



European
Commission

Horizon 2020
European Union funding
for Research & Innovation

Cyber Security PPP: Addressing Advanced Cyber Security Threats and Threat Actors



Cyber Security Threats and Threat Actors Training - Assurance Driven Multi- Layer, end-to-end Simulation and Training

D4.1: THREAT-ARREST Visualisation Tools v1[†]

Abstract: This deliverable reports on the status of developing the visualization tools of the THREAT-ARREST training platform. The visualization tools enable the trainees to have a clear view of the cyber system status and attacks upon it, which will be updated in real-time. It will also provide information about their assessment status and enable real-time interaction. We provide an overview on the different tools, their integration into the THREAT-ARREST platform and their current development status.

Contractual Date of Delivery	31/08/2019
Actual Date of Delivery	31/08/2019
Deliverable Security Class	Public
Editor	<i>Torsten Hildebrandt (SimPlan)</i>
Contributors	<i>Torsten Hildebrandt (SimPlan), George Bravos (ITML), Ludger Goeke (SEA), Hristo Koshutanski (ATOS), Fulvio Frati (UMIL)</i>
Quality Assurance	<i>George Hatzivasilis, Othonas Soultatos (FORTH), George Leftheriotis (TUV)</i>

[†] The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 786890.

The *THREAT-ARREST* Consortium

Foundation for Research and Technology – Hellas (FORTH)	Greece
SIMPLAN AG (SIMPLAN)	Germany
Sphynx Technology Solutions (STS)	Switzerland
Universita Degli Studi di Milano (UMIL)	Italy
ATOS Spain S.A. (ATOS)	Spain
IBM Israel – Science and Technology LTD (IBM)	Israel
Social Engineering Academy GMBH (SEA)	Germany
Information Technology for Market Leadership (ITML)	Greece
Bird & Bird LLP (B&B)	United Kingdom
Technische Universitaet Braunschweig (TUBS)	Germany
CZ.NIC, ZSPO (CZNIC)	Czech Republic
DANAOS Shipping Company LTD (DANAOS)	Cyprus
TUV HELLAS TUV NORD (TUV)	Greece
LIGHTSOURCE LAB LTD (LSE)	Ireland
Agenzia Regionale Strategica per la Salute ed il Sociale (ARESS)	Italy

Document Revisions & Quality Assurance

Internal Reviewers

1. *George Hatzivasilis and Othonas Soultatos (FORTH)*
2. *George Leftheriotis (TUV)*

Revisions

Version	Date	By	Overview
0.7, 0.8	27/08/2019	Editor	Addressing reviewer comments
0.6	19/08/2019	Fulvio Frati, UMIL	Added contents of Section “Trainee Access to Emulated Components”
0.5	03/08/2019	Hristo Koshutanski, ATOS	Added contents to Section “Overall GUI Approach”
0.4	31/07/2019	Torsten Hildebrandt, SimPlan	Added contents of Section “Visualizing Simulation and Emulation State”
0.3	26/07/2019	George Bravos, ITML	Contents of Section “Trainee Performance Assessment Visualisation” added
0.2	24/06/2019	Ludger Goeke, SEA	Added Section “GUI Concept for Gamification Tool”
0.1	01/06/2019	Editor	First Draft

Executive Summary

This deliverable is the first one reporting progress on task T4.1 of the THREAT-ARREST project. It reports on the work done from its start in month 4 until month 12 and also sketches some of the work that will be performed in the months ahead.

The visualization tools enable the trainees to have a clear view of the cyber system status and the attacks upon it, which will be updated in real-time. It also provides information about their assessment status and enable real-time interaction. In this deliverable we provide an overview on the various tools, their integration into the THREAT-ARREST platform and their current development status.

This deliverable also describes a first sketch of the dashboard provided by the Training Tool (see work package WP3). The dashboard serves as the central starting point for users of the THREAT-ARREST platform to access the various platform components as required by the overall workflow of a CTTTP training session. It integrates the UI of the Gamification Tool (see also task T4.2), the visualization offered by the Jasima Visualization Tool (used to show the current state of simulated -see work package WP5- and emulated -see work package WP2- cyber-system components of a training session; developed as part of task T4.1), as well as the UI offered by the Guacamole software to login to a virtual machine of the Emulation Tool and interact with it.

All of these components will be accessible using web-based technologies to ensure a good interactive user-experience. All of this functionality is achieved using web-based technologies, requiring only a web browser to be used by trainees.

Table of Contents

1	INTRODUCTION	9
2	OVERALL GUI APPROACH	10
3	TRAINEE PERFORMANCE ASSESSMENT VISUALISATION	12
3.1	TRAINEE PERFORMANCE ASSESSMENT UI: REQUIREMENTS	12
3.2	TRAINEE PERFORMANCE ASSESSMENT VISUALISATION: COMMUNICATION WITH OTHER COMPONENTS ..	13
3.3	TRAINEE PERFORMANCE ASSESSMENT VISUALISATION: WIREFRAMES	16
4	GUI CONCEPT FOR THE GAMIFICATION TOOL.....	24
4.1	GUI CONCEPT FOR PROTECT	24
4.2	GUI CONCEPT FOR AWARENESS QUEST	26
5	VISUALIZING SIMULATION AND EMULATION STATE.....	28
5.1	THE NEW JASIMA VISUALIZATION TOOL (JVT)	28
5.2	OVERVIEW AND ARCHITECTURE	28
5.2.1	<i>Communication Layer</i>	29
5.2.2	<i>Data Binding Layer</i>	29
5.2.3	<i>Presentation Layer</i>	30
5.3	SETTING UP A VISUALIZATION SCENARIO	30
5.4	EXAMPLE VISUALIZATION CONNECTING TO SIMULATION COMPONENTS	30
5.4.1	<i>Scenario and Component Definition</i>	31
5.4.2	<i>The Visualisation Scenario in Use</i>	33
6	TRAINEE ACCESS TO EMULATED COMPONENTS	38
7	CONCLUSIONS.....	40
	REFERENCES.....	41

List of Abbreviations

CTTP Cyber Training and Threat Preparation

GUI Graphical User Interface

IDE Integrated Development Environment

IoT Internet of Things

JVT Jasima Visualization Tool

PC Personal Computer

SSH Secure Shell

STOMP Simple (or Streaming) Text Oriented Messaging Protocol

SVG Scalable Vector Graphics

SWT Standard Widget Toolkit

UI User Interface

WP Work Package

List of Figures

Figure 1: THREAT-ARREST Platform Components Communications – Dashboard View ..	11
Figure 2: THREAT-ARREST sequence diagram towards trainees' assessment	15
Figure 3: THREAT-ARREST dashboard login screen	17
Figure 4: THREAT-ARREST dashboard users' summary for administrators	18
Figure 5: THREAT-ARREST dashboard: adding a user	19
Figure 6: THREAT-ARREST dashboard: trainees' list for trainers	20
Figure 7: THREAT-ARREST dashboard: trainees' status page for trainers	21
Figure 8: THREAT-ARREST dashboard: scenarios' list for trainers.....	22
Figure 9: THREAT-ARREST dashboard: personal dashboard for trainees	23
Figure 10: GUI of PROTECT at the start of the game.....	25
Figure 11: GUI of PROTECT with card on the player's hand.....	25
Figure 12: Dialog after an attack card has been drawn	26
Figure 13: Dialog for continuing the game after it has been paused.....	26
Figure 14: JVT layer architecture.....	29
Figure 15: Example Project Setup in Visual Studio Code	31
Figure 16: Scenario definition file	31
Figure 17: Example Visualization View	32
Figure 18: User Interface for Simulation Control and Scenario Visualization	33
Figure 19: The Scenario Visualization View	34
Figure 20: The Bridge Visualization View	35
Figure 21: The View "Ship's Position"	36
Figure 22: The Visualization View "2D charts", normal browser size	37
Figure 23: Visualization View "2D charts", small browser size.....	37
Figure 24: Emulation Tool REST response	38
Figure 25: Apache Guacamole login form.....	39
Figure 26: Example of SSH connection through Apache Guacamole	39

List of Tables

Table 1: Visualisation of profile and assessment information for Trainees.....	12
Table 2: Visualisation of profile and assessment information for Trainers	13

1 Introduction

The present deliverable documents and presents a first version of all THREAT-ARREST components and tools that have a major user interface (UI) visible to trainees or trainers during a training session:

- The central dashboard offered by the Training Tool
- The Gamification Tool
- The Jasima Visualization Tool
- The UI offered by the Emulation Tool.

The main goal is to design and develop a user-friendly visualization environment for the THREAT-ARREST training platform. The visualization tools will enable the trainees to have a clear view of the cyber system status and the attacks upon it, which will be updated in real-time. It will also provide information about their assessment status and enable real-time interaction (e.g. (Manifavas et al. 2014; Fysarakis et al., 2015)).

This deliverable includes the “Trainee Performance Assessment” section describing the UI of the Training Tool, which also offers a central dashboard to start and administrate a training session (see Section 3). Being the central dashboard, it will also serve as the central component for a user to access the other visualization tools of the THEAT-ARREST platform by either embedding or linking to it.

Section 4 provides a brief description of the user interface offered by the Gamification Tool. Further details on the Gamification Tool can also be found in the deliverable D4.2.

In Section 5, the visualisation tool of the Jasima simulator (the main module of the Simulation Tool detailed in D5.2) is described. It offers a flexible mechanism to show the state of the cyber system while a Cyber Training and Threat Preparation (CTTP) session is executed. It allows defining visual components for each user role and linking these components to real-time information from the simulated or emulated components of the THREAT-ARREST platform. The Jasima Visualization Tool (JVT) can be linked to other components using a flexible approach utilizing a message-oriented broker.

Section 6 briefly describes the UI that can be used to log in to virtual machines executed by the Emulation Tool (offering Secure Shell (SSH) or Remote Desktop). For details of the Emulation Tool readers are referred to deliverables D2.1 and D2.3.

Finally, a short conclusion is given in Section 7.

2 Overall GUI Approach

The THREAT-ARREST platform will be accessed via a web interface. The platform will collect information from the underlying tools internally and will present them in a unified and user-friendly manner to the user. This section presents the THREAT-ARREST architectural approach to provide an integral Graphical User Interface (GUI) experience. It provides an overall view of the different components offering a GUI to the various functional aspects of the platform and their communications.

The Training Tool's dashboard is the central component to access the platform, offering cybersecurity training functionality to both trainees and trainers through a GUI tailored to the project needs. The dashboard's GUI follows a co-design approach which involves both software engineers (technical partners in the project) and end-users of the platform (use case partners) in order to better address end-user experience and interactions with the platform for cybersecurity training.

An important aspect of the GUI co-design approach is the proper integration of the different components so that a unified layout and experience during training is offered. From the platform standpoint, such integration will make the platform dataflow and interactions with trainees and trainers more focused and central to the Training Tool functionality.

In this way, the dashboard will become a user's entry point to interact with the other platform components such as the Emulation Tool's Guacamole¹ Remote Desktop Gateway (for hands-on training, see D2.1-D2.3), the Gamification Tool's serious games for cybersecurity training (namely for social engineering training, see D4.2), the Visualisation Tool's JavaScript engine for progressive in-browser visualisation of the state of cyber system components (simulated/emulated).

During the second year of the project, the dashboard will be further developed to also address integration with other emerging components or functionality of the platform, such as the CTPP model editor's GUI.

