

RESISTO:

D5.2_INTERIM SOFTWARE DEFINED SECURITY SYSTEM AND DECISION MAKING MODULE



RESISTO

D5.2 – INTERIM SOFTWARE DEFINED SECURITY SYSTEM AND DECISION MAKING MODULE

| | | | |
|--------------------------|--------------------|-----|--------|
| Document Manager: | Federico Colangelo | RM3 | Editor |
|--------------------------|--------------------|-----|--------|

| | |
|-----------------------------|---|
| Project Title: | RESilience enhancement and risk control platform for communication infraSTructure Operators |
| Project Acronym: | RESISTO |
| Contract Number: | 786409 |
| Project Coordinator: | LEONARDO |
| WP Leader: | RM3 |

| | | | |
|------------------------|------------------------|-----------------|------------|
| Document ID N°: | RESISTO_D5.2_191128_01 | Version: | 1.0 |
| Deliverable: | D5.2 | Date: | 28/11/2019 |
| | | Status: | APPROVED |

| | |
|--------------------------------|---------------|
| Document classification | Public |
|--------------------------------|---------------|

| Approval Status | |
|---|--------------------------|
| Prepared by: | Federico COLANGELO (RM3) |
| Approved by: (WP Leader) | Marco CARLI (RM3) |
| Approved by: (Coordinator) | Bruno SACCOMANNO (LDO) |
| Approved by: (Advisory Board Leader) | NA |
| Security Approval (Security Advisory Board Leader) | NA |

CONTRIBUTING PARTNERS

| Name | Company / Organization | Role / Title |
|---|------------------------|--|
| Federico COLANGELO, Marco CARLI | RM3 | Postdoc Researcher, Associate Professor |
| Moisés VALEO, Jose SANCHEZ, Javier VALERA | INT | Senior Researchers, Electrical Engineers, Defence and Security Specialists |
| Andrei AVADANEI Lucian NITESCU | BSS | CEO & Senior Security Specialist, Communication Specialist |
| Jorge CARAPINHA, Paula CRAVO | ALB | Contributor |
| Alberto NERI, Emanuele AONZO | LDO | Contributor |
| Giuseppe CELOZZI, Cosimo ZOTTI | TEI | Contributor |

DISTRIBUTION LIST

| Name | Company / Organization | Role / Title |
|----------------|------------------------|----------------------|
| PMT | RESISTO CONSORTIUM | NA |
| Markus MULLER | EC DG REA | EC Programme Officer |
| General Public | NA | NA |

REVISION TABLE

| Version | Date | Modified Pages | Modified Sections | Comments |
|---------|------------|----------------|-------------------|--------------------------------------|
| 0.1 | 17.04.2019 | | All | Draft ToC |
| 0.2 | 14.06.2019 | | All | Additions and partners contributions |
| 0.3 | 28.06.2019 | | All | First draft |
| 0.4 | 02.06.2019 | | All | Reviewed with all WP partners |
| 0.9 | 15.07.2019 | | All | Final release for review |
| 1.0 | 28.11.2019 | All | All | Final Release |

COPYRIGHT STATEMENT



© 2018-2021 This document and its content are the property of the RESISTO Consortium. All rights relevant to this document are determined by the applicable laws. Access to this document does not grant any right or license on the document or its contents. This document or its contents are not to be used or treated in any manner inconsistent with the rights or interests of the RESISTO Consortium or the Partners detriment and are not to be disclosed externally without prior written consent from the RESISTO Partners. Each RESISTO Partner may use this document in conformity with the RESISTO Consortium Grant Agreement provisions.

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Program, under the Grant Agreement No 786409.

The views and opinions in this document are solely those of the authors and contributors, not those of the European Commission.

PROJECT CONTACT



LEONARDO
Via Puccini 2 – Genova (GE) – 16154 – Italy
Tel.: +39 348 6505565
E-Mail: bruno.saccomanno@leonardocompany.com

RESISTO PROJECT – PUBLISHABLE EXTENDED ABSTRACT

Communications play a fundamental role in the economic and social well-being of the citizens and on operations of most of the CIs. Thus, they are a primary target for criminals having a multiplier effect on the power of attacks and providing enormous resonance and gains. Also, extreme weather events and natural disasters represents a challenge due to their increase in frequency and intensity requiring smarter resilience of the Communication CIs, which are extremely vulnerable due to the ever-increasing complexity of the architecture also in light of the evolution towards 5G, the extensive use of programmable platforms and exponential growth of connected devices. The fact that most enterprises still manage physical and cyber security independently represents a further challenge. RESISTO platform is an innovative solution for Communication CIs holistic situation awareness and enhanced resilience (aligned with ECSO objectives). Based on an Integrated Risk and Resilience analysis management and improvement process availing all resilience cycle phases (prepare, prevent, detect, absorb, etc.) and technical resilience capabilities (sense, model, infer, act, adopt), RESISTO implements an innovative Decision Support System to protect communication infrastructures from combined cyber-physical threats exploiting the Software Defined Security model on a suite of state of the art cyber/physical security components (Blockchain, Machine Learning, IoT security, Airborne threat detection, holistic audio-video analytics) and services (Responsible Disclosure Framework) for detection and reaction in presence of attacks or natural disasters. Through RESISTO Communications Operators, will be able to implement a set of mitigation actions and countermeasures that significantly reduce the impact of negative events in terms of performance losses, social consequences, and cascading effects in particular by bouncing efficiently back to original and forward to operational states of operation.

EXECUTIVE SUMMARY

Software Defined Security (SDS) is a framework encompassing technologies that allow an approach to security in line with current networking technologies such as software defined networking. SDS aims at implementing abstract security functions that can enable better situational awareness concerning risks, ongoing attacks, and available mitigation strategies. It also enables dynamical reaction strategies to threats by deploying requirements-aware countermeasures.

This deliverable summarizes the activities performed for implementing the Software Defined Security framework in RESISTO. More specifically, techniques, software and novel algorithms are considered for coping with NFV and 5G technologies. Furthermore, systems for applying the SDS framework to physical security are considered. Key elements of the RESISTO SDS architecture are the mitigation framework and the orchestrator module. Ongoing activities as well as technologies that are currently being evaluated are reported.

This deliverable represents the current status of Tasks 5.2 and 5.3.

CONTENTS

| | |
|---|-----------|
| ABBREVIATIONS | 10 |
| 1. INTRODUCTION – PURPOSE OF THE DOCUMENT..... | 13 |
| 2. SDN, NFV, and 5G | 14 |
| 2.1. SDN and NFV | 14 |
| 2.1.1. SDN | 14 |
| 2.1.2. NFV | 15 |
| 2.1.3. Interworking with external systems - Northbound interfaces | 17 |
| 2.1.4. SDN and NFV in 5G | 20 |
| 2.2. SDN and NFV in RESISTO | 20 |
| 2.2.1. Open Source Platforms | 20 |
| 2.2.2. SDN controllers | 21 |
| 2.2.3. NFV Orchestrators | 23 |
| 2.2.4. 5G Mobile Core | 27 |
| 3. SECURITY IN THE SDN/NFV WORLD | 29 |
| 3.1. Attack surface evolution..... | 29 |
| 3.2. Novel security and resiliency approaches | 32 |
| 4. SOFTWARE DEFINED SECURITY | 34 |
| 4.1. The SDS framework | 34 |
| 4.2. SDS in RESISTO..... | 36 |
| 4.2.1. Mitigation framework..... | 37 |
| 4.2.2. Orchestrator module | 43 |
| 4.3. Extension of the SDS model..... | 44 |
| 4.3.1. Software Defined Radio Security | 45 |
| 4.3.2. Multi-objective countermeasure optimization | 46 |
| 4.3.3. Machine learning-based positioning for Physical Security | 49 |
| 4.3.4. Machine Learning-based cell fault detection | 50 |
| 5. CONCLUSION | 57 |
| 6. REFERENCES..... | 58 |

LIST OF FIGURES

| | |
|--|----|
| Figure 1: SDN interfaces [3]..... | 15 |
| Figure 2: ETSI NFV MANO Architecture | 16 |
| Figure 3: Main MANO reference points and respective ETSI NFV specs | 17 |
| Figure 2: ETSI NFV MANO Architecture | 19 |
| Figure 5: ONOS Web Interface, hosts list | 21 |
| Figure 6: ONOS Web Interface, topology summary | 22 |
| Figure 7: OSM interaction with VIMs and VNFs..... | 24 |
| Figure 8: SONATA architecture | 25 |
| Figure 9: SONATA Service platform architecture | 26 |
| Figure 10: Open5GCore architecture | 28 |
| Figure 11: SDS conceptual architecture | 35 |
| Figure 12: BPMN task and process | 38 |
| Figure 13: BPMN Process example | 38 |
| Figure 14: Example of process and task | 39 |
| Figure 15: Process designer web HMI..... | 39 |
| Figure 16: Activiti Engine | 40 |
| Figure 17: Detail of opened alarms | 40 |
| Figure 18: Execution of a user task..... | 41 |
| Figure 19: Workflow flow chart | 41 |
| Figure 20: Action list of a workflow..... | 42 |
| Figure 21: Altice Labs Cognition Pipeline | 51 |
| Figure 22: The Cognitive Process..... | 52 |
| Figure 23: ML methodology..... | 55 |

ABBREVIATIONS

| | |
|-----------------------|---|
| 2G, 3G, 4G, 5G | Second, third, fourth and fifth generation of mobile phone systems |
| 3GPP | 3rd Generation Partnership Project |
| API | Application Programming Interface |
| APT | Advanced Persistent Threat |
| BPM | Business Process Model |
| BPMN | Business Process Model and Notation |
| CI | Critical Infrastructure |
| CN | Core Network |
| DDoS | Distributed Denial of Service |
| DSS | Decision Support System |
| ETSI | European Telecommunications Standard Institute |
| eMBB | Enhanced Mobile Broadband |
| EMS | Element Management System |
| IETF | Internet Engineering Task Force |
| IoT | Internet of Things |
| KPI | Key Performance Indicator |
| LTE | Long Term Evolution (= 4G) |
| LTE-M | simplified term for LTE-MTC LPWA (Long Term Evolution Machine Type Communication Low Power Wide Area) |
| mMTC | Massive Machine Type Communication |
| MVNO | Mobile Virtual Network Operators |
| NFV | Network Function Virtualization |
| NFVO | NFV Orchestrator |
| NS | Network Service |
| OSM | Open Source Management and orchestration |
| RAN | Radio Access Network |
| RF | Radio Frequency |
| SDN | Software Defined Networks |
| SDR | Software Defined Radio |

| | |
|-----------------------|---|
| SDS | Software Defined Security |
| SLA | Service Level Agreement |
| UMTC | Ultra-reliable Low Latency Communication |
| VIM | Virtualized Infrastructure Manager |
| VNF | Virtual Network Function |
| VNF-FG | VNF Forwarding Graph |
| VNFM | Virtual Network Function Manager |
| VPN | Virtual Private Network |
| VR | Virtual Radio |
| WP | Work Package |
| 2G, 3G, 4G, 5G | Second, third, fourth and fifth generation of mobile phone systems |
| 3GPP | 3rd Generation Partnership Project |
| API | Application Programming Interface |
| APT | Advanced Persistent Threat |
| BPM | Business Process Model |
| BPMN | Business Process Model and Notation |
| CI | Critical Infrastructure |
| CN | Core Network |
| DDoS | Distributed Denial of Service |
| DSS | Decision Support System |
| ETSI | European Telecommunications Standard Institute |
| eMBB | Enhanced Mobile Broadband |
| EMS | Element Management System |
| IETF | Internet Engineering Task Force |
| IoT | Internet of Things |
| KPI | Key Performance Indicator |
| LTE | Long Term Evolution (= 4G) |
| LTE-M | simplified term for LTE-MTC LPWA (Long Term Evolution Machine Type Communication Low Power Wide Area) |
| mMTC | Massive Machine Type Communication |
| MVNO | Mobile Virtual Network Operators |

| | |
|------|--|
| NFV | Network Function Virtualization |
| NFVO | NFV Orchestrator |
| NS | Network Service |
| OSM | Open Source Management and orchestration |
| RAN | Radio Access Network |
| RF | Radio Frequency |

1. INTRODUCTION – PURPOSE OF THE DOCUMENT

This document reports the status of the activities on Tasks 5.2 and 5.3.

The main outputs of the tasks are the RESISTO Software Defined Security system and the Decision Support System.

The SDS system in RESISTO provides support to the operator in the selection of the appropriate actions for mitigating threats and risks and to simplify the deployment of those actions by coordinating networks, security functions, and physical resources.

The mitigation framework supports the operator in reacting to a threat by means of workflows, that are activated to cope with specific threats detected by the RESISTO architecture. Operators will be able to assess the situation visually through the RESISTO HMI, described in [1], and select the most appropriate workflow based on the threat as well as on the impact of the workflow's actions on the infrastructure. The RESISTO platform then executes the action in the workflow by means of the workflow manager module, described in this document. The document details the current state of the activities and plans for augmenting the mitigation framework with decision-support tools based on multi-objective optimization.

The orchestrator has the role of implementing action provided by the mitigation framework, basically by receiving a high-level action and translating it into a set of low-level commands and configurations. Ongoing activities include the development of multi-objective algorithms to find the most efficient way to deploy complex actions.

The document starts by reviewing Software-Defined Networking (SDN), Network Function Virtualization (NFV) and 5G architectures in order to highlight their role in RESISTO architecture. An overview of the evolution of security concepts is provided to understand strength, threats and opportunities derived from these technologies. SDS is first introduced as a conceptual framework, then its deployment in RESISTO is described.

The deliverable is structured as follows:

Section 2 introduces the concepts of SDN and NFV and the key differences they introduce with respect to legacy approaches to networking. The enabling role of these technologies for 5G is also presented. The most important open source tools are introduced. Section 3 deals with security in the SDN/NFV world. Specifically, threats that are exclusive to these paradigms are presented, discussing state-of-the-art solution to address them. Furthermore, a review on how SDN/NFV can be used to enhance network security is given.

Section 4 defines the concept of SDS and the ongoing activities. Finally, in Section 5 the conclusions of the deliverable are presented.

2. SDN, NFV, AND 5G

This section provides a brief introduction to SDN and NFV, and how these technologies can cooperate with external systems. The role played by SDN and NFV in 5G and 5G-enabled network concepts, such as network slicing, is briefly described. SDN and NFV components, developed under the umbrella of open source projects, are also concisely described in this section.

2.1. SDN and NFV

2.1.1. SDN

SDN is a network paradigm decoupling network control and forwarding functions by enabling programmable network control and the underlying infrastructure to be abstracted from applications and Network Services (NS).

SDN is based on four basic principles [4]:

- 1) **Separation of control and data plane:** Removing the control plane from network devices and implementing it in an external SDN controller significantly reduces the complexity of network devices, making them simpler and cheaper than CN devices whose distributed control plane functionality is implemented across millions of lines of code, and defined across hundreds of RFCs.
- 2) **Logically centralized control:** Control decisions are made on an up-to-date global view of the network state, rather than distributed in isolated behavior at each network hop. With SDN, the control plane acts as a single, logically centralized network operating system in terms of both scheduling and resolving resource conflicts, as well as abstracting away low-level device details, e.g., electrical vs. optical transmission. The SDN Controller summarizes the network state for applications and translates application requirements to low-level rules. This does not imply that the controller is physically centralized. For performance, scalability, and/or reliability reasons, the logically centralized SDN Controller can be distributed so that several physical controller instances cooperate to control the network and serve the applications.
- 3) **Programmability of NSs:** This principle permits a client to exchange information with an SDN controller, either by discovery or negotiation prior to the establishment of a service, or during the lifetime of a service according to changes in client needs or the state of the client's virtual resources. The network is programmable through software applications running on top of the Controller (communicating via Northbound Application Programming Interfaces (API)), which in turn interacts with the underlying data plane devices. The SDN controller, with complete knowledge of the network topology, controls a wide range of network devices within its administrative domain. By providing APIs, SDN enables the development of networking applications, e.g., traffic engineering, thus enabling network innovation. In contrast, CN devices are proprietary and closed, making it hard or impossible to develop innovative network applications.
- 4) **Open interfaces:** This implies the well-defined partition of functions and interfaces, and specifies that the interfaces be public and open to community definition. One value of SDN lies in the expectation that the Control to Data-Plane Interface (Southbound Interface) is implemented in an open, vendor-neutral and interoperable way. In the absence of a standard open interface, one of the main SDN advantages — the interchangeability of network devices and control planes — would be taken away.

In an SDN environment, the SDN controller plays a pivotal role by managing data flows in to the switches/routers 'below' (via southbound APIs) based on the policies and rules defined by the applications and business logic 'above' (via northbound APIs). The SDN Controller translates instructions or requirements from the SDN Application layer into commands to networking components. Conversely, the SDN controller extracts information about the network from the hardware devices and communicates back to the SDN Applications with an abstract view of the network, including statistics and events.

