ENERGY SHIELD

Integrated Cybersecurity Solution
for the Vulnerability Assessment, Monitoring and Protection of
Critical Energy Infrastructures

INNOVATION ACTION

H2020 Grant Agreement Number: 832907

WP5 TOOLKIT INTEGRATION

D5.6 TEST / QA REPORT

Document info	
Contractual delivery	31/12/2021
Actual delivery	31/12/2021
Responsible Beneficiary	SIMAVI
Contributing beneficiaries	PSI, SIGA, FOR, L7D, TEC, KT, CITY, KTH, NTUA
Version	1.1

This project has received funding from the European Union's H2020 research and innovation programme under the Grant Agreement No. 832907

DOCUMENT INFO

Document ID:	D5.6
Version date:	**31/12/2021**
Total number of pages:	51
Abstract:	The aim of this task is to ensure that the developed software integration platform is performing according to the specifications of WP1. A testing plan document was specified in task 5.1 based on user and system requirements documentation and the testing is run according to this plan. In order to be able to fulfil stakeholders' expectations and to meet up with user standards a strict quality assurance methodology will be enforced. This QA methodology will ensure that defects in the software product (platform) are prevented and/or addressed in a timely manner. This way, possible shortcomings, mistakes, or defects that may occur in the developed product (integrated software platform) may not only be prevented but also properly addressed may these occur. This task does not include the participation of end-users, who will be involved into live field tests in WP6 but will focus on stressing the prevention potential of testing, with defect detection and demonstration of capability as secondary goals, and a primary goal on finding requirements and design defects through early development of tests designs. Our approach is to carefully and systematically analyse requirements and to derive design-based coverage inventories, resulting in known and measurable test coverage matrix, and to schedule test-ware design in parallel with software design and even before coding, resulting in a test-driven development approach.
Keywords	cybersecurity, integration, toolkit, testing, common platform, QA, test plan, test cases, test scenarios

AUTHORS

Name	Organisation	Role
Iacob Crucianu	SIMAVI	Overall Editor
Lavinia Dinca	SIMAVI	Editor
Per Eliasson	FOR	Contributor
Hagai Galili	SIGA	Contributor
Christos Angelidis	KT	Contributor
Yisrael Gross	L7D	Contributor
Anna Georgiadou	NTUA	Contributor

REVIEWERS

Name	Organisation	Role
Ana-Maria Dumitrescu, Lavinia Dincă	SIMAVI	Overall Reviewer
Otilia Bularca	SIMAVI	QA Reviewer

VERSION HISTORY

V0.1	26/10/2021	ToC
V0.2	20/11/2021	Update the existing content
V0.3	15/12/2021	Collecting contributions from partners
V0.4	27/12/2021	Consolidating contributions received from partners
V0.5	29/12/2021	Overall and QA review
v1.0	31/12/2021	Final version

EXECUTIVE SUMARY

The current document reports on the progress of software development and platform integration in accordance with the architecture design anticipated in WP1 System Specifications & Architecture and WP5 Toolkit Integration, respectively. Thus, the user and system requirements of the EnergyShield toolkit are assessed against the integration and test plan (D5.1). Also, quality assurance methodologies are deployed to ensure that defects in the software product (platform) are prevented and/or addressed in a timely manner.

The approach proposed is based on careful and systematic analysis of requirements and testing of the available results both EnergyShield tools and toolkit.

The report starts from a summary of the **integration & deployment** approaches in the context of the overall project implementations and continues with a short description of the **testing** strategy while also providing details on the results obtained at M30.

Three approaches on testing Energy Shield the tools and toolkit are addressed:

- Testing the tools individually in a laboratory environment. The expected outcome of these tests would be demonstrating the functionalities of the individual tools
- Testing the tools placed in the toolkit in a laboratory environment. At this stage, the way the tools are interfaced and are communicating is tested by means of input and output information received from consumption points. Details of this are presented in D5.5 System release v3 [**ESH55**]
- Testing the tools integrated in the toolkit and placed in the OT environment. This is the most extensive approach and tests the functionality as designed and implemented in the final system.

This task does not include the participation of end-users, who will be involved into live field tests in WP6 Field Trials, but focuses on stressing the prevention potential of testing, with defect detection and demonstration of capability as secondary goals, and a primary goal on finding requirements and design defects through early development of tests designs.

A final version including all the results of the architecture, integration and testing alongside with updated documentation will be included in D5.7 Common software platform release, incl. user and developer documentation – final version that will be submitted at M34 after the completion of field trials.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ACRONYM	DESCRIPTION
AD	Anomaly Detection
API	Application Programming Interface
CI	Continuous Integration
D	Deliverable
DDoSM	Distributed Denial of Service Module
DSO	Distribution System Operator
EPES	Electrical Power and Energy System
FDD	Feature Driven Development
FR	Functional Requirement
HW	Hardware
IAM	Identity and Access Management).
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
JEE	Java Enterprise Edition
ISO	International Organization for Standardization
KB	Knowledge Base
MQTT	Message Queuing Telemetry Transport
NoSQL	Not only Structured Query Language
OS	Operating System
OT	Operational Technology
RDBMS	Relational Database Management System
REST	Representational state transfer
RBT	Risk Based approach to Testing
SBA	Security Behaviour Analysis
SGAM	Smart Grid Architecture Model
SIEM	Security Information and Event Management
SQA	Software Quality Assurance
TC	Test Case
TRL	Technology Readiness Level
TSO	Transmission System Operator
VA	Vulnerability Assessment
VM	Virtual Machine
WP	Work Package

1. INTRODUCTION

1.1. SCOPE AND OBJECTIVES

This work package focuses on the integrations of the tools developed in WP2, WP3 and WP4, and the common platform (including the software development kit) that is required to run and deploy the tools for the field trials.

Testing and quality assurance activities are also part of the current report and specific activities are detailed. The testing specification documentation will both stress out platform capabilities (functional and non-functional) in relation with all the defined use cases and provide details about using and installing each tool.

1.2. STRUCTURE OF THE REPORT

The report is structured into main sections as follows:

Section 1 describes the integration and testing approached proposed for EnergyShield including the timeline of activities.

Section 2 provides details about the integration and deployment activities starting from the architecture design, continuing with the description of the logical view and the technology support and describes how the testing and quality assurance activities will be performed for both tools and integrated toolkit.

Section 3 presents the results of testing both the tools (individually) and the toolkit based on scenarios drafted for IT or lab environed and for OT environment.

1.3. TASK DEPENDENCIES

This task uses outcomes of deliverables 1.1 (D1.1) technical requirements specification [**ESH11**] and from Deliverable 1.4 (D1.5) Architectural design [**ESH15**] which include details about the proposed use cases, tools integration and deployment possibilities and from deliverables 5.1 (D5.1) Integration and test plan [**ESH51**], 5.2: (D5.2) Common software platform release, incl. user and developer documentation [**ESH52**], 5.3 (D5.3) System release v1 [**ESH53**], 5.4 (D5.4) System release v2 [**ESH54**] and D5.5System release v3 [**ESH55**] reporting on the testing and toolkit development.

A final version including all the results of of the architecture, integration and testing alongside with updated documentation will be included in D5.7 Common software platform release, incl. user and developer documentation – final version that will be submitted at M34 after the completion of field trials.

2. INTEGRATION AND TESTING APROACHES

This section presents the integration approach specific to EnergyShield project, based on the system specifications and design documented in the reports submitted as part of WP1 System Specifications & Architecture deliverables 1.1 (D1.1) technical requirements specification [**ESH11**] and from Deliverable 1.4 (D1.5) Architectural design [**ESH15**] that provide details about proposed use cases, architecture design, tools integration and deployment possibilities.

2.1. INTEGRATION PERSPECTIVE

Different possibilities of integration are investigated in this section and different perspectives are considered to identify the most feasible solutions for EnergyShield. Based on the outcomes of Deliverable 1.5 (D1.5) System Architecture [**ESH15**] the EnergyShield integration concept is addressed from a logical view and by considering the available technologies.

A series of specific activities needs to be detailed and planned throughout project implementation. Figure 1, below, presents the major steps to be taken.

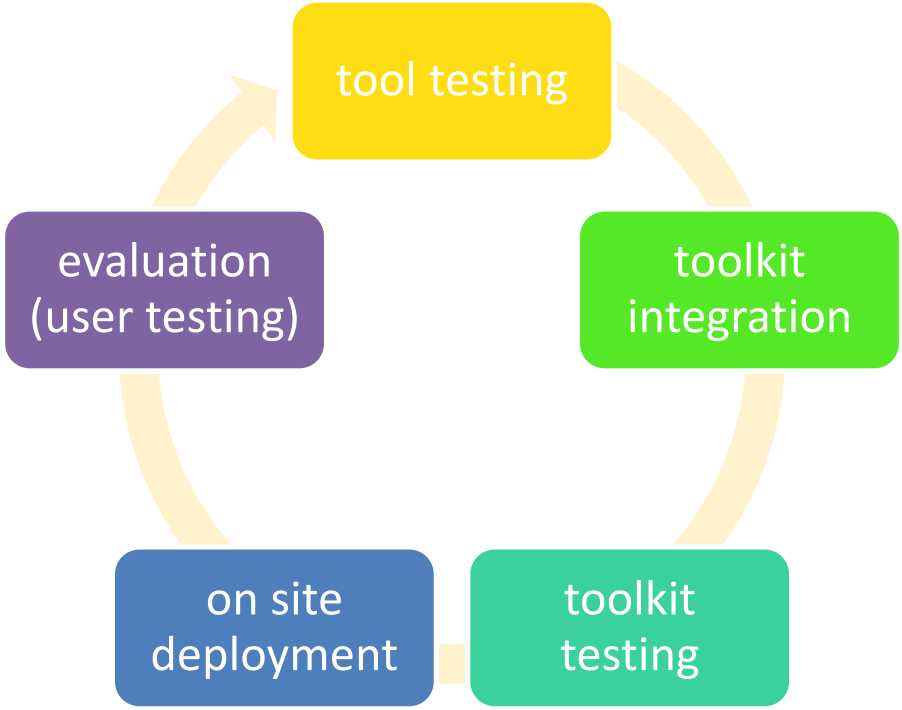

Figure 1. EnergyShield integration, deployment, and testing activities

Tools testing. Following the first release of EnergyShield tools (M12) the available features and capabilities are tested and mapped against the needs of the pilot cases provisioned in EnergyShield project.

Toolkit integration. The integrated design proposed for EnergyShield is multi-layered covering operating systems, middleware, database and IoT harvesting methods.

Toolkit testing refers to different testing from unit testing individual modules, integration testing an entire system to specialized forms of testing such as security and performance.

On site deployment includes all the operations to prepare EnergyShield system for assembly and transfer to the computer system(s) on which it will be run in production.

2.2. TIMELINE OF ACTIVITES

The integration plan has been organised in 7 phases (i.e., phase 0 to 6) as shown in Table 1, below, in which the development and integration will follow a stepwise plan to ensure that the components are individually developed and ready to be deployed it the common environment.

The timeline of tools release is synchronized with the planned release of the integrated demonstrator. With every release the integrated demonstrator progresses towards a demonstrator that can be deployed on pilot site for field trials.