¹ <https://guacamole.apache.org>

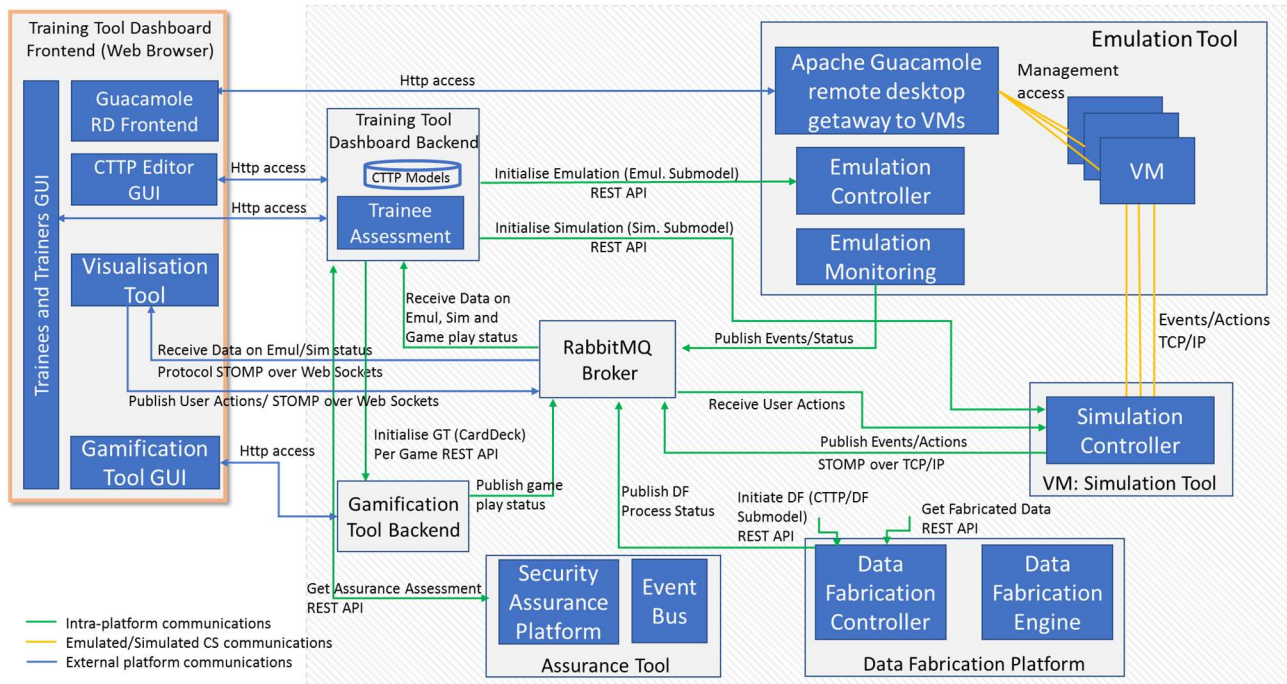


Figure 1: THREAT-ARREST Platform Components Communications – Dashboard View

Figure 1 shows the THREAT-ARREST Platform Components Communications with a particular emphasis on the various components that form part of the dashboard as well as their backend interactions. Details of the various components' interconnections and dataflow can be found in deliverables “D2.4 – Emulation tool interoperability module v1”, “D4.3 – Training and Visualisation tools IO mechanisms v1”, and “D5.3 – The Simulation component IO module v1”. There is a certain level of complexity in the realisation of the overall GUI concept, as it requires proper integration of the various individual GUIs, while handling their interactions with different backends. For instance, the Jasima Visualisation Tool's JavaScript engine requires the use of the Simple (or Streaming) Text Oriented Messaging Protocol (STOMP)² over the WebSockets protocol to receive event/state information from the THREAT-ARREST platform's message broker.

In the rest of the document, the GUI concepts of the above mentioned tools are presented, while the overall GUI and layout integration of the various components will be subject of the second-year's activities of work package 4 in general and task T4.1 in particular.

² STOMP: <http://stomp.github.io/>

3 Trainee Performance Assessment Visualisation

3.1 Trainee performance assessment UI: Requirements

The trainee performance assessment will be visualised in the THREAT-ARREST dashboard, based on specific requirements collected and aggregated from the THREAT-ARREST Pilots' end-users. The visualisation of that information comprises two main different fields: the screens for the trainees and the screens for the trainers.

Table 1 summarizes the information that will be available to the trainees. Each trainee will have access to his/her personal account profile information and status in terms of the training scenarios (available and completed) and relevant scores. With respect to the trainees' profiles, pilot-specific fields will also be used.

Table 1: Visualisation of profile and assessment information for Trainees

	Basic information	Health pilot	Energy pilot	Maritime pilot
Trainee's status	Available scenarios (including brief details about related pilot components, e.g. smart home router configuration, Internet of Things (IoT) devices installation, backend system administration, etc.)			
	Documentation and Help			
	Incomplete scenarios			
	Campaign scenarios (trainee can choose to complete subsequent scenarios as a whole campaign)			
	Score (and possibly prizes) per scenario completed			
	Total score			
Trainee's account profile	Personal details (name, expertise, email, years of experience)	Pilot-specific role, which can be: •Coder (a doctor or a biologist); •Technical staff	Pilot-specific role, which can be: •Homeowner, •Technician, •Administrator	Pilot-specific details: • Rank, • Previous Assignment to Vessel (name, type, vessel basic specs), • certificates
	Position in the organization			Pilot-specific additional detail: Department (for ashore personnel)
Pilot – specific screens			Illustration of the scenario scope on the pilot system architecture	

Table 2, on the other hand, summarizes the information that will be available to the trainers in each organization. Each trainer will be able to get an overview of all trainees in the organization (or his/her department); in the health pilots the trainer will also be able to get information about the score of the trainees per role, while in the maritime pilots per department and vessel. On top

of that, the trainers will also have detailed information about the trainees' performance and status per scenario and per trainee.

Table 2: Visualisation of profile and assessment information for Trainers

	Basic information	Health pilot	Energy pilot	Maritime pilot
Overview of trainees	Number of trainees, names of trainees, overall scores (based on trainee type)	Pilot-specific details: Score of trainees per role		Pilot-specific details: Score of trainees per department or per vessel
	Overview of trainees' ongoing activities (ongoing scenario real-time reporting)			
Trainee's profile (per trainee)	All information available in trainee's screen (available scenarios, incomplete scenarios (including % of progress), score per scenario completed, total score, account profile), and information per trainees' roles (different for every pilot)			
	Ongoing activities' performance			
	Feedback			
Overview of scenarios	Scenarios played, scenarios remaining, scenarios per level of difficulty			
Scenario's details (per scenario)	Assessment results, times executed, average scores, difficulty level, trainees that executed the scenario			

3.2 Trainee performance assessment Visualisation: Communication with other components

In order for the trainee and the trainer to obtain access to the respective profiles (and assessment outcomes), a detailed analysis has been carried out with respect to the sequence of the actions / messages to be exchanged between the various components of the THREAT-ARREST platform. The outcome of this analysis has been depicted in a detailed sequence diagram (see Figure 2) , according to which the THREAT-ARREST dashboard, being the main component of the THREAT-ARREST Training Tool, will be responsible for initiating all the relevant components for each training session and consequently aggregate all the information regarding the profiles and the assessment of the players/trainees.

In more detail, the process is initiated by the trainee, that requests information about the scenarios available for him/her, through the dashboard. The dashboard requests that information from the CTPP modeler, and provides the list to the trainee. The trainee chooses one scenario to be initiated, and the Training Tool initiates (if needed) the Gamification Tool, the Simulation and, last, the Emulation Tool. Based on these initializations, the Training Tool then initiates the visualisation component for the trainee to proceed with the scenario. Regarding assessment, the trainee initiates the process by requesting his/her current score for a specific scenario; the Assessment Tool initiates the necessary communications with the Gamification, Simulation and Emulation Tools to aggregate information regarding the trainee's status, and provides the necessary information via the dashboard.

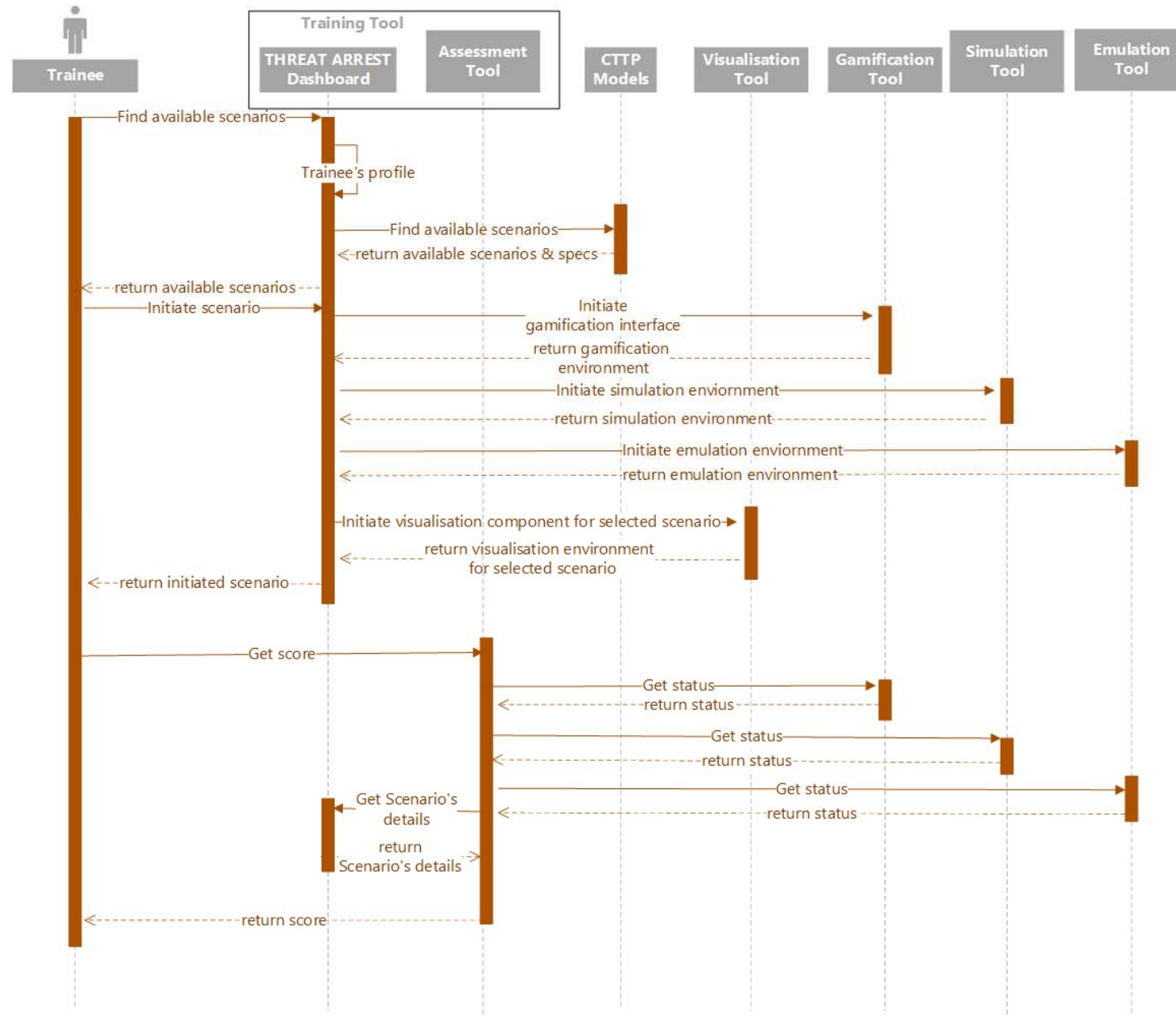


Figure 2: THREAT-ARREST sequence diagram towards trainees' assessment

3.3 Trainee performance assessment Visualisation: Wireframes

The visualisation of the overall trainee performance assessment and the training monitoring will be based on the THREAT-ARREST dashboard.

The administrator will login through a basic interface (Figure 3 – Login) and will be directed in a page providing an overview of the users already present in the system, with their names and roles (Figure 4 – Users’ Summary).

The administrator will be able to add a user and his/her details such as, the name, company, department, position in company, expertise, years of experience, username etc. (Figure 5 – Adding a user).

A trainees list will also be available, including information about the role of each trainee, his/her overall score, and the status of the scenarios that he/she has already completed (Figure 6 – Trainees’ list).



Additional information will be available for each trainee; the trainer/administrator will have access to details related to the company, the expertise, the role, the total score, the scenarios completed, the rank of the trainee within the company based on his/her score, as well as detailed information about the scores in each training scenario (competition status, times played, top score, average playing time etc.) (Figure 7 – Trainees’ status page).

Apart from the information related to each trainee, the administrator will have access to aggregated details related to the scenarios that are available: assessment results, times of execution, average score, difficulty level, number of trainees engaged etc. On top of that, an overview of the scenarios that are currently ongoing by trainees will also be available (Figure 8 – Scenarios’ list).

Finally, the trainee will have access to his/her personal dashboard, providing information about his/her profile as well as the scenarios enabled for him/her and the current status for each one of them: completion status, times played, top score, average playing time and total time played (Figure 9 – Personal dashboard).

The overview of the pages that have been currently designed is presented in the following table.

Page	Roles
Login	All
Users’ Summary	Administrator
Adding a User	Administrator
Trainees’ list	Trainers
Trainees’ status page	Trainers
Scenarios’ list	Trainers
Personal dashboard	Trainees


ThreatArrest Dashboard

Username


Password

Login

Figure 3: THREAT-ARREST dashboard login screen




ThreatArrest Dashboard




Users

Name	Username	Role	Enabled		
Giacomo Guilizzoni	giaco	Trainer	*	Edit	Delete
Marco Botton	marco	Admin	✓	Edit	Delete
Mariah Maclachlan	mariah	Trainee	⊞	Edit	Delete
Valerie Liberty	val	Trainee	✓	Edit	Delete
Giacomo Guilizzoni	giaco	Trainer	*	Edit	Delete
Marco Botton	marco	Trainer	✓	Edit	Delete
Mariah Maclachlan	mariah	Trainee	⊞	Edit	Delete
Valerie Liberty	val	Trainee	✓	Edit	Delete
Giacomo Guilizzoni	giaco	Trainer	*	Edit	Delete
Marco Botton	marco	Trainer	✓	Edit	Delete
Mariah Maclachlan	mariah	Trainee	⊞	Edit	Delete
Valerie Liberty	val	Trainee	✓	Edit	Delete
Giacomo Guilizzoni	giaco	Trainer	*	Edit	Delete
Marco Botton	marco	Trainer	✓	Edit	Delete
Mariah Maclachlan	mariah	Trainee	⊞	Edit	Delete
Valerie Liberty	val	Trainee	✓	Edit	Delete
Giacomo Guilizzoni	giaco	Trainer	*	Edit	Delete
Marco Botton	marco	Trainer	✓	Edit	Delete


 Add a User

Page 1 of 10

Figure 4: THREAT-ARREST dashboard users' summary for administrators



ThreatArrest Dashboard



Users

User Addition

Name

Surname

Company

Department

Position in company

Expertise

Years of experience

Username

User Type

Trainee

Admin

Trainee

Trainer

Scenario Role

Homeowner

Administrator

Homeowner

Technician

Scenarios

☐ Energy 01

☒ Energy 02

☒ Energy 03

..