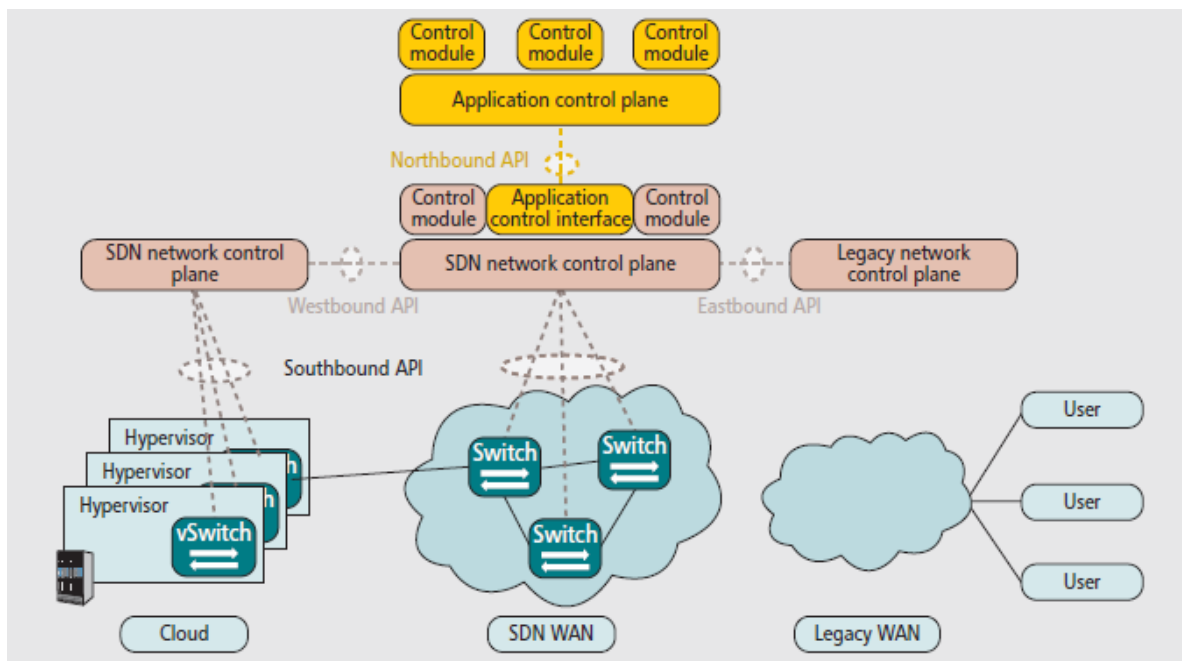


Figure 1: SDN interfaces [3]

2.1.2. NFV

NFV started in 2012 as an industry initiative to virtualize NSs traditionally run on proprietary, dedicated hardware. NFV virtualizes network functions and eliminates specific hardware. The network managers can add, move or change network functions at the server level in a simplified provisioning process.

ETSI has been the main driver of the NFV standardization efforts. The ETSI NFV MANO (Management & Orchestration) model [13], represented in **Errore. L'origine riferimento non è stata trovata.**, defines as main functional blocks:

- NFV Orchestrator (NFVO):
 - on-boarding of new NS, Virtual Network Function Forwarding Graph (VNF-FG), and VNF Packages
 - NS lifecycle management (including instantiation, scale-out/in, performance measurements, event correlation, and termination)
 - global resource management, validation and authorization of NFVI resource requests

- policy management for NS instances
- VNF Manager (VNFM):
 - lifecycle management of VNF instances
 - overall coordination and adaptation role for configuration and event reporting between NFVI and the Element Management System (EMS)
- Virtualised Infrastructure Manager (VIM):
 - controlling and managing the NFVI computing, storage and network resources, within one operator's infrastructure sub-domain
 - collection and forwarding of performance measurements and events

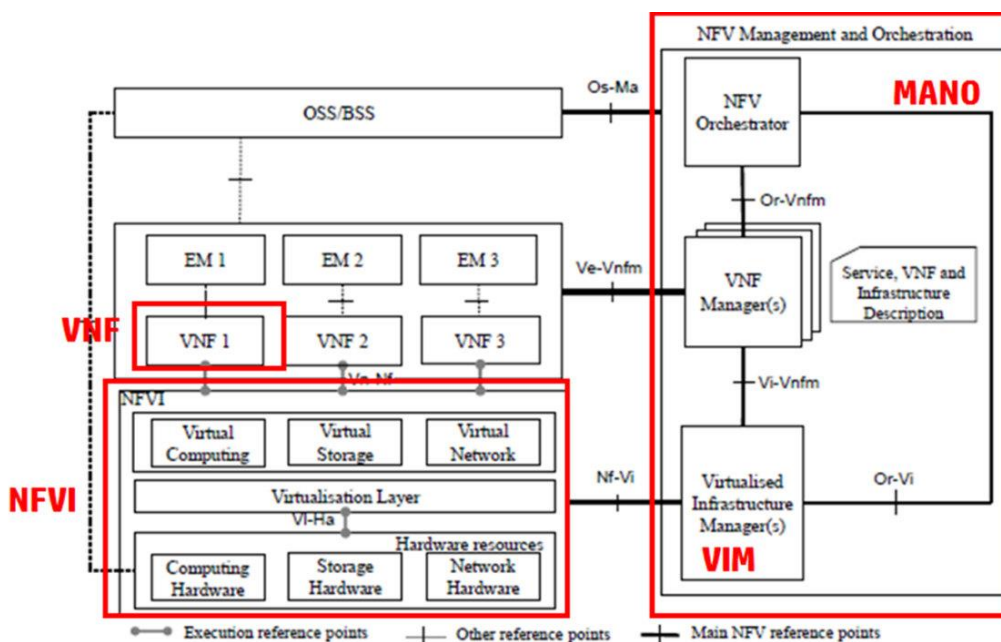


Figure 2: ETSI NFV MANO Architecture

The ETSI MANO model provides the framework to provision VNFs and manage the NFV infrastructure. It also helps components within NFV infrastructure communicate with existing OSS/BSS systems. MANO systems are responsible for managing virtualized infrastructure, such as cloud systems, communication and network infrastructure, including SDN, VNF (implemented as virtual machine or container images) and the full lifecycle of all these components.

The following list describes the interfaces a VIM uses it to allocate, manage, and control the NFVI resources.

- Or-Vnfm: defines how Communication between NFVO and VNFM happens , such as VNF instantiation and other VNF lifecycle-related information flow.
- Or-Vi: defines the direct NFVO-VIM communication to influence the management of the infrastructure resources, such as resource reservation for VMs or VNF software addition.

- Vi-Vnfm: defines the information exchange between VIM and VNFM, such as resource update request for VM running a VNF.
- Vn-Nf: This is the only reference point that doesn't have a management functional block as one of its boundaries. This reference point is meant to communicate performance and portability needs of the VNF to the infrastructure block.

It should be noted that although the NFV architectural framework identifies reference points between the NFVO and the VIM (Or-Vi reference point) and between the VNFM and the VIM (Vi-Vnfm reference point), ETSI does not currently provide API specifications for the corresponding interfaces. The assumption is that the overwhelming majority of VIM implementation is based on OpenStack, and thus the APIs exposed by a VIM are those specified by this open source community [8].

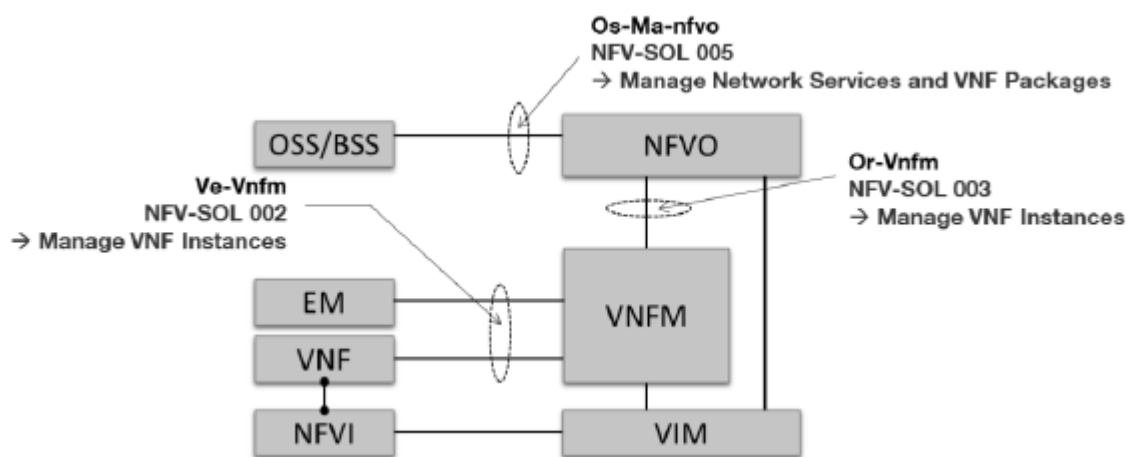


Figure 3: Main MANO reference points and respective ETSI NFV specs

Use of RESTful APIs for management and orchestration is specified by ETSI, except for the VIM northbound interfaces where use of OpenStack APIs is assumed, as noted before. ETSI APIs specifications are available both as Text and Tables and in OpenAPI format.

2.1.3. Interworking with external systems - Northbound interfaces

The interworking with SDN and NFV systems is usually performed through the northbound interface. A brief overview of SDN and NFV northbound interfaces is provided below.

NFV northbound interface: Os-Ma-Nfvo reference point

The Os-Ma-nfvo reference point is used for exchanges between the OSS/BSS and the NFVO and supports, among other things,

- NS Descriptor Management
- NS Lifecycle Management
- NS Performance Management

- NS Fault Management
- VNF Package Management

The following ETSI NFV specifications are relevant in relation to the OS-Ma-nfvo interface:

- ETSI NFV IFA013 defines the interfaces supported over the Os-Ma-nfvo reference point of the NFV-MANO architectural framework as well as the information elements exchanged over those interfaces.
- ETSI NFV SOL 005 defines the protocols and data models for the Os-Ma-nfvo reference point, in the form of RESTful API specifications. ETSI NFV SOL 005 specifies a set of RESTful protocol specifications and data models.

The following NS lifecycle management operations can be invoked through the NS Performance Management interface [7]:

- Create NS Identifier
- Instantiate NS
- Scale NS
- Update NS
- Query NS
- Terminate NS
- Delete NS Identifier
- Heal NS
- Get Operation Status
- Subscribe
- Query Subscription Information
- Notify
- Terminate Subscription

Furthermore, the NS Fault Management interface provides the following operations:

- Get Alarm List
- Acknowledge Alarm
- Subscribe
- Query Subscription Information
- Terminate Subscription
- Notify

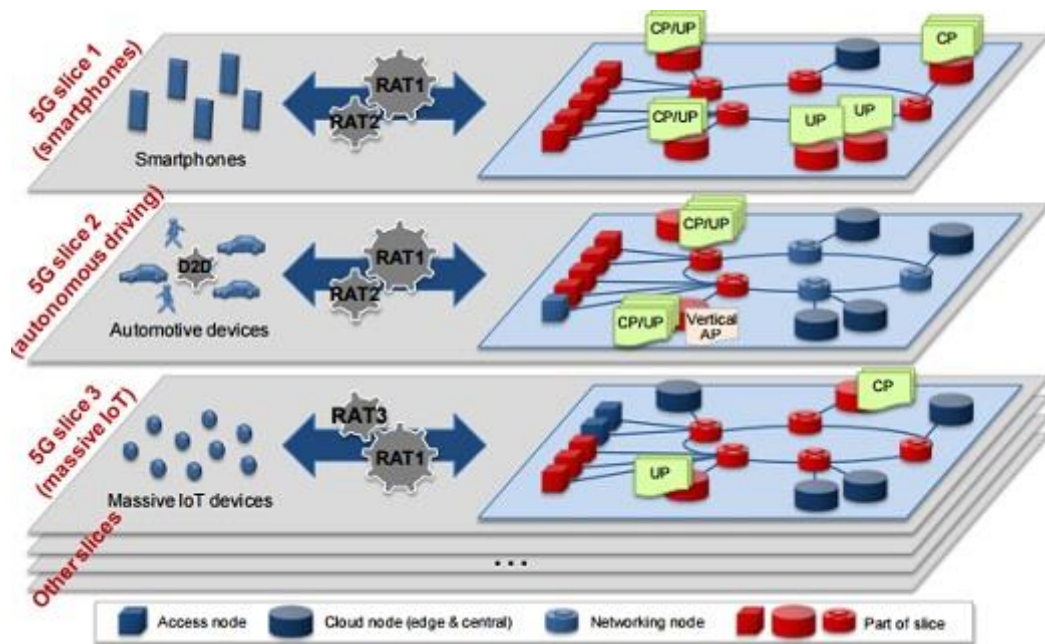


Figure 4: ETSI NFV MANO Architecture

SDN Northbound interface: REST APIs

SDN Northbound APIs must support a wide variety of applications, with different characteristics. A variety of interfaces exist in different places up the stack to control different types of applications via an SDN Controller.

Northbound APIs should present a network abstraction interface to applications and management systems at the top of the SDN stack and is hence considered to be one of the most important components of SDN Architecture. The standardization of Northbound API has been a topic of discussion, partly because of its very dynamic nature. While the SDN controller can directly adapt the behavior of the network, the application controller adapts the behavior of the application using the network. It can be implemented as part of a single application instance to a central entity for the entire network responsible for all applications.

SDN enables the exchange of information with applications running on top of the network. This information exchange is performed via the Northbound-API between the SDN controller and an “application control plane”. A universal, standardized Northbound API does not exist. Furthermore, the form and frequency of the exchanged information depend on the targeted application and network, such universal API is not useful.

Adopting a REST API for the SDN Northbound API has many benefits, including decentralized management of dynamic resources, support of heterogeneous clients, service composition, localized migration and scalability.

2.1.4. SDN and NFV in 5G

5G represents the next major phase of the telecommunication industry. A 5G network will be able to support Enhanced Mobile Broadband (eMBB), Massive Machine Type Communication (mMTC), and the provisioning of Ultra-reliable Low Latency Communication (UMTC) services.

Although the attention of 5G has been focused on developments in the radio system, some of the main innovations of 5G are based on NFV and SDN by enhancing the flexibility of mobile networks and the ability to support a wider variety of services. One of the most distinct 5G innovations, of which NFV and SDN are key ingredients, is the network slicing.

3GPP TS 23.501[9] defines a network slice as “a logical network that provides specific network capabilities and network characteristics”. Network slicing allows multiple virtual networks to be created on top of a common shared physical infrastructure, which can be customized to meet the specific needs of applications, services, devices, customers or operators.

In the case of 5G, a single physical network is sliced into multiple virtual networks that can support different radio access networks (RANs), or different service types running across a single RAN. Network slicing will be used to partition the CN, as well as the RAN.

One of the innovative feature of 5G compared to previous generations of mobile technologies is the so-called service-based architecture, as depicted in **Errore. L'origine riferimento non è stata trovata.**, whereby the control plane functionality and common data repositories of a 5G network are delivered by way of a set of interconnected Network Functions (NFs), each with authorization to access each other's services.

2.2. SDN and NFV in RESISTO

As an innovative security project for the communication CI, RESISTO must be able to operate with SDN and NFV technologies. As it has been reviewed, the next generation of mobile communications rely heavily on these technologies in order to achieve its objectives. More generally, SDN and NFV offer considerable advantages in the domain of security, as it will be reviewed in Section 3. Thus, while for the RESISTO platform it is not fundamental to have an SDN/NFV enabled infrastructure to command, it is important to be able to comply with these technologies as well as to be able to leverage in full the advantages they offer. The features based on SDN and NFV can thus be considered one of the more innovative components of the RESISTO platform.

2.2.1. Open Source Platforms

In this Section, the most relevant SDN, NFV and 5G open source software is reviewed. Open source software is fundamental for these technologies, as it poses the foundations for the development of interoperable software. In the context of RESISTO, open source solutions are favored as they are more upgradable, understandable and compatible with the standards by design.

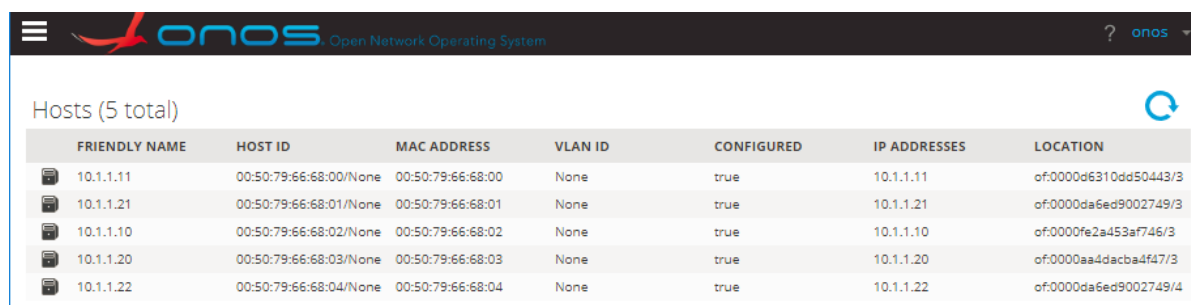
2.2.2. SDN controllers

ONOS

Open Network Operating System (ONOS) is an open source SDN controller [10].