While the first four phases of integration accommodate EnergyShield tools with part of the functionalities' releases, the last three phases refine the toolkit with all the functionalities available, via testing and quality control to ensure a smooth and effective on-site deployment. This report corresponds to Phase 5 – System Release v3 and leaves the final outcomes for Phase 6.

Table 1. EnergyShield integration phases and corresponding reports

Phase	Name	D	M12	M15	M19	M20	M26	M30	M34
	1st release of tools		*x*						
0	Integration plan	D5.1	x						
1	Toolkit concept	D5.2		x					
	2nd release of tools				*X*				
2	Toolkit v1	D5.3				x			
3	Toolkit v2	D5.4					x		
	Final release of tools							*x*	
4	Toolkit v3	D5.5						x	
5	Testing & QA	D5.6						x	
6	Toolkit – final version	D5.7							x

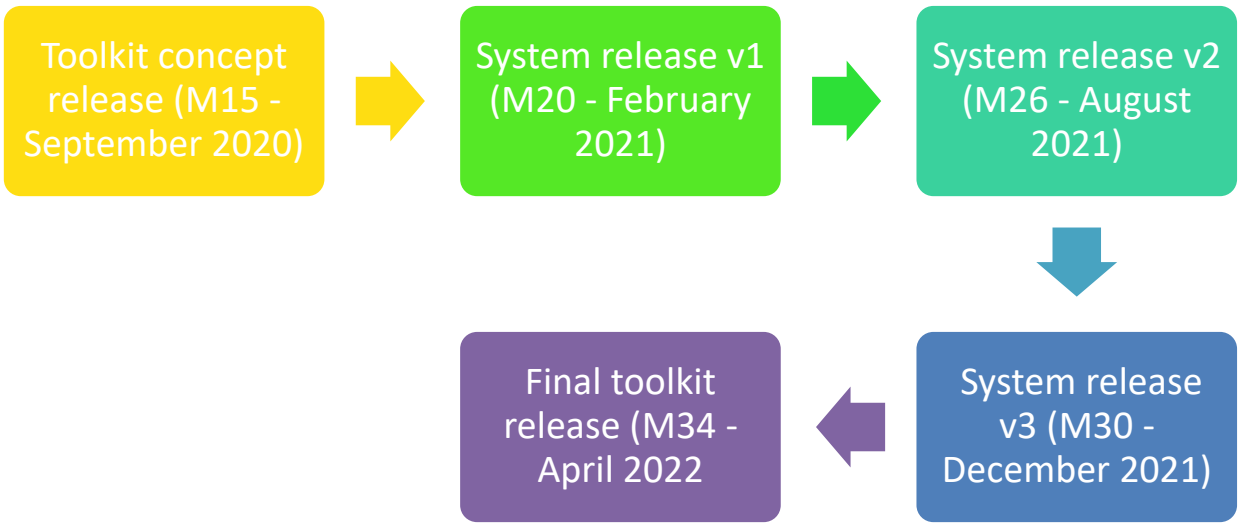

Figure 2. Toolkit demonstrator release timeline

Phase 0 – represents the integration stage at the moment of submitting this report, i.e. the design of the integrated prototype is ready and the tools have been released with the first set of functionalities and the integration activates are detailed and planned in accordance with the planned deliveries.

Phase 1 – builds the EnergyShield integration concept based on the 1st set of tools supporting documents (user manual and installation manual).

Phase 2 – represents the moment of releasing the first integrated EnergyShield toolkit including the first release of tools functionalities

Phase 3 – represents the second release of the integrated toolkit with the second set of functionalities released by technology providers

Phase 4 – is the third release of the toolkit including the final version of tools

Phase 5 – refers to performing the tests and quality assurance – presented in the current report

Phase 6 – releases the final version of the toolkit, ready for field trials.

Each phase presented above exploits the outcomes of the previous phase and integrates further components and capabilities as they are developed and released.

Every release of the tool will be functionally tested by the technology provider, while the system releases will be tested and evaluated by Practioners. Within WP6 Filed trials - that starts in M15 - the evaluation methodology will be defined and planned in three evaluation cycles considering the already planned tools and system releases.

2.3. TEST PLAN

The most suitable approach concerning the specific levels of testing in EnergyShield was investigated to address the needs of the pilots.

The testing methodologies to be applied at both tool and pilot solution level are unit testing, integration testing and performance testing.

- Functional testing (Black Box Testing) is the software testing method in which the internal structure/design of the tested item is not known to the testers; applicable to functional testing, scenario testing and simulations.
- Unit testing (White Box Testing) refers to the testing of the software solution's internal structure, design and coding and is applicable to unit level and integration level (e.g. web services).
- Integration Testing tests how parts of the system work are similar to some extent to unit testing. The major difference between those two testing approaches refers to the fact unit tests are isolated from other components, while integration tests are not.
- Performance Testing tests how the software application performs given the expected workload.

Testing strategy covers the major aspects of testing stage (concepts, principles, deliverables, and work-products) in the context of developing a typical software system. Two IEEE sets of standards are applicable for the testing stage in EnergyShield is ISO-IEC-IEEE 29119 – Software and system engineering – Software testing [**IEE16**].

Compatibility with the above-mentioned standards was the basis for choosing the two conceptual development models applicable in EnergyShield – V- model and Agile.

2.4. SPECIFIC SMART GRID AND OT CONSIDERATIONS

Traditional power grids distributed and managed the power from a central location, but with growing demands for energy as well as reliability and operations, an interconnected dynamic model was developed known as a Smart Grid (SG). Thus, testing on SG becomes very complex due to the multi-layer systems that need to function together as part of the SG. In the context of SG each layer needs to be tested separately as a single entity and as a whole system [**CHR18**]. Because SG is a cyber system that involves both hardware and software devices, modelling and simulations must be used to discover integration issues [**PAL14**], [**KHA15**].

In [**SCH18**] a brief testing summary as defined in literature is defined, please see table 2.

Table 2. Summary of SG testing from literature, as defined in [SCH18]

Literature reference	Testing area	How to test
Kok et al. [KOK11]	Power flow	Real using 1:1, scaled or simulated data.
Kok et al. [KOK11]	Data flows	Power grid only, information grid only, and combined scenarios.
Kok et al. [KOK11], Karnouskos and Holanda [KAR09]	Interaction capture	Large data volume information capture
Karnouskos and Holanda [KAR09], Wang et al. [WAN10]	Topological changes	Capture state before test, during test, and after test
Karnouskos and Holanda [KAR09]	Multi-agent systems	Test one entity by breaking it down into components
Karnouskos and Holanda [KAR09]	Simulator integration	Through APIs
Karnouskos and Holanda [KAR09], Hahn et al. [HAH13]	Entity classification	Classify each entity like: prosumer, consumer, transporter, network intruder, SCADA.
Hahn et al. [HAH13]	Network requirements	Network analysis, packet injection, simulate intruders
Wang et al. [WAN10]	Topology generation	Automatic, must test if the model autoscales
Wang et al. [WAN10]	Testing platform	The platform should support different SG topologies.

When testing SGs the information exchange testing process is usually very complex because there are multiple types of devices as detailed in table

Table 3. SG types of devices

Device	Device short description	Device details
A	Not connected to the grid, exchanging data	This type of devices are part of the SG and sending/receiving data to other parts of the grid or external entities. These devices can be both software only and hardware, e.g. weather station. These types of devices are control systems SCADA.

B	Connected to the grid, not exchanging data	This type of devices are the necessary grid hardware like power lines and transformers that don't have any network reporting capabilities, There are very important for the SG because they represent the physical SG topology and are required when all SG components must be tested.
C	Connected to the grid, exchanging data.	This type of devices are a combination of type A and Type b. When testing a simulation environment must be used to simulate both the power grid and information exchange,
D	Intruders (both virtual and physical)	This type of devices are used to simulate, study and test cyber-attacks on the SG network. Type D devices are mainly concerned with simulation and study of cyber-physical attacks on the SG.

As shown in the subchapter before, there are many testing methodologies, but the one chosen by us ISO/IEC/IEEE 29119 [**IEE16**] standard is the one that should be used to test SGs [**SCH18**]. By applying this standard, a risk-based approach will be chosen [**FEL14**]. The integrated testing process is shown below.

2.5. TESTING PROCESS

The appropriate agile testing techniques are specific to the following Agile testing areas:

- A1: Technology-facing tests that support the team
- A2: Business-facing tests that support the team
- A3: Business-facing tests that critique the product (focused on the acceptance of the product)
- A4: Technology-facing tests that critique the product (focused on the performance criteria of the product).

Analysing the specific features of each testing area, two of the agile testing areas were mainly addressed within, and tailored to this stage of EnergyShield project implementation:

The particularity of this type of testing is given by the conceptual focusing of testing on feature – *feature driven testing*, applying the same approach as for the feature driven development (FDD) [**FDD19**].

The testing process proposed for EnergyShield, based on agile methodology, and specifically, feature driven testing, considers the specific methodological requirements of the project and the conceptual architectural design of the EnergyShield toolkit.

The outcomes of the testing process include documents referring to:

- What was tested (using test cases) and test failures (bugs)
- Which bugs caused failures and when they can be addressed

Specific tasks for each team are registered in Jira - a specific application for issue tracking and general project management features.

2.5.1 TEST PLAN

A test plan provides the necessary planning information for a development organization to get ready for testing by allocating the appropriate resources early on at the project. By providing detailed requirements for hardware, software, and people as early in the lifecycle as possible, the quality of the final product is ensured minimizing the possibility of delays, additional costs and additional effort at later stages of the project.

The goal of a master test plan is to:

- Provide an overview of the testing activities at a detailed level, in compliance with the testing strategy
- Define the test environment requirements
- Define each level of testing, focus areas and testing techniques to be applied
- Define in detail the features / functionalities and test cases to be tested
- Define the testing tasks to be executed, and assign responsibilities
- Define the business and technical risks that can be addressed through each level of testing.

As mentioned earlier a risk-based approach (RBT) is considered for testing. RBT is a specialized testing mantra that prioritizes software tests based on their risk of failure calculated as an average between likelihood and impact. In theory one can define an infinite number of tests, RTB makes sure that the features with the highest risk of failure and most impact to the organisation are tested first and that accurate tests are done specifically for those features.

2.5.2 TEST CASE DESIGN SPECIFICATIONS

Test design specification addresses the features to be tested. A standardized form of designing test specifications undertakes the definition of high-level test cases that fulfil the defined business requirements and ensure traceability.

Also, a test Case is defined by the following items, pursuant to the guidelines of ISO/IEC/IEEE 29119 Software Testing [**IEE16**].

Table 4. Test case design

TC	***<ID>***	***<Title of the use case>***
UC	*<related use cases*	
FRs	*<related functional requirements>*	
Precondition	*<the setup needed for a test case to be executed successfully>*	
Test environment (optional)	*<hardware, software and network configuration needed>*	
TC Step (actions)	**Obtained result**	**Verdict**
1. <steps to be executed>	*<expected results>*	*<pass>, <fail>*

2.6. EXPECTED RESULTS

Expected testing results are very significant items within the testing process. The monitoring activity of testing results is retrieved in the test plan and is materialized by the elaboration of testing reports.