...

☐ Energy 10

Save User




Figure 5: THREAT-ARREST dashboard: adding a user

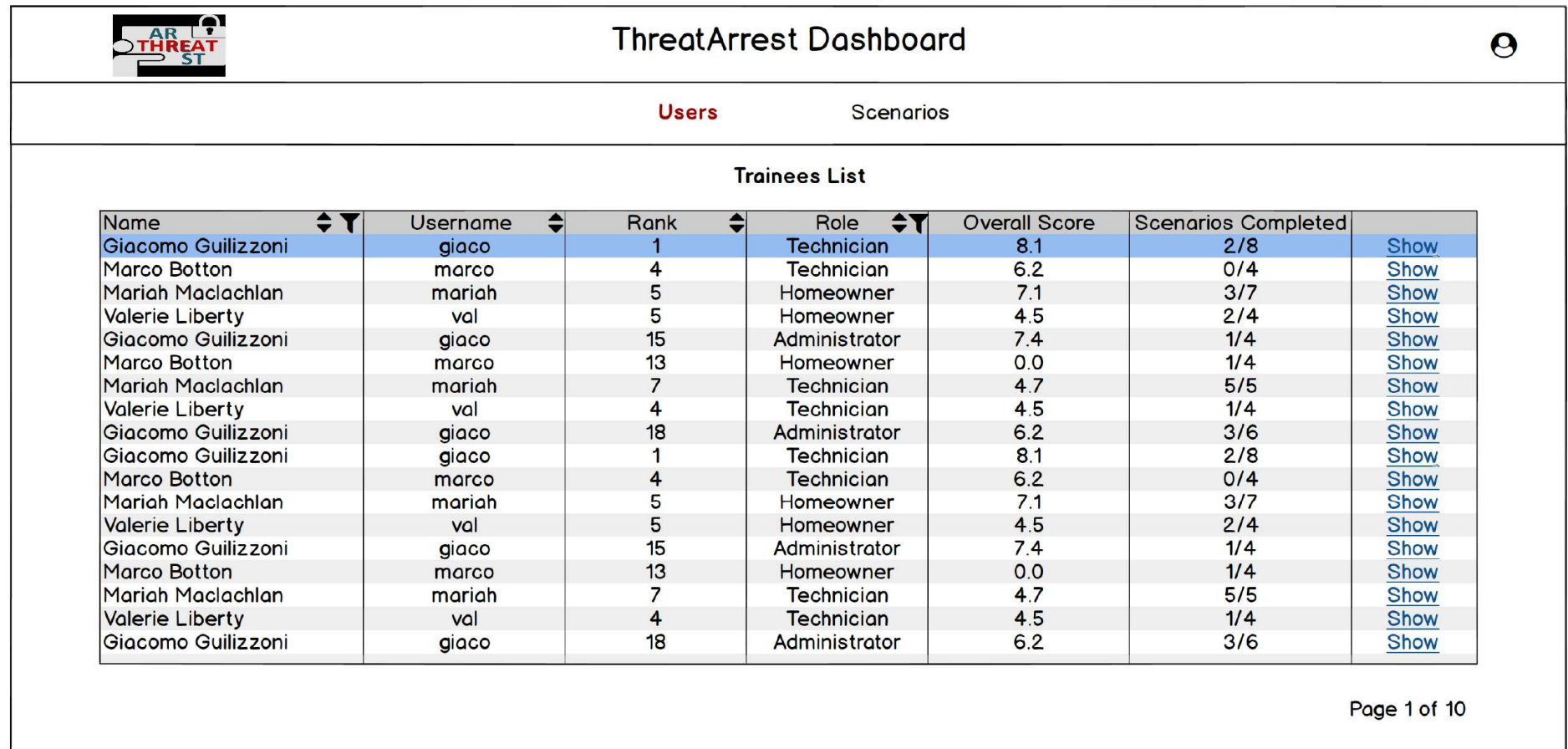


Figure 6: THREAT-ARREST dashboard: trainees' list for trainers

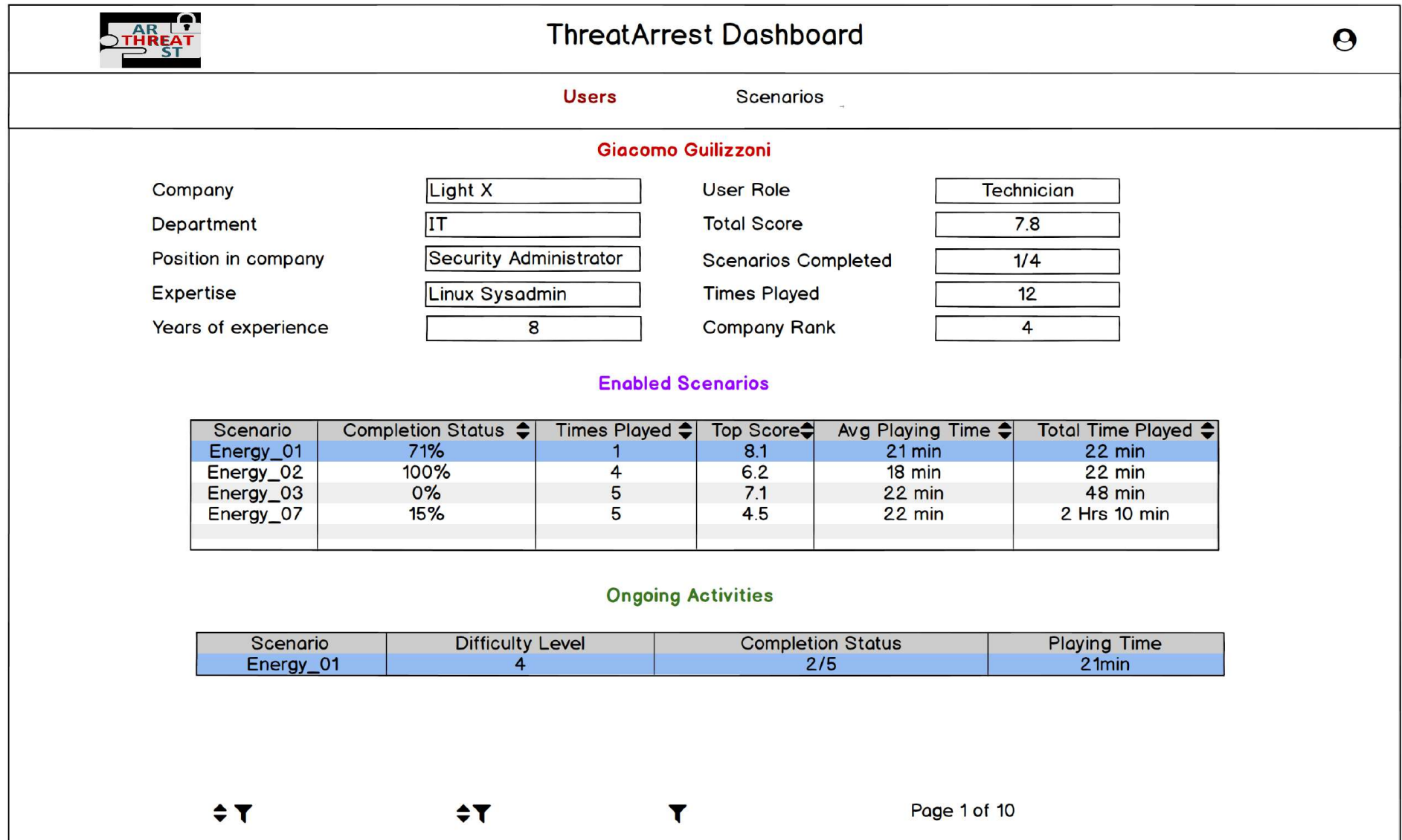


Figure 7: THREAT-ARREST dashboard: trainees' status page for trainers

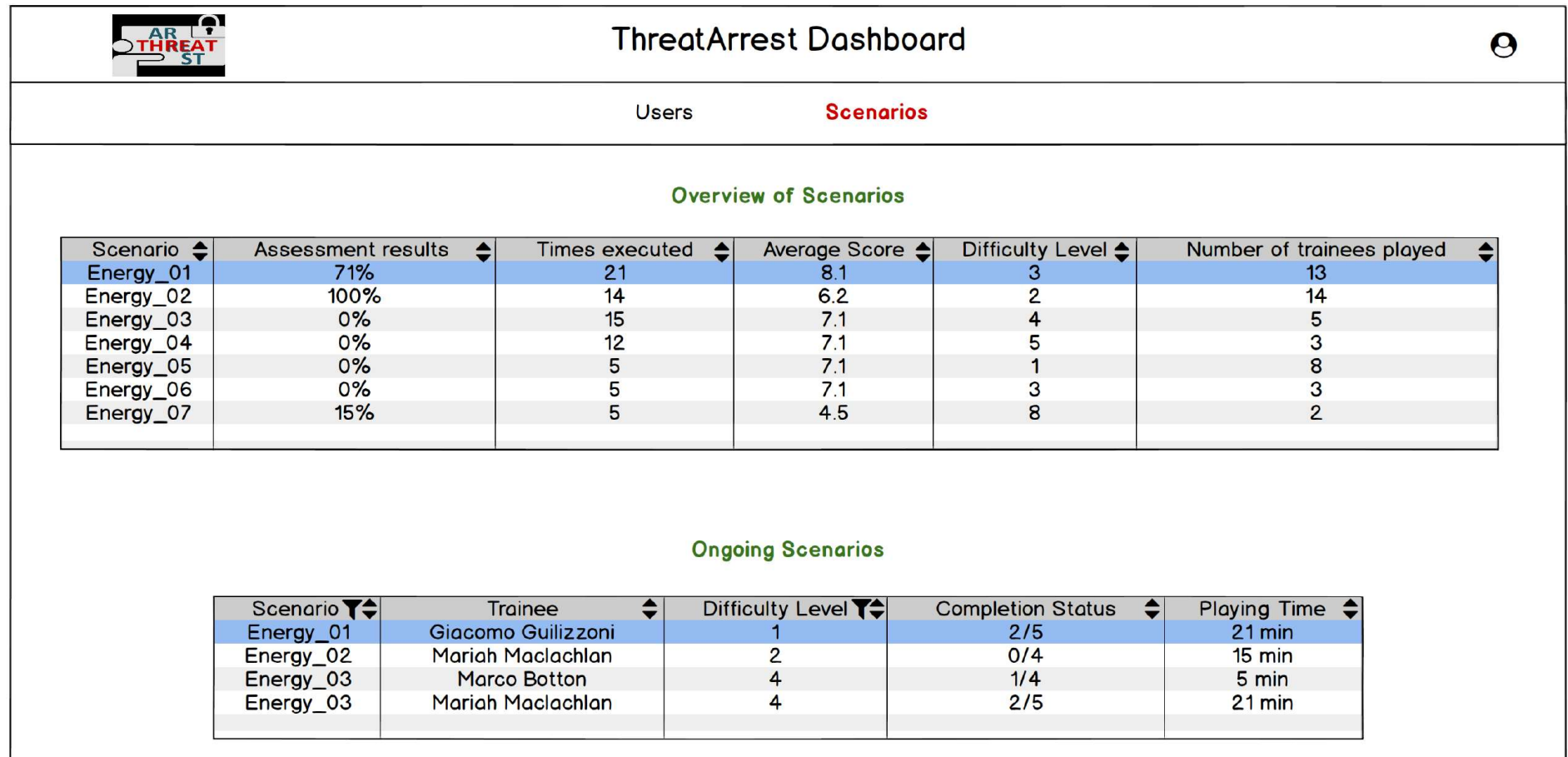


Figure 8: THREAT-ARREST dashboard: scenarios' list for trainers

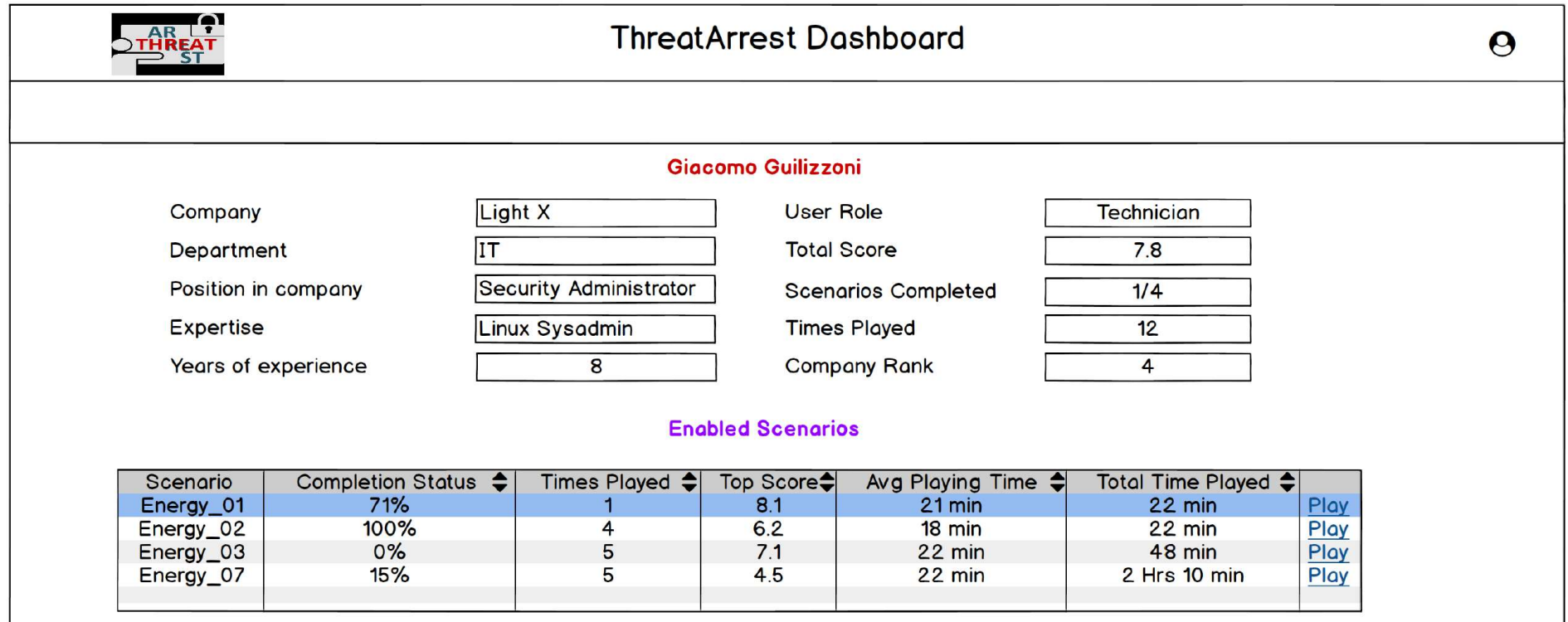


Figure 9: THREAT-ARREST dashboard: personal dashboard for trainees

4 GUI Concept for the Gamification Tool

The Gamification Tool is a part of the THREAT-ARREST training platform and it will provide the serious games AWARENESS QUEST and PROTECT³ for training users in subjects of social engineering (e.g. (Ferrera et al., 2018)). These serious games, also called gaming tools, will be provided in the form of online games. The following subsections describe the graphical user interfaces of the gaming tools.

4.1 GUI Concept for PROTECT

PROTECT is a serious online game for sensitizing people against social engineering attacks. The main game concept is that players are confronted with social engineering attacks. The task of the players is to prevent the success of an attack by selecting the appropriate defense behavior. PROTECT is implemented as a single player card game that realizes a patience game approach. A card deck in PROTECT includes cards that represent attacks and appropriate defences. Additionally, it contains special action cards (e.g. Joker cards). For a detailed description of the game concepts of PROTECT reference is made to D4.2 (Beckers, Bravos, Goeke, & Pape, 2019) in subsection 2.3.3.

The GUI of PROTECT is executed in a web browser. Currently, the tool is primary tested in the *Mozilla Firefox*⁴. It is planned that PROTECT will be compatible to further web browsers like for example *Google Chrome*⁵. PROTECT is implemented in JavaScript. In this context, the JavaScript library *jQuery*⁶ and the framework *Bootstrap*⁷ are used. The PROTECT GUI is especially designed to be displayed on mobile devices. Nonetheless, it can be displayed on personal computer (PC) monitors and laptop screens without any problems.

