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

D1.3: THREAT-ARREST platform's initial reference architecture [†]

Abstract: This deliverable presents the initial reference architecture of the THREAT-ARREST platform. It extends the overall architecture of DoA with two specific architectural views – the Data Flow View and the Component Communication View. Following the platform architecture, each main tool of the platform is specified in terms of its own architecture, specific components' functionality and I/O dependencies. As initial architecture, the two views together with the tools' specifications will provide an important insight on the data dependencies and communications among the platform tools and will successfully contribute to the next stage of project activities.

Contractual Date of Delivery	28/02/2019
Actual Date of Delivery	28/02/2019
Deliverable Security Class	Public
Editor	*Hristo Koshutanski (ATOS)*
Contributors	TUBS, FORTH, CZNIC, STS, UMIL, SIMPLAN, SEA, ITML, IBM, ATOS

[†] *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.*

Quality Assurance	*George Hatzivasilis (FORTH),* *Elvinia Riccobene (UMIL),* *Fulvio Frati (UMIL),* *Stelvio Cimato (UMIL).*

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 (FORTH),*
2. *Elvinia Riccobene (UMIL),*
3. *Fulvio Frati (UMIL),*
4. *Stelvio Cimato (UMIL).*

Revisions

Version	Date	By	Overview
0.7	28/02/2019	Editor	Addressed reviewers' comments.
0.6	25/02/2019	Editor	Added Introduction and Conclusions by ATOS. Updated THREAT-ARREST platform architecture Data Flow View and Component Communication View, and description of those by ATOS.
0.5	20/02/2019	SEA, UMIL, IBM, FORTH (contributors as in v0.3)	Updated Gamification Tool specification by SEA, Updated Emulation Tool specification by UMIL, Updated Data Fabrication Platform specification by IBM, Updated platform requirements analysis by FORTH.
0.4	14/02/2019	Editor	Added ATOS contribution to THREAT-ARREST platform architecture Data Flow View and Component Communication View, and description of those.
0.3	29/01/2019	Martin Kunc, Pavel Bašta	Added CZ.NIC contribution to the platform security requirements analysis
0.3	29/01/2019	Kristian Beckers, Sebastian Pape	Added contribution by SEA to the Gamification Tool specification.
0.3	29/01/2019	Vina Rompoti, George Bravos, Vassilis Chatzigiannakis	Added contribution by ITML to the Training Tool Specification.
0.3	29/01/2019	Torsten Hildebrandt, Dirk Wortmann	Added contribution by SIMPLAN to the Simulation Tool specification and the Visualisation Tool specification.
0.3	29/01/2019	Oleg Blinder, Michael Vinov	Added contribution by IBM to the Data Fabrication Platform specification.
0.3	29/01/2019	Konstantinos Fysarakis, George Spanoudakis	Added contribution by STS to the Assurance Tool specification.
0.3	29/01/2019	George Hatzivasilis	Added FORTH contribution to platform requirements and dependencies
0.3	29/01/2019	Ernesto Damiani, Fulvio Frati, Stelvio Cimato, Elvinia Riccobene	Added UMIL contribution to the Emulation Tool Specification.
0.2	22/01/2018	Marinos Tsantekidis	Added TUBS contribution to users and stakeholders analysis
0.1	15/01/2019	Editor	First Draft

Executive Summary

This deliverable presents the initial version for the THREAT-ARREST platform architecture, focusing on the **Data Flow** and **Component Communication** views between the underlying tools. It extends the architecture in DoA with relevant details on data dependence and communication among the tools, and provides initial specification of each main tool of the platform. As initial architecture, this document will serve as input to further project activities and developments. The work is developed under the task "T1.3 – Initial Platform architecture".

Table of Contents

List of Abbreviations

CM Certification Model

CIH Component’s Interaction Hub

cPPP contractual Public-Private Partnership

CSP Constraint Satisfaction Problem

CTTP Cyber Threat and Training Preparation

DoA Description of Action

I/O Input/Output

IAM Identity & Access Management

IT Information Technology

JSON JavaScript Object Notation (data-interchange format)

REST Representational State Transfer (cf. RESTful Web services)

SQL Structured Query Language

SRIA Cyber-Security Strategic Research and Innovation Agenda

TRL Technology Readiness Level

VM Virtual Machine

WP Work Package

List of Figures

List of Tables

1 Introduction

The THREAT-ARREST initial reference architecture forms an important input and part of the first milestone of the project. The initial architecture extends the architecture identified in the DoA by introducing two specific views – the Data Flow View and the Component Communication View.

