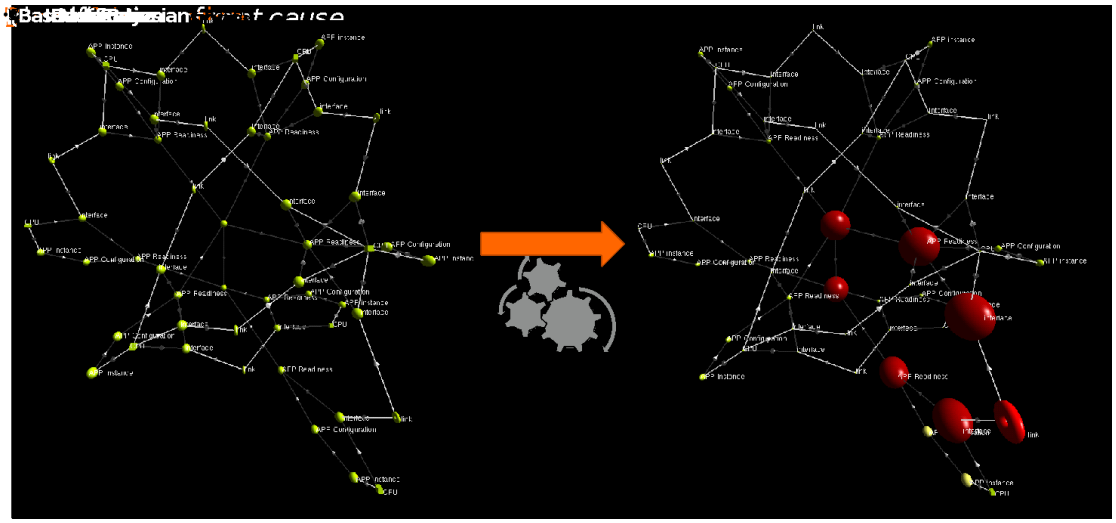

Figure 12 Root Cause Analysis process

Two scientific publications concerning this enabler demonstrate a methodology to grasp the network structure (topology-aware) in the article [30] and virtual links on the network structure (service-aware) in the article [32]. This network structure is given in the shape of DAG (directed acyclic graph) to later diagnose the root cause with Bayesian Networks. It can also detect topological changes and re- create from scratch the network graph taking those into account.

The demo paper and corresponding video [33] demonstrate how we can detect the network topology form a single SDN controller, detect topological changes and regenerate the graph when those changes are detected.

4.4.4 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.

4.4.5 Implementation

The RCA-VNF integrates different open-source software packages. The SDN controller is based on Floodlight [46] and the SDN infrastructure is emulated with Mininet [47] The Bayesian Network algorithm is based on the Kevin Murphy's Bayesian Networks Toolbox [48], running in MATLAB [49]. We implemented the Graphical User Interface (GUI) in Python with the Qt software library [50]. The fault propagation model is visualized in 3-D with UbiGraph [51], 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
- License: private license (internal process to externalize as open-source code under BSD license)

4.4.6 Residual challenges

The most important challenge to be considered consists of the lack of scalability of RCA, in particular, the Bayesian Network inference algorithm as it scales badly when the number of network elements increases. An example was Figure 11, with 30 seconds for 500 network elements.

4.5 GRALAF

4.5.1 Description of problem and challenges

In 5G systems as well as future networks, it is paramount to address the question of liability and responsibility for security incidents, faults and failures in 5G networks. In such a distributed and multi-party context, novel solutions for defining liabilities and detecting the causes of security breaches need to be developed for ensuring liable end-to-end delivery of 5G services. Liability management mechanisms need to be devised by exploiting graph-theoretic techniques to dynamically investigate the coverage of security objectives by systems in 5G networks. Such a dynamic liability analysis tool will be developed (for the multi-tenant environment case) and analysed for their usability and impact on security, the effectiveness of the dynamic approach on liability, and the compliance with the regulatory policies in future networks.

- **Grand challenges:** Agents' responsibilities and liabilities imputation/charging in the case of violations and security incidents in virtualized environment.
    - **Global target:** GRALAF will help to address these grand challenges by analysing and identifying the root cause and liability for security incidents in the virtualized communication environments.
- **Grand challenges:** Dynamic liability and root cause analysis (integrating provability, accountability and delegation concepts).
    - **Global target:** GRALAF will dynamically determine the liability of agents in 5G and future networks by analysing incidents that impair the operation using graph-based approaches.

4.5.2 State of the art

One related work is [36], where an Information System Security Risk Management meta-model including responsibility, accountability and commitment was used to create a multi-agent system-based architecture for broadcasting forecasts and alerts in a power distribution infrastructure. In [37], an adaptation of this model was proposed for a decision mechanism for incident reaction in telecommunications network, but it is not adapted for the 5G context.

A Security Panel was proposed in [38] as a platform regrouping risk managers and experts throughout the eSIM ecosystem and allowing them to collect the information required for their risks analysis. Giaretta et. Al. propose to use Security-by-Contract paradigm for fog-based IoT management [39]. The decision to add an IoT device in the local network, update or monitor it is taken by matching the IoT device's manifest with a security policy. Costa et. al. [40] show that Security-by-Contract paradigm can be extended to include models and KPIs for quantitative trust management. However, responsibilities are implicit. In [41], Lin et al. proposes a solution that captures anomaly propagation among different tenants.

For liability analysis, there are several advantages to causal models, in particular, graphical causal models [42]. First, a causal model provides a deeper understanding of a domain, by identifying the direct and indirect causes of certain events, which is not necessarily true for associative models such as Bayesian networks. Secondly, with causal models we can perform other types of reasoning that are not possible, at least in a direct way, with PGMs such as Bayesian networks [43]. These other inference situations are: (i) interventions, in which we want to find the effects of setting a certain variable to a specific value by an external agent (note that this is different from observing a variable); and (ii) counterfactuals, where we want to reason about what would have happened if certain information had been different from what actually happened. Being capable of statistical reasoning and probabilistic inference, graphical models have the advantages of encoding prior knowledge into the learning procedure, and producing explainable models that can be understood and effectively tuned [44].

4.5.3 Description of the solution

GRALAF extracts the network model from the network segment to be analysed based on the test case as shown in Figure 13. In GRALAF, a Causal Bayesian network (CBN) is used to model the network assets, in which each node represents a variable and the arcs in the graph represent causal relations; that is, the relation A → B represents some physical mechanism such that the value of B is directly affected by the value of A.

The liability model will be integrated with the network model to generate the system model. Since GRALAF will focus on MANIFESTS, one of the inputs will be those information sets. The exact MANIFEST fields to be digested will be determined in the ongoing project work (also related to T4.4 work). The security agents and monitoring system in the infrastructure will feed the network data well as trigger the liability analysis actions once incidents occur. The inference algorithm will process the graphical model and determine a list of liable entities with a likelihood score and liability value. The model will be updated according to the precision and accuracy of the output liability analysis.