A user starts a game of PROTECT from the dashboard. When the user has finished or paused a game he/she returns to the dashboard. Figure 10 shows the execution of the PROTECT GUI in a web browser after the instantiation of the game. At the start of the game, a dialog for changing the language of the game is displayed. The card deck, including the attack, defence, and special action cards, is positioned in the top right corner. It is shuffled automatically before each round of the game. A player can draw a card by double-clicking on the card deck. The game score and the remaining game time are represented in the bottom right corner. It is also possible to pause a game with the help of the *Pause*-button in the bottom left corner of the GUI. A game can also be cancelled and restarted. The corresponding *Restart*-button is positioned next to the *Pause*-button. The remaining lives of a player during a game are displayed by the pink heart symbols. Every time a player selects an incorrect defence card for an attack card, he / she loses a life. When a player has lost all his / her lives, the game is over.

³ <https://www.social-engineering.academy/en/>

⁴ <https://www.mozilla.org/en-US/firefox/>

⁵ <https://www.google.com/intl/en/chrome/>

⁶ <https://jquery.com/>

⁷ <https://getbootstrap.com/docs/3.4/javascript/>

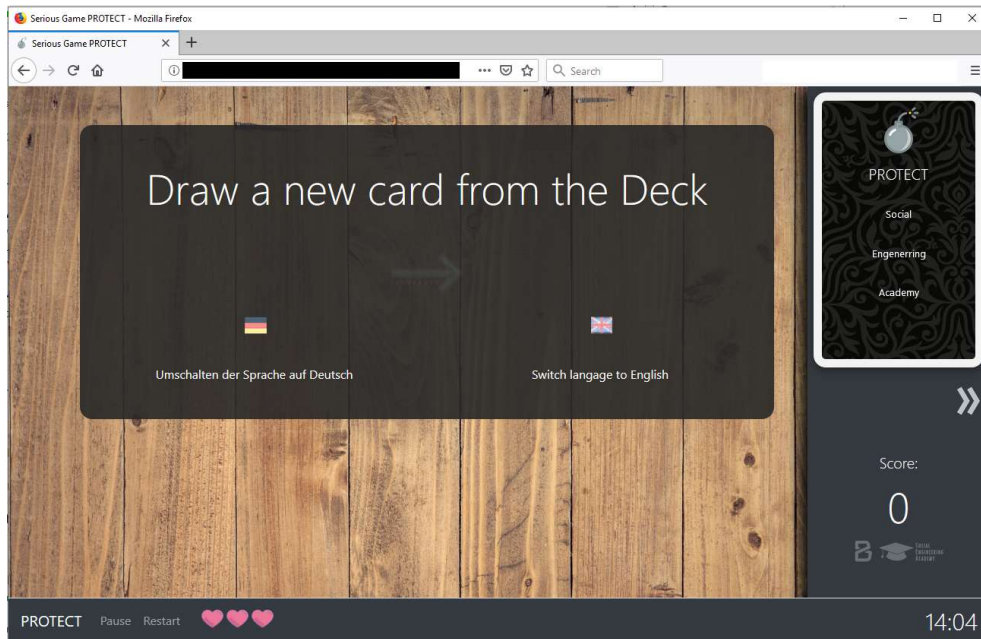


Figure 10: GUI of PROTECT at the start of the game

Figure 11 shows defence cards that a player has drawn on his/her hand. The hand of a player is represented by the table.

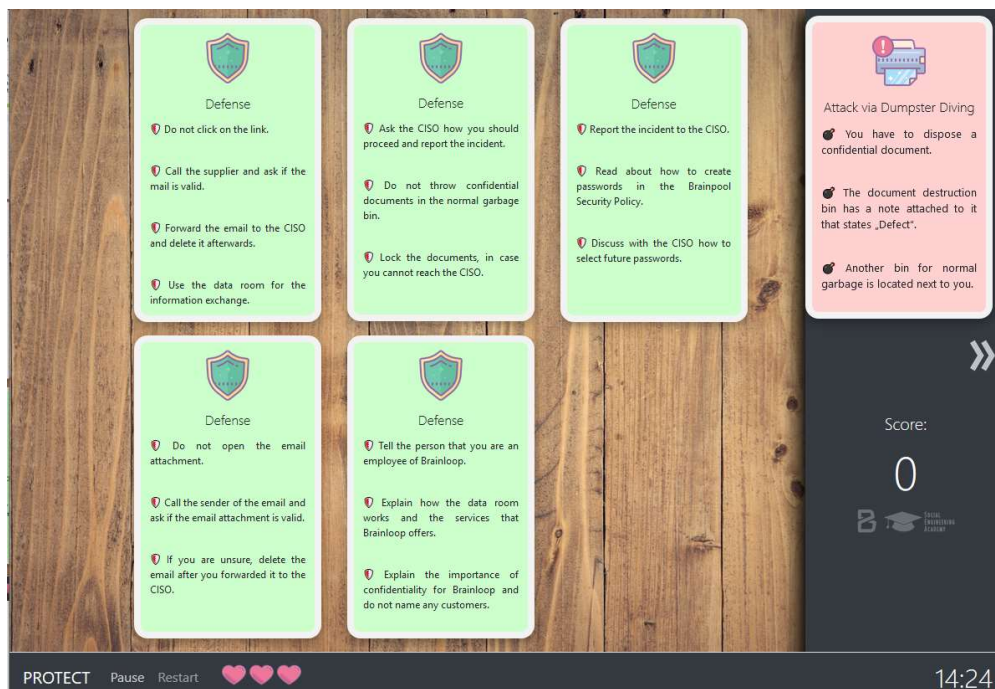


Figure 11: GUI of PROTECT with card on the player's hand

The GUI supports the player in the game flow with appropriate dialogs. For example, the dialog in Figure 12 is shown after the player has drawn an attack card. It requests the player to select a defense card after clicking on the *Select defense*-button.

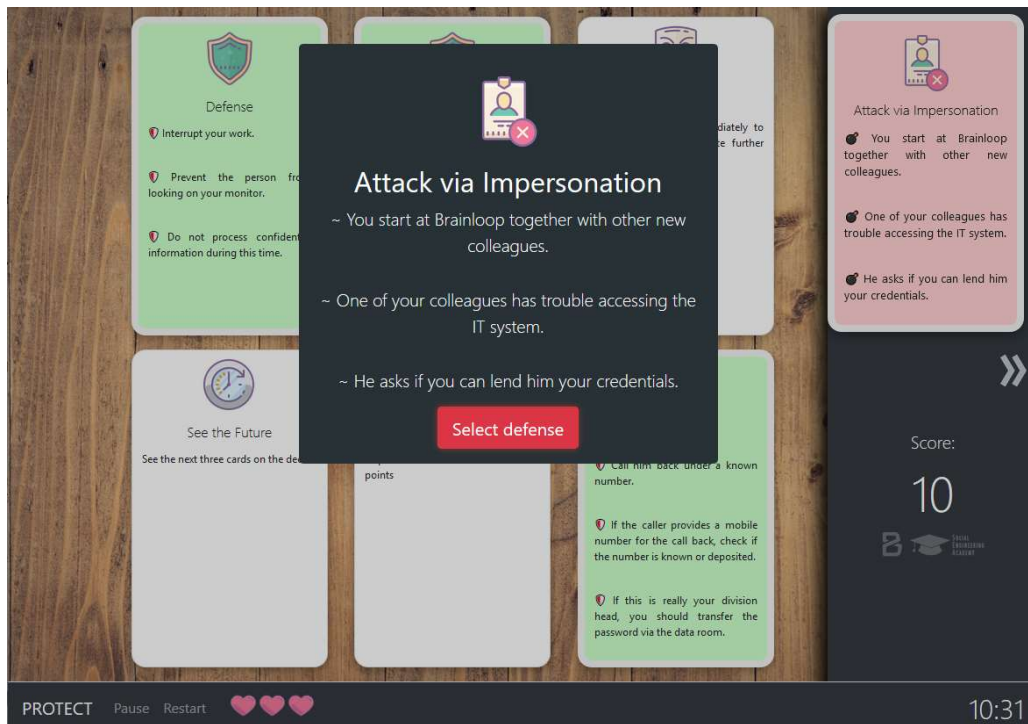


Figure 12: Dialog after an attack card has been drawn

In another example, the dialog in Figure 13 points out that the player can continue the paused game by clicking on the *Start*-button in the dialog.