The rationale for the data flow view is to give insights of data flows in the system and data dependencies among the tools of the platform – from users' interactions with the platform for training and evaluation, to simulation and emulation of system behaviour, and to security assurance assessment based on testing and monitoring.

As initial architecture, it has been also recognised the need for a greater level of details on the functional components of each tool of the platform and identify what these components interact for with the other components of the tools. Such component communication view is an essential step towards finer-grained specification of the platform and its integration (subject of Work Package (WP) 6), and I/O interconnections among the tools (subject of tasks T2.4, T4.6 and T5.4).

Importantly, and complimentary to the architecture views, this document details the specification of each main tool of the platform. It discusses the architecture of each tool, defines the components forming part of the architecture and their role, and interfaces or I/O dependences on other tools of the platform.

As initial architecture, we believe that the two views will provide an important insight on the data dependencies and communications among the platform tools and will successfully contribute to the next stage project activities where, for instance, I/O mechanisms and interfaces for tools' interoperation will be subject of development.

We note that as an initial architecture, each tool specification will be further analysed and refined along with the overall architecture as necessary to address further project activities and developments.

We followed a specific baseline of activities for the definition of the architecture and production of this report. We first analysed the users and stakeholders of the platform, the beneficiaries in a cyber-security ecosystem, both in a general case and also in particular to the three pilot sectors in the project, ref. D1.1 (THREAT-ARREST D1.1, 2018). We then performed analysis of platform's requirements identified in D1.2 (THREAT-ARREST D1.2, 2018) in the context of requirements' dependencies among tools, and identified data and communication dependences from requirements' specification phase.

Next, we iterated with each corresponding partner for specification of their tools to determine and finalise the initial platform architecture. Importantly, as a result of this iterative process with the tools' specifications, we refined and improved not only the overall architecture but also some individual tools' architectural aspects.

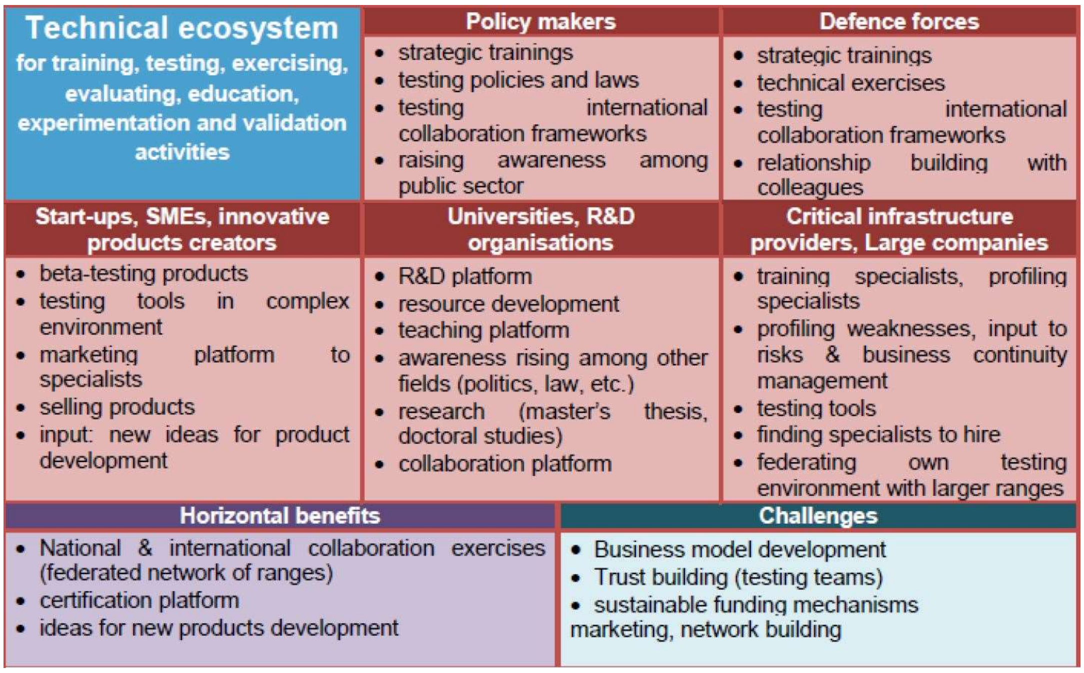
The rest of the document is structured as follows: **Chapter 2** overviews the users and stakeholders of the platform. **Chapter 3** presents the THREAT-ARREST initial reference architecture including the platform requirements analysis and the two extended architectural views – the Data Flow View and the Component Communication View. Next, **Chapter 4** presents the Assurance Tool specification, **Chapter 5** presents the Simulation Tool specification, **Chapter 6** the Emulation Tool specification, **Chapter 7** the Gamification Tool specification, **Chapter 8** the Training Tool specification, **Chapter 9** the Visualisation Tool specification, and **Chapter 10** the Data Fabrication Platform specification. Finally, **Chapter 11** concludes the document and outlines next steps related to the platform architecture.