It acts as an extensible, modular, distributed SDN controller providing the control plane for an SDN based network, managing network components, such as switches and links, and running software programs or modules to provide communication services to end hosts and neighboring networks. The main features of ONOS are:

- High availability and performance scaling



| FRIENDLY NAME | HOST ID | MAC ADDRESS | VLAN ID | CONFIGURED | IP ADDRESSES | LOCATION |
|---------------|------------------------|-------------------|---------|------------|--------------|-----------------------|
| 10.1.1.11 | 00:50:79:66:68:00/None | 00:50:79:66:68:00 | None | true | 10.1.1.11 | of:0000d6310dd50443/3 |
| 10.1.1.21 | 00:50:79:66:68:01/None | 00:50:79:66:68:01 | None | true | 10.1.1.21 | of:0000da6ed9002749/3 |
| 10.1.1.10 | 00:50:79:66:68:02/None | 00:50:79:66:68:02 | None | true | 10.1.1.10 | of:0000fe2a453af746/3 |
| 10.1.1.20 | 00:50:79:66:68:03/None | 00:50:79:66:68:03 | None | true | 10.1.1.20 | of:0000aa4dacba4f47/3 |
| 10.1.1.22 | 00:50:79:66:68:04/None | 00:50:79:66:68:04 | None | true | 10.1.1.22 | of:0000da6ed9002749/4 |

Figure 5: ONOS Web Interface, hosts list

ONOS is built following a distributed paradigm distribute system so it can run as a cluster, providing scaling when more computational resources are needed and ensuring services continuity. It has been shown that performances (in terms of throughput) scale linearly with the number of nodes while latency remains constant [11].

- Modularity

Over 135 platform extensions are available for ONOS, including traffic steering applications, network overlay applications, southbound providers. The ONOS applications, are written in Java as bundles that are loaded into the Karaf[42] OSGi container.

- Northbound abstraction

The ONOS applications exposes a high-level API to allows external applications to act on the network without the need for details on how the service will be performed. An example is the Intent Framework that allow to send a command like “Set up a connection between Host A and Host B” simply invoke the correspondent REST-API with the correct JSON structure.

- Southbound abstraction

ONOS already implements some provider to communicate with network device through a variety of protocols(e.g. P4[43], OpenFlow [44], NETCONF [45]SNMP [46]).

- Web User Interface

ONOS provides the view of the multi-layer network and allows the user exploration of network elements, connectivity, network state, network errors and more. Sample screenshot of the interfaces are shown in Figure 5 and Figure 6.

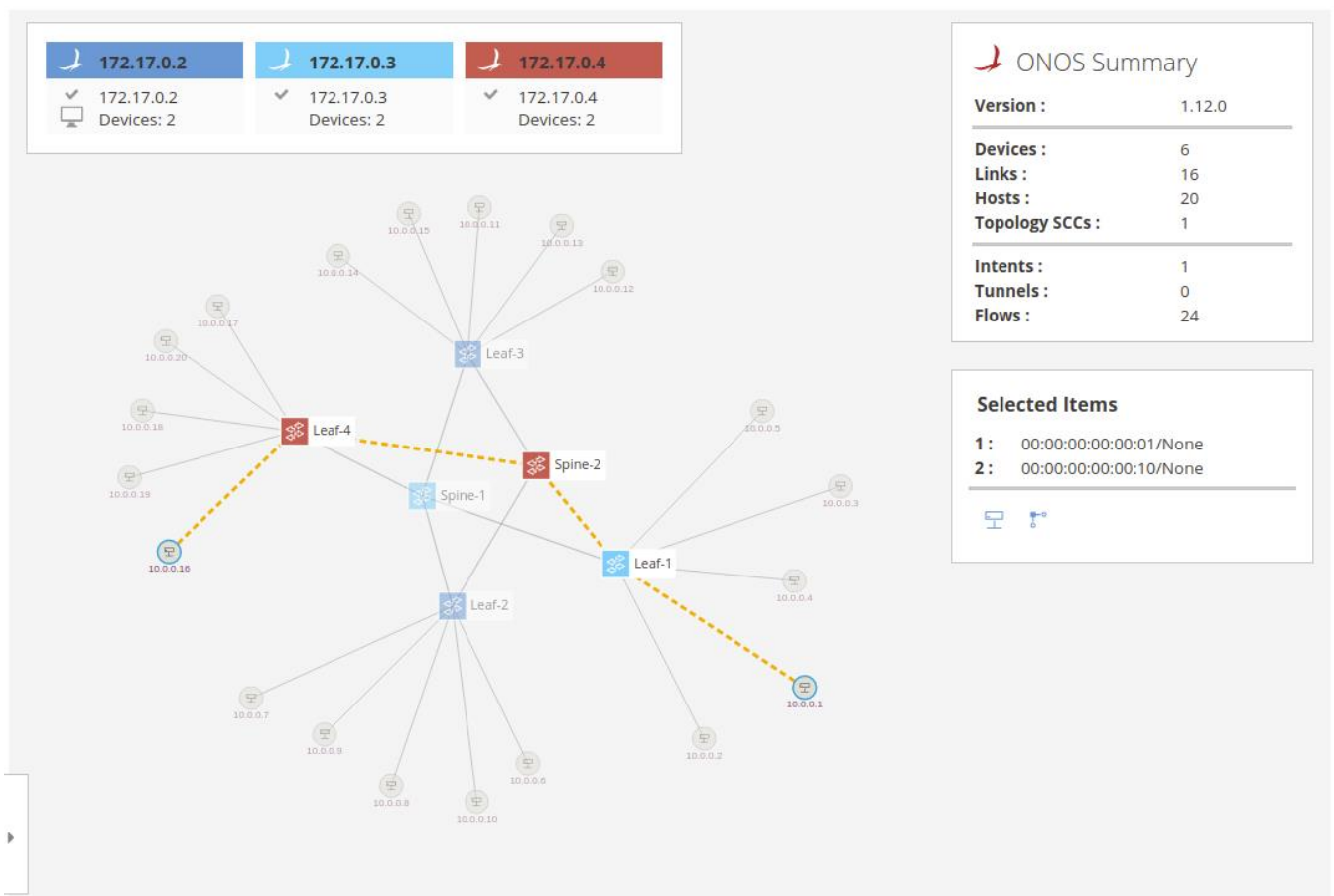


Figure 6: ONOS Web Interface, topology summary

2.2.3. NFV Orchestrators

OSM

Open Source MANO (OSM) is a framework consisting of a set of software functions, designed to guarantee the MANO functions in an NFV architecture. It is an open source project hosted by ETSI. The software implementation is aligned with ETSI NFV Information Models. ETSI is an operator-led community and then OSM is a high-quality product that meets the requirements of commercial NFV networks. Furthermore, ETSI will continue to define the standard for NFV environment and the communication with other software modules involved in network flow, so the new features introduced in OSM will probably become standard.

Figure 7 illustrates OSM interaction with VIMs and VNFs:

OSM consists of three logical components. Each logical component has an implementation based on an open source software. More specifically:

- Resource Orchestrator

The Resource Orchestrator is responsible of the NFVI infrastructure resource orchestration. The Resource Orchestrator is implemented as OpenMANO [47], a software developed by Telefonica, one among the main members of the ETSI ISG NFV MANO group. It allows the orchestration of resources through one or more VIM and is compatible with the most relevant solutions on the market and in open way

- VNF Configuration & Abstraction

The VNF Configuration & Abstraction manages the life cycle of VNFs. More specifically, it is in charge of run-time configuration and management. Open Source MANO leverages JUJU[48] as VNF Manager, a software developed by Canonical, an active member of the consortium for the development of OSM.

- Service Orchestrator

The Service Orchestrator primary task is the coordination of the Resource Orchestrator and the VNF Configuration & Abstraction. The Service Orchestrator manages the complete service life cycle of network, from the on-boarding of the descriptors to the management of the network service. As Service Orchestrator, OSM uses RIFT.ware, a Python software in that support numerous additional features. Examples are translation of the NS descriptors from various formats (YAML, XML, JSON, etc.), to a YANG data model, in order to make it compatible with OSM components.

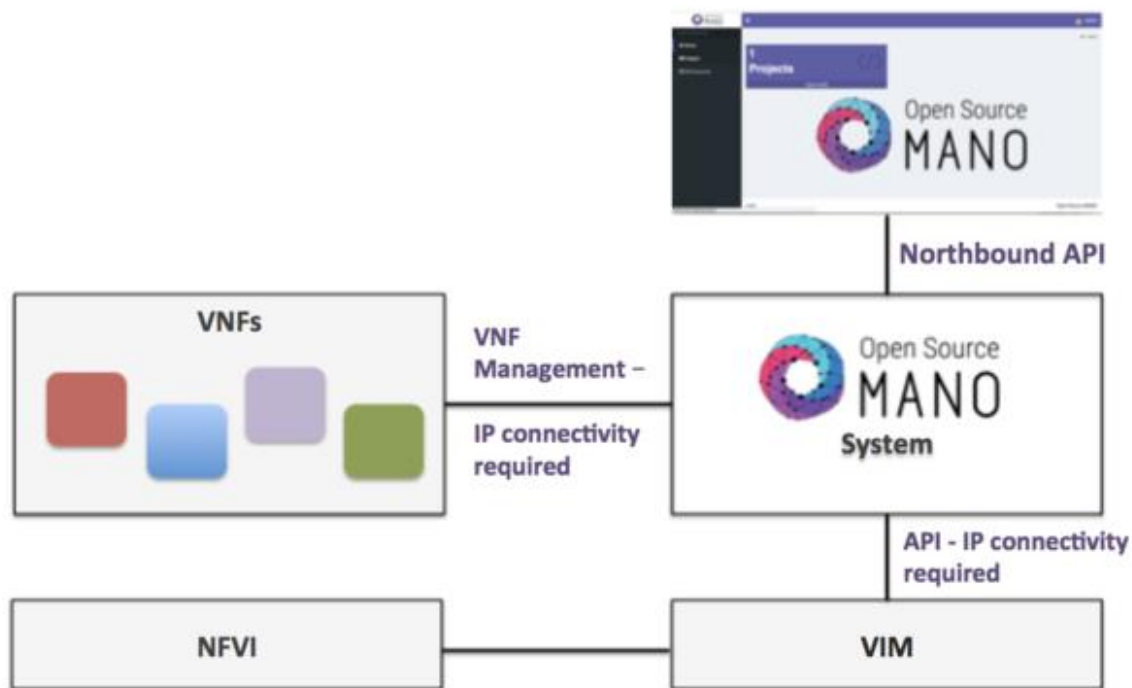


Figure 7: OSM interaction with VIMs and VNFs

It is possible to interact with OSM either through either the Northbound API or the Web user interface. The Web user interface includes a high-level interface for the establishment of NSs as well as various tools for configuring the network. Through the Web interface, VNF, NS and NetSlices can be defined on the infrastructure.

SONATA

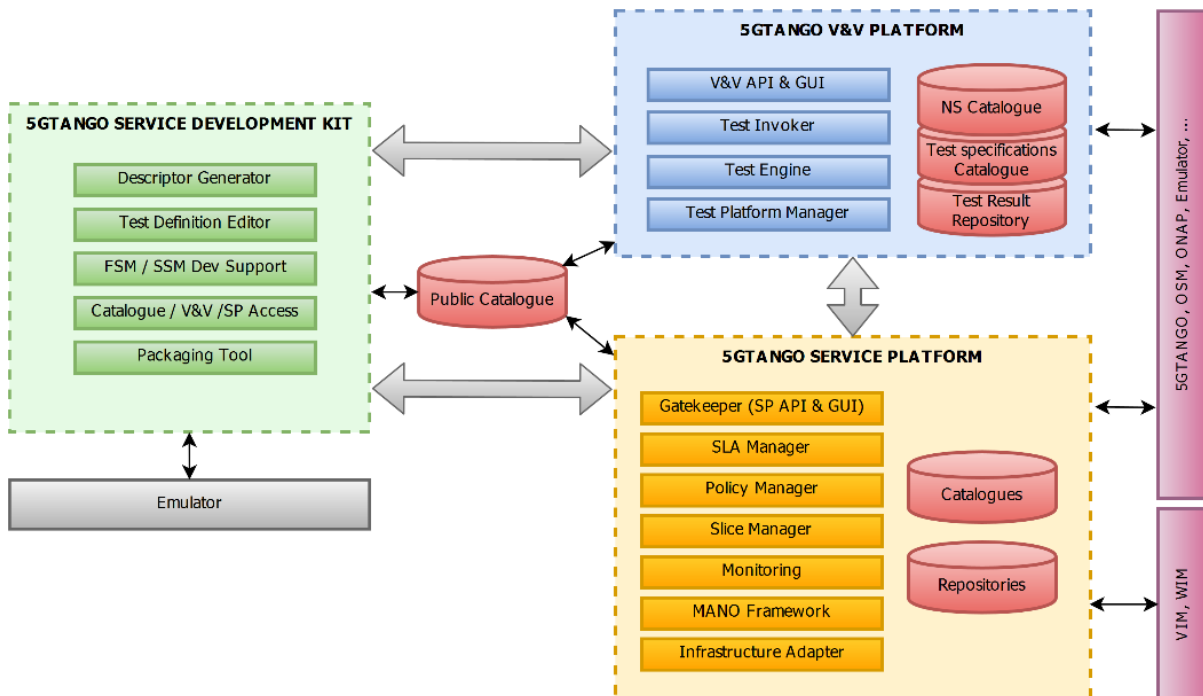


Figure 8: SONATA architecture

SONATA [13] is an open source Framework that has been developed in the scope of the SONATA 5G-PPP phase-1 project and extended in the 5GTANGO 5G-PPP phase-2 project. The framework has three main subsystems (as shown in Figure 8): (1) the Software Development Kit (SDK), which provides a set of applications to create packages, on-board NFV artifacts and provide CLI-based tools to manage the entire SONATA framework; (2) the Service Platform (SP), which implements a MANO platform that enables the management and orchestration of NSs, from on-boarding to instantiation, scaling or termination; and, (3) the Validation and Verification, which provides a platform to validate, test and benchmark VNFs/NSs before they go to production.

In particular, the Service Platform is a powerful MANO tool that enables operators, developers and customers to manage and orchestrate NSs in a flexible manner. The following is a list of the most relevant SONATA Release 4.0 features.

- MANO basic functions, enabling the on-boarding and lifecycle management of VNFs and NSs (VNFM/NFVO ETSI functions) by different roles (operators, developers and customers);
- Catalogue-driven, enabling the storage of VNFs and NSs, in a centralized Catalogue database (using package files, where software images, descriptors, etc. are included), to be later used for lifecycle management operations (e.g. instantiation, termination, etc.);

- Records Repository, stores runtime data in a Repository databased, where information about instances currently running as well as historic data can be obtained;
- Multi-VIM/WIM, enabling the support multiple VIM and WIM technologies, by using a plug-in approach that abstracts particular infrastructure technologies, facilitating the introduction new ones (Openstack, VTN are supported in this version);
- F/SSM, enabling the development of Function Specific Manager (FSM), for VNFs, and Service Specific Managers (SSM), for NSs, to create tailored behavior for lifecycle operations, according to VNF/NS specificities;
- Policy-driven, enabling the autonomous management (zero-touch service management –ZSM), through policies that take lifecycle management decision (e.g. deploying, scaling, healing) based on predefined rules, triggered by monitoring events (thresholds);
- SLAs Assurance, enabling the commitment of SLAs to NSs and checking whether any violation occurs;
- Monitoring, enabling the collection of monitoring data from VNFs and NS, feeding Policy, SLA and other components;
- RESTful APIs, exposing ETSI NFV aligned APIs for an external access to the SP capabilities, namely to on-board VNFs/NSs, instantiate/terminate NSs, get Catalogue data, get Repository records, or manage Policies and SLAs, among others;
- Management Portal, enabling the management by multiple roles (operators, developers and customers) of the main SP assets, on-boarding and managing the lifecycle of Network Slices, NSs and VNFs, as well as the managing Policies and SLAs.

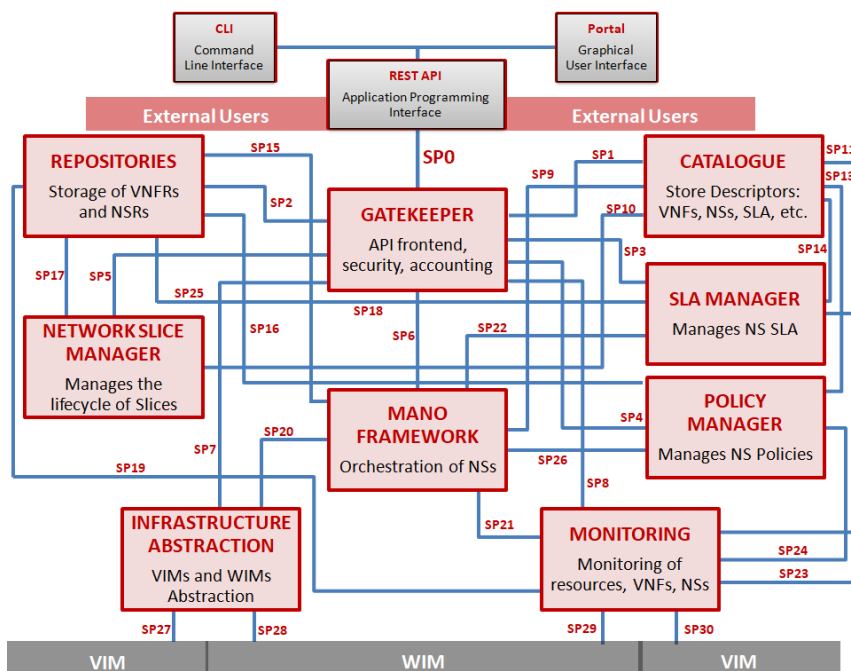


Figure 9: SONATA Service platform architecture

SONATA will soon make available the Release 5.0 (expected by end of July 2019), bringing significant improvements. The final architecture of the SP subsystem is depicted in Figure 9, where the main internal components (developed as microservices) and the interactions with each other are shown. In particular, the following are the main new features for this last Release:

- Network Slicing, enabling the deployment of Network Slices Instances (NSIs), created from Network Slice Templates (NSTs), which use a combination of NSs, either deployed in a single PoP or multi-PoP, to build the Network Slice;
- Deployment Flavours, enabling the design of different VNF and NS flavours (e.g. Gold, Silver, Bronze), making possible the instantiation of VNFs/NSs with different levels of performance and quality (e.g. the same VNF/NS can be deployed with 2 VMs or 10 VMs);
- Kubernetes as VIM, enabling the deployment of VNFs/NSs on top of Kubernetes-based (K8s) infrastructures (support also hybrid scenarios, e.g. mixed Openstack/K8s VNFs);
- Quality of Service, enabling the specification and enforcement of different levels of QoS, both within PoP environments and within WAN segments (QoS enforcement supports T-API and Openstack ecosystems);
- Ingress and Egress endpoint, enabling NSs to have ingress and egress WAN segments before the first/last VNF, making possible an NS to expand towards the customer premises, ensuring end-to-end QoS;

- Licensing, enabling the control of VNFs/NSs usage (support public, trial and private licenses);

Portal Enhancements, extending the number of operations supported via Portal.