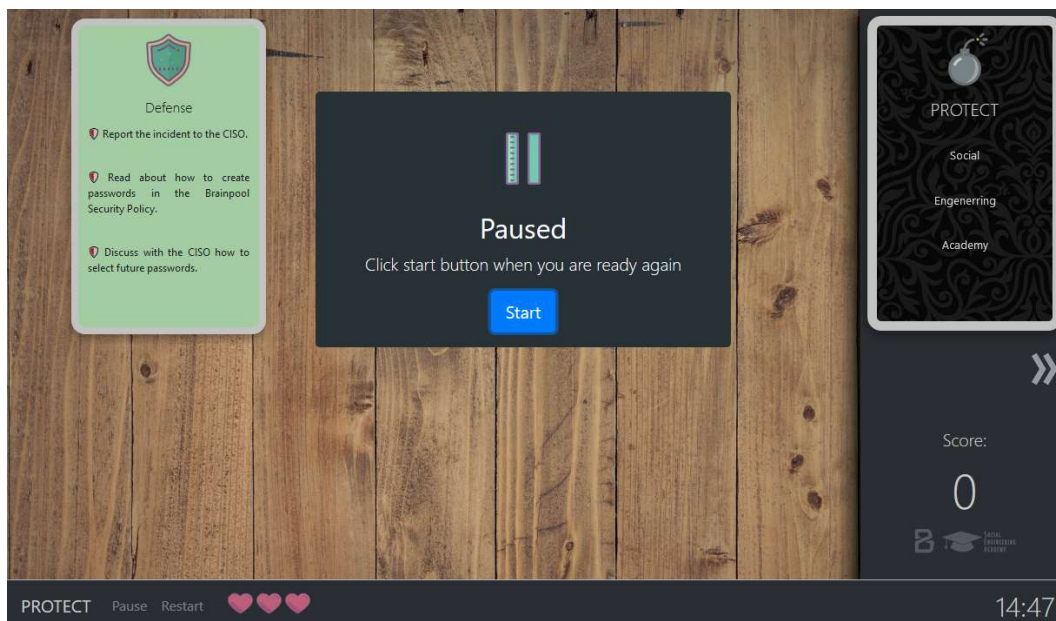


Figure 13: Dialog for continuing the game after it has been paused

4.2 GUI Concept for AWARENESS QUEST

AWARENESS QUEST will implement quizzes to assess the awareness of a company's employees regarding social engineering. Each quiz will consider a different social engineering scenario.

Since AWARENESS QUEST is currently in the conceptual phase (see (Beckers, Bravos, Goeke, & Pape, 2019), Section 2.2) only general statements about the GUI of AWARENESS QUEST can be made at the time this document is written.

The GUI of AWARENESS QUEST will also be browser-based and especially designed for mobile devices. More details will be provided in the next version of the visualization modules in D4.8 (due to M30).

5 Visualizing Simulation and Emulation State

5.1 The new Jasima Visualization Tool (JVT)

Jasima is a Java-based software library for discrete-event simulation, developed by SimPlan. In WP5, it is used to create simulated components of a cyber-system, like network components, IoT devices, or even attackers (e.g. (Hatzivasilis et al., 2019a; Hatzivasilis et al., 2019b; Hatzivasilis et al., 2017)). To allow a trainee to have a clear view of the cyber system covered by a training session (comprised of simulated and emulated components), a flexible and user-friendly way for the presentation is required. To achieve this, a completely new visualization tool/framework was developed largely depending on state-of-the-art Web technologies, as the current GUI components of Jasima, which are based on old UI technology such as the Java Standard Widget Toolkit (SWT)⁸ framework, turned out not to be flexible or future-proof enough.

5.2 Overview and Architecture

The Jasima Visualization Tool (JVT) is a visualization library written in JavaScript. It is utilizing state-of-the-art web technologies (like HTML5⁹, HTML Custom Elements¹⁰, and CSS3¹¹) to create flexible, easy-to-use visual representations showing the state of simulated components. The approach is flexible enough to also connect to other data sources. In THREAT-ARREST, this is required to show the state of emulated components in a unified environment together with information from simulated components.

A visualization scenario defined for JVT runs solely in a trainee's web browser as a Progressive Web App. It can either run standalone in a separate browser tab or window or be integrated directly in the dashboard on a dedicated tab using IFRAMEs or by directly including JVT's JavaScript library and the HTML Custom Elements it provides in the HTML code of the dashboard. When the scenario is completed, the user returns to his/her main page in the dashboard.

Its overall architecture is summarized in Figure 14. As can be seen in that figure, JVT's functionality can be grouped in three different areas that are described bottom to top in the following sub-sections, i.e., starting with the Communication Layer.

⁸ <https://www.eclipse.org/swt/>

⁹ <https://www.w3.org/html/>

¹⁰ <https://github.com/w3c/webcomponents/>

¹¹ <https://www.w3.org/Style/CSS/>

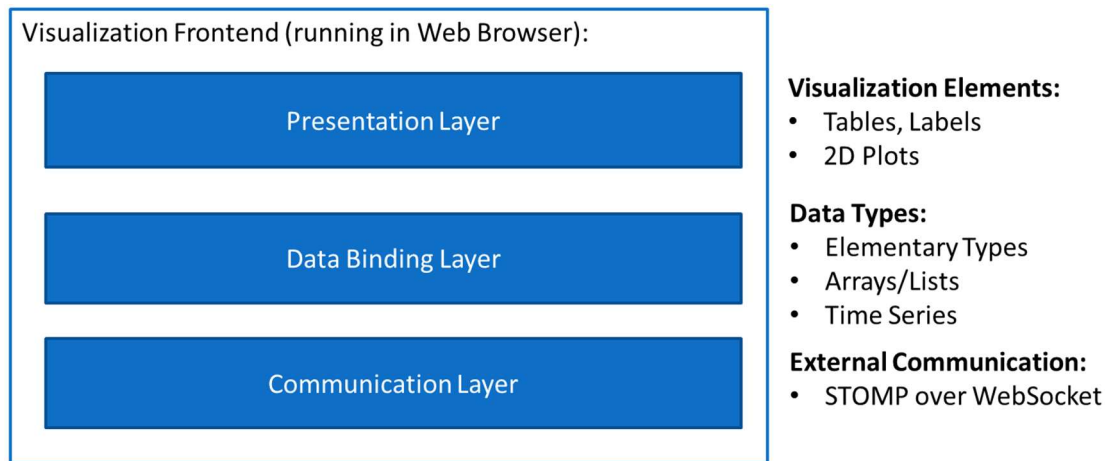


Figure 14: JVT layer architecture

In each layer, functionality is defined as JavaScript classes and functions. Especially the Presentation Layer makes strong use of a relatively new component technology called Svelte¹². It allows an efficient and easy-to-understand way to define Web components comprised of HTML elements (and other Svelte components), their styling as CSS3, and the functionality of these components using JavaScript. As a speciality of Svelte, the user interface gets reactive. This means that the framework keeps track of the data certain parts of the user interface depend on (for instance which value of the data model is displayed by a certain text element in the user interface). Whenever this data changes, all dependant parts of the user interface get updated automatically. This is similar to using formulas in a spreadsheet application: whenever contents of the cell a formula uses change, this cell gets updated to contain its new value.

5.2.1 Communication Layer

The communication layer is concerned with all functionality to connect to and maintain the connection to a data source(s) for a visualization scenario. In THREAT-ARREST, a message-oriented broker is used to asynchronously exchange state-updates from simulated or emulated components. The visualization tool connects to the broker using the WebSocket protocol¹³. Once this connection is established, the STOMP message protocol is used on this connection to subscribe to status update messages of certain platform components in total (using wildcards) or only for individual pieces of information (see also Figure 1). Data received this way is passed to the data binding Layer.

5.2.2 Data Binding Layer

The data binding layer acts as an intermediate data store for the components of the presentation layer. Presentation layer components can subscribe to certain pieces of information offered by a data source and get notified automatically by the data binding layer whenever this information changes.

The core functionality of the Data Binding Layer is to manage subscriptions for certain pieces of information and notifying subscribers about changes. This allows tracking which information is actually required by presentation-layer components, which is a useful feature as it allows requesting only for the required information from a data source.

¹²<https://svelte.dev>

¹³<https://tools.ietf.org/html/rfc6455>

The data binding layer also contains functionality to aggregate data, e.g. to form time-series. This could be used for instance to track how a certain simulated sensor value is evolving over time to be displayed in a line-chart.

5.2.3 Presentation Layer

The presentation layer contains all components that comprise the actual user interface of JVT. This library of visual components offers a set of generic UI components (like a top-level view element offering navigation between components) as well as components that can be used as building blocks of more complex visualization views. Components on the presentation layer provide functionality to subscribe to certain pieces of information from the data binding layer, and therefore, update their visual representation whenever a data source sends confirmation of the required information changing. Components in the library contain simple labels, but can also be complex objects, like charts or even a map component to show locations on a geographic map.

Unless stated otherwise, a component in the context of the JVT always refers to a visual part of the user interface, i.e., an element of the presentation layer.

5.3 Setting up a Visualization Scenario

Definition of a visualization scenario can usually be done with any Integrated Development Environment (IDE) or text editor suitable for handling projects based on HTML/JavaScript. In the following, the IDE “Visual Studio Code” from Microsoft will be used. In a build step (using a JavaScript-tool-set based on NPM¹⁴ and WebPack¹⁵), a single-page JavaScript application package is created usually only consisting of a single HTML, CSS and JavaScript file. This package can then be included, e.g. in the THREAT-ARREST dashboard, in order to use a JVT scenario.

Definition starts with a sample project provided by JVT (or checking it out from a version control system like GIT¹⁶). This project contains a working setup to create the final user interface using JavaScript build tools, like NPM and WebPack. It also offers access to sample components and currently contains the complete JVT source code to be used as a reference (in a future version the core JVT components will probably be moved to a separate JavaScript library). A concrete example using this setup is presented in more detail in the following subsections.

5.4 Example Visualization Connecting to Simulation Components

This section shows a comprehensive example comprised of defining a visualization scenario and presenting the visualisation components used in the example. The visualization tool currently connects directly to the simulation controller. For demonstration purposes the simulation controller provides basic broker functionality to provide the required data.

The example described below demonstrates some of the components required for the Smart Shipping and Energy Pilots.

¹⁴ The NodeJs Package Manager: <https://www.npmjs.com/package/npm>

¹⁵ <https://webpack.js.org/>

¹⁶ <https://git-scm.com/>

5.4.1 Scenario and Component Definition

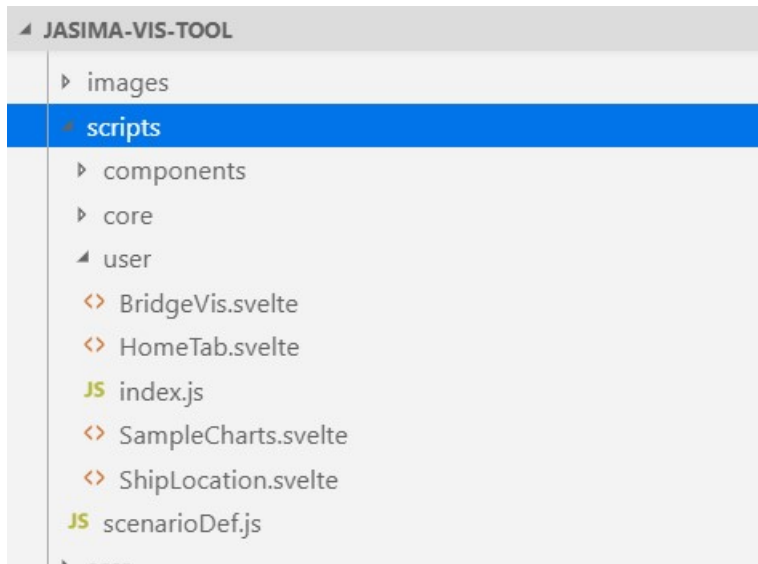


Figure 15: Example Project Setup in Visual Studio Code

The basic setup for the scenario definition is depicted in Figure 15, showing the contents of the “assets/scripts” sub-folder of the example project. The directory contains plain JavaScript files (file extension .js) as well as Svelte-Components (file extension .svelte). The subfolders “components” and “core” contain the JVT implementation (with the “components” folder containing the components of the presentation layer). The “user” folder contains complex view definitions that are specific to the current scenario and used here for the user role “user”.