A standardized structure for each type of report will be defined taking into consideration the rules and specifications of the IEEE Standard 29119, section IEEE 29119-3 [**IEE16**]:

A template for the results documentation (outcomes will be drafted and included in the updated version of this report. This should include at least details about:

- Test cases for each tool and the test execution report
- User and installation manuals for each tool
- Integration test cases for the integrated pilot solution and the execution report.

The proposed approach for testing should be considered as a guideline to deliver a complete and stable software application for the field trials. If constraints apply and the proposed methodology cannot be followed providing a piece of evidence of testing by other means also covers the testing requirements.

3. TESTING RESULTS

As detailed in the previous section 3 approaches were proposed on testing Energy Shield the tools and toolkit.:

a) **Testing the tools individually** in a laboratory environment. The expected outcome of these tests would be demonstrating the functionalities of the individual tools
b) **Testing the tools placed in the toolkit** in a laboratory environment. At this stage, the way the tools are interfaced and are communicating is tested by means of input and output information received from consumption points.
c) Testing the tools integrated in the toolkit and placed in the **OT environment**. This is the most extensive test and tests the functionality as designed and implemented in the final system.

The first approach refers to testing of tools and toolkit in the IT environment on the premises or lab of tool provider, while the last two one aim at testing the toolkit with and without OT interaction.

3.1. INDIVIDUAL TOOLS TESTING

For the individual tools testing EnergyShield tool providers have updated the list of functional requirements (as pe D1.5) and tested the availability of each feature released alongside with some details about the methodology applied to test tool functional and non-functional requirements. Also, details about the implemented features of tools are to be found in the tool corresponding reports, submitted in their final versions in M30 which also includes user manuals) toolkit integration of tools (further elaborated in D5.5)

3.1.1. SBA TOOL TESTING

SBA has been developed based on testing principles which include, among others:

- **Unit testing** referring to the testing of individual methods and functions of the classes, components and modules used by the SBA tool.
- **Integration testing** utilised to verify that different components, modules and services of the SBA tool interact and collaborate in a productive manner.
- **Functional testing** focusing on the business requirements of the SBA tool as described in detail in D154 System Architecture – final version and presented in [**ESH15**]
- **End-to-end testing** used to verify that a few key end-to-end user flows work as anticipated.

- **Performance testing** to ensure the reliability, stability, and availability of the SBA tool under significant workload (parallel user participation in evaluation campaigns and significant reporting extractions).

In all cases, testing automation has been pursued and achieved to a certain degree, ensuring a continuous integration life cycle of the tool.

Having ensured a certain quality and achievement score of the functional requirements, a SaaS version of the SBA tool (accessible at http://energyshield.epu.ntua.gr/) has been made publicly available to both pilots of the EnergyShield project and to other interested parties, e.g., pilots from collaborating EU projects, scientific workshops, security agencies, etc.

All features of the SBA tool have been implemented as planned and details about the implemented features alongside with sample test case are provided in Table 5).

Table 5 SBA Functional and Non-Functional Requirements

FR_ID	Requirements' description	Existing	Release 1	Release 2	Release 3
SBA_FR_1	The system should provide a basic questionnaire and framework for security culture assessment	X			
SBA_FR_2	The system should provide an anonymisation service that supports suppression or generalisation processing				X
SBA_FR_3	The system should provide the ability to assess the socio-cultural behaviour of an organisation against cyberthreats based on both organisational and individual level		X		
SBA_FR_4	The system should provide the ability to view the results of the assessment aggregated per specified cybersecurity culture domains		X		
SBA_FR_5	The system should provide a questionnaire mechanism to gather the input for the assessment		X		
SBA_FR_6	The system should provide a mechanism to the managers of the assessment to create new assessment campaigns where they select their target group (list		X		

	of employees to include) and domains to include				
SBA_FR_7	The system should enable campaign managers to manage the list of users and groups to include in the SBA platform		**X**		
SBA_FR_8	The system should allow end-users to conduct self-assessments on specified cybersecurity culture domains		**X**		
SBA_FR_9	The system should map the results of the assessment of different cybersecurity culture domains to cybersecurity threats based on the MITRE ATT&CK knowledge base			**X**	
SBA_FR_10	The system will provide tests and games mechanisms to assess further cybersecurity culture domains (mail phishing games / password-related games / security IQ)			**X**	
SBA_FR_11	The system should recommend existing security training programs / actions that correspond to the results of the assessment			**X**	
SBA_FR_12	The system should provide the results of the evaluation in REST API endpoints		**X**		
SBA_NFR_1	Maintainability - The questions from the questionnaires and tests of the system are not hardcoded but can be dynamically updated in each software release		**X**		**X**
SBA_NFR_2	Maintainability - The assessment domains can be dynamically updated for each software release		**X**		**X**
SBA_NFR_3	Security - The system can be deployed within the private networks of the organisations as local installations		**X**		
SBA_NFR_4	Security - All input data is protected from unauthorised access		**X**		
SBA_NFR_5	Capacity - The system should be able to handle at least 100 simultaneous transaction requests				**X**

SBA_NFR_6	There should be user documentation for all the functionalities of the tool				**X**
SBA_NFR_7	There should be documentation for the usage of API based on OpenAPI / Swagger Documentation			**X**	
SBA_NFR_8	The system should support internationalisation and localisation				**X**

SBA – end-to-end Sample test cases

Table 6. SBA. TC-01

Test Case	**01**		
FRs	SBA_FR_1, SBA_FR_3, SBA_FR_5, SBA_FR_6, SBA_FR_7		
Precondition	A subscribed user should perform the test.		
TC Step (actions)		**Obtained result**	**Verdict**
1	Login to SBA tool as "manager".	Landed in the “My Dashboard” view with advanced preview privileges.	Success
2	Change the language preference (via the dropdown menu on the upper right corner menu).	User interface appeared in the language selected (Italian in this case).	Success
3	Select "Users" from the side menu.	“Users” view appeared presenting a table with all users’ details.	
4	Select "Add User".	“Create New User” wizard appeared.	Success
5	Follow the wizard steps to create a new user.	Filled all required fields guided by the tool to complete the step.	Success
6	Select "Groups" from the side menu.	“Groups” view appeared presenting a table with all groups’ details.	Success
7	Select "Add Group".	“Create New Group” wizard appeared.	Success
8	Follow the wizard steps to create a new group.	Filled all required fields guided by the tool to complete the step.	Success
9	Select "Questionnaires" from the side menu.	“Questionnaires” view appeared presenting a list of all available questionnaires.	Success
10	Examine different dimensions and domains divided into 2	Selecting specific dimensions and domains, questionnaires list is filtered and updated accordingly.	Success

	levels ("organization" & "individual").		
11	Identify questionnaires of interest.	Selected specific questionnaires from the filtered list.	Success
12	Change the language preference and examine changes.	When Italian was selected, the questionnaire examined was presented in Italian.	Success
13	Select "Campaigns" from the side menu.	“Campaigns” view appeared presenting a list of all available campaigns along with their status.	Success
14	Select "Create new campaign".	“Create new campaign” view appeared.	Success
15	Fill in all required fields (questionnaires, users, campaign details) using entities created or located in previous steps.	Completed all required information guided by the tool and successfully created a campaign.	Success
16	Logout from the SBA tool.	After confirming the sign out decision, logged out from the tool.	Success

Table 7 SBA- TC-02

Test Case	**02**		
FRs	SBA_FR_1, SBA_FR_3, SBA_FR_5, SBA_FR_8, SBA_FR_9, SBA_FR_10, SBA_FR_11		
Precondition	A subscribed user should perform the test.		
TC Step (actions)		**Obtained result**	**Verdict**
1	Try invoking '/users', '/campaigns' or any other relative path without having logged in to the SBA tool.	Redirected to the “Sign In” page.	Success
2	Login to SBA tool as "simple user".	Landed in the “My Dashboard” view with limited preview privileges (restricted to information related to the user).	Success
3	Verify that side menu is filtered down to submenus available to simple users.	Side menu did not present the “Campaigns”, “Tests”, “Questionnaires”, etc. menu items.	Success
4	Try invoking "/users', '/campaigns' or any other relative path of	An error message appeared informing user of insufficient privileges.	Success

	submenu not available in the side menu.		
5	Select "Self Evaluation" from the side menu.	“My Self Evaluation History” list appeared.	Success
6	Select "Execute a New Assessment".	“Self Evaluation” view appears offering the possibility to select any of the available questionnaires in the tool.	Success
7	Randomly select any of the available questionnaires and run the survey.	Selected “Employee Climate I” questionnaire and completed the survey guided by the tool.	Success
8	Select “Rrecommendations” from the side menu.	“Recommendations” view appeared displaying a list of free online games related to cyber-security.	Success
9	Select an available test or game to execute.	A new tab redirects you to the game.	Success
10	Logout from the SBA tool.	After confirming the sign out decision, logged out from the tool.	Success

Table 8. SBA – TC-03

Test Case	**03**		
FRs	SBA_FR_2, SBA_FR_4, SBA_FR_12		
Precondition	A subscribed user should perform the test.		
TC Step (actions)		**Obtained result**	**Verdict**
1	Login to SBA tool as "manager".	Landed in the “My Dashboard” view with advanced preview privileges.	Success
2	Perform a rest call to any of the relative paths: • 'api/metrics/organization' • 'api/metrics/campaign/<campaign_id>/' • 'api/metrics/user/<user_id>/' • 'api/metrics/group/<group_id>/'"	Using an external tool performed all presented REST calls.	Success
3	Select "Reports" from the side menu.	“Reports” view appeared.	Success

4	Fill the filtering pane (report, level, period).	Selected to generate an organisation report at an individual level for the last 6 months.	Success
5	Examine generated graphs.	Graphs were updated to present the filtering criteria.	Success
6	Invoke "Calculation Info" and review anonymized data used for reporting.	A pop up window appeared listing information related to the filtered data.	Success
7	Logout from the SBA tool.	After confirming the sign out decision, logged out from the tool.	Success

VA TOOL TESTING

3.1.2. VA TOOL QUALITY ASSURANCE

3.1.2.1. Almost all development of the EnergyShield features of the VA tool have been added as standard features to Foreseeti's securiCAD product. This mean that the quality assurance process also follows that of the Foreseeti standard development process. This section contains a summary of this process with focus on the VA tool development.

3.1.2.2. GUIDING PRINCIPLES

The three key principles on which the quality assurance is based are:

- *Defect avoidance by a "shift left" mindset* - placing appropriate quality assurance measures as early in the development flow as possible to avoid high costs of late defect discovery
- *Continuous delivery* - built releasable artifacts continuously by keeping the production branch of release quality and automate the release process
- *Continuous improvement* – be a learning organization, encouraging introspection and inventiveness from the team to improve processes and activities related to product quality.

3.1.2.3.

SQA MEASURES BY PHASE

Requirements - Requirement review is part of the backlog grooming and sprint planning efforts. The team must have a clear understanding of the requirements but also help the Product Owner avoid mistakes. The Product Owner also has a responsibility to be flexible and sponsor stories to reduce technical debt.