2 THREAT-ARREST Users and Stakeholders

It is acknowledged that the successful implementation of the project and the sustainability and wider exploitation of its outcomes will be maximised if they are most relevant to actors in the rapidly evolving market for cybersecurity solutions and if they take into account the incentives, roles and limitations of the actors the project is trying to influence. Figure 1 shows potential beneficiaries in a cyber-security ecosystem. Targeted actors can be grouped in the following groups, covering most of the ecosystem defined in the European Cyber-Security Strategic Research and Innovation Agenda (SRIA) for contractual Public-Private Partnership (cPPP) (SRIA-cPPP, 2019):

- Network operators and managers of sensitive infrastructures, to help them understand the additional capabilities of THREAT-ARREST with respect to existing solutions;
- Security designers and managers, to help them value how THREAT-ARREST can improve the security of the network infrastructure and provide additional detection and reaction capabilities;
- Academic and industrial R&D community interested in/dealing with cyber-security, to make them aware of the project results and to potentially incorporate them as part of more a complex/complete solution or product;
- Public and private organisations, including SMEs that can be potential customers of the THREAT-ARREST solution;
- policy makers (EU and national cyber-security authorities), to inform them about the progress accomplished by THREAT-ARREST with respect to improved security features of modern networks and how they can contribute to the overall security of an ICT infrastructure.

Furthermore, THREAT-ARREST will build strong collaborations with cPPPs and related initiatives as pointed out in the European Cyber-Security cPPP Strategic Research & Innovation Agenda, exploiting (i) its partners' memberships and (ii) its partners' existing active involvement in ongoing projects within such initiatives.



Technical ecosystem
for training, testing, exercising, evaluating, education, experimentation and validation activities

Policy makers
- strategic trainings
- testing policies and laws
- testing international collaboration frameworks
- raising awareness among public sector

Defence forces
- strategic trainings
- technical exercises
- testing international collaboration frameworks
- relationship building with colleagues

Start-ups, SMEs, innovative products creators
- beta-testing products
- testing tools in complex environment
- marketing platform to specialists
- selling products
- input: new ideas for product development

Universities, R&D organisations
- R&D platform
- resource development
- teaching platform
- awareness rising among other fields (politics, law, etc.)
- research (master's thesis, doctoral studies)
- collaboration platform

Critical infrastructure providers, Large companies
- training specialists, profiling specialists
- profiling weaknesses, input to risks & business continuity management
- testing tools
- finding specialists to hire
- federating own testing environment with larger ranges

Horizontal benefits
- National & international collaboration exercises (federated network of ranges)
- certification platform
- ideas for new products development

Challenges
- Business model development
- Trust building (testing teams)
- sustainable funding mechanisms marketing, network building

Figure 1. Potential beneficiaries in the Cyber-Security Ecosystem

Further classifying the actors involved with THREAT-ARREST, they are distinguished in two categories, (i) users and (ii) stakeholders.

Under the first category fall those persons or organizations who directly interact with the THREAT-ARREST system (platform), and therefore, they are within the THREAT-ARREST system boundaries. Referring to deliverable D1.1 (THREAT-ARREST D1.1, 2018) where the pilot systems were defined, examples of users can be found for each of the use-case applications:

- Residential consumers, smart energy solutions providers (smart energy system);
- Clinicians, patients, public authority agents, system administrators, compliance auditors (healthcare system);
- Ship owners, on-board personnel, shore-side personnel (smart shipping system).