The basic contents of a simulation scenario are defined in the file “scenarioDef.js”. The file used in the example is shown in Figure 16. It first imports all required components of the scenario. These could be standard components or components defined specifically for a certain scenario.

```

JS scenarioDef.js
assets > scripts > JS scenarioDef.js
1  // import all visualization views required
2  import { HomeTab, BridgeVis, SampleCharts, ShipLocation } from "../user"
3  import ExpResultsGeneric from "../components/ExpResultsGeneric.svelte"
4
5  // for each possible user role: export an array of visualization options
6  export default () => ({
7    user: [
8      // visualization tabs for user role
9      { name: 'Home', component: HomeTab, icon: "fas fa-home" },
10     { name: 'Bridge Visualization', component: BridgeVis, icon: "fas fa-ship" },
11     { name: "Ship's Position", component: ShipLocation, icon: "fas fa-map-marked-alt" },
12     { name: "2D Charts", component: SampleCharts, icon: "fas fa-chart-line" },
13     { name: "Experiment Results", component: ExpResultsGeneric, icon: "fas fa-poll" },
14   ],
15   trainer: [
16     // visualization tabs for trainer role to be added here
17   ],
18 })

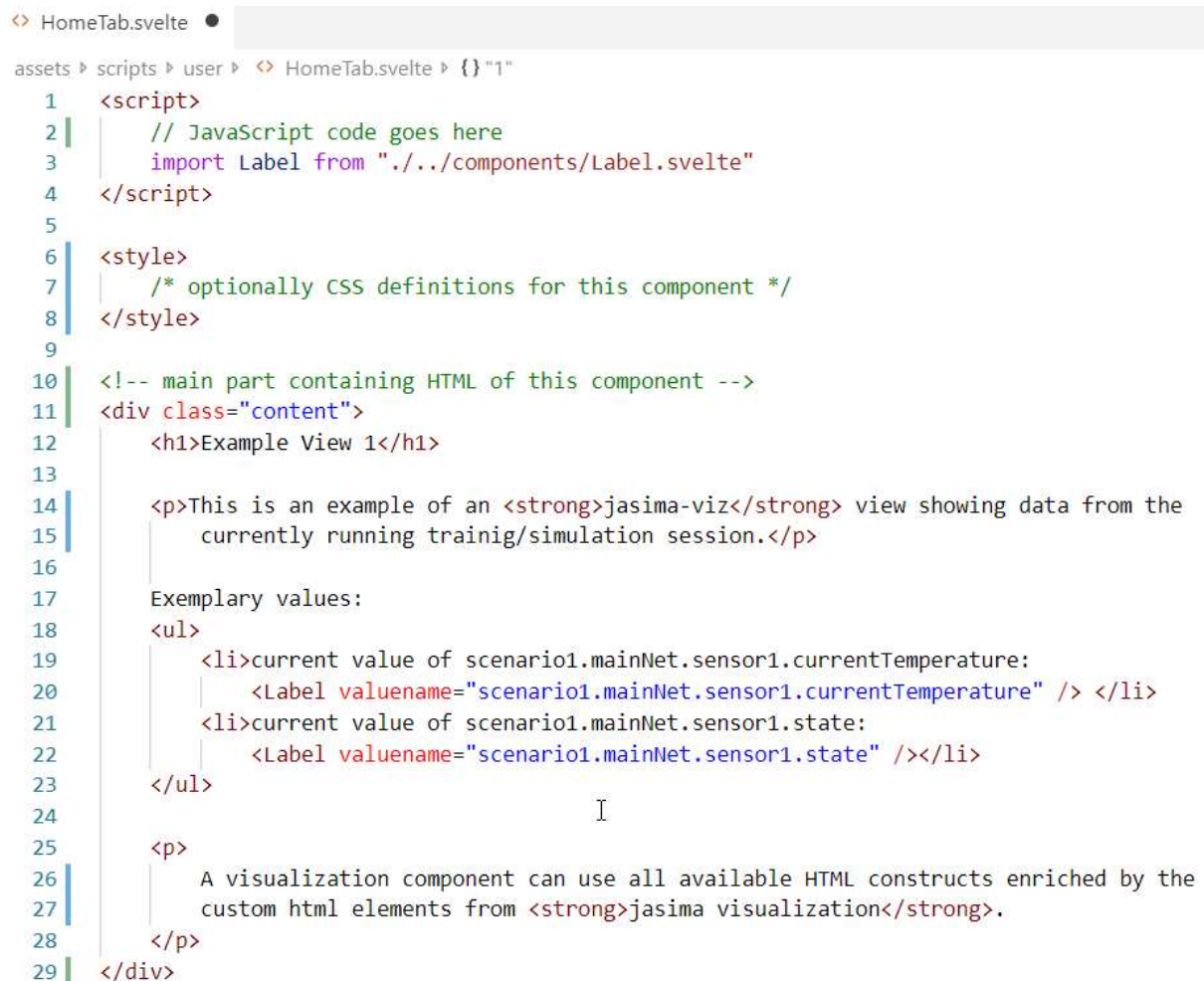
```

Figure 16: Scenario definition file

The main part of the scenario definition file defines for each user role the components their scenario view is comprised of. The example only defines views for the role “user”. Another role “trainer” is defined, but no views are assigned to this role at the moment.

As an example, visualization view “HomeTab.svelte” is shown in Figure 17. It consists of three main parts. First, enclosed in script-tags (lines 1 to 4), there is a JavaScript section to import components used in the third part (after line 8) and to provide the JavaScript code that the components require to work. In the example, only the “Label” component from the component library is imported. The (optional) second part (lines 6 to 8) contains CSS code to style the component as required. The main part starting at line 9 defines the HTML-Code the component consists of. It can also contain other components, as shown in the example with the “Label”-components used in lines 20 and 22 of Figure 17. A Label component shows a value from a data source. The value to be displayed is defined using the “valuenam” attribute. In the example, both labels show information from a simulated component. Which information is available to be displayed depends on the data source, as a visualization view usually is closely linked to a particular training scenario.

Visualization views will usually be defined once for a particular training scenario and then be used for many training sessions using this scenario.



```

1  <script>
2  |   // JavaScript code goes here
3  |   import Label from "../components/Label.svelte"
4  </script>
5
6  <style>
7  |   /* optionally CSS definitions for this component */
8  </style>
9
10 <!-- main part containing HTML of this component -->
11 <div class="content">
12 |   <h1>Example View 1</h1>
13 |
14 |   <p>This is an example of an <strong>jasima-viz</strong> view showing data from the
15 |     currently running trainig/simulation session.</p>
16 |
17 |   Exemplary values:
18 |   <ul>
19 |     <li>current value of scenario1.mainNet.sensor1.currentTemperature:
20 |       <Label valuenam="scenario1.mainNet.sensor1.currentTemperature" /> </li>
21 |     <li>current value of scenario1.mainNet.sensor1.state:
22 |       <Label valuenam="scenario1.mainNet.sensor1.state" /></li>
23 |   </ul>
24 |
25 |   <p>
26 |     A visualization component can use all available HTML constructs enriched by the
27 |     custom html elements from <strong>jasima visualization</strong>.
28 |   </p>
29 </div>

```

Figure 17: Example Visualization View

5.4.2 The Visualisation Scenario in Use

When the example using the visualization scenario is executed, the web browser shows a screen similar to Figure 18. The UI contains two parts. One for the visualization scenario as described before (“Scenario Visualization”) and an additional one “Simulation Controller” concerned with connecting to the controller, specifying and executing a simulation run.

This “Simulation Controller”-part of the user interface is just for the purpose of the stand-alone example using simulation and visualisation only. In the final THREAT-ARREST system architecture, this functionality will be triggered by the Training Tool as part of the overall training workflow.

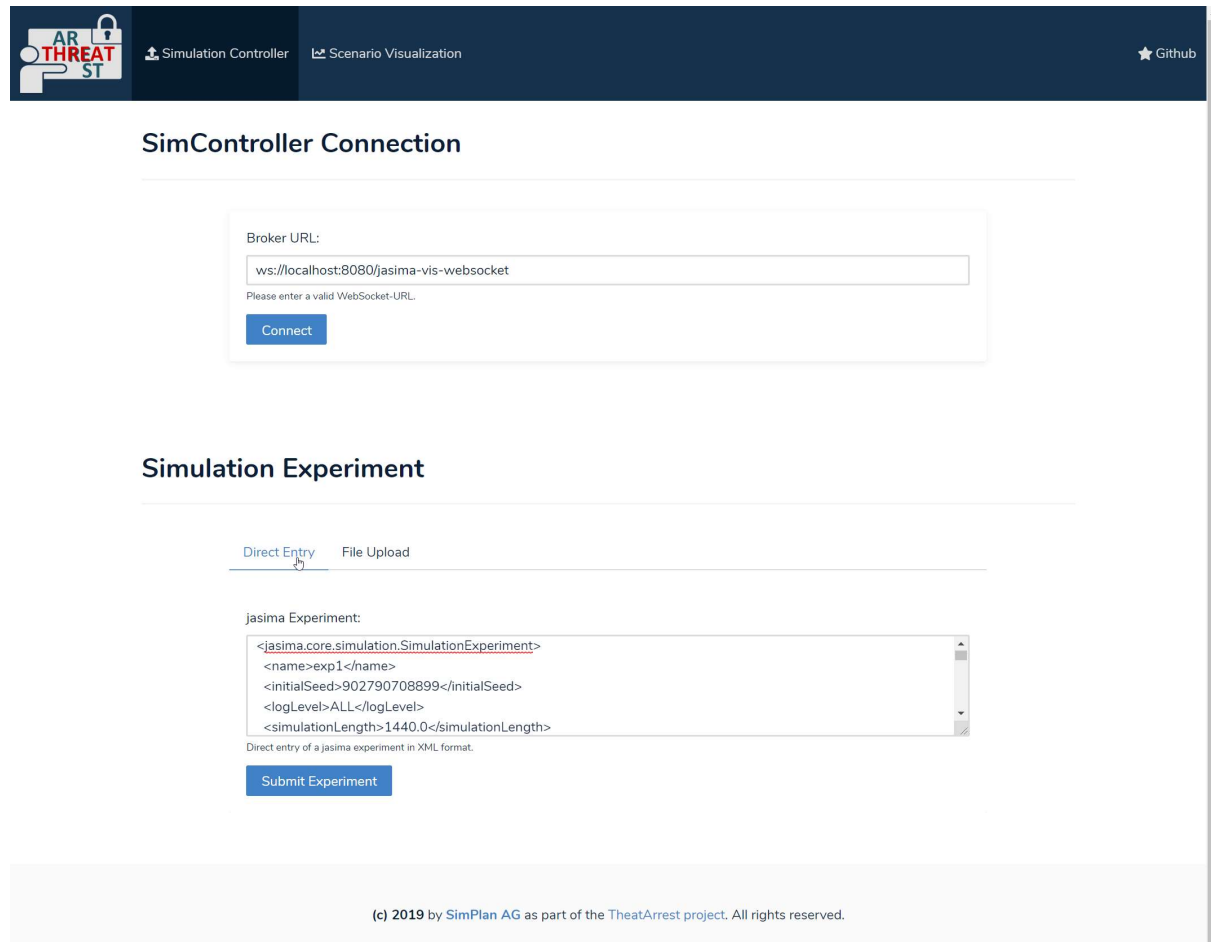


Figure 18: User Interface for Simulation Control and Scenario Visualization

The Simulation Controller can be reached using a Web-Socket connection. In the UI, the proper URL can be configured and the JVT connection to the SimController can be established. The simulation scenario can be specified in the section “Simulation Experiment”. For simplicity, here we chose to just copy-and-paste the XML-file required by the Jasima simulator in the text-box, so it can then be submitted to the simulation controller (see also D5.2 for a description of the expected file format). After submitting it, we can switch to the “Scenario Visualization” page (Figure 19). In future versions of the THREAT-ARREST platform this XML-file will be derived from the CTP simulation sub-model (see also deliverable D3.1).

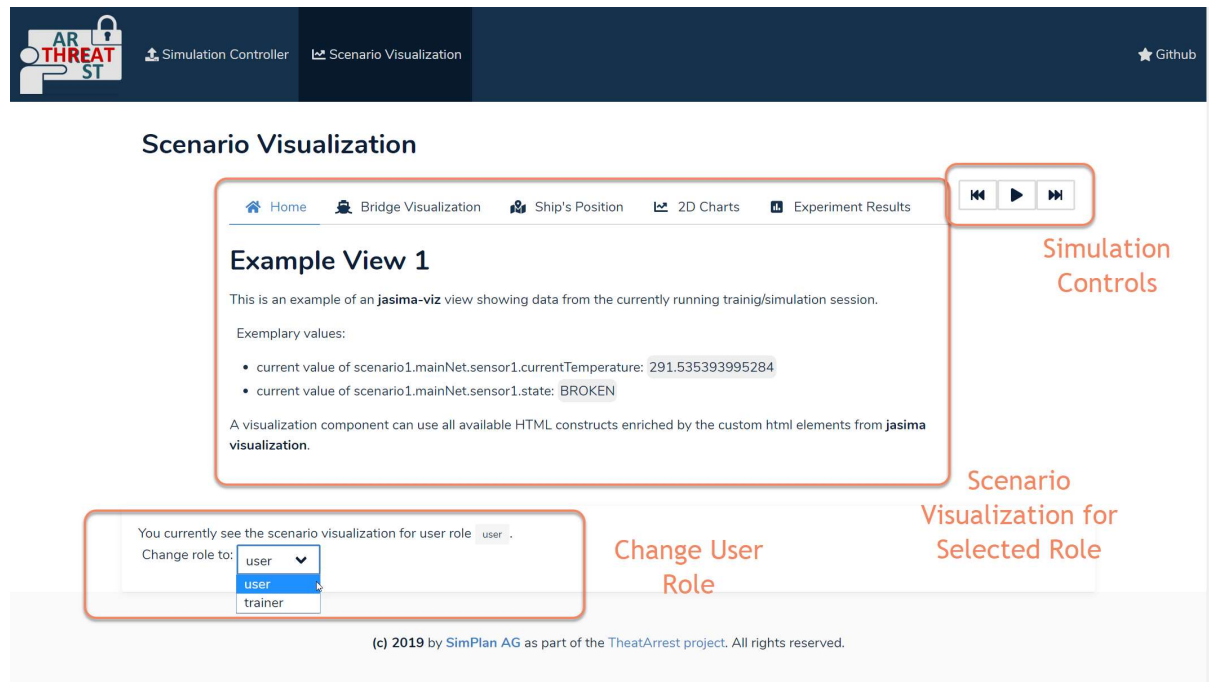


Figure 19: The Scenario Visualization View

The “Scenario Visualization” consists of three parts, highlighted in Figure 19. “Simulation Controls” are buttons to trigger the run/pause/reset-operations of the simulation controller. The section “Change User Role” can be used to view the visualization as defined for a particular user role. Both of these parts will probably not be provided if the JVT is utilized via the THREAT-ARREST dashboard. The main part finally contains the visualization scenario that we defined for the selected user role (in the example: “user”).