Design - Design reviews are woven into the development workflow where the team will tackle major design decisions as a group, either by inserting spikes into the sprint

backlogs or by including collaborative design activities in the development of the story.

Implementation - During implementation, several quality assurance measures are taken:

- *Peer assistance* – collaboration is always encouraged
- *Management of technical debt* – leave the code in better shape than you found it!
- *Code review* – mandatory "four-eyes" for all code before merge to production branches
- *Code quality control* – use of static analysis tools to stay coding standard compliant and catch errors before merge to production branch.
- *Feature testing* – manual test of new features before merge to production branch.
- Automated testing – continuous regression tests on both feature branches and production branches.

The implementation phase is defined to be the development activities until a patch is merged into a production branch.

Testing - The testing phase is defined to mean testing activities performed on the production branches and is mainly focused on automated system tests and regression test suites.

Release - A release package will go through a manual regression test in addition to the automated tests. The manual regression test suites are primarily aimed at covering the functionality that is not fully covered by the automated test suites.

3.1.2.4. SUPERIMPOSED SQA MEASURES

In addition to the specific activities by phase, we also employ quality assurance measures that are superimposed on top of the development workflows.

Software configuration management (SCM) is an important part of the quality assurance framework. At foreseeti the SCM includes:

- **Source code control** – This is handled through the Gerrit management tool though which we develop new features through short-lived "patches" outside of the Master branch until passing all controls and it is merged into the production branches.
- **Dependency management** – We handle open-source dependencies through the Artifactory tool to ensure that only vetted and approved dependencies are used in a controlled fashion in the build process.
- **Build management** – Automation of build and test activities is done through the Jenkins CI system. Jenkins is responsible for supporting build and tests of Patch branches (before merge to production branch) as well as automated builds and daily auto-testing (system regression tests) on the production branches.

- **Defect tracking** - Security scanning of dependencies is done through the Debricked and Snyk vulnerability scanning tools.

CAUSAL ANALYSIS AND FEEDBACK

Issues are tracked in our JIRA work tracking/bug management system. We do periodical analysis of issues, trace root causes and use this for analysis and as a basis for process improvements.

3.1.2.5 All features of the VA tool have been implemented as planned and details about the implemented features are provided in Table 9.

Table 9. VA Functional and Non-Functional Requirements

ID	Requirement	Phase	Status
FR2.1	The organization using the platform must be able to allow the organizational data access constraints and controls to be represented in the VA system's access Identity and access management system	Existing	Pre-existing. Minor extension done in the project
FR2.2	A security analyst must be able to inspect, analyze and manage the infrastructure models	Existing	Pre-existing but only for the old, static threat modeling language securiLang.
FR2.3	The VA system must be able to simulate models for general IT environments	Existing	Pre-existing but only for the old, static threat modeling language securiLang. A brand new MAL-based language and MAL support was needed.
FR2.4	A security analyst must be able to inspect attack paths and cyber exposure reports general IT environments	Existing	Pre-existing but only for the old, static threat modeling language securiLang.
FR2.5	When there is a need to perform a cyber analysis of a general IT system, it must be possible for a security analyst to manually model the environment	Existing	Pre-existing but only for the old, static threat modeling language securiLang.
FR2.6	In order to analyse both IT and OT parts of there must be a basic MAL-based Energy Sector language with combined support for both environments	R1	Implemented as planned. A first version was used in release 1 and a significantly updates version is ready for release 2 of the VA tool.
FR2.7	The VA system must be able to simulate models for the Energy Sector expressed in the specific domain-	R1	Implemented as planned. In addition to the MAL support in the

	specific, MAL-based threat modeling language		VA tool, significant updates of the MAL tooling itself had to be done.
FR2.8	When a security analyst wants to manually model an Energy Sector environment, it must be possible to do so in the VA module web modeler interface	R1	Implemented as planned. Full support for model management and creation of MAL-based languages.
FR2.9	When a security analyst wants to get insights into the cyber characteristics of an analyzed environment, the VA must provide exposure reports and Critical Path attack paths for simulations of models in the Energy Sector threat modeling language	R1	Implemented as planned. Full support for simulations of MAL-based languages.
FR2.10	When a security analyst wants actionable feedback, the system must provide basic suggested mitigations for the Energy Sector language	R2	Implemented as planned. Full support for suggested mitigations for MAL-based languages.
FR2.11	In order to do detailed analysis of both IT and OT parts of there must be an advanced MAL-based Energy Sector language with combined support for both environments	R2	Implemented as planned in D2.1/D2.5. This MAL-based language is default in the VA Tool.
FR2.12	For continuous security exposure monitoring of an environment, the VA tool must have infrastructure support to build automated model creation	R2	Implemented as planned. A complete “parser” framework for automatic model generation and automation.
FR2.13	For continuous security exposure monitoring of an environment, it must be possible to automate the analysis workflows	R2	Implemented as planned. The securiCAD Enterprise SDK and Webhook features of securiCAD allows for full workflow automation. The Enterprise SDK is also implementing the VA tool adapter, linking the VA tool to the Kafka message broker.
FR2.14	When a security analyst wants actionable feedback, the system must provide advanced suggested mitigations beyond missing defenses for the Energy Sector language	R2	Partially implemented. In addition to missing defenses, there is now support for recording information on possible security controls in the MAL based languages. Next step is structural mitigations, but this is beyond the scope of EnergyShield.
FR2.15	In order to support integration to other EnergyShield platform tools the VA tools will need integration APIs	R2	Implemented as planned. This is supported in the form of the VA tool adapter, linking the VA tool to the Kafka message broker.

FR2.16	In order match real-world experiences and requirements, we must update the MAL-based Energy Sector language during and after field testing experiences	R3	Not yet implemented. This is a post-pilot and penetration testing activity.
NFR2.1	In order to support deployment and use, the VA tool must have support for installation, backups and logging	Existing	Pre-existing. Several improvements for operations have been made including extended platform support and the ability to use an external PostgreSQL database.
NFR2.2	In order to support domain-specific threat modeling language, the VA must support the Meta Attack Language framework	R1	Implemented as planned. This was a major undertaking and also spread into major improvements to the MAL platform itself.
NFR2.3	A user of the VA web GUI should have an EnergyShield branded experience	R1	Implemented as planned. A standard branding package exists for EnergyShield.
NFR2.4	As a VA tool demonstrator, the solution must be packaged so that it can be easily deployed, run, and demonstrated	R1	Implemented as planned. The VA tool development was completely possible to be made as standard extensions to Foreseeti's securiCAD plus a small integration module to the platform.
NFR2.5	In order to support continuous analysis of large environments, the simulation performance for a 10k asset model should support near real-time simulations	R3	Implemented as planned. This is already supported. And we have successfully run the VA tool in test environments of 5 – 10x this size with optimized deployment configuration.
NFR2.6	In order for the VA to integrate into the EnergyShield platform, the VA must be adapted to the authn/authz/logging and operational requirements of the platform	R3	Implemented as planned. This is already supported by adding SSO and federated logging through standard support of OpenID Connect and SAMLv2.

3.1.3.

AD TOOL TESTING

The AD tool is implemented and operates on real process data taken from a real EPES asset. The EPES asset monitored by the AD tool is defined as an operational site with relevant machinery (e.g., power plant, sub-station etc.), and it must have an ICS/SCADA with a PLC that manages the operational process with its end devices (sensors and actuators - analogue and discrete IOs).

TEST SET-UP. To test and demonstrate the tool capabilities in a lab environment, the above-mentioned process data can be available to the tool in two manners:

1. Physical testbed:

The testbed in the lab will be a small-scale physical model of an EPES asset with a real ICS/SCADA managing the asset's operational process. The tool will be installed physically on the ICS. The electrical signals of the testbed will be used for the process data, as the tool really operate when installed in the field.

2. Virtual testbed:

Real process data of a real EPES asset in the field will be recorded for a sufficient period and will be replayed as a dataset to be used by the AD tool as a replacement of the physical environment. The tool will be installed without its hardware layer on a dedicated server and will use the dataset as input data instead of real electrical signals from the field. This recorded process data can also be generated back to electrical signals with a special simulator (if available in the lab), for higher similarity to the tool operation on-site in the field.

TEST PLAN. After the tool is installed and implemented, the following steps should be taken to test the AD tool in the lab:

1. Learning phase:

The tool will learn the normal behavior of the process it is monitoring. The process will run in its normal operation, without interference, for a sufficient period of time. This period length is set according to the complexity and the amount of monitored IOs by SIGA. The formed dataset will contain electrical signals from the ICS/SCADA of the monitored asset, either collected by a physical installation of the tool or by receiving these electrical signals pre-recorded from a real EPES asset.

2. Testing phase:

The testing phase will take place after the AD tool has learned the normal process and will usually consist of various simulated cyber-attacks on the ICS/SCADA (e.g., Man-in-the-Middle type of attacks) for the tool to detect. These cyber-attacks will simulate a situation in which the attacker has reached the controller (e.g., PLC, RTAC etc.) and is now changing the process parameters and thresholds, trying to cause the asset to fail and malfunction.

These attacks can be performed in two different manners (in correlation with the test set-up method):

a. In a physical testbed, the test team (i.e., red team) will perform the attack in real time by changing the sequence of operation of the ICS/SCADA of the testbed physical asset. This can be conducted by changing parameters or the logic diagram of the controller, causing the process to behave abnormally.

b. In a virtual testbed, the test team will prepare an attack dataset that will be streamed into the tool as the operation of the process during the attack. This dataset will contain simulated data that is representing the abnormal behavior of the process.

The AD tool will detect the anomalies in the process that have been simulate by the test team and raise alert in the dashboard and send it to any other interface required (e.g., SIEM).

Important note: If the test team wish to test the tool with several attack scenarios, it is recommended that the different scenarios will be performed separately from one another, with a gap of normal process operation between them, to allow clarity on which alert was raised for which attack.

3. Reporting Phase:

After the test the AD tool will issue a report of all the attacks that were detected with visualization and information on each attack.

3.1.4 DDoSM TOOLS TESTING

While the generic Ammune API-DDoS profile for the "smart meter" APIs is straight forward, as generated mostly from a technical analysis of the simulated requests/replies profiles, it is worth understanding its BL profile characteristics. The "smart meter" application contains major BL bottlenecks.

- **Smart meter update** - Requires to update the DB and additional data structures that are used to estimate the power consumption. Data update in distributed systems is usually a potential bottleneck.
- There are however worst-case scenarios for this functionality as **Trying to update non-registered smart meter reading** - Heavy-duty call in which the server is round-tripping to the DB and can't do any calculations in cache
- **"Read region power consumption"** - Heavy-duty call in which the server is round tripping the DB

Ammune complexity estimation of these API calls (see above) is easily obtained as influencing of the overall model, while an attack on these APIs generally will make more impact, though these requests may come at a low frequency ("Low and Slow").