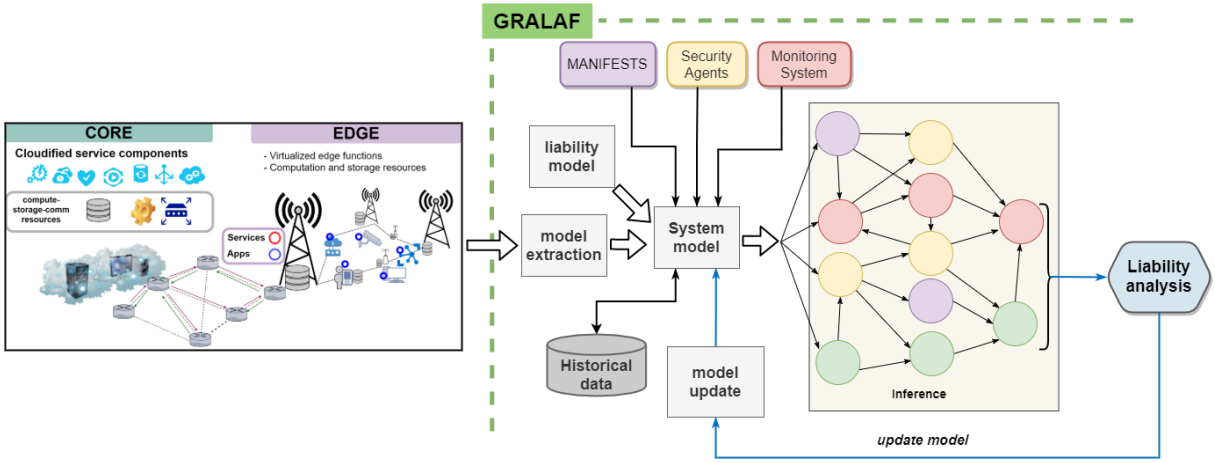

Figure 13 GRALAF system model

4.5.4 Efficiency factors

This tool is planned to be integrated into MANIFESTS in T4.4 and evaluated for the following metrics:

- **Scalability:** This metric will quantify the complexity of the GRALAF with respect to varying problem sizes (hardware resources, dependency links, service components and liability relations). Here the time complexity will be measured as well as theoretical analysis.
- **Precision**: Two candidate metrics are *Precision at top K* (the probability that top K network elements given by GRALAF actually are the root causes of each anomaly case) and *Mean Average Precision (MAP)* quantifies the overall performance of the method over the set of tested anomalies and incidents [35].
- **Response time:** The response time for liability analysis for different tests will be measured for GRALAF. This has to be minimized for efficient operation.

Another way to evaluate efficiency of GRALAF will be via the interaction between MANIFEST and other different enablers of the project, in particular liability and trust enablers for WP4.

4.5.5 Development outcomes

The alpha version of GRALAF is available at: https://github.zhaw.ch/sous/Gralaf. The development of GRALAF is still ongoing based on the collaboration with other enablers and use case implementations.

4.5.6 Implementation

GRALAF is implemented and tested in the testbed setup of Test Case 5 described in D5.2, which provides 2 NFV infrastructures representing the edge and the core networks, an NFV Orchestrator (i.e., OSM), and a network slice manager (i.e., Katana slice manager) managing a simulated operator's network slice. The current implementation of GRALAF ingest monitoring data from the slice manager Katana and the orchestrator OSM, and create a Causal Bayesian Network (CBN) as illustrated in Figure 14.

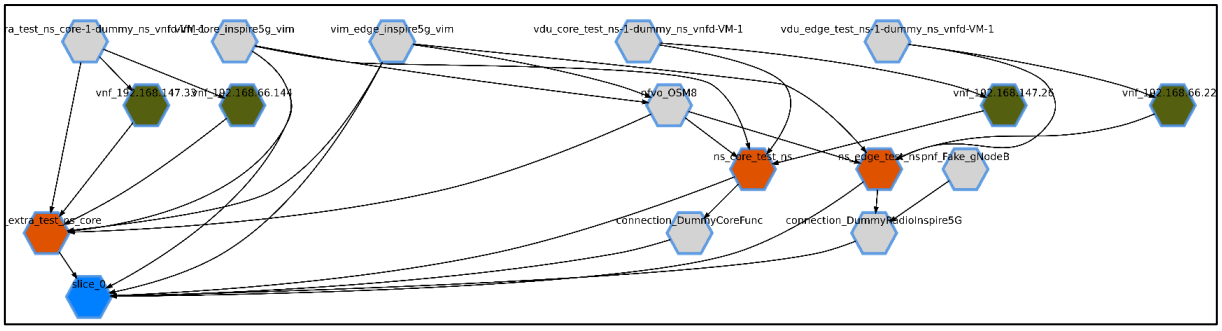

Figure 14 Graphical model (CBN) of a sample testbed (TC5) at a specific time t.

The CBN considers the topology of the infrastructure and founds its causal correlations on the connections of the different network's components on both the logical (or virtual) and physical layers. The former layer monitors the following components: network slices, network services (NSs), VNFs, virtual links, virtualization deployment units (VDUs) and the resources allocated with the VIM (i.e., both NFVIs of the testbed use an OpenStack VIM). The latter layer includes the physical network functions, the radio access network (RAN), NICs, switches, links, and connection ports (currently in development). The broad view of the graph combined with its granularity allows to acquire and identify the failure propagation in an incident, thus dating back to the component that originated the failure.

The CBN is generated every second, allowing to obtain the health status of the overall infrastructure at near-real time and record it in JSON-formatted files. Nodes are displayed with different colours based on the resource type they represent, and edges get thicker on stronger correlations. Nodes have different attributes not displayed in the graph and available only in the JSON-formatted model. The graph is created using the CausalNex Python library[5], which provides an analytics module and a graphical module. The first is used to extract causal correlations between variables of a dataset, learning the graph structure of the CBN, and to improve the graph using the expert domain knowledge. The second allows to customize and display the CBN.

4.5.7 Associated publications and patents

Currently, there are no publications or patents based on GRALAF work. However, a paper is being developed consolidating the liability concepts and providing a research overview related to the work in T4.3.

4.5.8 Residual challenges

The most evident open challenge is the scalability of GRALAF. In many practical cases, the number of components/indicators to be taken into the analysis could be very large. This situation can lead to a big volume of data processed and very complex algorithms to run. Similarly, the historical data and liability model have to be managed for size and be kept concise. Moreover, the representative strength

[5] https://papers.nips.cc/paper/8157-dags-with-no-tears-continuous-optimization-for-structure-learning.pdf, https://causalnex.readthedocs.io/

of the network model is another residual challenge. This is also related to how model generation will be performed. The translation of a 5G environment should be realistic and should include critical elements in the network and service infrastructure. We are planning to cooperate with other partners and utilize existing enablers for system modelling, if possible.

5 Liability enablers integration into INSPIRE-5Gplus architecture

The High-Level Architecture (HLA) of INSPIRE-5Gplus shown in Figure 13 serves a framework to integrate the different liability enablers described in Section 4 in 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 of liability enablers and the identified interactions.