JVT creates a navigation tab for each of the views defined in “scenarioDef.js”. When the user clicks on a particular tab the respective view component is created and activated. In the example shown above our sample component “HomeTab.svelte” (see Figure 17 and Figure 19) is active. As a result, the component’s HTML is rendered by the browser. The two “Label” components we used in the list (lines 20 and 22 in Figure 17) are shown in Figure 19 using the grey background. As the simulation was running when taking the screenshot, the values shown always reflect the current simulation state. Any change in the simulation state results in a STOMP message containing the new value. This message finally is published by the broker and transmitted to the JVT. As a result, all view elements/components showing this value will be updated to show or use the most recent values.

In the following, the rest of the example views will be shown, each demonstrating certain capabilities of the JVT.

The “Bridge Visualization View” (inspired by the maritime scenario in THREAT-ARREST, currently described in D2.1, D2.2, and D 5.2) is shown in Figure 20. It contains a static image of a ship’s bridge containing two active areas indicated by the red ellipses. This internally uses an SVG¹⁷ area map to define active areas. If a user clicks on them, several actions could be triggered. Examples would be opening a different visualization view (either by changing the tab or displayed as an overlay of the image) or opening a connection to a certain virtual machine in the emulated environment. By using HTML5 and state-of-the art web technologies as the basis for JVT, all features of this powerful platform for presenting visual content can be used easily.

This is also demonstrated by the “Ship’s Position” view illustrated in Figure 21. This figure shows a map containing three markers: two for the source/destination ports from Heraklion to

¹⁷ Scalable Vector Graphics: <https://www.w3.org/TR/SVG2/>

Piraeus, respectively, and a third one for the current position of the simulated ship. As the simulation run progresses, the ship's marker is updated periodically to reflect the latest position of the ship being synchronized with the position of the simulation component representing the ship in the simulation tool.

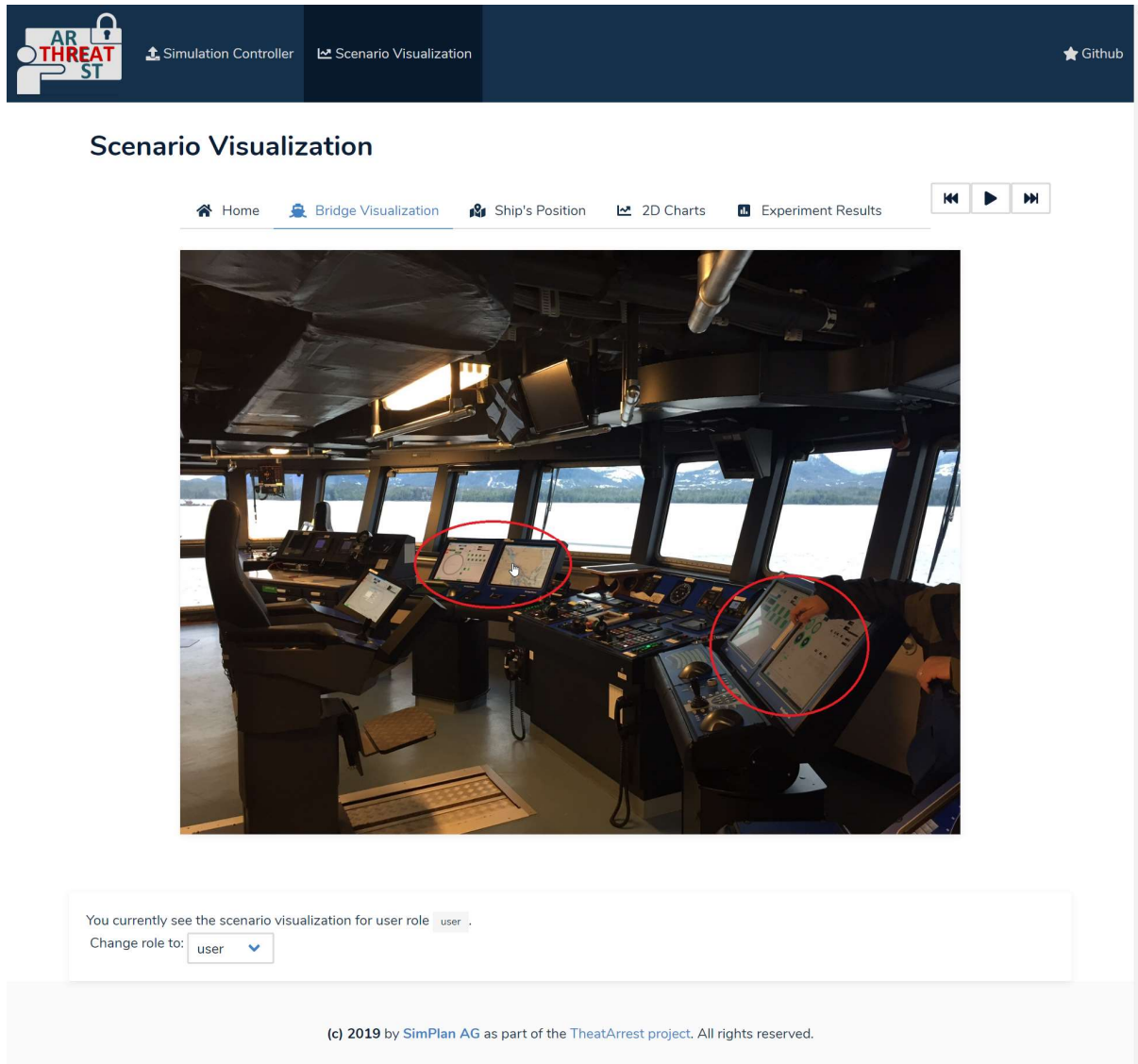


Figure 20: The Bridge Visualization View

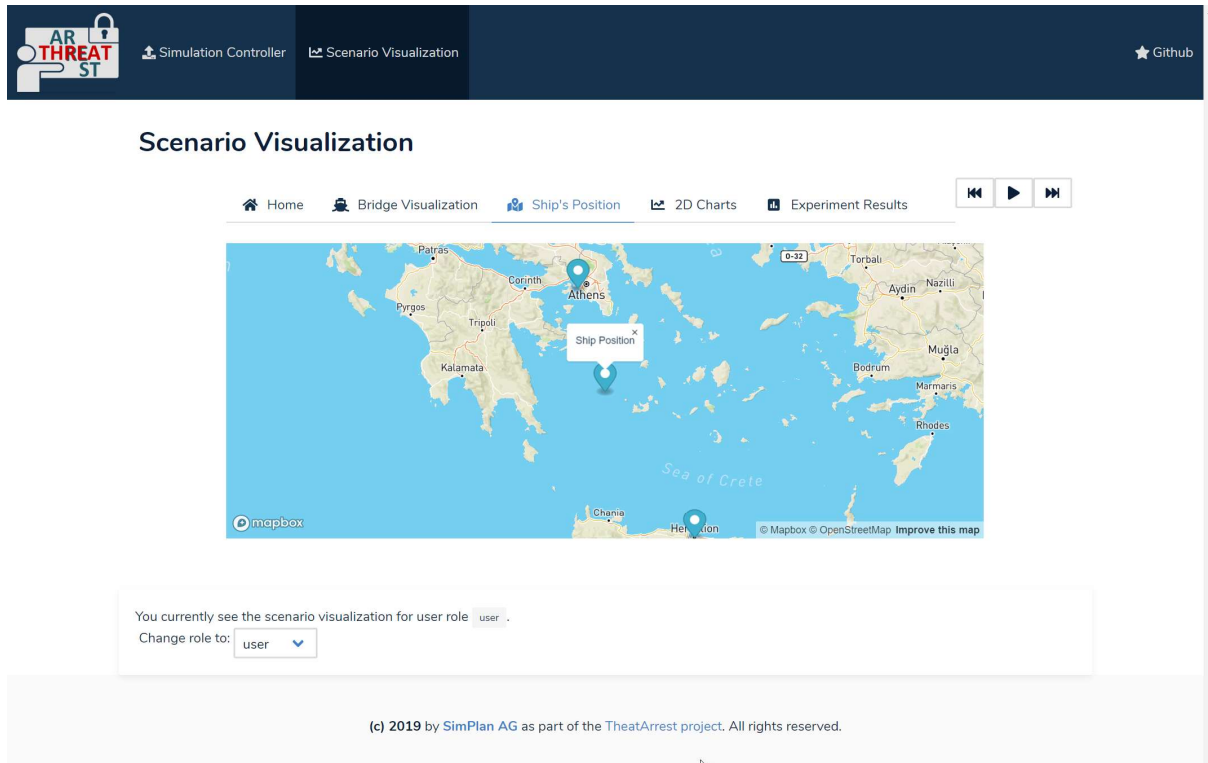


Figure 21: The View "Ship's Position"

The Web Platform offers a large number of very powerful JavaScript-libraries to create charts. JVT will also support the most important chart types and make them accessible for easy use as part of the visualization scenarios. As two simple examples, the ColumnChart and BarChart-components offer the respective chart types with data linked to a data source such simulation or emulation. An example using these two chart types is shown in Figure 22. Internally this uses the ApexCharts library¹⁸. In future versions of JVT, we want to also support some of the more advanced chart types offered by this library, like line charts, heat maps, radar charts, etc. Similar to using ApexCharts, it should also not be too difficult to add support for other charting or even 3D-visualization libraries in the future, as they could offer interesting visualization options to be used in a training session.

The example charts in Figure 22 also demonstrate another feature offered by state-of-the-art web technologies: they also consider mobile devices and offer the so-called "fluid designs" that can easily be used on mobile devices, such as tablets or smart phones in addition to normal computers. To demonstrate this, the two example charts are duplicated and organized in two columns. If shown on a large screen, they are shown as a 2×2 grid, but if the screen-size is reduced to the size of a smartphone (Figure 23), the layout changes to a single column, still allowing the charts to be readable despite the small screen size. When using the reduced size, the main navigation bar is also collapsed to a drop-down-menu that is only opened when the user clicks on the icon in the top-right position.