Table 10. DDoSM Functional and Non-Functional Requirements

FR_ID	Requirements description	Existing	Release 1	Release 2	Release 3
DDoS_FR_1	Building a Smart Grid DDoS attack simulation and regular normal traffic scenarios.				X
DDoS_FR_2	Design Ammune User Interface provide the user with dashboard to visually display the DDoS attacks it identifies AND the functionality to manage protection module of Ammune.	X	x	x	
DDoS_FR_3	Create Ammune documentation for user Interface management in English	X	x		
DDoS_FR_4	install Ammune SW in electrical grid production and provide protection from Web DDoS attacks.	X			x

DDoS_NFR_1	Syslog Access and Alert messages local sent to syslog servers over UDP/TCP/ HTTPS, All changes to Ammune configuration are logged and can be accessed by administrator.	X			
DDoS_NFR_2	Performance: Ammune identifies DDoS attacks within 60 sec. For most complex attacks - Mitigation of attack starts within 90 sec drops to. Within 5 minutes 90% of the attack mitigated.			x	X
DDoS_NFR_3	Availability: Ammune 24/7 availability using high availability cluster (Active-Active) for all modules. Local management is accessed with web browser HTTPS connection over port 8443. Ammune can also be securely accessed over ssh	X			
DDoS_NFR_4	Reliability: Ammune runs internal watchdogs that constantly monitors reliability of the installation and automatically running remedial actions such as restarting unhealthy modules or opening more machines	X			
DDoS_NFR_5	Recoverability: Ammune state is stored in high availability DB and can thus be resumed at any time Recovery within 2 minutes				X
DDoS_NFR_6	Robustness: Ammune internal watchdog process checks all system modules and reports to Ammune Management	X			X
DDoS_NFR_7	Integrity: Ammune communication with the db is secured. Automatic housekeeping of assures db integrity and disk space. DB backup internal Ammune or external disk	X			
DDoS_NFR_8	Maintainability: Ammune is designed to provide as high availability reverse proxy, it is easily integrated with any std cloud or on-premise environment				X
DDoS_NFR_9	Documentation: Administration Guide, Installation Guide and Integration Guide			x	X

3.1.5.

SIEM TOOL TESTING

SIEM tool is installed in laboratory conditions, as a stand-alone tool. That means that SIEM solution (Cluster) must be installed in a Server, to monitor it. Other than that, Agents must be installed in other endpoints of the laboratory, such as:

- Laptops
- Desktops
- Other Servers
- VMs

These endpoints will be automatically sent logs and events to the main Server (main host), where the SIEM solution will be installed. As a result, the test will be performed based on real-time data (syslogs, syschecks), coming from all the aforementioned endpoints. Furthermore, having this data inside the main cluster, adapted and

customized functionalities (log analysis, dashboards and others described in “D4.4 SIEM tool final release”, can be tested too.

The laboratory conditions and endpoints’ requirements are depicted in the figure below.

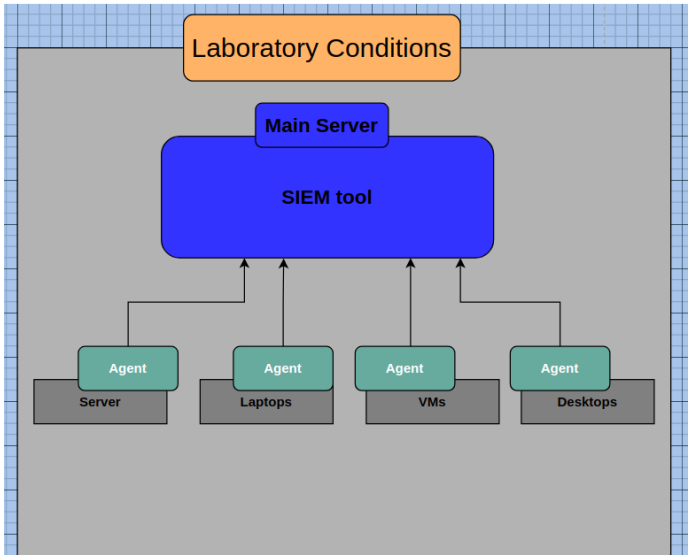

Figure 3. SIEM tool PHASE 1 laboratory conditions

SIEM tool will integrate with the Rest toolkit via Apache Kafka and REST APIs. This depends on the communication ways of the other tools. SIEM has adjusted, customized, and developed a data pipeline mechanism (logstash) for integration via Apache Kafka, in order to produce and consume data to the Apache Kafka Topics. However, since there are tools that are not integrated via Apache Kafka, SIEM tool will make REST API requests (Post and GET HTTP methods), in order to communicate with them.

The test data coming from the SIEM solution will be real-time data, coming from all the monitoring endpoints, where agents are installed (see Phase 1 testing). These data are going to be analysed and mapped to specific rules, in order to be correlated and indicate awareness about endpoint’s health and safety. As a result, the rest tools will obtain as input SIEM’s outcomes (data from indices), in order to finally adjust them to their tools’ environment.

The test data coming from the rest of the tools to SIEM, will be their outcome data, which will be consumed by Logstash or REST APIs and finally stored in Elasticsearch indices in the SIEM environment.

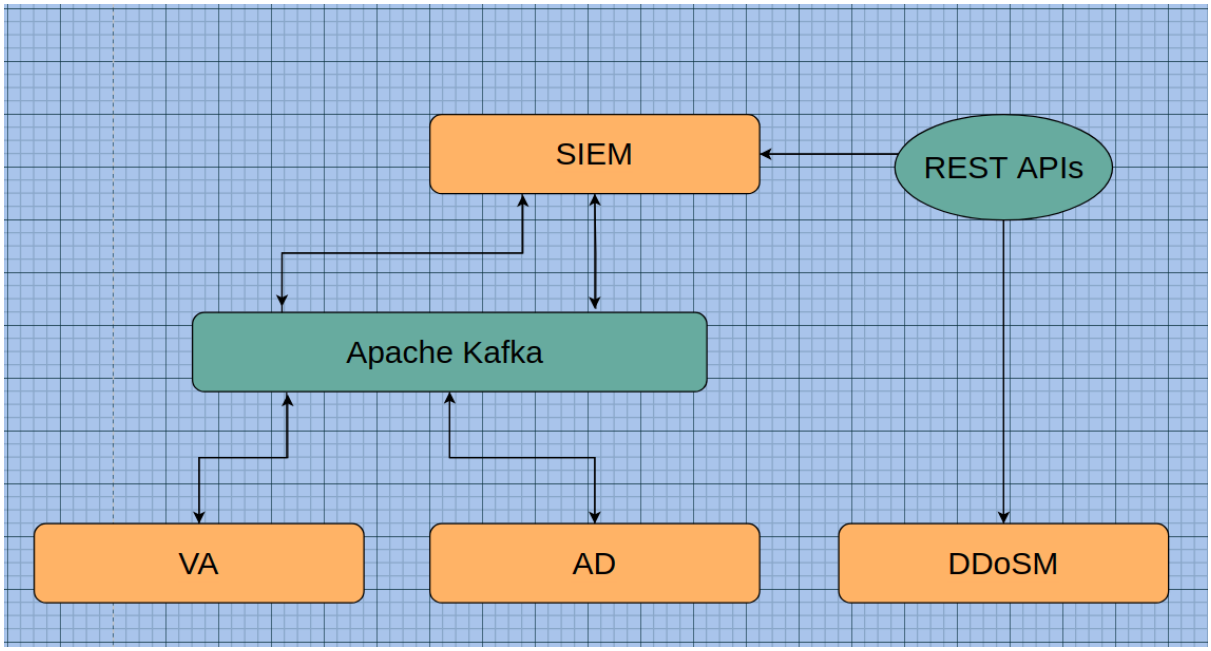

Figure 4. SIEM tool in PHASE 2 testing

SHORT DESCRIPTION OF THE TESTING METHODOLOGY

3.1.5.1 After the deployment of SIEM solution and the agents in the monitoring endpoints, the testing methodology could be addressed by:

- Verifying that Elasticsearch is running by visiting the REST API in https://SIMAVI's_VPN_IP:9200, where details such as the cluster name, uuid, among others will be revealed.
- By trying to connect to Elasticsearch API, the user will be asked to provide credentials. Furthermore, when the user tries to log in, credentials will be requested too. This is a way of testing XPACK and authentication in the SIEM solution
- In order to test the health of the system, the easiest way is to navigate into the "Stack monitoring" where the health of SIEM-cluster (Elasticsearch & Kibana) will be revealed (green indications mean that the system's health is fine and there is no disk space issues).
- The next phase is the testing of queries. This can be achieved with curl command inside the "Dev tools" or by a terminal (connected via ssh to the system). Testing queries can be made for searching the number of the installed agents, retrieving data from an elasticsearch index, among others.
- The testing methodology could be also applied by using tools such as Postman or/and Cerebro or/and Logstest, in order to make REST API calls.
- For the integration testing, the methodology is to request via testing queries the incoming indices (data and logs from the other tools).
- Another testing methodology to be applied, is the testing on the capabilities of SIEM, especially the Active Response. Other capabilities, such as log data analysis, file integrity monitoring could be addressed by making queries

(instead of only viewing on the final screen their functionality). However, the Active Response can be tested only when an attack attempts to invade the system. “D4.4 SIEM tool final release” has demonstrated the testing methodology in “Chapter 3.1 Countermeasures”. Using the methodology from this chapter, attacks such as Brute force, Shellshock attack and SQL injection will challenge the SIEM environment to fire its rules on them and trigger the active response, in order to face them.

Table 11. SIEM DDoSM Functional and Non-Functional Requirements

FR_ID	Requirements description	Existing	Release 1	Release 2	Release 3
SIEM_FR_1	Collection of log files: The SIEM collect data from the Agents which will be installed to the operating system	x	x		
SIEM_FR_2	Auditing who-data: This information contains the user who made the changes on the monitored files and also the program name or process used to carry them out.			x	
SIEM_FR_3	Active response: Perform various countermeasures to address active threats, such as blocking access to an agent from the threat source when certain criteria are met.	x		x	
SIEM_FR_4	Agentless monitoring: Agentless monitoring allows you to monitor devices or systems with no agent via SSH, such as routers, firewalls, switches and linux/bsd systems. This allows users with software installation restrictions to meet security and compliance requirements.				x
SIEM_FR_5	System Event Monitoring: The SIEM is able to monitor the events which will be sent from Agents	x		x	
SIEM_FR_6	System Alerting: The SIEM will provide Alerts when will detect suspicious changes on the system				x
SIEM_FR_7	Ability to build specific alert conditions: SIEM will have specific alerts based on requirements	x	x		
SIEM_FR_8	Intuitive visualization interface, with log processing abilities, data management & alert monitoring. Data exploration & visualization, with the ability to produce several charts like Heatmaps, histograms and pie charts.				x
SIEM_FR_9	Ability to build specific alert conditions: SIEM will have specific alerts based on requirements>		x		
SIEM_FR_10	Intuitive visualization interface, with log processing abilities, data management & alert monitoring. Data exploration & visualization, with the ability to produce several charts like Heatmaps, histograms and pie charts.>	x		x	x
SIEM_NFR_1	User authentication against active directory				x

SIEM_NFR_2	Support for Role - based Access Control			**x**	
SIEM_NFR_3	Support for Multi-Tenant Environments				**x**
SIEM_NFR_4	Performance analyzer able to provide System Diagnostic insights				**x**
SIEM_NFR_5	Open SSL and TLS 1.2 support for regulatory compliance reasons				**x**

3.2. ENERGYSHIELD TOOLKIT TESTING RESULTS

Considering the particularities of the system, a phase test is proposed to demonstrate how the tools integrated in the EnergyShield toolkit are placed in the Operational technology (OT) environment, and how they can influence OT behaviour, especially in relation with Smart Grid type operations.