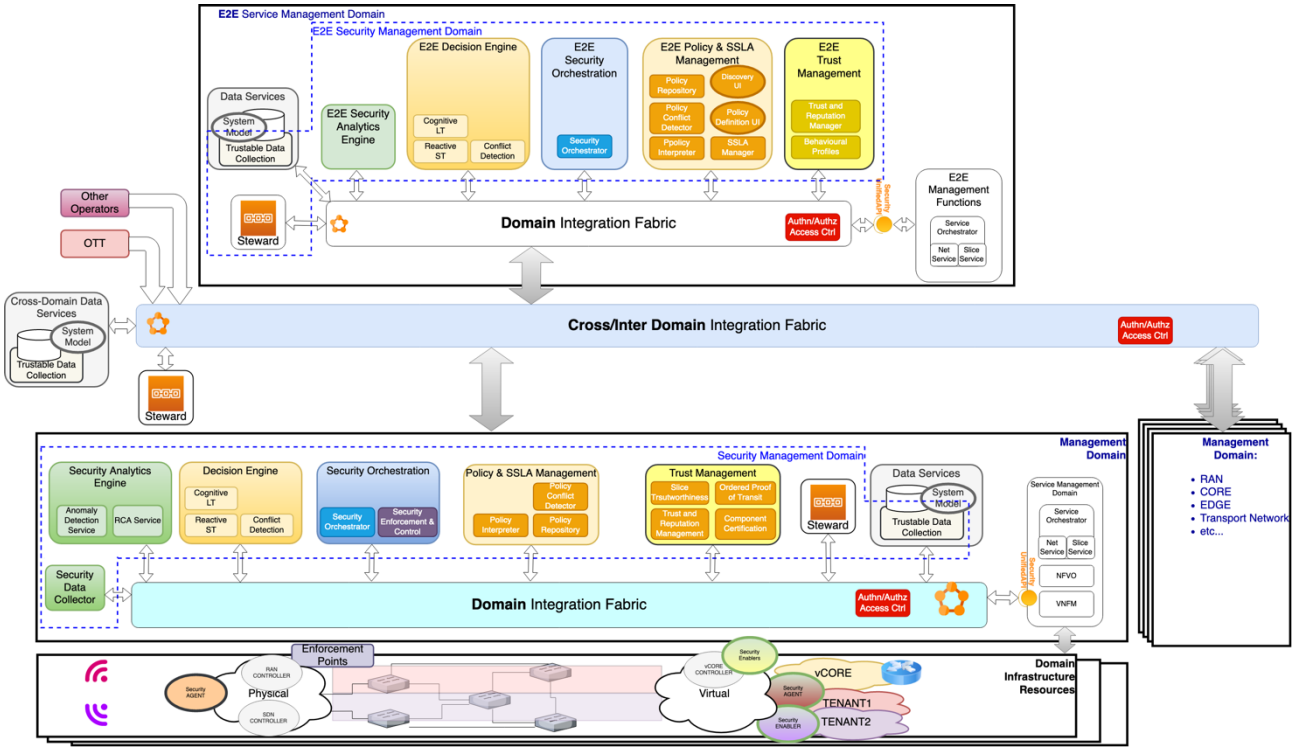

Figure 15 INSPIRE-5Gplus High Level Architecture (HLA)

5.1 Remote Attestation (RAP)

5.1.1 Enabler placement and interactions

The service Orchestrator within the service management domain (see Figure 16) can ask the RA module to run a deep attestation validation on a designated network node. Before that, it can check if the target is RA compliant or request the list of the available and RA-active network nodes. The service Orchestrator can also ask to clean up a give network node from all the data and software related to RA. Finally, the service orchestrator has two options: (i) it gets the result of the RA module and analyses it by its own means and data or (ii) ask the RA module to analyse the results. The second option implies that the RA module has the necessary data to make the analysis (Given by the service Orchestrator, or retrieved on the network from, e.g., Security Data collector, Decision engine, Data services).

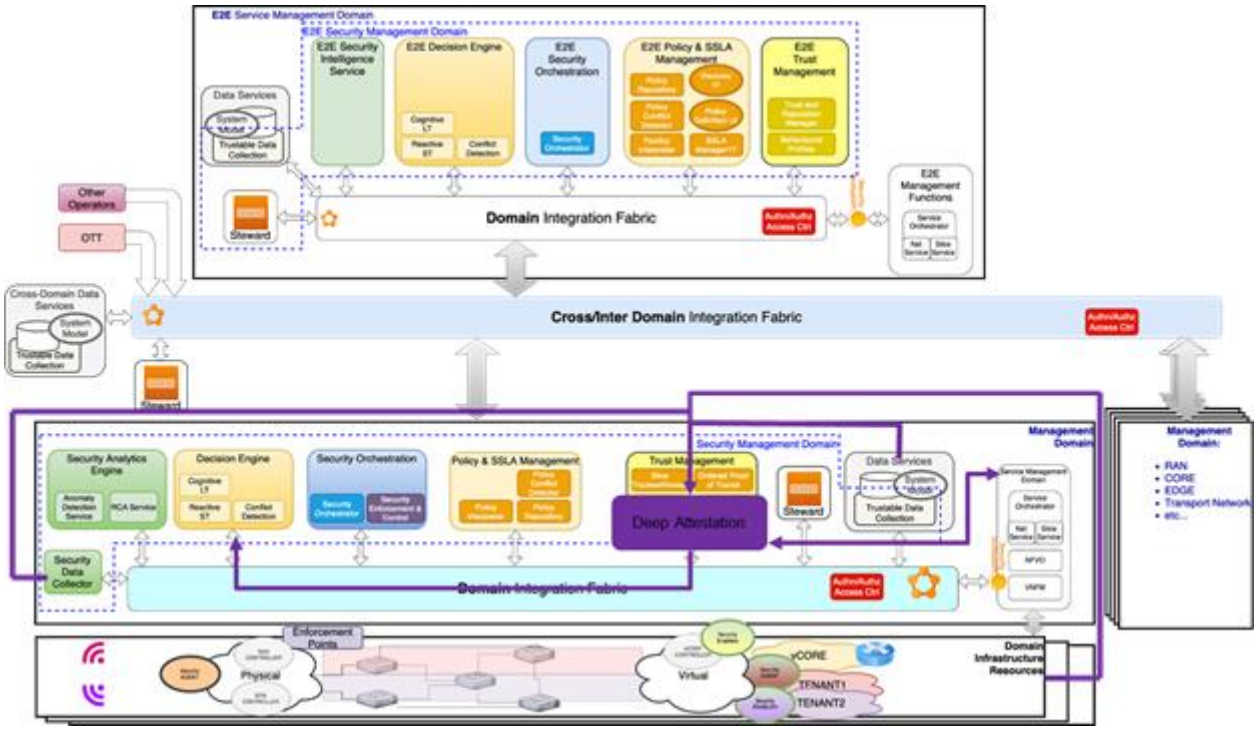

Figure 16 RA placement in INSPIRE-5Gplus HLA

5.1.2 Interface specification

The external RA server interfaces are shown in the Table 6 and in Figure 17. We distinguish two classes of APIs: APIs enabling operations related to targets (e.g., adding a target to the list of targets that can be attested by the RA server, returning the list of available targets, etc) and APIs enabling operations about the attestations (e.g., running an attestation, returning computed attestations about a designated target, etc). Internal interfaces, between RA server and RA agents will be over TLS protocol. Internal interfaces, between RA agents and the underlining hardware will be based on TPM standards [45].

Table 8 Remote Attestation API

Method	URL	Required Data Objects	Returned Data Object
Operations related to targets			
GET	/api/targets/	---	JSON list
POST	api/targets/	{ID, name, host, port}	---

GET	/api/targets/{id}	--	JSON list / error 404
DELETE	/api/targets/{id}	--	Ok (204) / Error (404)
PUT	api/targets/{id}	{id, name, host, port}	Ok (204) / Error (404)
Operation related to attestations			
GET	api/attestations/	--	JSON list
POST	/api/attestations/	{id, name, target_id}	OK (201)
GET	/api/attestations/ targets/{id}	--	JSON list
GET	/api/attestations/ {id}	--	JSON list / ERROR (404)
DELETE	/api/attestations/ {id}	--	OK (204) / ERROR (404)
PUT	/api/attestations/ {id}	{id, name, target_id}	OK (204) / ERROR (404)