2.2.4. 5G Mobile Core

Open5GCore

The Fraunhofer FOKUS Open5GCore toolkit [14] is practical implementation of the 3GPP 5G CN. It mirrors in a prototype form the 3GPP Release 15 for the CN functionality and its integration with 5G New Radio (Standalone and Non-Standalone). The Open5GCore serves as a consistent basis for 5G testbed deployments for trials and pilots.

Open5GCore implements the new 5G components as standalone, independent of the previous 4G EPC functionality. Through this, Open5GCore enables a fast and targeted 5G innovation, hands-on fast implementation and realistic evaluation and demonstration of new concepts and use case opportunities.

Open5GCore Rel. 4 includes a large level of newly implemented functions developed on top of an accelerated software platform:

- Integration with 5G New Radio: NSA [S1-MME, S1-U] and SA [N1, N2, N3]
- Implementing control-user plane split – PFCP [N4]
- First implementation of Service-Based Architecture Features [HTTP/2, OpenAPI, REST]
- Data path diversity supporting local offloading and backhaul control
- Maintaining backwards compatibility with LTE and NB-IoT [IDD and NIDD]

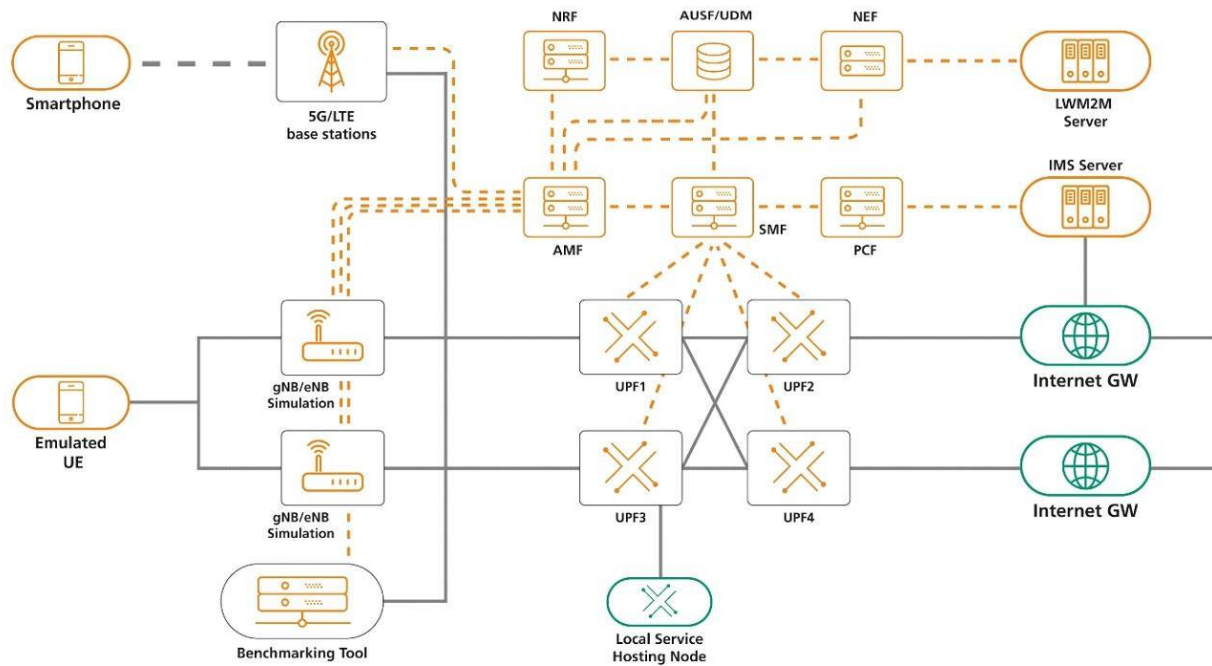


Figure 10: Open5GCore architecture

Open5GCore Rel. 4 integrates with 5G New Radio Stand-Alone (SA) and Non-Stand-Alone prototypes and off-the-shelf LTE access networks enabling immediate demonstration of different features and applications and supporting the current need to have a genuine 5G CN in addition to the evolved EPC one.

Open5GCore runs on top of common hardware platforms and can be deployed with containers or virtual machines on top of a large number of virtualization environments. The required hardware for a testbed setup, highly depends on the expected capacity.

Open5GCore is highly customizable, enabling the deployment of instances addressing the needs of the specific use cases. The source code license option extends the offer with ultimate flexibility for easy customization and prototype developments.

3. SECURITY IN THE SDN/NFV WORLD

In this Section, security in the context of SDN, NFV, and 5G networks is introduced. The section is organized into two parts. Sub-section 3.1 presents the new attack surfaces introduced by these technologies. Specifically, attacks that are unique to SDN networks and NFV architectures are introduced. Attacks on virtualization are also considered, since this is a core technology for NFV. A review of the state of the art concerning available solutions as well as open issues is also given,

Sub-section 3.2 discusses new solutions as well as opportunities enabled by SDN/NFV technology. Essentially, this Section deals with the new solutions to old issues, such as providing isolation, Distributed Denial of Service (DDoS) mitigation etc.

3.1. Attack surface evolution

The attack surface of a system is the combination of all possible interface (e.g. inputs, protocols, services). SDN/NFV introduce novel architectural elements, as discussed in Section 2. Each framework has its own architectural element, but there are logical similarities. As an example, the SDN application plane and the NFV OSS/BB layer both communicate with a control plane through a northbound API. Communications are made for different purposes and with different protocols, but they share the same model (i.e. application requesting a virtual service through an interface). Thus, in analyzing attack surface, it is useful to make shared considerations on these elements. In the following, the logical functions attack surface is discussed:

- **Applications plane:**

Threats at this level are due to corrupted/malicious applications. This threat category is not novel in the sense that is basically a classic application security problem. However, the impact is novel, in the sense that the corruption of an application results in abuse of the services offered northbound from the control plane (i.e. modifying forwarding in the case of SDN, or instantiating services and slices in NFV).

In terms of impact, attacks from the application planes can impact any of the infrastructure capabilities, depending on the authorization mechanism in place. Three configurations can be distinguished:

- Read-only access to the infrastructure state. In this case, consequences are limited, even if the information could be used to for preparing another attack
- Modifying the configurations only partially (e.g. only for selected nodes, service types). In this case, the attack's impact is proportional to the infrastructures that can be modified, but it is in principle possible to maintain some service leveraging the more protected part of the infrastructure.
- Modify configurations arbitrarily. In this case, the attacker has basically full control over the architecture.

Finally, malicious applications can also be used as a starting point to launch an attack on the control plane (e.g. buffer overflow to the controller).

- **Northbound interfaces:**

Threats in this category are related to vulnerabilities in the northbound protocols. Basically, a vulnerable northbound API can either allow unauthorized application to interact with the control plane or authorized application to perform unauthorized modifications.

- **Control plane:**

The control plane (i.e., SDN controller, NFV orchestrator, and VIM) represents a single point of failure for the architecture. The control plane is accessible through northbound and southbound interfaces as well as from administrative interfaces.

The control plane represents the main security concern, as it is basically vulnerable to classic attacks to OS/applications. This means that corruption of the control plane gives the attacker total control over the network. While this can cause all types of fault in the infrastructure, such as a complete DDoS, this situation can also be exploited in a less evident. As an example, an attacker could employ an advanced persistent threat (APT) that remains latent until needed and is thus very difficult to detect.

A less severe attack to the control plane is a DDoS based on saturating its computational resources, such as sending a consistent number of packets with spoofed IP addresses in an SDN network. Another potential scenario is data leakage, such as flow rule discovery (side channel attacks on input buffer), credentials management (keys, certificates for each Logical Network), forwarding policy discovery (such as packet processing timing analysis) or data modification, for instance flow rule modification that modifies packets, using Man-in-the-Middle attacks. The attacks are also a risk for data plane (SDN)

- **Southbound interface:**

As for the northbound interface, threats in this category are connected to improper authentication or authorization of the control plane. Basically, if the control plane devices are not properly authenticated, malicious modification can be made to the infrastructure. It should be noticed that this type of threats requires a direct connection to the underlying elements.

While this issue can be solved by implementing proper authentication between network elements (e.g. mutual authentication between the controller and a switch before accepting a new flow rule), it is important to note that for this type of communications communication overhead can be critical, and thus there are strong constraints on the complexity of the authentication protocols.

- **Data plane (SDN):**

Threats to the data plane vary in severity depending on the type of devices that are in use, If the switches are bare metal (i.e. they only forward traffic according to the flow rules) then the threat landscape is more limited. In this case, an attacker can try to perform an indirect DDoS to the control plane by sending a large amount of traffic to the switches.

Another possibility is to poison the topology discovery process. This attack is possible when the topology discovery does not provide adequate authentication of the data plane devices thus allowing the attacker to insert devices in the topology, redirect traffic flows and allow eavesdropping.

Many SDN implementations based on popular open source controllers were found vulnerable to these attacks [21]. However, if the switches are bare metal devices, the mitigation strategy can only be implemented in the controller.

On the other hand, if the data plane is composed of more complex (i.e. programmable) devices the same consideration made for the control plane apply, as the software stack can be attacked and corrupted. However, while compromising the control plane gives total control over the network, having control of a single data plane device has a smaller potential impact.

- **Virtualized infrastructure (NFV):**

Threats in this category are related to virtualization technology. Basically, they refer to the capability of an attacker to leverage a virtual machine instance to interfere with the security. For example, the attacker can generate a DoS attack by exhausting the physical nodes resources with malicious service instances.

Threats to isolation and virtualization mechanism should also be considered. Vulnerable virtualization mechanisms can enable data breach and disruption of services. While many of these risks are avoidable by following best practices [19], some of the vulnerabilities cannot be effectively addressed [17], representing an inherent threat to virtualized architecture that run un-trusted software, even if in a separate container/VM.

The status of the underlying nodes should also be considered, given that if the host machine is compromised, all the services running on it will be compromised as well.

Based on the aforementioned considerations and on the relevant literature [19][20][28], a set of best practices to cope with the new attack surface can be derived.

Table 1 provides a list of such best practices, divided by the domain of interest.

| Entity | Best Practices |
|---|--|
| Application Plane (SDN and NFV) | <ul style="list-style-type: none"> • Verify running applications (IDS, anomaly detectors) • Allow only application by trusted developers • Maintain a patch policy & procedure and update applications when new releases are available • Use strong encryption |
| Northbound interface (SDN and NFV) | <ul style="list-style-type: none"> • Enforce mutual authentication on the communications between applications and control plane • Verify authorization level for issued commands • Log accesses and requests |
| Control Plane (SDN controller and NFV orchestrator) | <ul style="list-style-type: none"> • Sanitize input from application layer • Hardening of the server running the control plane services • Minimization of the application running in the same server • Make administrative interfaces available only on private networks • Thorough logging of operations • Mutual authentication enforced in interfaces with other control plane elements • ACL-like mechanisms to control access to the control plane • Monitor for north and southbound |

| | |
|--|--|
| | <p>DoS (e.g. applications, devices blacklisting)</p> <ul style="list-style-type: none"> • Device registry to avoid topology attacks • Redundancy • Maintain a protected set of resources to be used for critical services • Change vendor default passwords, use strong passwords and frequently update them. • Use best practice when configuration is done, verify at least once a year if configurations recommendation changed • Develop Integrity protection mechanisms, code signing |
| Southbound interfaces (SDN and NFV) | <ul style="list-style-type: none"> • Enforce mutual authentication on the communications between underlying elements and the control plane |
| Forwarding Plane | <ul style="list-style-type: none"> • Device registry of authorized control plane devices • |
| Virtualized infrastructure (includes server running the control plane) | <ul style="list-style-type: none"> • Hardening of the hypervisors (see control plane best practices) • Separation of VNF and management traffic • Separation of VNF traffic and instances based on their security requirements • Secure storage of VNF memory, even in case of crash • Encryption of VNF virtual disks • Integrity check for VNF images |

Table 1 - SDN/NFV security best practices

3.2. Novel security and resiliency approaches

The attack surface introduced by SDN/NFV has been reviewed in the previous Section. In legacy networks, many security issues could be traced back to the lack of centralized control and awareness of the security functions. As an example, consider a security service (e.g. firewall, packet inspector) deployed in a legacy network: the effectiveness of the service is constrained by the capability of the other network element to route traffic through it. If a router is misconfigured or compromised, the effectiveness of the service is jeopardized. On the other hand, having a global view of the network situation means that unusual and malicious patterns in network traffic can be detected and analyzed

effectively. As an example, for DDoS mitigation and detection, the controller can perform periodic measurements through telemetry protocols. This opens up the possibility for network administrators to detect attacks timely and react accordingly, but also for controller to have algorithms and policies that enable automatic reaction. While this scenario appears appealing, it is important to notice that these capabilities have no software implementation to date. Effort is needed in order to leverage these opportunities. More specifically, while SDN/NFV represent the enabling factors for this security procedures, how these features are implemented is actually a critical point. New vulnerabilities could be introduced or performances could be degraded, if the development and validation processes are not properly handled. Generally speaking, algorithms must also be evaluated in terms of performances, pros and cons. Examples of SDN/NFV based security protocols and algorithms follow: In [21], a tool for detecting attacks through the SDN data plane aimed at the control plane is introduced. More specifically, the proposed solution monitors for the attacks on the topology and the DoS introduced in the previous paragraph by considering flow graphs. A flow graph is a visual representation of the connections between two endpoints (e.g. two hosts in a network). Basically, the proposed solution is a combination of a rule-based and anomaly-based Network intrusion detection system.

Central topics in network security are DDoS detection and mitigation, since DDoS are one of the most common attacks in the recent years [22]. Concerning detection, a variety of methods have been proposed. Some algorithms leverage the sudden spike in the distribution of the source IP addresses as a symptom of an incoming attack [24][25]. These approaches are labeled as entropy-based, as they monitor the information entropy of the distribution of the IP addresses. Other approaches leverage machine learning algorithms [23][26][27] to detect incoming attacks. These approaches can be further classified based on the algorithm and the features selected. Mitigation solutions have been proposed as well. In terms of mitigation, the novelty lies more in the automatic deployment of the solution than in the solution itself. This is due to the fact that, while existing mitigation techniques exist, their deployment during an attack is challenging. In [28], a modular, distributed scheme for mitigation is proposed, where multiple SDN controllers deployed over different domains (customer and provider) cooperate to enforce a mitigation strategy. Basically, the customer side monitors the traffic while the actual mitigation strategy (that can be loaded as a module) is enforced in the provider network.

The aforementioned approaches could appear as SDN-based solutions, with little involvement of NFV technologies. However, while the core technology involved might be SDN, this type of solution is an example of combinations of SDN and NFV functions (as the algorithms are NSs in many cases). Especially for DDoS, coordination across multiple network domains is essential for effective mitigation. Thus, an orchestrated NFV-like approach to security is the natural framework for integrating these solutions. The idea of security orchestration has been proposed in the past both in industry [29] and research fields [30][31]. However, proposed solutions are still in early development. To summarize, SDN/NFV offer a large number of advantages to the security domain, even considering the novel attack surface introduced by the architectures. Commercial efforts are mostly focused on transferring existing security functions and algorithms in SDN/NFV networks. In Academia, on the other hand, there has been an effort to design novel functionalities that exploit these architectures. These solutions, while being cleverly designed, are not easily integrable with the current software stack that enables SDN/NFV.

SDS, described in Section 4, is an attempt at modelling the security workflow, providing a clear framework to integrate existing security solutions as well as integrate novel algorithms to address future security challenges.

4. SOFTWARE DEFINED SECURITY

In this Section, the SDS framework is introduced. Specifically, the concept of SDS and its logical elements are introduced, with examples of past experimentation with the framework. The role of SDS in RESISTO and its adaptation based on end-users' feedback is reviewed. In more details, modification to the mitigation framework will be introduced, given the necessity of including an operator in the workflow. Finally, the ongoing research on extending SDS will be introduced. This will include experimentation with multi-objective optimization-based re-routing, virtual radio elements and enhanced mitigation DSS.