The proposed phase test approach follows three main stages:

1. Testing the tools placed in the toolkit in a laboratory environment. At this stage, the way the tools are interfaced and are communicating is tested by means of input and output information received from consumption points.
2. Testing the tools integrated in the toolkit and placed in the OT environment. This is the most extensive test and tests the functionality as designed and implemented in the final system.

The proposed approach considers each tool's unique functionalities and particularities. Thus, the next sub-chapters are presenting in detail how each phase will be applied to each tool, and finally for the integrated toolkit.

3.2.1.1. PHASE 1 TESTING

For this phase there is NO interaction with OT. The tests target the interaction between tools, and less considering individual tools functionality.

To run this testing phase, the message broker mechanism should be up and running. The dialog between tools is considered to be part of the message exchange governed by the message broker.

Each tool will produce messages for some specific topics and will consume messages from other topics.

- Vulnerability Assessment (VA) tool.

The outputs are for SIEM.

It consumes information from SBA, AD, SIEM.

To test the outputs, simulations will be run on the model. The outputs, placed on output message queues (topics) will be tested against test data.

To test the inputs, test data will be placed in the input message queues (topics). The model will be updated based on that data. Then the simulation will be run, and again, the results placed in the output topics will be tested against test data.

- Security Behaviour Analysis (SBA).

The outputs are for VA.

SBA shall only indirectly interact with SIEM.

To test the outputs, the tool will collect standard (test) data from OT users. Then the data placed in the output topics will be tested against test data.

- Anomaly Detection (AD) tool**.** This is a hardware tool.

The outputs are for VA, SIEM

It consumes information from SIEM

To test the outputs, the tool will be connected to a laboratory device. Then the data placed in the output topics will be tested against test data.

To test the inputs, test data will be placed in the input message queues (topics). The toll will be run, and outputs will be compared with expected test data.

- DDoS Mitigation (DDM) tool.

The outputs are for SIEM

Inputs are from SIEM

To test the outputs simulated attacks will be sent. Messages placed in output queues will be tested against test defined data.

To test the inputs, test data will be placed in the input message queues (topics). The tool will be run and outputs will be compared with expected test data.

- **SIEM Tool**.

The outputs are for VA, AD, DDoSM

Inputs are from VA, AD, DDoSM

To test the outputs, the application will be run with the user entering test data. Then the output topics will be read and data compared with test data.

To test the inputs, test data will be placed in the input message queues (topics). The tool will be run and outputs will be compared with expected test data.

Test the full path.

This will follow several full paths to be tested. Our proposal is:

3.2.1.2.

1. AD->VA->SIEM->AD
2. AD->VA->DDM->SIEM->DDM
3. SBA->VA->SIEM

PHASE 2 TESTING

For this phase **there IS interaction with OT**. The tests are focused on the interaction between tools, and in relation with OT.

To run this testing phase the message broker mechanism should be up and running. The dialog between tools is considered to be part of the message exchange governed by the message broker.

Each tool will produce messages for some specific topics and will consume messages from other topics.

The installation in the OT will be performed so that this will not degrade the functionality.

This is the most extensive test phase and includes:

a) Testing the placement of the tools in OT and how they influence OT normal functionality
b) Testing the tools against the expected results when placed in OT
c) Registering OT behaviour when tested tools are working.
d) Testing the integrated toolkit placed in OT, for the way the tools are interacting inside toolkit and between toolkit and OT.
e) Testing how smart grid components are influenced by the tool

The points from a) to e) are explained for each tool and for the integrated toolkit in the following.

- **Vulnerability Assessment (VA) tool**

a) The tool will be installed on o computer inside OT. Considering the specification of the tool it will not interact directly with OT component.
b) The tests will be the same as that for the Phase 2.
c) It is expected that the OT will be not directly influenced by the presence of the tool alone. But when integrated it is expected to indirectly influence the OT via SIEM.
d) The interaction between tools will be similar with that defined in Phase 2.
e) The smart grid is expected not to be influenced by the functionality of this tool alone. But when integrated is expected to indirectly influence the smart grid via SIEM.

The outputs are for SIEM

It consumes information from SBA, AD (via SIEM), SIEM.

- **Security Behaviour Analysis (SBA)**

a) The tool will be installed on a computer inside OT. Considering the specification of the tool it will not interact directly with OT component.
b) The tests will be the same as that for Phase 2.
c) It is expected that the OT will be not directly influenced by the presence of the tool alone. But when integrated is expected to indirectly influence the VA.
d) The interaction between tools will be similar with that defined in Phase 2.
e) The smart grid is expected not to be influenced by the functionality of this tool alone. But when integrated is expected to indirectly influence the smart grid via VA.

The outputs are for SIEM (via VA).

- **Anomaly Detection (AD) tool.** This is a hardware tool.

a) The tool will be installed inside OT. Considering the specification of the tool it will interact directly with OT component. After installing the tool, the OT components considered will be tested and compared with their functionality without AD.
b) The tests will be the same as that for the Phase 2.
c) It is expected that the OT will be not influenced by the presence of the tool alone. But when integrated is expected to indirectly influence the OT via VA, SIEM.
d) The interaction between tools will be similar with that defined in Phase 2.
e) The smart grid is expected not to be influenced by the functionality of this tool alone. But when integrated is expected to indirectly influence the smart grid via VA, SIEM.

The outputs are for VA, SIEM

It consumes information from SIEM

- **DDoS Mitigation (DDoSM) tool**

a) The tool will be installed on a VM that can receive the traffic, before reaching the OT. Considering the specification of the tool it will interact directly with the IT infrastructure of the OT After installing the tool, the OT components considered will be tested and compared with their functionality without DDM tool present. No differences should be found.
b) An attack against the OT will be simulated. As the attacks comes from the external IT, this simulation is to be performed from the external IT. It is expected that the tool will stop the attack, the external attacker will be isolated, and the OT will continue to function normally.
c) It is expected that the OT will be influenced in the way that the attack will be stopped, meaning that it will not reach the OT, or if reaching at the beginning, it will be stopped before producing effects. As a DDoS attack can have cascading effects, it is expected that OT will react even by stopping the information from smart meters.
d) The interaction between tools will be similar with that defined in Phase 2.
e) The smart grid is expected to be influenced by the functionality of this tool in the sense that, according with DDoS definitions, some functionality will be changed to avoid the attack.

The outputs are for SIEM

Inputs are from SIEM

- **SIEM Tool**.

a) The tool will be installed on o computer inside OT. Considering the specification of the tool it will interact directly with OT component. After installing the tool, the OT components considered will be tested and compared with their functionality without AD

b) Different commands will be issued from SIEM tool, according with the test data defined. The behaviour of OT will be registered and compared with functionality without SIEM tool.
c) It is expected that the OT will be influenced in the way that the OT will behave according with the commands sent.
d) The interaction between tools will be similar with that defined in Phase 2.
e) The smart grid is expected to be influenced by the functionality of this tool in the sense that, according with SIEM definitions, some functionality will be changed.

The outputs are for VA, AD, DDM

Inputs are from VA, AD, DDoSM

- **Test the full path.**

When testing the full path, in OT environment the assumption is that all the tools are already installed, and the behaviour of the OT with each individual tool functioning alone is known.

This will follow several full paths to be tested. Our proposal is:

Protect the OT - Command smart grid when an anomaly is detected

1. AD->VA->SIEM->AD->OT

This test will simulate an anomaly. The AD tool should detect, send a message to VA and SIEM, and SIEM should act to command the smart grid accordingly.

Protect the OT - Avoid a DDoS Attack, and add elements to whitelist

2. DDM->SIEM->DDM-OT

This test will simulate an attack. The DDoSM tool should detect, send a message to SIEM, and SIEM should consider low risk, and tell this to DDoSM. Then another attack will be simulated, a high risk one. This time DDoSM is expected to command the smart grid accordingly.

Improve the system - Adjust VA parameters based on SBA

3. SBA->VA->SIEM ->OT

This test will conduct a behaviour analysis. Based on the result the VA model will be updated, and the result will be sent to SIEM. SIEM will assess the information and ask the administrator to adjust OT parameters.

Protect the OT - Avoid a Brute force, Shellshock and SQL injection Attack, and add elements to host-deny list

SIEM-> REST tools

This test will obtain data from the three above attacks on the monitoring endpoints to the SIEM environment. SIEM will actively respond to them by denied the host that makes this attack, as long as send the alert and the respond to the rest of the tools.

3.2.2. SBA

The SBA tool was tested with the practitioners from both Italian and Bulgarian Practitioner.

Italian Pilot. As described in detail in ***D6.1 Offline field trial report***, the SBA tool was submitted to a significant cluster of beta-testers and early adopters who were selected to assess the business value and applicability of the questionnaires and games implemented. Each questionnaire of the SBA tool was analysed to evaluate if it is directly applicable to the IRETI test case or not, or if minor modifications were needed (e.g., regulatory adjustments to the Italian framework). In a second phase, the same analysis was carried out at a single question level.

Initially, the interested business units and employee roles were individuated within the company focusing on the Advanced Meter Management (AMM) personnel, managers and operators, remote control system responsible personnel, IT referents, cybersecurity team, DSO top management, and DSO technical and OT staff. Secondly, questionnaires were characterised by setting the relation among units, roles and groups of questions.

At a second stage, tests and questionnaires were submitted to “early adopters” (e.g., one person per significant unit individuated in the previous phase) so as to gather valuable feedback regarding both SBA’s content, applicability and usability.

Once questions were categorised and modified accordingly to IRETI requirements, all SBA questionnaires were translated into Italian since the vast majority of IRETI staff is native Italian speakers and lifting the language barrier was expected to assist in the overall security assessment.

Having concluded this testing phase with the assistance of the Italian pilot, SBA was ready to be submitted to a broader sample of users and practitioners in order to assess the overall company’s cyber and physical security culture.

Bulgarian Pilot. The SBA tool was demonstrated to representatives of the Bulgarian pilot who subscribed to the SaaS version so as to familiarise with and explore the capabilities of the tool.

At a second phase, various actors in the Bulgarian energy value chain (TSO, DSO, generation plants, prosumer, etc.) were involved in a more detailed testing of the applicability and usability of the tool. During this phase, specific roles were assigned to different partners so as to analyse in detail the possibilities offered via the assessment mechanism of the tool.

Having adjusted and improved the tool based on the Bulgarian pilot’s feedback, we proceeded in translating all available questionnaires, thus fulfilling SBA localisation and internationalisation goals.

Other Applications. During the COVID-19 crisis, the CSC framework was used to design a cyber-security culture assessment campaign targeting critical infrastructures [**GES20**, **GEA20**]. Its revealing findings [**GED20**] provided significant feedback to the participating EU organisations. Insights and recommendations towards enforcing their cyber-security resilience were offered, further contributing to this research domain.