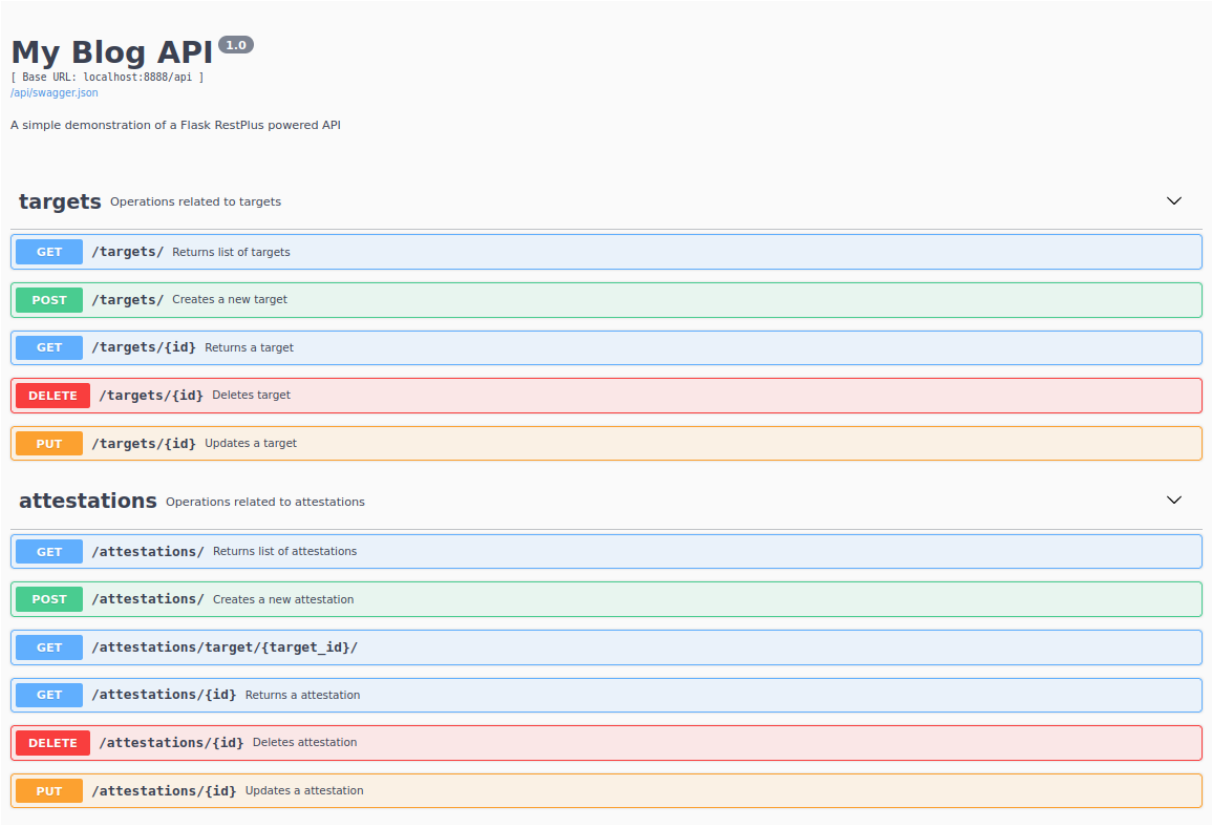

Figure 17 [RA enabler] REST API overview

5.2 Root Cause Analysis (RCA - M)

5.2.1 Enabler placement and interactions

The enabler can be placed in a server, virtual machine, or container. It needs access to the collected data provided by probes, system, and application loggers, etc. Its placement in INSPIRE-5Gplus HLA is shown in Figure 15.

There is no particular advantage of having APIs between the RCA and the security visualization and management applications. There can be several ways that the MMT-Operator can collect reports from RCA: i) RCA generate reports in csv/json that the MMT-Operator reads; ii) RCA writes directly to a collection in the MongoDB of the MMT-Operator; and iii) RCA publishes reports via a communication bus (e.g., Redis, Kafka).

Once the report provided to the MMT-Operator, an external enabler or application can get the results provided by the RCA via the MMT-Operator in a similar way.

5.2.2 Interface specification

Currently the RCA module has not been integrated into the MMT framework. We define the API of MMT-Operator to get root causes in csv or json format sent from the RCA module. The detailed message is yet to be completely specified. It is possible to support all types of communications means, such as through the use of MongoDb, Redis or Kafka servers.

The RCA-M is part of the Decision Engine, as shown in Figure 18. It receives information from the Security Analytics Engine or the SSLA assessment module; historical root cause-related information from the Data Services and stores new information; and finally it notifies the Security Orchestrator so that corrective actions can be taken.

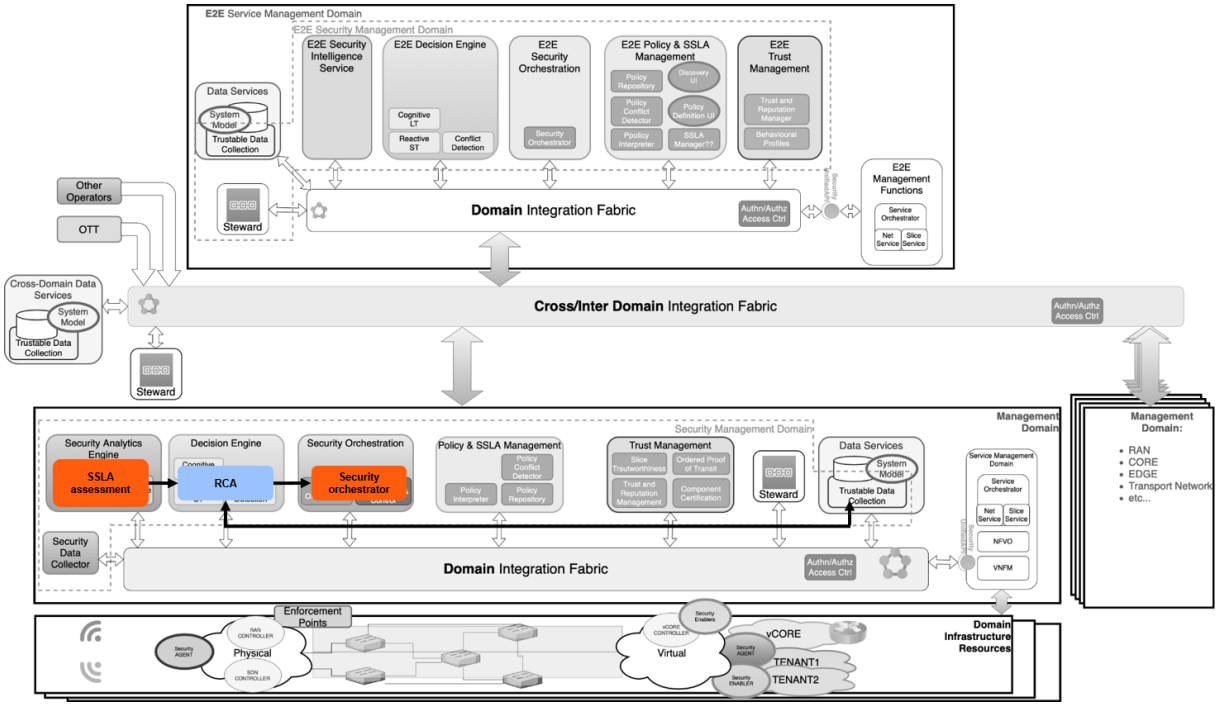

Figure 18 Location of RCA-M in the INSPIRE-5Gplus HLA

The initial version of the API is shown in the following Table 9. The API allows providing the RCA module with all the information extracted and the analysis obtained by monitoring the system, retrieving the all the possible root causes, and interrogating the RCA to determine the most probable

cause of a detected security breach. Optionally, the RCA module can store and retrieve historical information in the Data Store (not included in the API described here).

Table 9 Root Cause Analysis API

Method	URL	Required Data Objects	Returned Data Object
POST	**/api/causes**	Request Body: An array of causes in json (to be defined)	200: OK 400: Invalid input
GET	**/rca/**	None	Return all alerts and the potential root-causes suggested by RCA 200: OK 404: Not found
GET	**/rca?property=propertyId**	Params `propertyId` (type: string): the security alert Id requested by Security Orchestrator `root-cause` (type: json) (to be defined)	Return the root-cause related to the threat in the alert identified by 'alertId' 200: OK 404: Not found

5.3 Root Cause Analysis – VNF (RCA - VNF)

5.3.1 Enabler placement and interactions

RCA-VNF will be integrated into *Security Analytics Engine* in INSPIRE-5Gplus HLA as shown in Figure 19.