4.1. The SDS framework

SDS refers to a novel approach to security, enabled and inspired by modern networking technologies. One of the key advantages of SDN is the decoupling of the network functional planes. This separation, together with the *softwarization* of the control plane, enables the application of the classical software development workflow, where a solution can be developed once and then reused or updated, to networking. SDS shares similar vision: to develop a clear security workflow, composed of well-defined and separated logical functions, where security functions can be integrated and orchestrated. Key benefits of this approach are:

- **Scalability:**

Scalability and upgradability are essentially inherited from the underlying networking technology. Basically, to scale security function and services (e.g. deploy more instances of anomaly detector to address a traffic burst) is useless if the security resources are virtualized.

- **Upgradability:**

Being SDS based on a clearly defined, modular, architecture, it is easy to replace/upgrade any of the logical functions. This is due to the fact that SDS entities are built with modularity in mind, i.e. as a way to integrate different functionalities in a coherent manner, with clear endpoint.

- **Understandability:**

Making security understandable is crucial. Misconfigurations and various errors are a leading cause of security breaches. As an example, in 2018 approximately 35% of the Healthcare sector breaches were caused by misconfigurations [36]. This type of incident can be mitigated by building workflows and systems that are easy to understand and act on, so operators can have better awareness of the ongoing security status of the infrastructure.

- **Virtualization and abstraction of the security functions:**

Being able to virtualize and abstract further reduces the threats coming from misconfigurations. Basically, security actions are expressed as a set of high-level actions (e.g. separation policies for network traffic, or blacklisting). The SDS framework takes care of translating an abstract action into a set of low-level configurations to be applied to the nodes.

It should be noticed that it is possible to apply SDS in a variety of contexts (e.g. smart grids, physical infrastructures) to develop a unified model. As a matter of fact, the various security domains (i.e. physical, network, system...) cannot be treated as independent, as they can interact. SDS is designed to support a holistic approach to security, focusing on virtualization of resources.

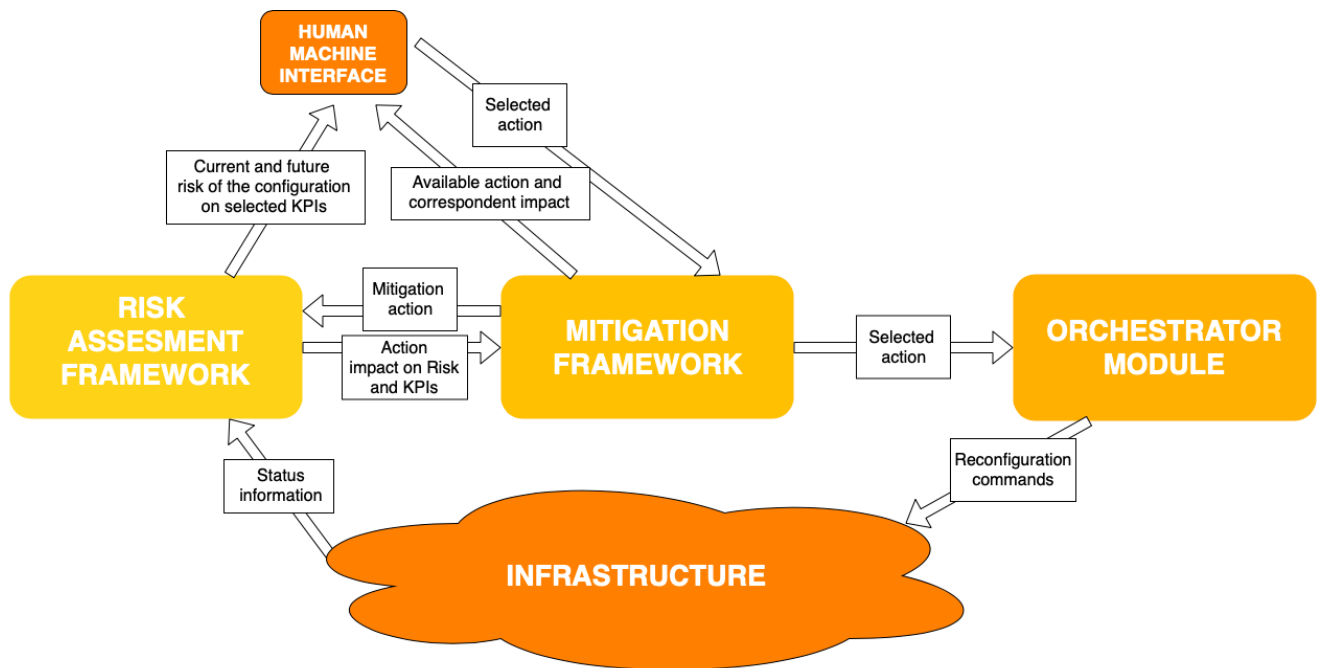


Figure 11: SDS conceptual architecture

The SDS conceptual architecture is shown in Figure 11. As can be seen, there are three main components: the risk assessment framework, the mitigation framework and the orchestrator. The key idea is to divide the security workflow into three logical steps: first, assess the current situation, evaluating the need for intervention based on the risk associated to a particular configuration. Second, evaluate possible strategies for reaction and rank them considering the impact on the system's Key Performance Indicators (KPI). The countermeasure is then selected, either by an operator or by an automated system. Third, apply the countermeasure, translating a high-level action into a low-level set of commands that are issued to the single apparatuses.

In more details, the SDS logical elements can be described as follows:

- **The risk assessment framework** has the purpose of evaluating the current status of a system, leveraging a variety of possible data and sensors. Basically, the framework is modular and allows to integrate different functionalities, based on the use case (i.e. the specific type of infrastructure). Example of modules that can be leveraged to assess the risk are network anomaly detector and sensors measurements (e.g. network probes, equipment degradation status). The measures are used to predict the potential impact of the current situation on the system KPIs. The risk assessment framework serves two main purposes. First, it enables system-wide situational awareness in complex situation. This means that operators can evaluate the impact of a certain event (e.g. anomaly, fault of a device, etc.) given the potential cascading effects and given the correlation between various other events. Second, risk evaluation can be used to evaluate the impact of a modification of the configuration, and thus support the other SDS elements in providing a ranking of the available actions.

An example of instance of Risk assessment framework can be found in the ATENA project [37]. In ATENA, the risk is assessed by means of the CISI Apro simulator. The platform allows for the integration of different variables and models the interdependency of different components of the

system. The output is the evaluation of risk as well as a prediction of the availability of key components.

- **The mitigation framework** is in charge of determining actions to mitigate risk and impact on the infrastructure's KPI. The idea is that an action that mitigates the effect of an attack often has a detrimental effect on performances. For example, during a jamming attack, the radio system could be made to use a more resilient modulation scheme (e.g. a QAM modulation with smaller constellation), improving robustness to noise but deteriorating the throughput. While the mitigation module can in principle function without human intervention, operators generally refuse to handle complete control of the system to software systems, given that malfunctions can happen and have harmful consequences. Thus, the mitigation framework is designed as a Decision Support System (DSS). The available actions are evaluated by multi-objective optimization to balance between impact on KPIs and benefits of an action, submitting the risk associated with the post-action configuration to the risk assessment framework. A list of actions (and the correspondent gain in risk and impact on performances) is then presented to the operator which elects which action is most fitting for the current situation
- **The orchestrator module** has the purpose of actuating the actions selected by the mitigation framework. Basically, the idea is to abstract the high-level action or policy from the low-level commands that are needed to actuate the action. The main idea is to enable abstraction in the actuation of mitigation actions, removing the associated complexities. Furthermore, the orchestrator provides a way to optimize the application of a countermeasure. As an example, re-routing a flow (a potential consequence of a mitigation action) in a network topology can be achieved in many ways, depending on the complexity of the topology. In this case, multi-objective optimization can be applied to find the optimal route for every flow, depending on the associated service. While this procedure can in principle be associated with the mitigation purpose, it is useful to separate the two phases. Basically, mitigation focuses on high-level KPIs (e.g. risk, overall vulnerability, performances on a higher level) while orchestration focuses on lower-level KPIs (e.g. bandwidth, energy consumption). This configuration provides better abstraction of the security workflow (and thus better understandability), and simplifies the optimization problems, potentially yielding better solutions in terms of time and efficiency.

As an example of orchestrator module, consider an SDN network. As a mitigation action, consider the removal of a node from the routing. This scenario, analyzed in the ATENA project, can be considered as a part of a more complex situation, where a node should not handle a certain flow (e.g. the traffic from a slice with isolation requirements). In this case, an alternative route has to be computed for the flow, and the correspondent forwarding rules have to be installed in the network elements. The orchestrator thus acts by communicating with the SDN controller and providing the new forwarding rules to be installed in the data plane.

4.2. SDS in RESISTO

The SDS framework, in RESISTO, is part of the advanced security services in the short-term control loop in RESISTO platform. In more details, as mentioned in D6.1, SDS is focused on the evaluation of the risk, the selection of the most appropriate mitigation options, based on a what-if analysis and the actuation of the selected mitigation actions.

The high-level analyses performed on the signals extracted from the already in-place (or added) probes monitoring the physical and cyber security status of the infrastructures allow to dynamically estimate the presence of cyber-physical anomalies. By feeding the risk-predictor, described in D4.4

[2], it is possible to evaluate the impact of an issue on the complete TLC infrastructure, also considering cascade effect. This represents an instance of the SDS risk assessment framework.

The output of the analyses is used for supporting the decision-making process, through a what-if analysis tool for determining the best reconfiguration. The reaction and mitigation strategies are described by means of workflows (composed of directives to teams, physical devices activation, software activation, etc.). The operator will select the most appropriate workflow that will be sent to the orchestrator. This sub-system represents an instance of the SDS mitigation framework. Research on this component is still ongoing, with the aim of supporting the operator with visualization of actions and their impact. The described system represents the current version and is subject to change. Details are given in Section 4.2.1.

The selected action is then applied by the orchestrator. The orchestrator oversees communication with the underlying resource managers (e.g. SDS controller, NFV orchestrator, etc), issuing commands to them to apply the action. The orchestrator provides an optimization of the selected actions implementation, based on the action type, the underlying infrastructure and the type of service. The orchestrator provides a fast application of the action to minimize Decision-Making Time (from D3.6 [3], KI 2.2) and the Average Mitigation Time (from D3.6 [3], KI 2.2), two of the RESISTO's KPIs. This represents an instance of the SDS orchestrator module. Research on the orchestrator implementation is still ongoing. The described system represents the current version and is subject to change. Details are given in Section 4.2.2

One of the purposes of RESISTO is also to extend the SDS model in terms of its current capabilities (develop better algorithmic solutions) and its applicability to other domains (e.g. physical security). Section 4.3 summarizes the current situation and offers an overview of future activities.

4.2.1. Mitigation framework

The mitigation framework is based upon the workflow engine. The workflow engine is a tool to support operators for managing critical infrastructure (CI) security. It is based upon a Business Process Management (BPM) model. The workflow engine is useful for the configuration and execution of automatic or semi-automatic processes, consisting of sequences of actions, which can be triggered by defined events. A workflow is a conditional sequence of actions. Actions can be executed either by the security operator or security team or be triggered automatically. More specifically, an action can:

- Be performed directly by the control room operator.
- Address a procedure activating a reaction team.
- Trigger an actuator (e.g., lock a physical gate, perform a complex action on the communication network).

Given a certain alarm/event, the most appropriate workflow is activated.

The workflows definition is based on Business Process Model and Notation (BPMN). BPMN is a standard for modelling of business processes that provides a graphical representation of the processes like activity diagrams designed using UML. The goal of BPMN is to provide support for the management of business processes to both "technical" and end users via a notation that is intuitive but also able to represent complex processes.

Here four passages can be identified:

1. Event job start (left circle).
2. Two *tasks* to be executed in sequence by a user.
3. Event of the end of the process (right circle with thick border).
4. Running stream represented by arrows that connect the elements.

Every BPMN process can be represented on an XML file.



Figure 12: BPMN task and process

In accordance to BPMN standard, a workflow is designed based on two concepts:

- Task
- Process

A BPMN task is an atomic activity which represents work that is not broken down. On the contrary, process represents work that is broken down to a finer level of detail (Figure 12).

A task is created when the work in the process cannot be broken down to a finer level of detail. For instance, sending an email or requesting the operator to insert some data may be tasks.

An example of process is shown in Figure 13.

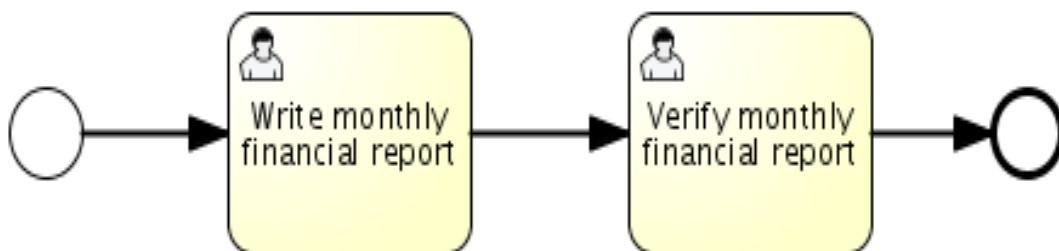


Figure 13: BPMN Process example

You use a process when you want to model the internal details of work in a lower-level process diagram. The process defines the timing for task execution. It may contain a block of parallel tasks, alternative blocks to be executed depending on session variables and so on. An example of process and task is depicted in Figure 14.

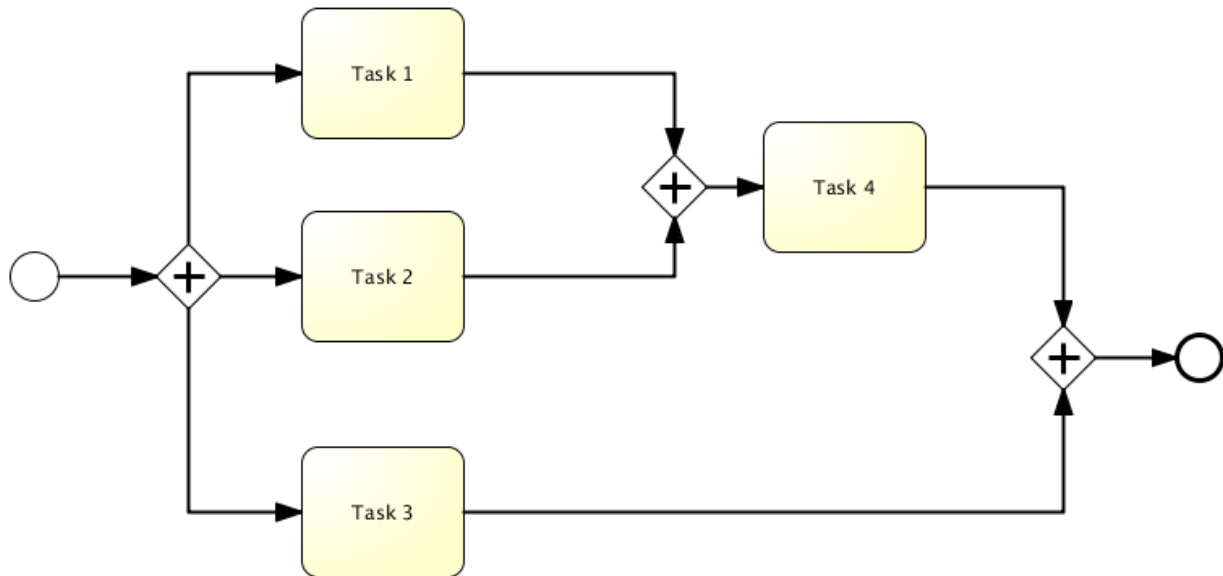


Figure 14: Example of process and task

Workflow definition is performed off-line. A graphical user interface (Figure 15) allows the definition of the most appropriate workflow to face a specific event/alarm type.