This scientific effort inspired SPHINX, an EU project aiming to enhance the cyber protection of the Health and Care IT Ecosystem [**SPH19SPH19**] and triggered a collaboration activity with EnergyShield. More specifically, the Cyber Security Culture framework assisted SPHINX security specialists in the design of a two-phase security awareness campaign targeting health sector personnel.

The CSC and its implementation tool, SBA, were evaluated and exploited in both wide application scenarios while gaining recognition by IT and security specialists of different business domains. The feedback provided throughout the process assisted in improving our methodology and approach towards end-users of different business domains and industries.

VA

3.2.3.

The VA tool can support a range of use-cases, from proactive manual analysis of the security posture of an EPES/ICS system in a PDCA fashion to a near-real-time, continuous analysis in a SOC or similar operational context.

The former is the fundamental use case for the tool in EnergyShield, while the latter is a very compelling operational solution enabled by the Kafka message broker and SIEM. The operationally integrated context is very exciting in its opportunity to provide situation awareness to SOCs, allowing the analysts to analyse effects of known or suspected breaches and threat hunting, but also requires site-specific auto-modelling capabilities as well as careful tuning to be practically usable in stressful incidents response situations.

The VA tool is primarily a tool for manual security analysis, done by security analysts at an intermittent basis. There are many possibilities to automate and tighten the integrations and make the VA a continuous and automated tool, providing constant situation awareness to SOCs/CSIRTs.

3.2.4.

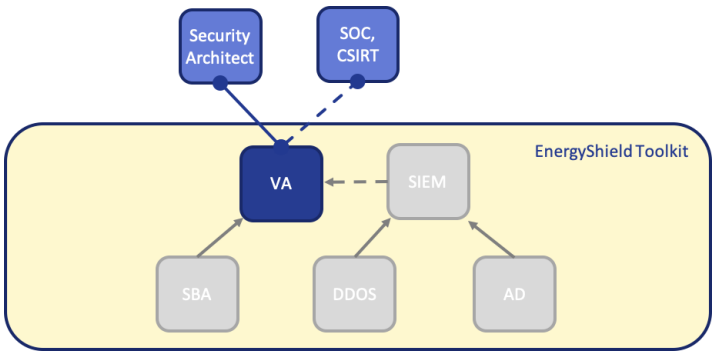

Figure 5. VA Integration

AD

The deployment of the AD tool in the Italian pilot in Martinetto substation in Turin involved the installation of SIGA hardware platform (i.e., SigaBox) in the substation, with the SIGA software and ML engine running on the platform computing device. For this purpose, IRETI designed and installed devices for the connectivity among the

SigaBox (the hardware platform of SIGA's AD tool) and the grid. This part mostly consists in the installation of transducers and voltage/current converters to gather measurements of the sub-station process from the field and transforming them into SCADA electrical signals that can be inserted as input to the SigaBox. IREN's ICT and cybersecurity provided the SigaBox cellular connectivity and the integration with IREN firewalls and security tools. The following table lists the variables monitored by the devices in field:

Table 12. List of all Monitored IOs (Input/Output) in the Sub-Station

No.	Type	Tag	Description and Function	Calibrated Value Range & Units			Signal Type		
				Min	*Max*	*Units*	*Min*	*Max*	*Units*
1	DI	G87512-A119-S407	Disturbance recorder	Off	ON		0	24	V
2	DI	G87512-A119-S408	Switch status linea Tofane	Off	ON		0	24	V
3	DI	G87512-A119-S408	Switch status linea Massaua	Off	ON		0	24	V
4	DI	G87512-A119-S408	Switch status linea Veronese	Off	ON		0	24	V
5	DI	G87512-A119-S408	Switch status linea Potenza	Off	ON		0	24	V
6	DI	G87512-A119-S408	Switch status linea Carrara	Off	ON		0	24	V
7	AI	G87512-A119-S408	Current marsurement linea Tofane	0	400	A	0	10	V
8	AI	G87512-A119-S408	Current marsurement linea Massaua	0	400	A	0	10	V
9	AI	G87512-A119-S408	Current marsurement linea Veronese	0	400	A	0	10	V
10	AI	G87512-A119-S408	Current marsurement linea Potenza	0	400	A	0	10	V
11	AI	G87512-A119-S408	Current marsurement linea Carrara	0	400	A	0	10	V
12	AI	G87512-A119-S407	Tensione omopolare	0	22000	V	0	8.33	V
13	AI	G87512-A119-S407	Tensione di fase	0	22000	V	0	8.33	V

The SigaBox is supplied as a closed box with all hardware components installed and pre-wired inside, with terminals ready to be connected to the electrical signals from the transducers (see Figure 1). The SigaBox consists of the following components:

a) **Isolators** - The isolators (isolated transmitters) are responsible for duplicating the electrical signals with full isolation and no risk to the monitored electrical signal and send it to the DAQ.
b) **DAQs (Data Acquisition Devices)** - DAQs are highly flexible devices converting analogue and discrete electrical signals into digital information. The DAQ send the information to the IPC by using MODBUS protocol over TCP/IP.
c) **IPC (Industrial PC)** - The IPC is a computing platform suited and certified for an industrial environment with sufficient computing power to run SIGA's light edge agent, a local DB service. All the analytical tasks and computing are done on the main server, although if needed it can also be performed on the IPC.

d) **Cellular Router** – The router is connected to the IPC and is providing to it an internet connection, allowing remote usage of the SIGA dashboard and alerting to users and to the EnergyShield platform.

Figure 6. The SigaBox installed within IRETI's primary substation.

Figure 6 is a scheme presenting the connections of the wires between the SigaBox and the SCADA of the substation, and the inner connections of the SigaBox components. Figure 7 is a front view drawing of the SigaBox.

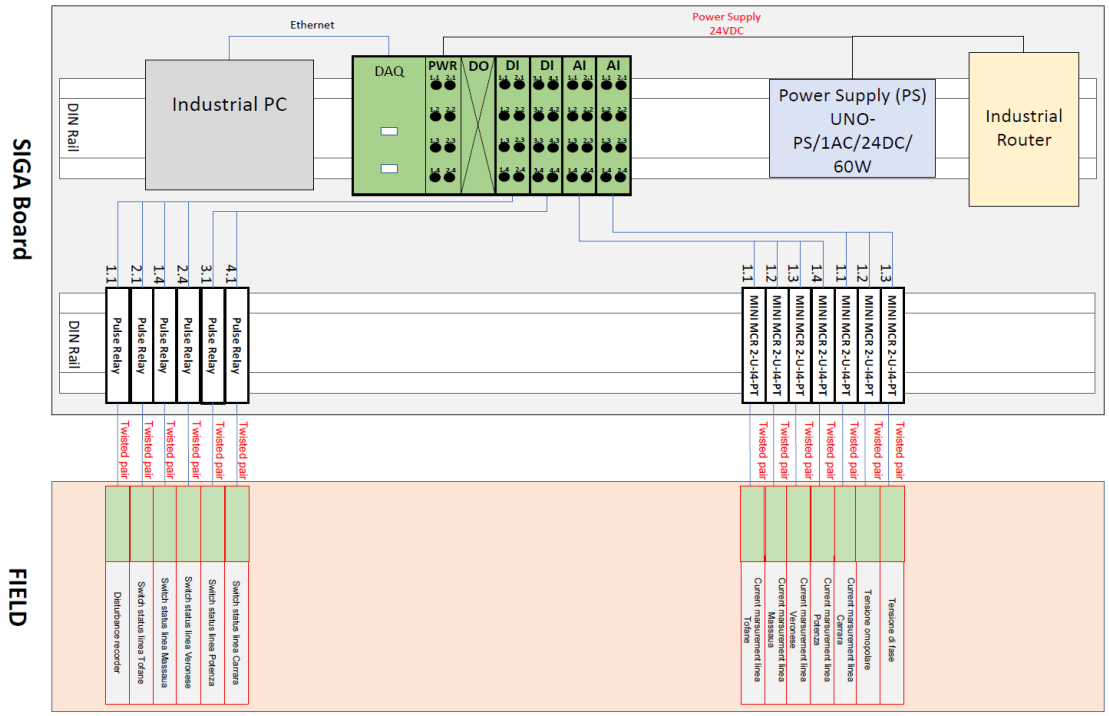

Figure 7. Schematic of the electrical connections in the SigaBox and to the SCADA

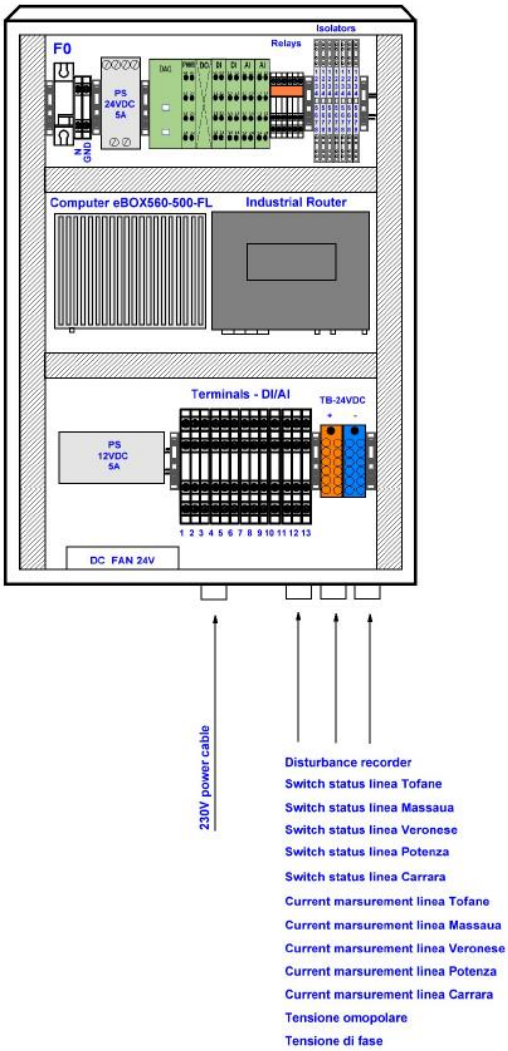

Figure 8. Front View Drawing of the SigaBox installed in Italian Pilot

The AD tool installed in the pilot is now in its learning phase, in which the ML models are learning the normal behaviour of the process. Once the learning period will be complete, the tool will start to detect anomalies, caused by abnormal behaviour of the substation operational process. These anomalies can potentially be caused by a cyber-attack performed on the SCADA, in which the attacker is trying to manipulate the substation process operation and harm the machinery, causing a breakdown of the substation. This can of course stop the electricity transfer out of the sub-station and in some cases endanger the safety of the substation and even risking human life.

The AD tool dashboard (GUI) in the Italian pilot is enabling the users to see all visualizations of the process, alerts on anomalies, perform analysis and forensics and all other dashboard functions. Figure 4 is a screenshot of the main screen of the dashboard, the Asset screen, in which the user can see a high-resolution graph of the process data in real-time.

Figure 9. AD tool dashboard main screen in the Italian Pilot

3.2.5. DDOSM

Alerts and security incidents generated by Ammune can be forwarded to other tools such as SIEM, SOAR, Ticketing systems, etc.

There are special integration patterns such as for Splunk, Arc-sight and ELK. Special integration pattern was added to support the Energy Shield project.