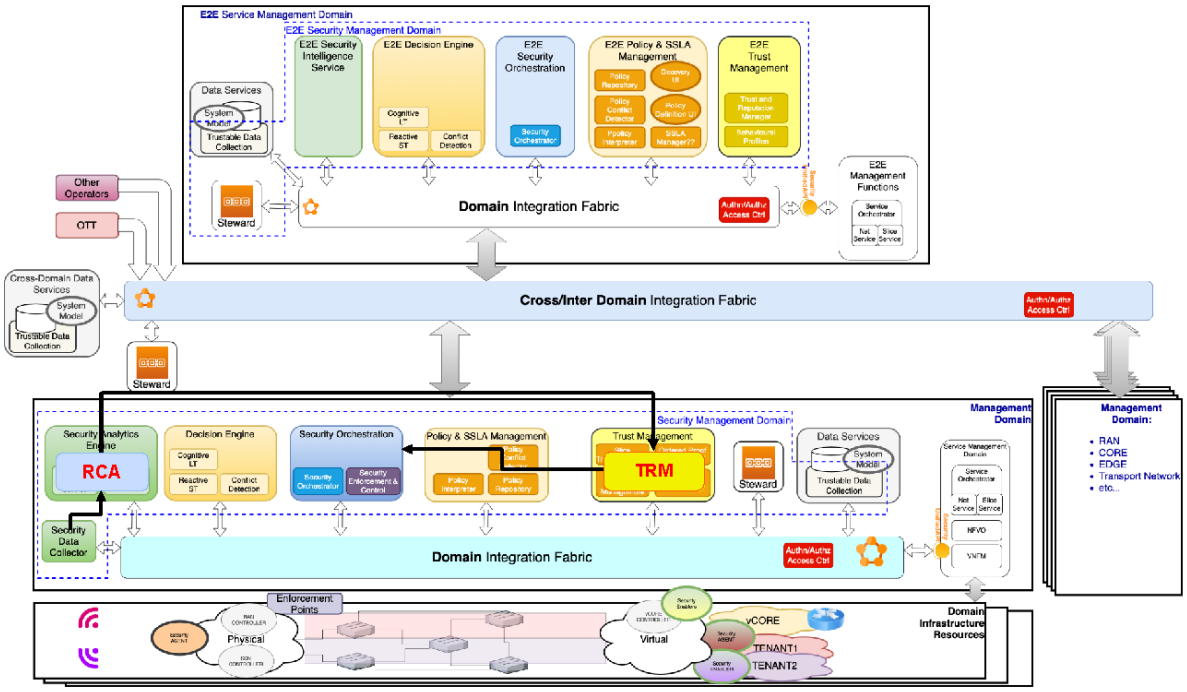

Figure 19 RCA-VNF placement in INSPIRE-5Gplus HLA

The RCA is located inside the Security Engine and gets information from the Security Data Collector and interacts with eTRM inside the Trust Management block. It gets the network topology from the SDN controller and diagnoses the root cause. It sends the network graph and probabilistic graph to the eTRM to compute reputation based on those probabilities.

5.3.2 Interface specification

Table 10 RCA-VNF API

Method	URL	Required Data Objects	Returned Data Object
GET	http:/security/rca/graph/{timeSlot}/networkgraph	–-	JSON object
GET	/security/rca/graph/{timeSlot}/probabilisticgraph	---	JSON object
GET	/security/rca/probability/{timeSlot}/node/{nodeId}	node_id	Probability value
GET	/security/rca/probability/{timeSlot}/link/{sourceNodeId}/{targetNodeId}	source_node_id target_node_id	JSON object

The interface specification is listed in Table 10.

1) The RCA API **provides the dependency graph** of the network topology seen by the SDN controller corresponding to a given timeslot:
    - timeslot: integer describing the current time we would like to get the topology (dynamic)
2) The RCA API **provides the probabilistic graph** of the network topology seen by the SDN controller corresponding to a given timeslot after the RCA is computed over the dependency graph
    - timeslot: integer describing the current time we would like to get the topology (dynamic)
3) The RCA API **provides the probability of fault in a network element root cause** of the network corresponding to a given timeslot after the RCA is computed over the dependency graph
    - `nodeId`: the identity of the network node
    - `linkId`: sourcenodeId, targetnodeId

```
nodes: [
          {
            "type": "controller",
            "probability": [0.5],
            "id": 0
          },
          {
             "type": "switch",
            "probability": [0.5],
            "id": 1
          },
          {
             "type": "host",
            "probability": [0.5],
            "id": 4
          }
]
```

Figure 20 JSON with the returned probabilistic graph object containing nodes (simplified)

```
links: [
         {
           "type": "control link",
           "probability": [0.0],
           "source": 0,
           "target": 1
         },
         {
```

```
            "type": "control link",
            "probability": [0.0],
            "source": 0,
            "target": 2
          },
          {
            "type": "switch-switch link",
           "probability": [0.0],
            "source": 2,
            "target": 3
          },
          {
            "type": "host-switch link",
            "probability": [0.0],
            "source": 3,
            "target": 6
          }
]
```

Figure 21 JSON with the returned probabilistic graph object containing links (simplified)

As additional information to Figure 20 and Figure 21, the concrete YAML file swagger of the RCA-VNF is provided in Annex B.1.

5.4 MANIFEST

5.4.1 Enabler placement and interactions

The MANIFEST is a document which will be consumed by INSPIRE-5Gplus Policy & SSLA management module as shown in Figure 22. The MANIFEST Class will be managed in the *Policy and SSLA Management* and interact with the *Security Orchestration* and *Discovery* tool. The specific MANIFEST instance will be embedded in the Trust Management by the *Security Orchestration*. The MANIFEST instance will be developed in the context of T4.4 because it contains observations made by the INSPIRE-5Gplus. For inter-domain operation, this structure will be replicated following the component design pattern.

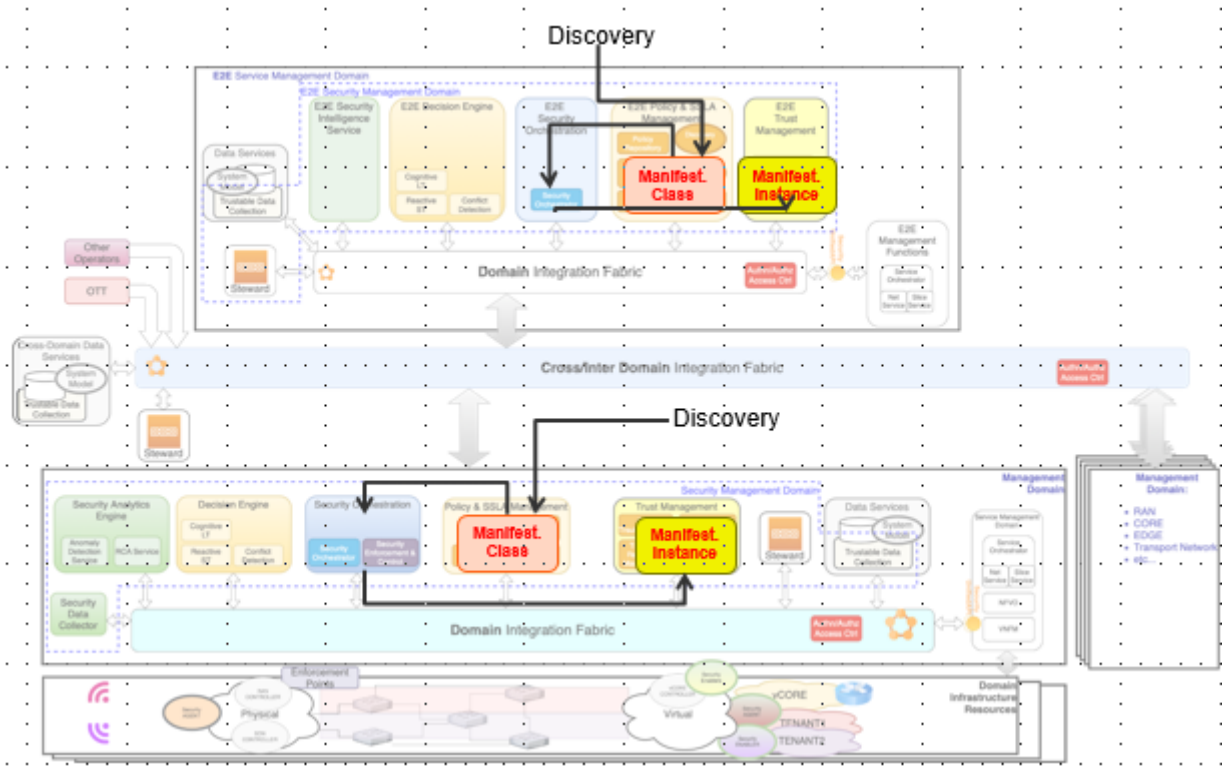

Figure 22 MANIFEST placement in INSPIRE-5Gplus HLA

5.4.2 Interface specification

The LASM Referencing Service and the LASM Analysis Service allow MANIFEST elements to be exchanged data. Their interfaces are shown in Table 11. They will be designed to enable API based communication from the MANIFEST Class and Instance in the INSPIRE-5Gplus architecture.