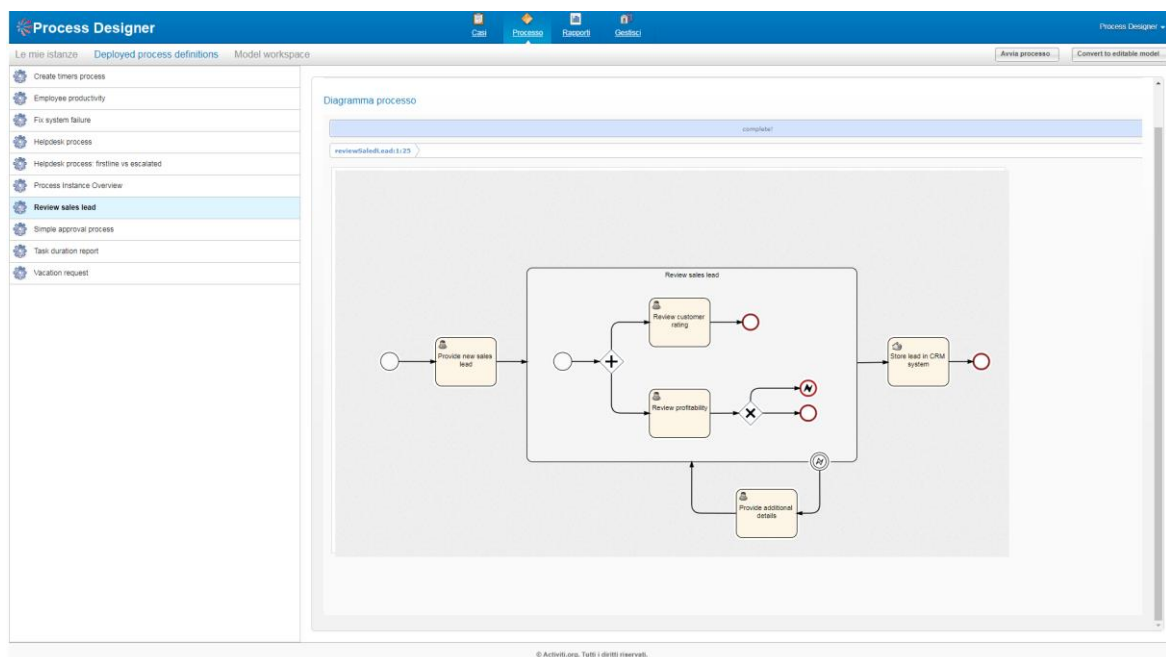


Figure 15: Process designer web HMI

The output of the workflow's definition is a set of XML files. The files are stored in a workflow engine own database available for the workflow execution engine (Figure 16). Workflow redefinition, upgrading and refining are allowed during the system lifecycle and no software coding activity is strictly needed.

At runtime, the workflow execution is carried out via Activiti, an open-source workflow engine written in Java that can execute business processes described in standard BPMN 2.0. At run time, the Workflow Manager receives the alarm trigger for activating a specific sequence of tasks on a specific ActiveMQ queue. The relevant workflow is executed by the Activiti engine.



Figure 16: Activiti Engine

The specific alarm lifecycle (New, In progress, Closed) is tracked and managed through the publication of its state updates on the corresponding ActiveMQ topic in order to be also displayed on the HMI.

The alarm management interface is shown in Figure 17.

| SINOTTICO ALLARMI | | | | | | | |
|-------------------|---------------|-----------------------|-------------|--------------|--------------|------------|--------------|
| 8 | Alarms | Manage Alarm | Close Alarm | Assign Alarm | Write Report | Mute Alarm | Clear Filter |
| | Source | Date | Status | Type | Location | Operator | |
| 3 | Moxa 2214 CH0 | 10/23/2018 3:45:45 PM | NEW | Moxa Alarm | Zona2 | | |
| 3 | Moxa 2214 CH0 | 10/23/2018 3:10:26 PM | NEW | Moxa Alarm | Zona2 | | |
| 3 | Moxa 2214 CH0 | 10/23/2018 3:05:32 PM | NEW | Moxa Alarm | Zona2 | | |
| 3 | Moxa 2214 CH0 | 10/23/2018 3:03:34 PM | NEW | Moxa Alarm | Zona2 | | |
| 3 | Moxa 2214 CH0 | 10/23/2018 2:53:30 PM | NEW | Moxa Alarm | Zona2 | | |
| 3 | Moxa 2210 CH0 | 10/23/2018 2:52:06 PM | NEW | Moxa Alarm | Zona1 | | |
| 1 | | 10/23/2018 2:16:21 PM | NEW | Help Alarm | | | |

Figure 17: Detail of opened alarms

From the interface it is possible to manage an alarm and this operation can run a workflow. The workflow can have two kind of tasks:

- Automatic tasks: task that run automatically without any interaction with the user;
- Manual task: task that need a human interaction to be completed.

If a manual task is running the operator has to perform a manual action on the HMI to complete the task. The user operation can be for example a click on a button write a text, select a checkbox, combo or radio button (Figure 18).

Figure 18: Execution of a user task

When a task is completed by the user the next task will be presented, when all the tasks are completed the user can proceed to close the alarm. The interface shows the workflow flow chart (Figure 19) with the indication of the task in progress to give the user an indication of the workflow progress and also a list of the action completed (Figure 20).

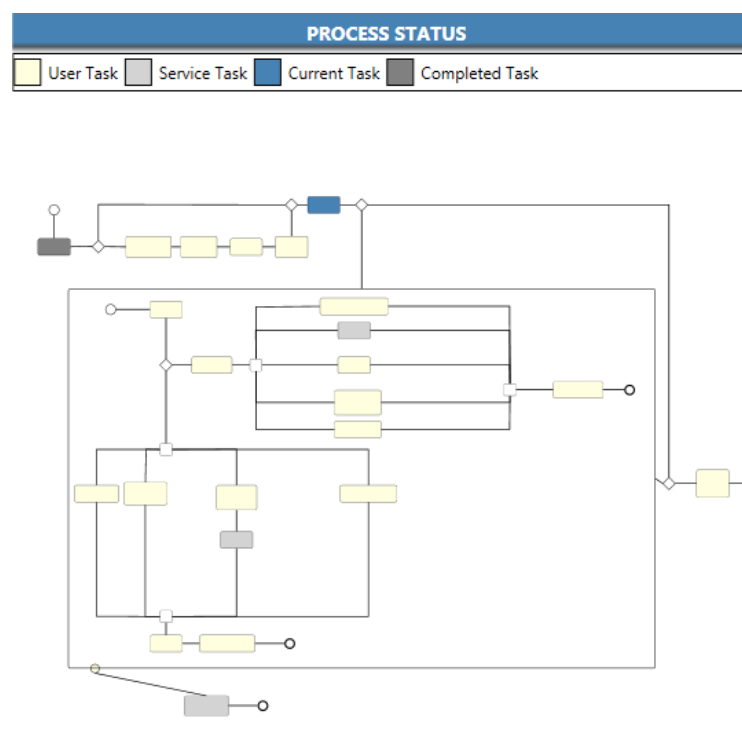


Figure 19: Workflow flow chart

All operations performed by the user on the interface and all automatic tasks are recorded in the database and can be investigated through the Action report.

In RESISTO the Workflow engine will support operator to react to detected alarms, detected from the Physical/Cyber Event Correlator, with workflows specifically designed to face each type of alarm. Some workflow tasks aim to support the operator in decision-making, suggesting evaluating:

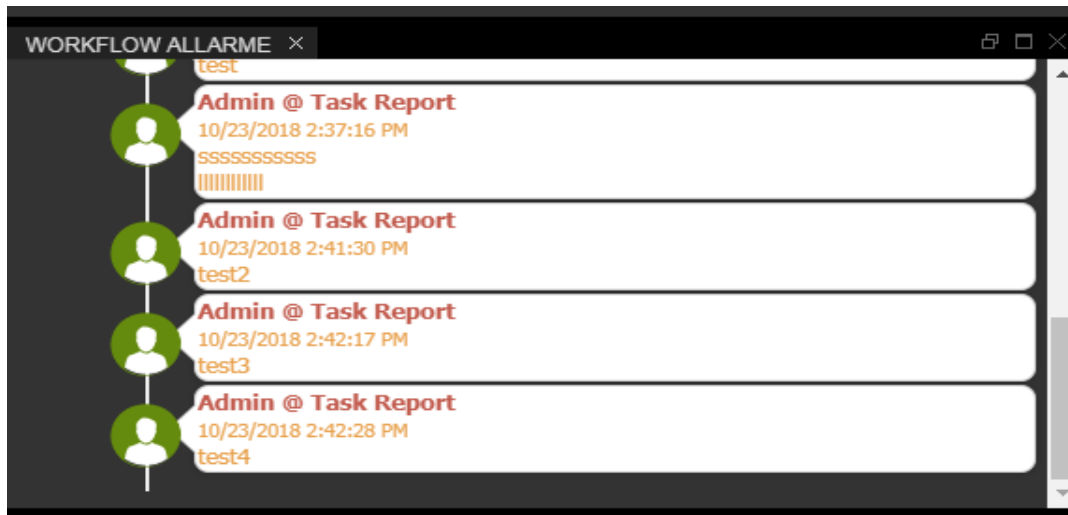


Figure 20: Action list of a workflow

- The situation and the alarm impact prediction provided by Risk Predictor.
- Different mitigation action alternatives, performing a what-if analysis.

Other workflow tasks and processes support the operator to react to the alarm by:

- Suggesting direct operations to be done by the operator himself.
- Commanding operations to security and technical team on filed by mean of instant message system provided by the Emergency warning Communication Function (EWCN).
- Requesting the application network actions, implemented by the orchestrator.
- Triggering a physical actuator (e.g., lock a physical gate).

During workflow execution the engine can request services to EWCN and to the orchestrator through REST protocol commands. Any actuator, equipment or Internet of Things (IoT) device providing a REST services can be controlled through workflow engine.

For the RESISTO project the workflows necessary for processing the alarms will be implemented. Each of the managed alarms will have a corresponding workflow or more, depending on its complexity.

The workflow engine represents a powerful tool to deal with alarms with a straightforward mitigation procedure. It is also a mature and dependable tool, a critical feat in the security domain. As part of the activities of task 5.2, augmentations of the current framework are being evaluated. More specifically, activities on the mitigation framework are currently focused on two objectives:

- To provide a complete spectrum of solution for alarms that can be handled with different actions. This is especially useful when dealing with certain cyber threats, that can have many different mitigation strategies.
- To provide tools to rank actions based on their impact on the infrastructure KPIs and their impact on the overall risk.

In this way, the operator will have full situational awareness of the mitigation strategies available, enabling more effective and secure security workflows. Clearly, in this situation, solutions can generally not be considered optimal, as there are multiple KPIs that do not allow contemporary maximization. To provide decision support to the operator, a multi-objective optimization framework for evaluating and ranking the possible actions is being evaluated. Multi-objective optimization, described in Section 4.3.2, provides a framework to combine different indicators and find solutions that are balanced with respect to each. In the case of the decision support system (DSS) (previously described as optimal decision-making module), it is however non-trivial to determine the cascading impact of changes over the infrastructure. Activities in this sense are still ongoing and will be reported in WP5 final deliverable.

4.2.2. Orchestrator module

In RESISTO, the orchestrator activates mitigation actions and countermeasures in response to commands from the mitigation framework. Actions and countermeasures are dependent on the context in which the RESISTO platform is deployed. As an example, countermeasures on a datacenter network can be very different from the ones available to a radio access site. At datacenter level, multiple network segments must be orchestrated with virtual nodes running services. On the other hand, a radio access site could manage virtualized radio hardware and a smaller network portion. Thus, actions and countermeasures are different based on the use-case. While a network built on a modern paradigm enables a better approach to security, as reviewed in Section 3, the concept of the SDS orchestrator can be adapted to different situations, while still reaping the benefits of the SDS approach.

Three different configurations of the orchestrator are being developed. The first will demonstrate SDS in the context of a NFV/5G enabled network. The second will show SDS in an SDN network, demonstrating the advancements made in the project in the development of SDS. Finally, the use of Orchestrator over a legacy network will be demonstrated. In all the configurations, the orchestrator will be based on open source software. This enables the best compatibility with existing solutions, as well as better upgradability. More specifically, this is how the orchestrator will be deployed in the configurations:

- **Orchestration in an NFV+SDN network**

In this configuration, the overall network is composed of physical infrastructures (computational, storage, networking), virtualized by a VIM. There are one or more underlying SDN networks, commanded by the corresponding controllers. The network can provide NFV services as well as some 5G services (e.g. slicing).

The orchestrator is built upon the ETSI Open Source MANO (OSM) definition (<https://osm.etsi.org>). The basic role of the orchestrator is to implement actions from the mitigation framework. Actions can include the definition and management of VNF, NS and Network slices on the TLC infrastructure. The orchestrator will define Network slice and NS templates that satisfy security constraints (e.g. do not use a certain physical resource) and optimize the configurations to minimize impact on the service. The algorithms will include support of hard constraints, such as Service Level Agreements (SLA) specified thresholds.

- **Orchestration in an SDN network**

In this configuration, the overall network is composed of an SDN-enabled data plane and an SDN controller. Clearly, this represents a smaller scale with respect to the previous one.

The orchestrator is based on Node-red and leverages ONOS as Network controller. In this case, the orchestrator can define and validate network topologies, recalculating routes that satisfy specific constraints. It will be possible to blacklist network elements, either completely or for certain type of services. Furthermore, it will be possible to support a of level trust for each node and define routing policies accordingly. This will allow to implement an ACL-like system, able to perform fast and effective reconfiguration of the data plane.

Reconfiguration will be based on the input from the mitigation framework. In this way, the network is able to react to ongoing attacks effectively by following a pre-defined reaction strategy. Furthermore, the definition of new reaction strategies to support un-anticipated threats will also be available.

- **Orchestration in a legacy network**

In this case, the does not offers any standardized way to perform dynamic reconfiguration. Clearly, this represents the worst-case scenario for the SDS approach, as effectively nothing can be software-defined.

In this case, the insertion of a limited number of SDN devices in critical interconnection points will be suggested. In this way, it will be possible to leverage the advantages of an SDS approach with a minimal investment. The orchestrator configuration will be the same as the one specified for SDN networks. In this case, however, the possibility to optimize deployment of countermeasure by orchestrating the behavior of all network elements is lost. ACL-like behavior is still possible, albeit dependent on the placement of the switches, that effectively become a security middlebox.

4.3. Extension of the SDS model

The main activity of Task 5.2 is the extension and development of the SDS model. Specifically, two main objectives have been specified:

1. consider physical security;
2. provide interface between SDS and legacy networks;
3. support 5G communications (slicing) as well as general multi-tenant scenarios.

As described in the previous Section, the mitigation framework is able to control physical actuators by means of the workflow engine. Furthermore, as part of the WP4 activities, physical intrusion detection system based on machine learning re being developed. In this Section, further research activities on support of SDS to physical security functions are presented. Specifically, Section 4.3.1 presents efforts made to virtualize radio functionalities and sensors, so that they can be included in a unified SDS platform.

For objective 2, an approach based on the insertion of SDN devices has been developed.

Concerning objective 3, Section 4.3.2 describes the efforts to provide a multi-objective based optimization procedure for the application of the countermeasures. The ongoing activities aim at providing solutions that optimize the deployment of action, considering the requirements specific to the type of service involved as well as hard constraints imposed by SLAs.

Finally, Sections 4.3.3 and 4.3.4 describe additional research work carried out concerning machine learning-based physical security, that could be integrated into the SDS model.

4.3.1. Software Defined Radio Security

In this section, an outline of the functionality and software-defined capabilities of RAN-MONITOR from the perspective of Software Defined Radio Security is given. RAN-MONITOR is one of the sensor-based tools being developed and used in RESISTO for CI physical protection.

RAN-MONITOR is a cellular radio access threat detection and alarm generation stand-alone system for massive deployment which allows all cells and operators to be monitored in real time in one target area.

The architecture of RAN-MONITOR is based on two main modules needed for the radio signal processing and detection. One is a multi-band, multi-RAN cellular modem and the other is a Software-Defined Radio receiver. The most complex and computationally expensive operation takes place in the latter so the focus will be on this part. A Software Defined Radio (SDR) is a radio system where some key hardware-based components such as filters, amplifiers, modulators or demodulators are implemented through software on a computer.

RAN-MONITOR is a system that can be used by several concurrent end-users in real-time. Therefore, in order to improve the multi-cell radio spectrum capabilities of RAN-MONITOR (i.e. number of cells to be monitored in real-time) and/or to reduce number of SDR nodes complexity, the radio monitoring resources of SDR devices should be virtually shared.

The main application could be creating a slice for each operator in a shared-RAN multi-operator scenario or creating multiple slices for the same operator in order to monitor multiple radio spectrum signals from different cells at the same time. Virtual Radios (VR) (front-ends) should be associated to each of the mentioned radio cell slices.

In order to achieve this, a radio virtualization scheme should be implemented to achieve such flexibility, enabling multiple VRs to be instantiated on top of one or more shared SDR by means of abstraction and slicing methods. In other words, According to [33] radio virtualization is the process of abstracting a physical radio and slicing it into VRs holding certain corresponding functionalities and isolating each other.

This scheme is made up of two main entities: the physical front-ends and a physical orchestrator.

The physical front-end includes all the hardware RF components whereas the orchestrator oversees abstracting the RF front-end physical resources into Virtual Radios. The orchestrator can be interfaced with external systems for remote real-time configuration.

The main requirements the radio virtualization scheme for RAN-MONITOR should have are summarized below:

- **Coexistence:** The orchestrator should coordinate and check that multiple VR instances can coexist on the same physical unit (RF front-end).
- **Isolation:** The orchestrator should coordinate and check that any VR misconfiguration does not affect or interfere with other coexisting VRs.
- **Dynamic Configuration:** Radio cell slices (VR instances) should be created, destroyed or modified by the orchestrator according to dynamic system needs.
- **Physical-like capabilities:** VR instances should have the same set of configuration functionalities as its core physical front-end including but not limited to center frequency, bandwidth and gain.

- **Small Computational Overhead:** In terms of system CPU efficiency the computational overhead created by the radio virtualization implementation should not translate into a significant amount of computational overhead, rather this figure should be kept to a small percentage of regular physical-only operation.