THREAT-ARREST stakeholders are those persons or organizations that have an influence on or are affected by the THREAT-ARREST system, but they are outside the project system boundaries. In the use-cases, in deliverable D1.1, examples of stakeholders are:

- Energy production enterprises, security service providers to energy production ICT systems (smart energy system);
- Public healthcare agencies, hospitals and medical practices, regulatory authorities and decision makers (healthcare system);
- Maritime technology and service providers, regulatory and standardisation bodies in the maritime industry, shipping management entities (smart shipping system).

Figure 2 depicts examples of relationships between such actors and the THREAT-ARREST platform for the three pilot sectors considered in the project.

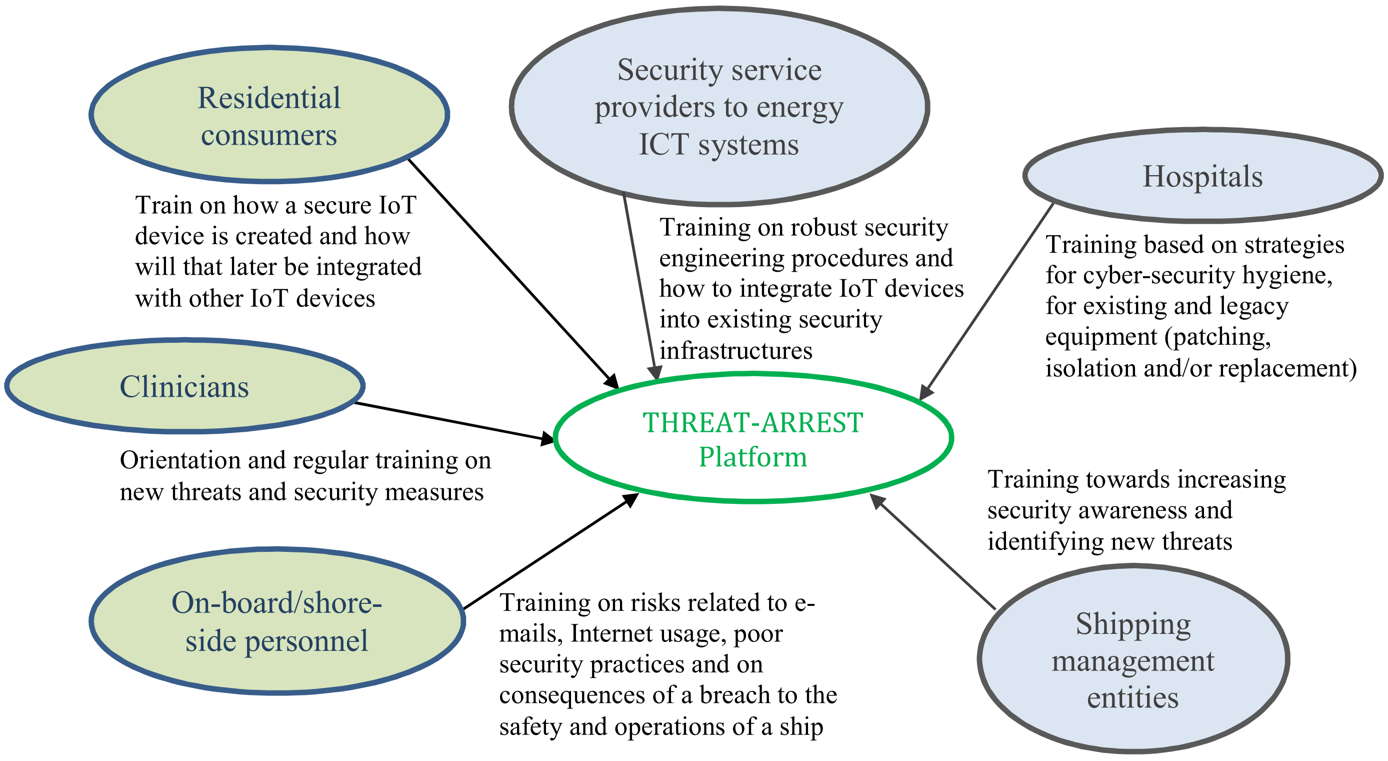

Residential consumers
Security service providers to energy ICT systems
Hospitals
Train on how a secure IoT device is created and how will that later be integrated with other IoT devices
Training on robust security engineering procedures and how to integrate IoT devices into existing security infrastructures
Training based on strategies for cyber-security hygiene, for existing and legacy equipment (patching, isolation and/or replacement)
Clinicians
THREAT-ARREST Platform
Orientation and regular training on new threats and security measures
Training towards increasing security awareness and identifying new threats
On-board/shore-side personnel
Training on risks related to e-mails, Internet usage, poor security practices and on consequences of a breach to the safety and operations of a ship
Shipping management entities