¹⁸ <https://apexcharts.com/>

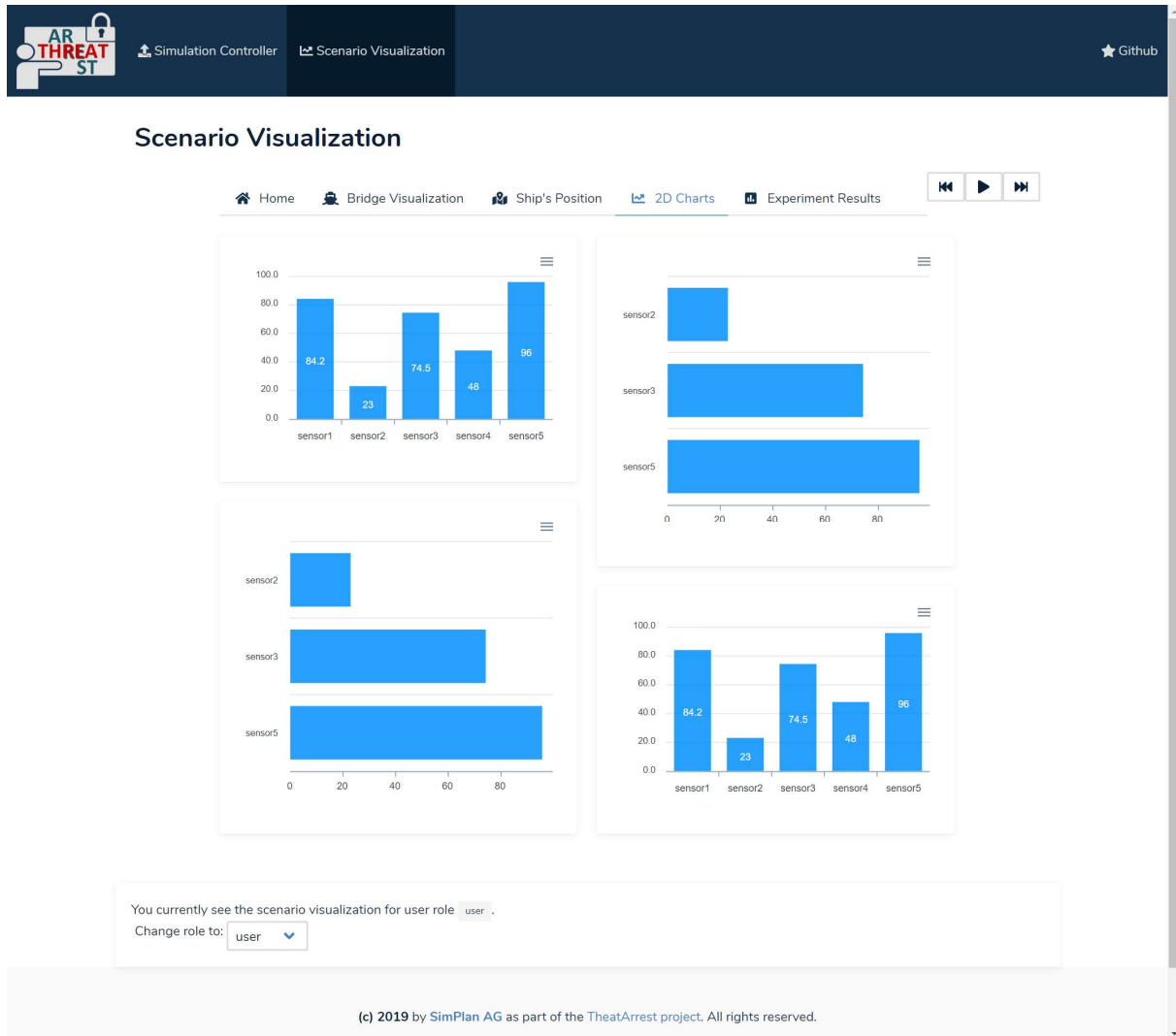


Figure 22: The Visualization View "2D charts", normal browser size

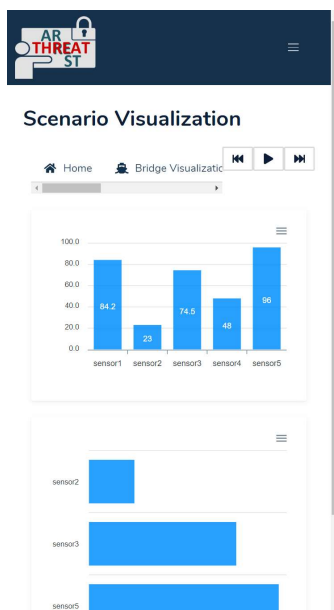
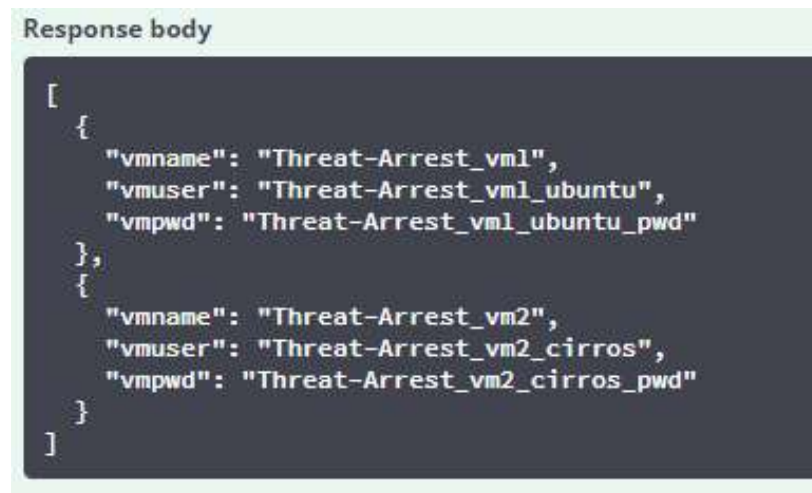


Figure 23: Visualization View "2D charts", small browser size

6 Trainee Access to Emulated Components

The access to the Emulated components, created and instantiated by the Emulation Tool within the OpenStack environment, is granted through the Apache Guacamole¹⁹ gateway provided by the Emulation Tool itself (also see deliverable D2.3).

The steps of the interaction between the Visualization Tool and the Emulation Tool are the following. First, the Training Tool/dashboard requests the Emulation Tool for the deployment of the training scenario selected by the trainee, as indicated in Section 3. The Emulation Tool, when the deployment is completed and the VMs are ready to be accessed, then sends as REST reply the name of the deployed machines, and specific username and password to be used for the access (see Figure 24).



```
[
  {
    "vmname": "Threat-Arrest_vm1",
    "vmuser": "Threat-Arrest_vm1_ubuntu",
    "vmpwd": "Threat-Arrest_vm1_ubuntu_pwd"
  },
  {
    "vmname": "Threat-Arrest_vm2",
    "vmuser": "Threat-Arrest_vm2_cirros",
    "vmpwd": "Threat-Arrest_vm2_cirros_pwd"
  }
]
```

Figure 24: Emulation Tool REST response

Then, the Dashboard can access the Guacamole gateway within the Emulation Tool. The access is granted using the username and password provided in the response. The Training Tool can either give the username and password to the trainee and let him/her access it, or provide direct access as a common HTTP response.

At this point, the Dashboard will open a web frame that contains the SSH console, in case of a Linux VM, or a Remote Desktop view, in case of Windows VM.

Figure 25 shows the Guacamole login form and Figure 26 provides an example of access to a Linux VM. Please note that the provided URL is the URL assigned to the OpenStack infrastructure installed within the THREAT-ARREST framework. Any single VM is accessed through the main OpenStack URL. The connection is then forwarded by the Guacamole gateway to the specific VM in a transparent way. The Emulation Tool provides facilities to automatically create and cancel user connections to the VM.

¹⁹ <https://guacamole.apache.org/>

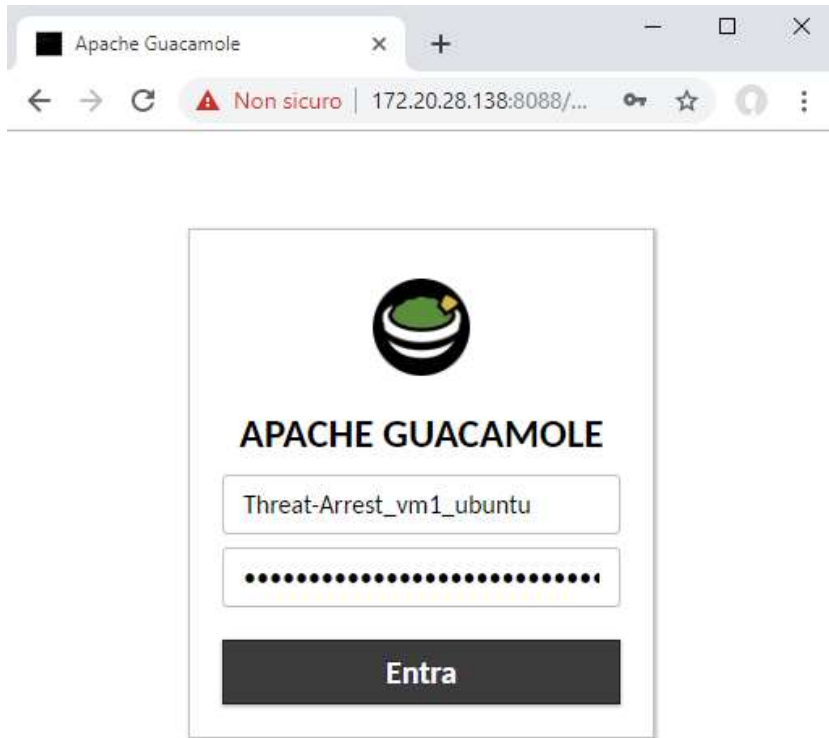


Figure 25: Apache Guacamole login form

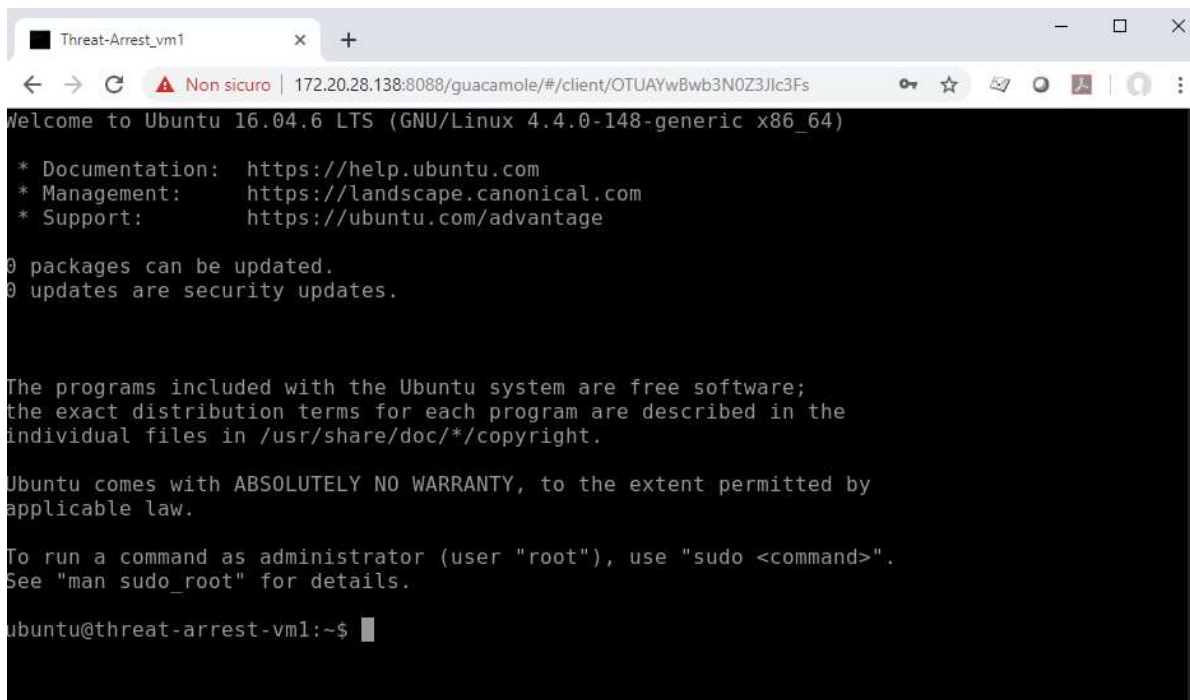


Figure 26: Example of SSH connection through Apache Guacamole

7 Conclusions

This deliverable documents the current state of the THREAT-ARREST components that offer major parts of the user interface visible to users of the THREAT-ARREST platform. As such, they are very important to ensure usability and user-acceptance of the platform.

This deliverable is the first one reporting progress on task T4.1 of the project. It reports on the work done from its start in month 4 until month 12 and also describes some of the work that will be performed in the months ahead.

All of these components will be accessible using web-based technologies to ensure a good interactive user-experience. The dashboard, which is part of the Training Tool, will offer the central starting point of platform users allowing them to access the UI's of other THEAT-ARREST platform components as required by the overall workflow of a CTP training session. It integrates the UI of the Gamification Tool, the visualization offered by the Jasima Visualization Tool (used to show the current state of simulated and emulated cyber-system components of a training session), as well as the UI offered by the Apache Guacamole gateway to login to a virtual machine of the Emulation Tool and interact with it. All of this functionality is achieved using web-based technologies, requiring only a web browser to be used by trainees.

References

- [1] Beckers, K., Bravos, G., Goeke, L., & Pape, S. (2019). *D4.2: THREAT-ARREST serious games*.
- [2] Ferrera, E., et al., 2018. IoT European Security and Privacy Projects: Integration, Architectures and Interoperability. CRISTin – SINTEF, Next Generation Internet of Things. Distributed Intelligence at the Edge and Human Machine-to-Machine Cooperation. Book Chapter 7, pp. 207-292.
- [3] Fysarakis, K., et al., 2015. RT-SPDM: real-time security, privacy and dependability management of heterogeneous systems. Human Aspects of Information Security, Privacy and Trust (HCI International 2015), 2-7 August 2015, Los Angeles, CA, USA, Springer, LNCS, vol. 9190, pp. 619-630.
- [4] Hatzivasilis, G., et al., 2019a. WARDOG: Awareness detection watchdog for Botnet infection on the host device. IEEE Transactions on Sustainable Computing – Special Issue on Sustainable Information and Forensic Computing, IEEE, vol. 4, pp. 1-15.
- [5] Hatzivasilis, G., et al., 2019b. MobileTrust: Secure Knowledge Integration in VANETs. ACM Transactions on Cyber-Physical Systems – Special Issue on User-Centric Security and Safety for Cyber-Physical Systems, ACM, vol. 4, issue 3, Article no. 33, pp. 1-15.
- [6] Hatzivasilis, G., et al., 2017. SecRoute: End-to-End Secure Communications for Wireless Ad-hoc Networks. 22nd IEEE Symposium on Computers and Communications (ISCC 2017), IEEE, Heraklion, Crete, Greece, 03-06 July 2017, pp. 558-563.
- [7] Manifavas, C., et al., 2014. DSAPE – Dynamic Security Awareness Program Evaluation. Human Aspects of Information Security, Privacy and Trust (HCI International 2014), 22-27 June, 2014, Creta Maris, Heraklion, Crete, Greece, Springer, LNCS, vol. 8533, pp. 258-269.