- Ammune as a software-based solution can be deployed as VM or dockers.
- Ammune can be controlled by APIs and integrated in CI/CD processes.
- Ammune could be integrated with API-GWs to receive traffic log feed for analysis. Log feeds could be fetched from other sources as well, such as SIEM or web logs.
- Ammune can web hook its mitigation commands to FWs, IPS devices, API-GW, WAFs and more.
- Ammune can integrate with packet brokers as source of traffic feed. In this case special adapter module extracts logs and forwards to Ammune main engine.

DDoS attack simulations are made including normal background traffic in order to simulate realistic attack scenarios on the Smart Meter APIs.

Attack #1 - Flooding the server with bogus smart meter update requests, where smart meter id and reading is selected randomly

Summary: Ammune was able to perform efficient mitigation in 30 seconds from the attack initiation. As the botnet sources rotate, Ammune was able to update "on the

fly" its mitigation policy, without any further degradation of service. Figure 10 presents Ammune dashboard print screen during attack #1.

Figure 10. Ammune Dashboard during Attacl#1

Attack #2 - Flooding the server with read region power consumption (heavy requests)

Summary: Ammune started an efficient mitigation in 30 seconds from the attack initiation. As the botnet sources were rotated, Ammune modified on-the-fly its mitigation policy, without any further degradation of service. Figure 11 presents Ammune dashboard print screen during attack #2.

Figure 11. Ammune dashboard during Attack#2

Attack #3 – Combination of attack #1 and attack #2

Summary: Ammune starts to mitigate the attack after 30 seconds from its initiation, which is the experience “setup time” for a visible impact of the attack on API activity,

under the simulation conditions. As the botnet changed it sources, Ammune modified on-the-fly its mitigation policy without allowing any further degradation of service. Figure 12 presents Ammune dashboard print screen at attack #3.

Figure 12. Ammune dashboard during Attack#3

3.2.6. SIEM TOOL TESTING

SIEM tool will be installed in the same host as the rest of the toolkit. This host must be a central Server inside the laboratory and combine PHASE 1 and PHASE 2 testing, meaning that SIEM will be installed in a Server and the integration with the rest toolkit will be enabled (e.g there are going to be producers and consumers).
Agents communicating with SIEM Cluster (as PHASE 1 indicated) will be installed in monitoring endpoints inside OT. These endpoints could be Desktops, Laptops, Servers, VMs and they must interact directly with OT component. These OT components could be RTUs, PMUs, Hydro-Plant or other critical assets of the EPES infrastructure.
Within this approach, the testing could be addressed as following:

- Monitoring Endpoints connected with OT assets (testing log analysis, file integrity monitoring, events management)
- Known attacks will be made in the monitoring endpoints where the agents are installed. If an attacker gains access to e.g a laptop connected with an OT asset, then he can easily make a malicious impact in the whole infrastructure. SIEM warns but also actively responses to this possible attack, in order to protect the sector (testing Active Response).
- It is expected the aforementioned endpoints included/connected on the OT to interact to the threat
- The interaction between tools will be similar with that defined in Phase 2.

- In order to test the “Learning and Sharing Mechanism”, there is the need to provide an SMTP server, as long as EPES sector’s actors and roles. If these requirements are provided, as long as emails of the actors, then email notifications will be sent to whole sector.

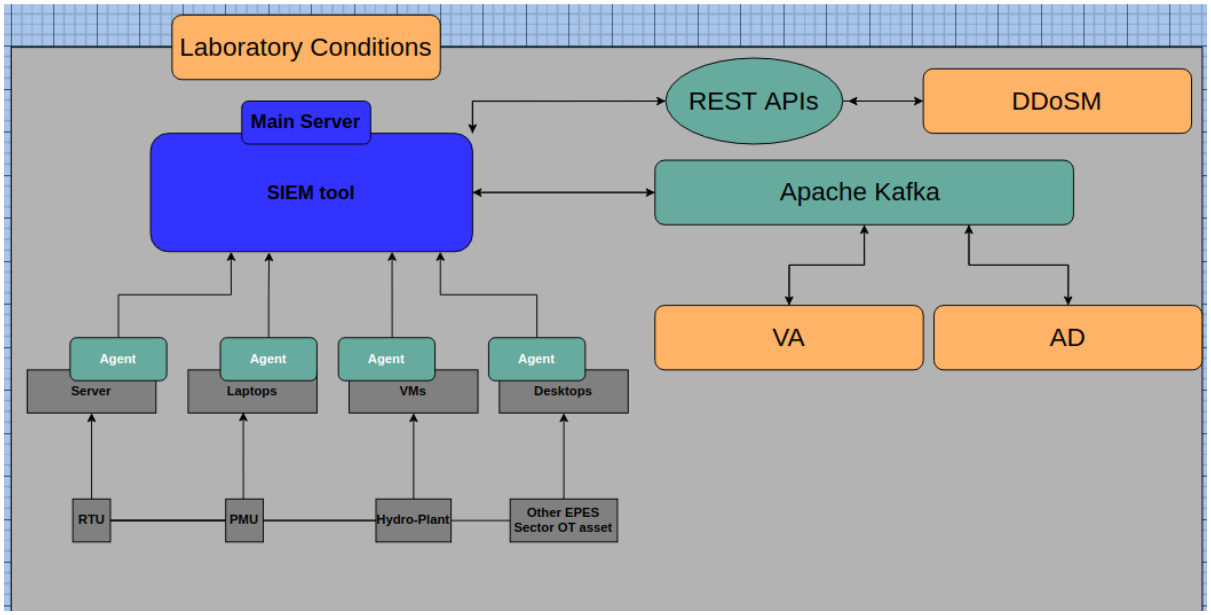

Figure 13. SIEM tool in PHASE 3 testing

4. CONCLUSION

The present document reports on the progress of software development (EnergyShield individual tools and toolkit), on the applied quality assurance methodology, while also documenting testing results.

Based on the previously achieved outcomes in terms of technical requirements, use cases descriptions, architecture design and the last release of the EnergyShield tools, the integration platform development, this test and quality assurance report provides relevant updates considering the implementation of tools and toolkit features and functionalities. All functionalities and features provisioned in the analysis phase [**ESH11**] are available and detailed in the corresponding tool reports submitted in their final version at the end of M30.

Considering the guidelines proposed in D5.1 for testing, pieces of evidence of testing both individual tools and toolkit are provided alongside with details about the anticipated scenarios for testing within the OT environment.

Three types of results on testing Energy Shield the tools and toolkit are presented: (1) availability of functional and non-functional requirement per each EnergyShield tool and a user manual (included as annex as part of the final reports on tools, (2) toolkit integration of tools (further elaborated in D5.5) and (3) scenarios about testing the toolkit within OT environment.

A final version including all the results of the architecture, integration and testing alongside with updated documentation will be included in D5.7 Common software platform release, incl. user and developer documentation – final version that will be submitted at M34 after the completion of field trials.

REFERENCES

[CHR18] Chren, S.; Rossi, B.; Bühnova, B.; Pitner, T. Reliability data for smart grids: Where the real data can be found. In Proceedings of the 2018 Smart City Symposium Prague (SCSP), Prague, Czech Republic, 24–25 May 2018.

[ESH11] Energy Shield Consortium (2019) D1.1 Technical requirements specification

[ESH15] EnergyShield Consortium (2021) D1.5 System Architecture final update

[ESH51] EnergyShield Consortium (2020) D5.1 Integration and test plan

[ESH52] EnergyShield Consortium (2020) D5.2 Common software platform release, incl. user and developer documentation

[ESH53] EnergyShield Consortium (2021) D5.3 System release v1

[ESH54] EnergyShield Consortium (2021) D5.4 System release v2

[ESH55] EnergyShield Consortium (2021) D5.5 System release v3

[FDD19] Agile Modeling (2019), http://agilemodeling.com/essays/fdd.htm, accessed in March 2019

[FEL14] Felderer, M.; Ramler, R. Integrating risk-based testing in industrial test processes. Softw. Qual. J. 2014, 22, 543–575.

[GEA20] Georgiadou, Anna, Spiros Mouzakitis and Dimitris Askounis. “Designing a Cyber-security Culture Assessment Survey Targeting Critical Infrastructures During Covid-19 Crisis.” ArXiv abs/2102.03000 (2021): n. pag.

[GED20] Georgiadou, A., Mouzakitis, S. & Askounis, D. Working from home during COVID-19 crisis: a cyber security culture assessment survey. Security Journal (2021). https://doi.org/10.1057/s41284-021-00286-2

[GES20] Georgiadou, Anna, Spiros Mouzakitis and Dimitrios Askounis. “Towards Assessing Critical Infrastructures Cyber-Security Culture During Covid-19 Crisis: A Tailor-Made Survey.” ArXiv abs/2012.13718 (2020): n. pag.

[IEE16] IEEE 29119-5-2016 - ISO/IEC/IEEE International Standard - Software and systems engineering - Software testing -- Part 5: Keyword-Driven Testing , https://standards.ieee.org/standard/29119-5-2016.html,

[HAH13] Hahn, A.; Ashok, A.; Sridhar, S.; Govindarasu, M. Cyber-Physical Security Testbeds: Architecture, Application, and Evaluation for Smart Grid. IEEE Trans. Smart Grid 2013, 4, 847–855

[KAR09] Karnouskos, S.; Holanda, T.N.D. Simulation of a Smart Grid City with Software Agents. In Proceedings of the 2009 Third UKSim European Symposium on Computer Modeling and Simulation, Athens, Greece, 25–27 November 2009; pp. 424–429.

[KHA15] Khaitan, S.K.; McCalley, J.D. Design techniques and applications of cyberphysical systems: A survey. IEEE Syst. J. 2015, 9, 350–365.

[KOK11] Kok, K.; Karnouskos, S.; Ringelstein, J.; Dimeas, A.;Weidlich, A.;Warmer, C.; Drenkard, S.; Hatziargyriou, N.; Lioliou, V. Field-testing smart houses for a smart grid. In Proceedings of the 21st International Conference and Exhibition on Electricity Distribution (CIRED 2011), Frankfurt, Germany, 6–9 June 2011.

[PAL14] Palensky, P.; Widl, E.; Elsheikh, A. Simulating cyber-physical energy systems: Challenges, tools and methods. IEEE Trans. Syst. Man Cybern. Syst. 2014, 44, 318–326.

[SCH18] Martin Schvarcbacher, Katarína Hrabovská, Bruno Rossi & Tomáš Pitner (2018) Smart Grid Testing Management Platform (SGTMP) Appl. Sci. 2018, 8, 2278, DOI: 10.3390/app8112278

[SPH19] SPHINX Project EU. SPHINX Project EU. SPHINX.,1 January 2019, Available online: https://sphinx-project.eu/. (Accessed on 19 June 2021).

[WAN10] Wang, Z.; Scaglione, A.; Thomas, R.J. Generating Statistically Correct Random Topologies for Testing Smart Grid Communication and Control Networks. IEEE Trans. Smart Grid 2010, 1, 28–39.

DEVELOPING THE CYBER-TOOLKIT THAT PROTECTS YOUR ENERGY GRID

www.energy-shield.eu