In summary, Radio virtualization has the following main benefits:

- **Automation and SDN Interoperability:** The orchestrator can be further extended to include machine learning algorithms that enabled increased operation performance and automation of the tool and interoperability in a SDN platform.
- **Fast Solution Development Cycle:** Only software programming is required for creating and modifying multiple radios (slices)
- **CAPEX and OPEX Reduction:** The amount of physical resources for monitoring a target area is reduced by using radio virtualization which in turns means reduced capital expenditure. Another benefit is that maintenance of the system can be more automated and simpler, thus reducing OPEX.
- **Efficient Utilization of Hardware Resources:** The front-end resources not being used in a regular operation mode can be assigned to several virtual radios.
- **Multi-radio, Operator and Band in a Single Device:** A single fully-featured SDR front-end enables operating in multiple scenarios in terms of technologies, frequencies, etc.

Although radio virtualization is an emerging topic there have been some interesting efforts in the scope of H2020 projects so far, including the following: a hypervisor for SDR platforms, known as HYpervisor for software Defined Radio (HyDRA - in WiSHFUL project) [35] and a demo showcase running two Wi-Fi, eight Zigbee and one BLE concurrent instances in 40MHz bandwidth from the same set of FPGA resources (Xianjun Jiao) in ORCA project [34].

4.3.2. *Multi-objective countermeasure optimization*

In RESISTO, novel algorithms for security orchestration are being developed. Specifically, research on the orchestrator has been carried out with the following ideas in mind:

- Develop solutions exploiting existing systems.
- Develop solutions addressing novel threats (e.g. attacks on virtualization techniques).

The focus has been on re-routing problems where security is involved, their extension to complex SDN use-cases and their generalization to networks for NFV/5G.

The first activities were devoted to the re-routing problems, studied in ATENA, and their application in complex topologies, with different service types. Specifically, if the analyzed topology is limited, the orchestrator would have to compute a new path for a flow, leveraging protocols that minimize and aggregated performance indicator (e.g. cost of a link). However, depending on the topology and the type of services, aggregating all parameters in a single cost indicator can reduce efficiency. Basically, as the network topology grows (i.e. number of nodes and links), the number of configurations satisfying the constraint imposed by the mitigation framework grows. Different links and different nodes have different performances while every service has its own constraints (e.g. a voice service flow, that requires low latency but can tolerate higher packet losses). This information is lost by aggregating the cost. This worsen the ability to engineer traffic, has a higher fault probability due to over-use of the “optimal” path and generally to a wasteful use of resources.

Many requirements can be considered. From the point of view of the service provider, beside avoiding SLA infringements (i.e. providing the agreed service level), other aspects are relevant:

- Energy efficiency

Network nodes may need a considerable amount of electrical energy to function. Optimizing the overall number of switched-on nodes in a topology may have a considerable impact on power consumption and thus represents an important indicator to optimize.

- Workload of selected link and nodes

For each node and link, the maximum workload that can be handled is defined. However, equipments reliably operate only up a percentage of the maximum workload capacity. From the reliability point of view, redundancy mechanisms (e.g. switching on a secondary node) should be used when the actual load reaches the limit. This represents a conflicting objective with the energy efficiency one.

- Isolation constraints

As described in Section 3, an increasing number of successful attacks have been demonstrated against logical isolation mechanisms (virtualization, VPN, sharing the same physical infrastructure). From the security point of view, critical services (e.g. mission critical communications during an emergency) should not share the physical infrastructure with untrusted services. Different services have specific security requirements, that can require various degrees of isolation. This can impact energy efficiency as well as performances for involved, non-critical flows.

- Trust

As described in Section 3, the rise of programmable network elements means that more and more elements can be compromised via software attacks such as APTs. It is thus advisable to deploy anomaly detection algorithms that evaluate the behavior of nodes and output a trust metric as a result. Minimizing the probability of a data breach would require leveraging trustable network elements for services with strict security requirements.

It is not possible to optimize all the indicators simultaneously, as many of them are conflicting. In case of multiple, conflicting, objective functions the concept of Pareto-optimality is used. Basically, in a Pareto-optimal solution, it is not possible to improve any of the objective functions without worsening the performances on one of the others. The set of all the Pareto-optimal solution is called the Pareto front or set.

The definition of Pareto-optimal solution can be complemented with the notion of dominated solutions. Basically, given two solutions for a multi-objective optimization problem P x_1 and x_2 , x_1 is said to dominate x_2 if x_1 is a more efficient solution to the problem P for every indicator considered. It is better to consider non-dominated and Pareto-optimal solutions, as considering only the latter can yield solutions that are stable point in the optimization (analogous of the concept of local minimum).

Multi-objective optimization enables approaching the routing problem without having to neglect any of the indicators and including SLA's constraints by means of inequality constraints (i.e. imposing that a certain indicator must be lower/greater than a certain value). The problem can be specified with the following notation:

$$\begin{aligned} & \text{minimize } (f_1(x), f_2(x), \dots, f_n(x)) \\ & \text{subject to } g(x) < n, \dots \end{aligned}$$

To solve multi-objective optimization problems where the number of possible solutions is too high to be evaluated completely, evolutionary algorithms can be used. Basically, evolutionary algorithms are based upon the evolution mechanisms found in nature. They find solution by iteratively evaluating the fitness of individuals (that represent possible solutions to the problem), mating the individual (i.e. combining the features of two different individuals) and mutating randomly the features of the individual with a small probability. The problem is specified by the following functions:

1. Population generator:

A random set of individuals is generated. Here, an individual represents a possible solution of the problem P. In this case, an individual is a path for a flow. Special care must be taken to ensure that the individuals are composed of valid paths (i.e. sequence of nodes that have links between them). Individuals are characterized by their genome (i.e. a vector where each element is a gene, a node in the path).

2. Selection strategy:

The selection strategy represents the ensemble of criteria used to choose which individuals should be used to breed the offspring. It is usually a function of the fitness.

3. Fitness evaluation:

The fitness of the solution is evaluated, based on the functions f_i . In this case, functions could be total latency, maximum nodes workload etc.

4. Crossover:

New individuals are generated by mating (i.e. combining) individuals in the current population. Special care must be taken to ensure that the resulting individuals are still valid paths. Fitting crossover functions must be selected or created for this purpose. Individuals to be mated can be selected according to various criteria (e.g. individuals with high fitness, diversity).

5. Mutation:

The mutation phase introduces random changes to the individuals' genome with low probability. Even in this case, care must be taken so that the mutation phase does not generate non-valid paths.

The ongoing experimentation aims at determining routes at SDN level and Virtual Network Function Forwarding Graph at NFV level by means of multi-objective optimization-based routing. This type of action supports different security and resilience scenarios:

- A part of the network is not available anymore due to accidents (e.g. fire), faults or natural disasters. New paths must be computed with the exclusion of a segment of the network.
- Some of the nodes become unavailable due to security reasons. Possibilities include compromised nodes or physical isolation constraints for critical communications.

The benefits of this approach to routing are not limited to reaction strategies, however. More generally, the proposed algorithm is a way for determining effective configurations for the network while being able to specify soft and hard constraints.

4.3.3. Machine learning-based positioning for Physical Security

In this section, a summary of the performance and capabilities of RADIOFILTER from the perspective of Machine Learning is given. RADIOFILTER is one of the sensor-based tools being developed and used in RESISTO for Critical Infrastructure (CI) physical protection which detects and roughly locates unauthorized devices using probabilistic techniques.

Networks are intrinsically insecure from unauthorized users, hence Wireless Local Area Network (WLAN), according to [0], has become a promising choice for indoor positioning as the only existing and established infrastructure, to localize the mobile and stationary users indoors. However, since WLAN has been initially designed for wireless networking and not positioning, the localization task based on WLAN signals has several challenges. Among the WLAN positioning methods, WLAN fingerprinting localization has recently achieved great attention due to its promising results.

In general, the RSS exploitation approaches are divided into two broad categories: model-based (path loss) and model-free (radio map) approaches. The radio map based techniques, also called fingerprinting techniques, make use of monitoring sensors deployments in indoor areas.

The fingerprinting methods are based on the singularity of the radio signals received in different positions, due to the problems of propagation in the complex interior environment.

There are two types of fingerprinting methods: (i) deterministic and (ii) probabilistic. A single RSS fingerprint may not be a sufficient representation of the data because of the time-varying nature of indoor propagation. The performance of deterministic localization approaches can be improved if instead of a single representative RSS fingerprint, all fingerprints are used. In probabilistic approaches, the whole ensemble of RSS fingerprints is utilized to provide statistical characteristics of the area.

RADIOFILTER is a system that can be used by several concurrent end-users in real-time. It is focused on the access control layer to the Wi-Fi medium (MAC) for the analysis of wireless networks and attempts to provide a reliable and accurate indoor localization service. The architecture of RADIOFILTER is based mainly in two modules: detection and localization.

For the detection, RADIOFILTER, through the monitoring sensors, will passively scan all packet traffic of which you can extract identifying information and signal levels from devices and access points.

For the localization, the probabilistic technique known as Fingerprint Matching supported by Machine Learning algorithms will be used. RADIOFILTER consists of offline and online phases. The basic idea is to create a radio RSS (Received Signal Strength) map of the interior of the building in the offline phase to establish the problem of localization as an optimization problem in which given the RSS values in a particular localization, a learning phase is carried out that will help, later, to achieve the estimated localization in the online phase.

First of all, a set of predefined points, referred to as Reference Points (RPs), also called landmarks, are selected. During an offline phase, a survey is conducted and multiple copies of RSS measurements are read at each RP from available sensors throughout a time interval. The database of fingerprints for all RPs makes a radio map for the whole area.

Then during online phase, the user observes RSS measurements at their location and applies algorithms to associate these measurements to the radio map entries finding similar fingerprints, and using associated RP locations for estimating the user's position. The learning data is fed into a machine learning algorithm that can do classification with probability. There are currently 10 classifiers that would be potentially enabled: Classifier implementing the k-nearest neighbors vote, Support Vector classification (linear), Support Vector classification (gamma), Decision tree classifiers, Random forest classifiers, Multi-layer perceptron classifier, AdaBoost classifier, Gaussian Naive Bayes, Quadratic discriminant analysis and Extended Naive Bayes. Random forests could be a good candidate algorithm for doing this type of classification in RADIOFILTER as it has the highest specificity and sensitivity of any other algorithms listed above providing the most probable location according to the knowledge acquired in the learning phase.

Fingerprinting emerges as a straightforward and plausible alternative offering both accuracy and ubiquity (all modern smartphones come with Wi-Fi capabilities) and Machine learning algorithms have been playing an increasingly important role for fingerprint-based systems to estimate the user position.

Different antenna configurations have been studied, evidencing a promising capability of directional antennas to enhance the performance of RSS fingerprinting-based indoor localization system, reducing the number of required monitoring sensor and also increasing the localization success

4.3.4. Machine Learning-based cell fault detection

As part of the research on the extension of SDS, Machine Learning (ML) algorithms for the detection of attacks on the mobile infrastructure are being developed. More specifically, the aim is to predict a *mobile cell unavailable* event by detecting a timely sequence of events that precede it. To this aim, data gathered from the Altice Portugal infrastructure will be analyzed to build ML models that can improve the communication infrastructure resilience.

In this multi-actor environment and technically agile context, in order to leverage business value from the Data science & ML technologies, urges the specification of a well-known cognitive process methodology (set of reference processes) across the organization for enabling the successful implementation of cognitive projects. Through the cognitive process methodology, each cognitive project implements a specific cognitive use-case. This is the main objective of the Cognitive Process Methodology described herein.

A set of key principles are identified for the Cognitive Process Methodology:

1. Simplicity: only the required set of components to support the several methodology processes must be defined, avoiding unnecessary components that increase the methodology complexity and hamper its understanding;
2. Extendibility: easy integration of new components and interactions that were not foreseen during the initial stage of the methodology definition must be supported;
3. Artefacts Oriented: a set of clear outputs must be provided in each component of the methodology;
4. Business Driven: the produced cognitive artefacts must provide added-value for the business units;
5. Technology Agnostic: the specified methodology must be independent of the underlying applied technologies;

6. Use-Case Driven: the full cognitive use-case lifecycle must be supported (design, modeling/implementation, deployment, supervision) must be supported;
7. Stakeholders inclusiveness: all actors (data scientists, product managers, customer manager, unit directors, etc.) roles and responsibility must be well-defined;
8. Industry Aligned / Not Disruptive: well-known and highly adopted DS methodologies by the community, such as Cross-industry standard process for data mining [39], Knowledge Discovery in Databases [40], and Microsoft Azure Team Data Science Process [41], must be used as a basis for the cognitive process methodology definition.

Determining what is the correct input data for the ML process is a major task. In this work, real data is received from several sources, but not all can be used or have the same relevance. For this task, four data sources are being used:

- Fault events, i.e. alarms generated by telecommunications equipment and systems;
- Problems events, e.g., events were reported by regular users of the network;
- Planned interventions events. This type of events may cause other events to appear, e.g., replacing a telecommunications equipment may generate fault events;
- Network inventory;

These raw data sources are submitted to a cognition pipeline where it is analyzed, selected and structured, a data analysis phase needed before a data management phase. At this stage, raw data is transformed to create other formats, or views of the data, that are more suitable for the problem under analysis. Part of this transformed data is then used for the training of the ML models. The other part of the transformed data is used for testing the models.

Errore. L'origine riferimento non è stata trovata. depicts the Altice Labs Cognition Pipeline described above.

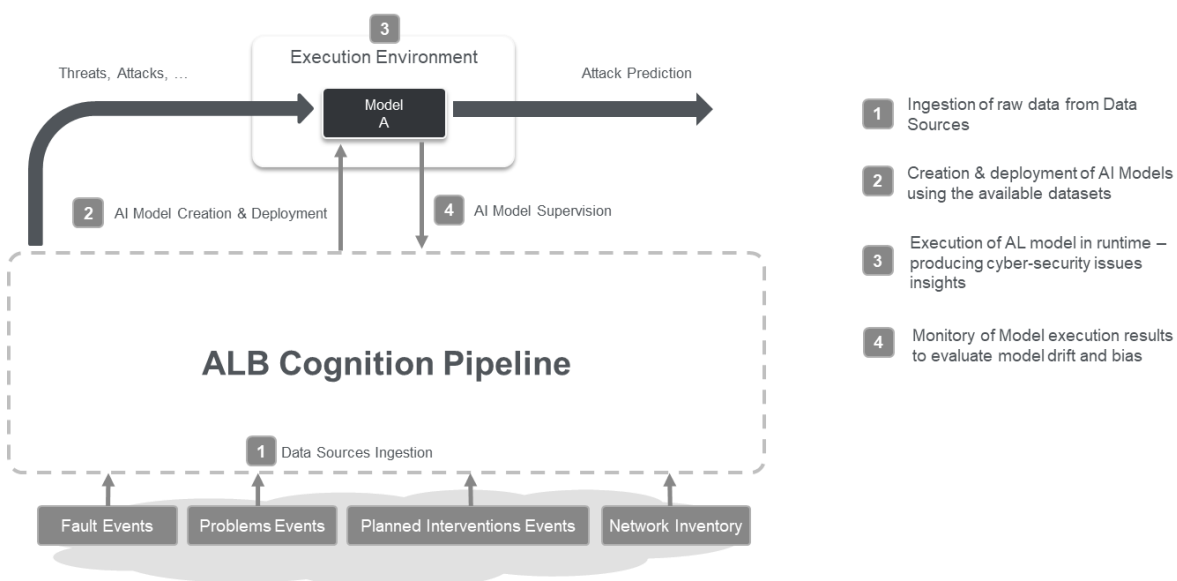


Figure 21: Altice Labs Cognition Pipeline

A generic, illustrative representation of the cognitive process used is given in Figure 22, showing its circular nature, in which are depicted the main stages that are typically executed, namely:

- Business Understanding Stage
- Data Acquisition & Transformation Stage
- Modeling Stage
- Deployment Stage

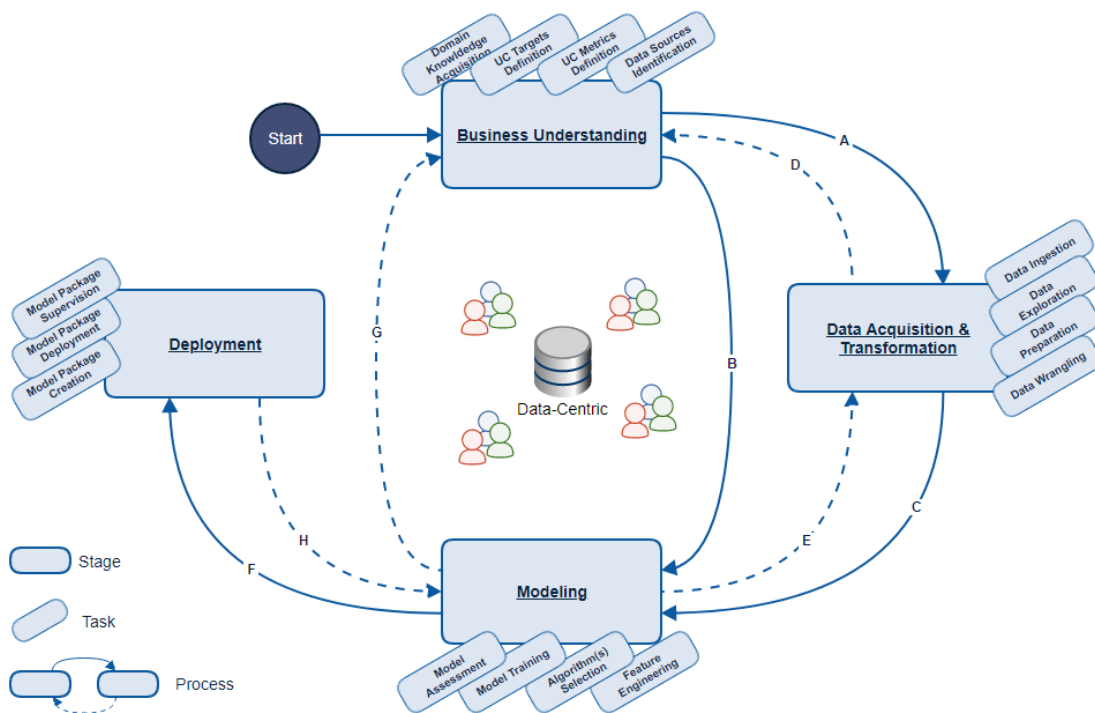


Figure 22: The Cognitive Process

The described methodology provides an overview of the life cycle of a cognitive project. It contains the stages of the project, their respective tasks and relationships between these tasks (or processes).