Table 11 APIs of the LASM Referencing Service and the LASM Analysis Service

REST endpoint	Method	URL	Required Data Objects	Returned Data Object
Referencing Service	GET	/referencingpolicy	--	Text file with SWRL rules
	POST	/referencingpolicy	Text file with SWRL rules	--
	POST	/manifestclass	Multipart(zip)	JSON(string :manifestclass_ID, string: manifestclass_status)

	PUT	/manifestclass/{ID}	--	JSON(string: manifestclass_ID, string: manifestclass_status)
	GET	/manifestclass/{ID}/status	--	Status in plaintext
	GET	/manifestclass/{ID}/file	--	Manifestclass_JSON
		/manifestclass/findby	JSON(map: key/criteria)	Array(string: manifestclass_ID)
Analysis Service	POST	/manifestinstance/	JSON(string: manifestclass_ID, string: instance_ID, JSON: instanciation)	--
	PUT	/manifestinstance/{instance_ID}	JSON(map: metrics/action)	--
	GET	/manifestinstance/{instance_ID}		JSON(map: metrics/value)
	GET	/manifestinstance/{instance_ID}/file	--	

5.5 GRALAF

5.5.1 Enabler placement and interactions

GRALAF will be integrated into *Security Analytics Engine* in INSPIRE-5Gplus HLA and closely work with LASM for liability analysis as shown in Figure 23.

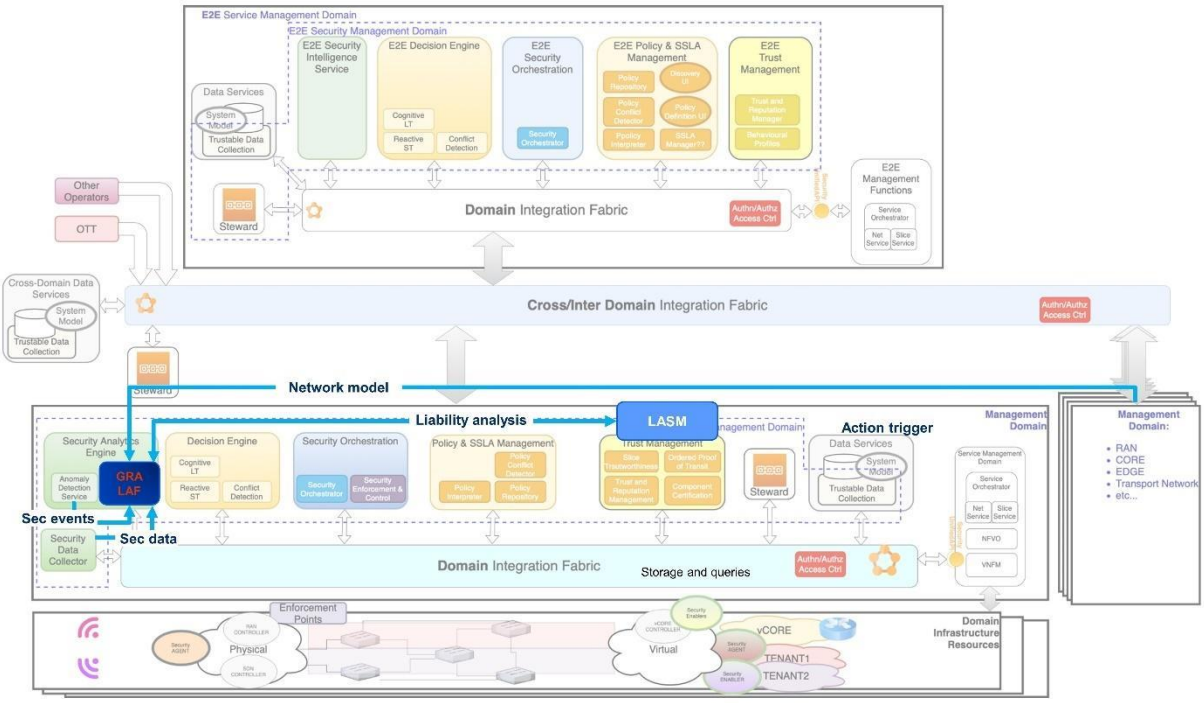

Figure 23 GRALAF placement in INSPIRE-5Gplus HLA

5.5.2 Interface specification

The preliminary interface specification for GRALAF is shown in Table 12. The purpose of the API is to ingest data, mainly network model, and generate the graph model for further analysis via GRALAF algorithms. Moreover, the results will be provided to the end user(s) or other elements in the architecture.

Table 12 GRALAF API

Method	URL	Required Data Objects	Returned Data Object
GET	liability/gralaf/graph/networkmodel	––	JSON object
GET	liability/gralaf/graph /probabilisticgraph	---	JSON object
GET	liability/gralaf/graph /probability/node/<nodeID>	nodeID	Probability value
GET	liability/gralaf/graph /probability/link/<sourceNodeID>/<targetNodeID>	sourceNodeID targetNodeID	JSON object
GET	liability/gralaf/graph/liabilityList	---	JSON object

References

[1] Ross J. Anderson, “Liability and computer security: Nine principles.” In European Symposium on Research in Computer Security, pp. 231-245. Springer, Berlin, Heidelberg, 1994.

[2] 5GPPP, “5G Infrastructure Public Private Partnership (5GPPP) the next generation of communication networks and services,” http://superfluidity.eu/wp-content/uploads/5GPPP-brochure-draft02.pdf.

[3] Pawani Porambage, Gürkan Gür, Diana Pamela Moya Osorio, Madhusanka Liyanage, Andrei Gurtov, Mika Ylianttila, “The roadmap to 6G security and privacy", IEEE Open Journal of the Communications Society, 2021.

[4] F. Akyildiz, A. Kak, and S. Nie, “6G and beyond: The future of wireless communications systems,” IEEE Access, vol. 8, pp. 133 995–134 030, 2020.

[5] ITU-T FG-NET-2030. “Network 2030 A blueprint of technology, applications and market drivers towards the year 2030 and beyond”, May 2019, online: https://www.itu.int/en/ITU-T/focusgroups/net2030/Documents/White_Paper.pdf.

[6] SMA, “Network slicing use cases requirements,” https://www.gsma.com/futurenetworks/wp-content/uploads/2020/01/2.1Network-Slicing-Use-Case-Requirements-Booklet-1.pdf, 2020.

[7] D. Mulvin, “The legal and political battles of Y2K,” IEEE Annals of the History of Computing, 2020.