1. Business Understanding

In the Business Understanding stage, the main goal is to acquire knowledge about the problem domain and specify the cognitive use case (UC) in its several dimensions. The following tasks are executed at this stage:

- Domain Knowledge Acquisition Task: Understand the UC objectives and requirements from a business perspective;

- UC Targets Definition Task: Convert the acquired knowledge into a cognition problem definition;
- UC Metrics Definition Task: Define the metrics to evaluate the cognition problem results, as well as the functional and non-functional requirements;
- Data Sources Identification Task: Identify and describe the required data sources to execute the UC.

This stage's transitions are:

- **Transition A** (Business Understanding Stage → Data Acquisition & Transformation Stage): after the cognitive UC is defined, the next step is to acquire, explore, prepare and transform the data for creating the ML models;
- **Transition B** (Business Understanding Stage → Modeling Stage): during the cognitive UC definition, it might be decided as next step to change the feature selection strategy (this transition assumes the data was already acquired and transformed in a previous iteration);

The output artefacts of this stage are:

- Problem domain terminology report;
- UC technical perspective report, including:
 - UC targets;
 - UC metrics & requirements/technical constraints (functional and non-functional).

2. Data Acquisition and Transformation

The goal of this stage is to manage the complete data lifecycle (ingest, explore, prepare, transform/wrangle).

These are the tasks executed at this stage:

- Data Ingestion Task: Data sources integration and collection;
- Data Exploration Task: Data understanding during the whole life of the stage;
- Data Preparation Task: Attributes selection, cleaning, restructuring, etc. towards the final dataset format;
- Data Wrangling Task: Data encoding, normalization, transformation, etc. towards the final dataset format.

This stage's transitions are:

- **Transition C** (Data Acquisition & Transformation Stage → Modeling Stage): after data transformation is completed, the next step is to work on the model training and evaluation;
- **Transition D** (Data Acquisition & Transformation Stage → Business Understanding Stage): during the data transformation process, due to unsatisfactory results, it might be required as next step to integrate a new data source, redefine the UC target, etc.

This stage produces the following output artifacts:

- Data preparation and transformation pipeline report, including:
 - Data sources description;

- Pipeline specification description;
- Dataset description;
- Data quality report;
- Pipeline instantiation;
- Dataset.

3. Modeling

The objective of the modeling stage is to select, apply and calibrate several modeling techniques to achieve optimal values.

The tasks executed at this stage are:

- Features Engineering Task: use-case features identification;
- Algorithm(s) Selection Task: select the most appropriated algorithm(s) for training;
- Model Training Task: apply selected algorithms for training the model;
- Model Assessment Task: evaluate and calibrate the algorithm to achieve optimal values.

This stage transitions are:

- **Transition E** (Modeling Stage → Deployment Stage): after the model is created and evaluated the next step is to deploy it in a production environment;
- **Transition F** (Modeling Stage → Data Acquisition & Transformation Stage): during the model training and/or evaluation, it might be required as next step to use a different strategy to prepare and/or transform the data;
- **Transition I** (Modeling Stage → Business Understanding Stage): during the model generation and/or evaluation, it might be required as next step to acquire further information about the problem domain, redefine the model target, integrate a new data source, etc.

The produced artefacts are:

- Modeling report
 - Features set description report;
 - Selected ML algorithm(s) report;
 - Algorithm configuration description report per model (e.g. model description, parameters setting, etc.);
 - Model assessment report (e.g. summarize results of this task, list qualities of generated models (e.g., in terms of accuracy, etc.), rank their quality in relation to each other, rank their quality facing the functional requirements and technical constraints defined in BU stage, etc.);
- Generated modeling instances
 - Model input transformation
 - Model
 - Model output transformation

4. Deployment

The objective of this stage is to deploy the model in a production environment.

The main tasks are:

- Model Package Creation Task: Create the model package artefacts;
- Model Package Deployment Task: Deploy the model;
- Model Package Supervision Task: Supervise the model.

The stage transitions are:

- **Transition H** (Deployment Stage → Modeling Stage): during the model execution phase, the model can be drifted/biased and no longer be useful; therefore, the next step is to train and generate a new model.

The produced artefacts:

- Deployment and supervision plan report;
- Model package instantiation;
- Model deployment details report;
- Status dashboard that displays the system health and key metrics;
- Final report consolidating all the project information and deliverables.

Figure 23 depicts the ML methodology adopted.

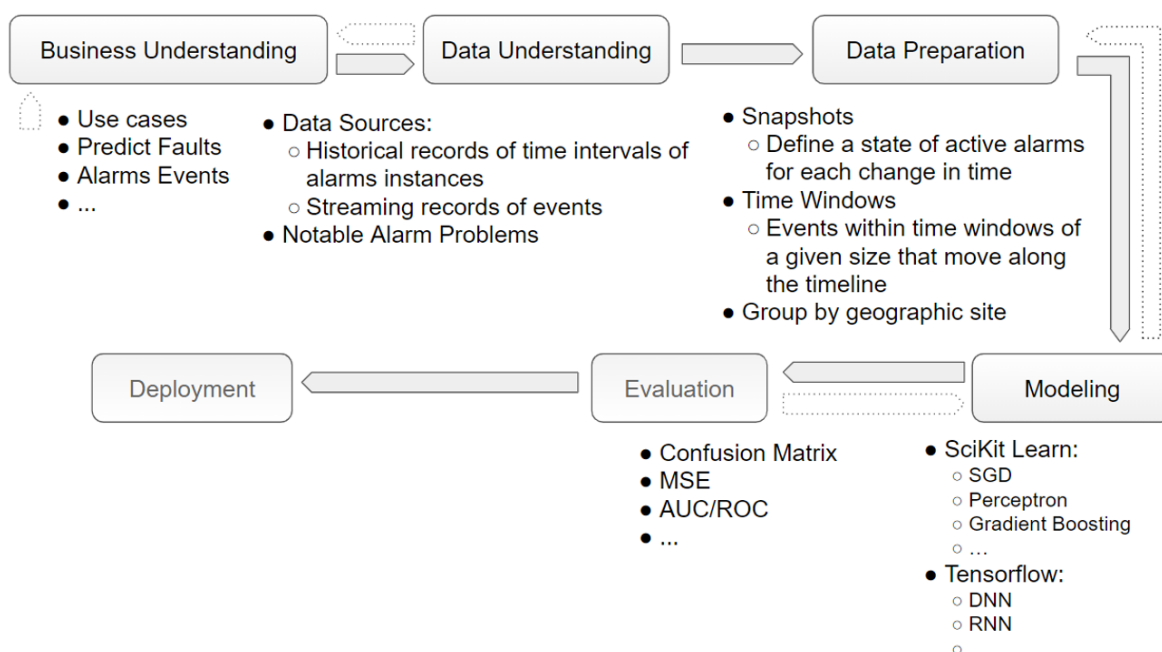


Figure 23: ML methodology

In the context of the modeling done, we used as data source the historical alarms data provided by Altice Portugal. Data can be read from a framework or via scripts. Some metadata normalization, validation of values or outlier exclusion might occur in this stage.

Data wrangling has a strong dependence on the modeling stage.

Two processes were defined in our context, both grouped by the geographic code:

- Snapshots of the current state of the network
 - Snapshots target labels defined by scanning 2 hours ahead of their definition.
- Time windows of the events
 - Target label defined within the defined time window

As this is an ongoing process, the modeling, evaluation, and deployment blocks shown in Figure 23 are not yet concluded.

5. CONCLUSION

In this document the current status of the efforts on Task 5.2 and 5.3 has been reported.

The current implementation of the mitigation framework, based upon workflows defined with a business logic-like language, has been presented. Currently, this implementation is able to support the operator with one or more workflow for every alarm situation. Future developments include the evaluation of multi-objective optimization strategies to support the operator with various solution, each having a profile of the impact on the different KPIs. In order to reach this goal, functions determining the impact of each action on the various indicators will have to be built by leveraging end-user's expertise.

The orchestrator framework supports different configurations based on the underlying network technology. It is able to execute abstract actions coming from the mitigation framework, modifying the configurations of the network. Algorithms for optimizing the deployment of actions in SDN networks, based on multi-objective optimization, are currently being developed. A set of KPIs to be used in the optimization has been determined. Future activities will extend this paradigm also to NFV networks.

Technologies to extend SDS to the physical world (beside the workflow engine) are also being studied. Specifically, the possibility to interface SDS elements directly with SDR elements is being investigated as a possible future development of the framework.

Overall, the Task are producing satisfying results and contributing to the technical and scientific state of the art.

6. REFERENCES

| INDEX | REFERENCE |
|-------|---|
| [1] | RESISTO Deliverable 6.1 - SW Architecture Definition |
| [2] | RESISTO Deliverable 4.4 – Complete propagation analysis |
| [3] | RESISTO Deliverable 3.7 – KPIs, Quantities and metrics for cyber-physical risk and resilience of telecom CIs |
| [4] | Open networking foundation, SDN architecture Issue 1.1 2016 – https://www.opennetworking.org/wp-content/uploads/2014/10/TR-521_SDN_Architecture_issue_1.1.pdf |
| [5] | D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," in <i>Proceedings of the IEEE</i> , vol. 103, no. 1, pp. 14-76, Jan. 2015. |
| [6] | M. Jarschel, T. Zinner, T. Hossfeld, P. Tran-Gia and W. Kellerer, "Interfaces, attributes, and use cases: A compass for SDN," in <i>IEEE Communications Magazine</i> , vol. 52, no. 6, pp. 210-217, June 2014. |
| [7] | Tijare, Poonam & Vasudevan, Deepika. (2016). The Northbound APIs of Software Defined Networks. 10.5281/zenodo.160891. |
| [8] | B. Chatras, "On the Standardization of NFV Management and Orchestration APIs," in <i>IEEE Communications Standards Magazine</i> , vol. 2, no. 4, pp. 66-71, December 2018. |
| [9] | 3GPP Specification # 23.501, System architecture for the 5G system: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144 |
| [10] | The ONOS project - https://onosproject.org/ |
| [11] | ONOS performance whitepaper - http://onosproject.org/wp-content/uploads/2017/08/ONOS_Performance_White_Paper-2.pdf |
| [12] | ETSI Open source management and orchestration - https://osm.etsi.org/ |
| [13] | SONATA Open Source Github repositories, https://github.com/sonata-nfv . |
| [14] | Open5GCore – The Next Mobile Core Network Testbed Platform; https://www.open5gcore.org/ |
| [15] | S. Scott-Hayward, S. Natarajan and S. Sezer, "A Survey of Security in Software Defined Networks," in <i>IEEE Communications Surveys & Tutorials</i> , vol. 18, no. 1, pp. 623-654, Firstquarter 2016. |
| [16] | Enisa, Security aspects of virtualization - https://www.enisa.europa.eu/publications/security-aspects-of-virtualization |

| | |
|------|--|
| [17] | Spectre vulnerability - https://nvd.nist.gov/vuln/detail/CVE-2017-5754 , https://nvd.nist.gov/vuln/detail/CVE-2017-5715 |
| [18] | Meltdown vulnerability - https://nvd.nist.gov/vuln/detail/CVE-2017-5753 |
| [19] | S. Lal, T. Taleb and A. Dutta, "NFV: Security Threats and Best Practices," in <i>IEEE Communications Magazine</i> , vol. 55, no. 8, pp. 211-217, Aug. 2017. |
| [20] | Enisa, Threat Landscape and Good Practice Guide for Software Defined Networks/5G – https://www.enisa.europa.eu/publications/sdn-threat-landscape |
| [21] | Dhawan, Mohan, et al. "SPHINX: Detecting Security Attacks in Software-Defined Networks." <i>NDSS</i> . Vol. 15. 2015. |
| [22] | Verizon data breach report, 2018 https://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_en_xg.pdf |
| [23] | Yunhe Cui, Lianshan Yan, Saifei Li, Huanlai Xing, Wei Pan, Jian Zhu, and Xiaoyang Zheng. 2016. SD-Anti-DDoS. <i>J. Netw. Comput. Appl.</i> 68, C (June 2016), 65-79. DOI: http://dx.doi.org/10.1016/j.jnca.2016.04.005 |
| [24] | K. Kalkan, L. Altay, G. Gür and F. Alagöz, "JESS: Joint Entropy-Based DDoS Defense Scheme in SDN," in <i>IEEE Journal on Selected Areas in Communications</i> , vol. 36, no. 10, pp. 2358-2372, Oct. 2018. |
| [25] | R. Wang, Z. Jia and L. Ju, "An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking," <i>2015 IEEE Trustcom/BigDataSE/ISPA</i> , Helsinki, 2015, pp. 310-317. |
| [26] | T. V. Phan and M. Park, "Efficient Distributed Denial-of-Service Attack Defense in SDN-Based Cloud," in <i>IEEE Access</i> , vol. 7, pp. 18701-18714, 2019. |
| [27] | L. Yang and H. Zhao, "DDoS Attack Identification and Defense Using SDN Based on Machine Learning Method," <i>2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN)</i> , Yichang, China, 2018, pp. 174-178. |
| [28] | Sahay, Rishikesh, et al. "ArOMA: An SDN based autonomic DDoS mitigation framework." <i>Computers & Security</i> 70 (2017): 482-499. |
| [29] | Verizon SDN-NFV Reference Architecture. Originally available at http://innovation.verizon.com/content/dam/vic/PDF/Verizon_SDN-NFV_Reference_Architecture.pdf . Currently unavailable. A copy is available at: https://docplayer.net/25746317-Sdn-nfv-reference-architecture.html |
| [30] | Montida Pattaranantakul, Yuchia Tseng, Ruan He, Zonghua Zhang, and Ahmed Meddahi. 2017. A First Step Towards Security Extension for NFV Orchestrator. In <i>Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFVSec '17)</i> . ACM, New York, NY, USA, 25-30. |

| | |
|------|---|
| [31] | M. Pattaranantakul, R. He, Q. Song, Z. Zhang and A. Meddahi, "NFV Security Survey: From Use Case Driven Threat Analysis to State-of-the-Art Countermeasures," in <i>IEEE Communications Surveys & Tutorials</i> , vol. 20, no. 4, pp. 3330-3368, Fourthquarter 2018. |
| [32] | Modern WLAN Fingerprinting Indoor Positioning Methods and Deployment Challenges, Ali Khalajmehrabadi, Nikolaos Gatsis, and David Akopian. |
| [33] | C. L. a. R. Yu, Wireless network virtualization: A survey, some research issues and challenges, <i>IEEE Communications Surveys &</i> , 2015. |
| [34] | I. M. W. L. a. F. A. P. d. F. Xianjun Jiao, Radio hardware virtualization for coping with dynamic heterogeneous wireless environments. |
| [35] | J. R. L. A. D. C. B. B. Maicon Kist, SDR Virtualization in Future Mobile Networks: Enabling Multi-Programmable Air-Interfaces. |
| [36] | Verizon 2018 Data Breach Investigation Report https://enterprise.verizon.com/resources/reports/DBIR_2018_Report.pdf |
| [37] | ATENA - Advanced Tools to assEss and mitigate the criticality of ICT compoNents and their dependencies over Critical InfrAstructures, European H2020 project - https://www.atena-h2020.eu |
| [38] | Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. <i>IEEE transactions on evolutionary computation</i> , 6(2), 182-197 |
| [39] | Cross-industry standard process for data mining - https://www.datasciencecentral.com/profiles/blogs/crisp-dm-a-standard-methodology-to-ensure-a-good-outcome |
| [40] | Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. 1996. The KDD process for extracting useful knowledge from volumes of data. <i>Commun. ACM</i> 39, 11 (November 1996), 27-34. |
| [41] | Microsoft Azure Team Data Science Process Documentation, https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/ |
| [42] | Apache Karaf application runtime - https://karaf.apache.org |
| [43] | P4 telemetry protocol - https://p4.org |
| [44] | McKeown, Nick, et al. "OpenFlow: enabling innovation in campus networks." <i>ACM SIGCOMM Computer Communication Review</i> 38.2 (2008): 69-74. |
| [45] | Netconf protocol - https://tools.ietf.org/html/rfc6241 |
| [46] | SNMP protocol - https://tools.ietf.org/html/rfc1157 |
| [47] | Telefonica OpenMano - http://www.tid.es/long-term-innovation/network-innovation/telefonica-nfv-reference-lab/openmano |
| [48] | JUJU - https://docs.jujucharms.com |