[8] EC Report with recommendations to the Commission on a civil liability regime for artificial intelligence, https://www.europarl.europa.eu/doceo/document/A-9-2020-0178_EN.html

[9] EC Report on the safety and liability implications of Artificial Intelligence, the Internet of Things and robotics, https://ec.europa.eu/info/sites/info/files/report-safety-liability-artificial-intelligence-feb2020_en_1.pdf

[10] ETSI NFV Release 4 Security; Security Management Specification, https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=58648

[11] L. Condamin, J.-P. Louisot, and P. Naim, Risk quantification - Management Diagnosis and Hedging. John Wiley and Sons, 2007.

[12] Ari Takanen, Petri Vuorijarvi, Marko Laakso, Juha Roning, “Agents of responsibility in software vulnerability processes,” Ethics and Information Technology, 2004.

[13] Kari Kostiainen, Alexandra Dmitrienko, Jan-Erik Ekberg, Ahmad-Reza Sadeghi, N. Asokan, “Key attestation from trusted execution environments”, TRUST 2010: 30-464, pp.295–304, 02 2018.

[14] Yue Yu, Huaimin Wang, Bo Liu, Gang Yin, “A trusted remote attestation model based on trusted computing”, TrustCom/ISPA/IUCC 2013: 1504-1509.

[15] Ludovic Jacquin, Adrian L. Shaw, Chris I. Dalton, “Towards trusted software-defined networks using a hardware-based Integrity Measurement Architecture”, IEEE NetSoft 2015: 1-6.

[16] Harrison Mfula, Jukka Nurminen, “Adaptive Root Cause Analysis for Self-Healing in 5G Networks”, 136-143. 10.1109/HPCS.2017.31, 2017.

[17] P. Muñoz, Isabel de la Bandera Cascales, Emil J. Khatib, Ana Gomez Andrades, Inmaculada Serrano, Raquel Barco, “Root cause analysis based on temporal analysis of metrics toward self-organizing 5G Networks”, IEEE Transactions on Vehicular Technology. 66. 1-1. 10.1109/TVT.2016.2586143, 2016.

[18] Maha Mdini, Gwendal Simon, Alberto Blanc, Julien Lecoeuvre, “ARCD: a Solution for Root Cause Diagnosis in Mobile Networks", CNSM, 280-284, 2018.

[19] Maha Mdini, Gwendal Simon, Alberto Blanc, Julien Lecoeuvre, “Introducing an Unsupervised Automated Solution for Root Cause Diagnosis in Mobile Networks”, IEEE Trans. Netw. Serv. Manag. 17(1): 547-561, 2020.

[20]Marcello Pelillo, Similarity-Based Pattern Analysis and Recognition, Springer, 2013, isbn: 978-1-4471-5628-4.

[21]Richard Lowry, Concepts and Applications of Inferential Statistics, Vassar College, 2008

[22]C. Gaber, J.S. Vilchez, G. Gur, M. Chopin, N. Perrot, J.L. Grimault, J.P. Wary, "Liability-aware security management for 5G," 2020 IEEE 3rd 5G World Forum (5GWF), pp. 133-138, Sept. 2020.

[23]ETSI ETSI GS NFV-IFA 011 V2.1.1 (2016-10), NFV-IFA 011 VNF Packaging Specification, ETSI Std., 2016.

[24]ETSI, NFV-IFA 014 Network Service Templates Specification, ETSI Std., 2016.

[25]B. Moran, H. Tschofenig, and H. Birkholz, “SUIT CBOR manifest serialisation format (draft),” IETF, Jul. 2019.

[26]E. Lear, R. Droms, and D. Romascanu, “RFC 8520 – Manufacturer Usage Description Specification,” IETF, Mar. 2019.

[27]C. Hounkonnou, “Active Self-Diagnosis in Telecommunication Networks”. PhD thesis. Université de Rennes 1. July 2013.

[28]M. Steinder and A. S. Sethi. “End-to-end service failure diagnosis using belief networks”. In Network Operations and Management Symposium, NOMS 2002, pages 375-390, 2002

[29]L. Bennacer, L. Ciavaglia, et.al., “Optimization of fault diagnosis based on the combination of Bayesian Networks and Case-Based Reasoning,” in IEEE NOMS, 2012, vol., no., pp.619,622, 16-20 April 2012.

[30]J. Sanchez, I. Grida Ben Yahia, N. Crespi, “Self-modeling based diagnosis of software-defined networks,” Workshop MISSION 2015 at 1st IEEE Conference on Network Softwarization, London, 13-17 April 2015.

[31]P. Bahl, R. Chandra, et. al., “Towards highly reliable enterprise networking services via inference of multi-level dependencies,” in SIGCOMM, 2007.

[32]Jose Manuel Sanchez Vilchez, Imen Grida Ben Yahia, Noel Crespi. Self-Modeling based Diagnosis of Services over Programmable Networks. 2nd conference on Network Softwarization (Netsoft2016), Jun 2016, Seoul, South Korea.

[33]Topology-Aware Self-Diagnosis framework, presented in Orange Labs Research exhibition 2015, Paris, France. Video available at: https://www.youtube.com/watch?v=xNudu48quRM

[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]J. Weng, J. H. Wang, J. Yang and Y. Yang, "Root Cause Analysis of Anomalies of Multitier Services in Public Clouds," in *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1646-1659, Aug. 2018, doi: 10.1109/TNET.2018.2843805.

[36]G. Guemkam, C. Feltus, P. Schmitt, C. Bonhomme, D. Khadraoui, and Z. Guessoum, “Reputation Based Dynamic Responsibility to Agent Assignement for Critical Infrastructure,” in 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, vol. 2, 2011, pp. 272–275.

[37]C. Bonhomme, C. Feltus, and D. Khadraoui, “A multi-agent based decision mechanism for incident reaction in telecommunication network,” in ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010, 2010, pp. 1–2.

[38] C. Gaber, J.-L. Grimault, C. Loiseaux, M. Hajj, L. Coureau, and J.-P. Wary, “How increasing the confidence in the eSIM ecosystem is essential for its adoption.” [Online]. Available: https://hellofuture.orange.com/en/how-increasing-the-confidencein-the-esim-ecosystem-is-essential-for-its-adoption/

[39]A. Giaretta, N. Dragoni, and F. Massacci, “IoT Security Configurability with Security-by-Contract,” Sensors, vol. 19, no. 19, p. 4121, 2019

[40]G. Costa, N. Dragoni, L. Aliaksandr, F. Martinelli, F. Massacci, and M. Ilaria, “Extending Security-by-Contract wth Quantitative Trust on Mobile Devices,” International Conference on Complex, Intelligent and Software Intensive Systems, 2010.

[41]J. Lin, Q. Zhang, H. Bannazadeh and A. Leon-Garcia, "Automated anomaly detection and root cause analysis in virtualized cloud infrastructures", Proc. IEEE/IFIP Netw. Oper. Manage. Symp. (NOMS), pp. 550-556, Apr. 2016.

[42]D. Dandona, M. Demir and J. J. Prevost, "Graph Based Root Cause Analysis in Cloud Data Center," 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE), Budapest, Hungary, 2020, pp. 549-554, doi: 10.1109/SoSE50414.2020.9130526.

[43]Y. Matsuo, Y. Nakano, A. Watanabe, K. Watanabe, K. Ishibashi and R. Kawahara, "Root-Cause Diagnosis for Rare Failures Using Bayesian Network with Dynamic Modification," *2018 IEEE International Conference on Communications (ICC)*, Kansas City, MO, USA, 2018, pp. 1-6, doi: 10.1109/ICC.2018.8422955.

[44]H. Wang, W. Cai, J. Li and K. He, "Exploring Graphical Models with Bayesian Learning and MCMC for Failure Diagnosis," *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Beijing, China, 2020, pp. 151-156, doi: 10.1109/ASP-DAC47756.2020.9045154.

[45]TCG, Trusted Platform Module Library Family "2.0" Specification - Parts 1-4 and Code, Revision 1.59

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

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

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

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

[50]Qt software package for Python, [Online]. Available:http://www.qt.io/download/

[51]UbiGraph 3-D graph representation tool, [Online]. Available: http://www.ubietylab.net/ubigraph/

[52]VLC Media Player, [Online]. Available: http://www.videolan.org/vlc/

[53]B. Dorsemaine, J. Gaulier, J. Wary, N. Kheir and P. Urien, "A new approach to investigate IoT threats based on a four layer model," 2016 13th International Conference on New Technologies for Distributed Systems (NOTERE), 2016, pp. 1-6, doi: 10.1109/NOTERE.2016.7745830.

[54]Bernhard A. Koch, "Liability for Emerging Digital Technologies: An Overview" *Journal of European Tort Law*, vol. 11, no. 2, 2020, pp. 115-136. https://doi.org/10.1515/jetl-2020-0137

[55]Christiane Wendehorst, "Strict Liability for AI and other Emerging Technologies" *Journal of European Tort Law*, vol. 11, no. 2, 2020, pp. 150-180. https://doi.org/10.1515/jetl-2020-0140

[56]Shankar Siva Kumar, R., O'Brien, D. R., Albert, K., & Vilojen, S. (2018). Law and Adversarial Machine Learning. arXiv e-prints, arXiv-1810

[57]Diego de Siqueira Braga, Marco Niemann, Bernd Hell, “Survey on Computational Trust and Reputation Models”, ACM Computing 2018

[58]Reinaldo A. C. Bianchi and Ramón López De Mántaras, “Should I trust my teammates? An experiment in Heuristic Multiagent Reinforcement Learning”, IJCAI'09, W12: Grand Challenges for Reasoning from Experiences

[59]Massimo Felici and Carmen Fernandez-Gago, “Cloud Accountability: Glossary of Terms and

Definitions", p.291—306, Springer International Publishing, 2015

[60]John Kingston, "Artificial intelligence and legal liability", International Conference on Innovative Techniques and Applications of Artificial Intelligence, 2016

[61]D. D. S. Braga, M. Niemann, B. Hellingrath, and F. B. D. L. Neto (2018). Survey on computational trust and reputation models. ACM Computing Surveys (CSUR), 51(5), 1-40.

APPENDIX A - Implementation Details

Appendix A.1 RCA-VNF (Swagger API)

The issue is the need of strong proofs of isolation when VNF of different sensitivities are operated on the same physical resource[6], e.g., an IoT management VNF with a VNF embedding Lawful Interception features. A new way to address these challenges will be to organize an orchestration under affinities constraints. In the scope of WP4, we will propose algorithms / theory to demonstrate the proposed approach efficiency. The Security by Orchestration will be delivered inside WP4 (TRL3-4, for Kubernetes environment) and will be implemented over the OLP MEC infrastructure (WP3, with a TRL 4-5) for an OpenStack environment.

```
openapi: "3.0.0"
info:
  description: "REST API for the Root Cause Analysis (RCA)-Orange"
  version: "0.2"
  title: "RCA API"
  contact:
    email: "jose2.sanchez@orange.com"
paths:
 /security/rca/graph/{timeSlot}/networkgraph:
    get:
     operationId: "api.security.rca.networkgraph"
      summary: "returns a network graph"
     description: ""
     parameters:
        - in: "path"
          name: "timeSlot"
         schema:
           type: "string"
         required: true
          description: "The R name"
      responses:
        "200":
          description: "Success"
          content:
            application/json:
              schema:
               $ref: "#/components/schemas/networkgraph"

        "400":
         description: "Invalid status value-string"
      tags:
        - "rca"
 /security/rca/graph/{timeSlot}/probabilisticgraph:
    get:
     operationId: "api.security.rca.probabilisticgraph"
      summary: "returns a probabilisticgraph"
      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/pgm"
       "400":
         description: "Invalid status value"
```

[6] https://www.fftelecoms.org/app/uploads/2021/02/fftelecoms_referentiel_objectifs_securite_virtualisation.pdf

```
      tags:
        - "rca"
  /security/rca/probability/{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/NodeProbabilityResponse"
       "400":
         description: "Invalid status value"
     tags:
       - "rca"
 /security/rca/probability/{timeSlot}/link/{sourceNodeId}/{targetNodeId}:
    get:
     operationId: "computeLinkProbability"
      summary: "Compute the probability 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/LinkProbabilityResponse"
       "400":
         description: "Invalid status value"
     tags:
       - "rca"
```

```
components:
  schemas:
   networkgraph:
      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 Network graph"
        nodes: [
          {
           "type": "controller",
           "id": 0
          },
          {
            "type": "switch",
           "id": 1
          },
          {
            "type": "switch",
           "id": 2
          },
          {
            "type": "switch",
           "id": 3
          },
          {
            "type": "host",
           "id": 4
         },
          {
            "type": "host",
           "id": 5
          },
          {
            "type": "host",
           "id": 6
          }
          ]
        links: [
          {
           "type": "control link",
           "source": 0,
           "target": 1
          },
          {
           "type": "control link",
           "source": 0,
           "target": 2
          },
          {
           "type": "control link",
           "source": 0,
           "target": 3
          },
          {
           "type": "switch-switch link",
           "source": 1,
           "target": 2
          },
          {
           "type": "switch-switch link",
           "source": 2,
           "target": 3
```

```
        },
        {
         "type": "host-switch link",
         "source": 1,
         "target": 4
        },
        {
         "type": "host-switch link",
         "source": 2,
         "target": 5
        },
        {
         "type": "host-switch link",
         "source": 3,
         "target": 6
        }
        ]

  NodeProbabilityResponse:
     type: "object"
    properties:
      timeSlot:
         type: "integer"
        format: "int32"
       nodeId:
         type: "integer"
        format: "int32"
      probability:
         type: "number"
        format: "float"
     example:
       timeSlot : 3
       nodeId : 1
      probability : 0.1
  LinkProbabilityResponse:
     type: "object"
    properties:
      timeSlot:
         type: "integer"
        format: "int32"
       nodeId:
         type: "integer"
        format: "int32"
      probability:
         type: "number"
        format: "float"
     example:
       timeSlot : 3
      sourceNodeId : 3
      targetNodeId : 1
      probability : 0.11
   Node:
     type: "object"
    properties:
       id:
         type: "integer"
        format: "int64"
       type:
         type: "string"
      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"
         items:
```

```
          type: "number"
          format: "float"
    pgm:
      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 PGM"
        nodes: [
          {
           "type": "controller",
           "probability": [0.5],
            "id": 0
          },
          {
            "type": "switch",
           "probability": [0.5],
           "id": 1
          },
          {
            "type": "switch",
           "probability": [0.5],
           "id": 2
          },
          {
            "type": "switch",
           "probability": [0.5],
           "id": 3
          },
          {
            "type": "host",
           "probability": [0.5],
           "id": 4
         },
          {
            "type": "host",
           "probability": [0.5],
           "id": 5
          },
          {
            "type": "host",
           "probability": [0.5],
           "id": 6
          }
          ]
        links: [
          {
           "type": "control link",
           "probability": [0.0],
           "source": 0,
           "target": 1
          },
          {
           "type": "control link",
           "probability": [0.0],
           "source": 0,
           "target": 2
         },
          {
           "type": "control link",
           "probability": [0.0],
           "source": 0,
           "target": 3
          },
```



```
{
 "type": "switch-switch link",
 "probability": [0.0],
 "source": 1,
  "target": 2
},
{
 "type": "switch-switch link",
"probability": [0.0],
 "source": 2,
 "target": 3
},
{
 "type": "host-switch link",
 "probability": [0.0],
 "source": 1,
 "target": 4
},
{
 "type": "host-switch link",
 "probability": [1.0],
 "source": 2,
 "target": 5
},
{
 "type": "host-switch link",
 "probability": [0.0],
 "source": 3,
 "target": 6
}
]
```


…….