Figure 2. THREAT-ARREST Pilots-specific Users and Stakeholders relationship diagram

3 THREAT-ARREST Platform Architecture

We will first recall the platform architecture high-level view as identified in the DoA (THREAT-ARREST DoA, 2018). The goal is to introduce and familiarise the readers with the scope of the platform and its main functional components/tools.

Based on the high-level platform overview, we will present an initial analysis of the dependencies of the tools' requirements identified in D1.2 followed by two specific views of the architecture – the **Data Flow View** and the **Component Communication View**. As initial architecture, we believe that the two views will provide an important insight on the data dependencies and communications among the platform tools, and will successfully contribute to the next stage project activities where, for instance, I/O mechanisms and interfaces will be subject of development.

3.1 Platform Architecture High-level View

The high-level view of the platform architecture is presented followed by a short description of the role of each of the main tools in the platform. The presentation of the architecture and tools follows the DoA. The aim is to make the deliverable self-contained and self-explanatory.

The THREAT-ARREST platform will offer training on: (i) known and new advanced cyber-attack scenarios, (ii) how to make effective and systematic use of different security tools developed to detect and/or respond to cyber-attacks in all the different layers of the implementation stack of a cyber-system, (iii) taking different types of actions against cyber-attack. The platform will comprise six key components, as can be seen in Figure 3.

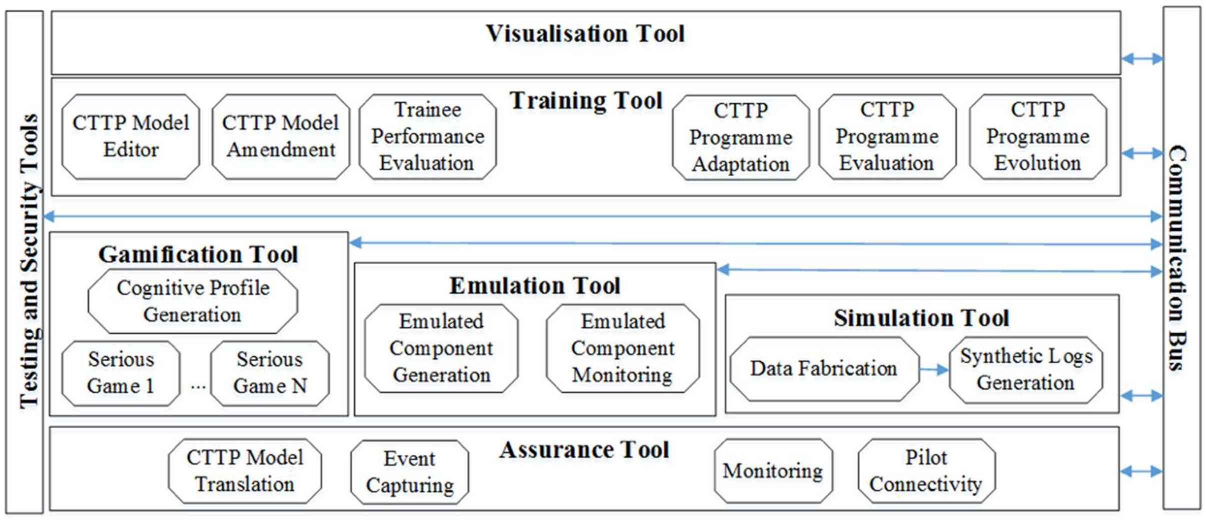


Figure 3. THREAT-ARREST Platform Architecture High-level View (THREAT-ARREST DoA, 2018)

The assurance tool supports the continuous assessment of the security of the cyber-system through the combination of runtime monitoring and dynamic testing, in order to provide information about the status of the actual cyber-system. It also collects runtime system events and generates alerts that provide the basis for setting up realistic simulations.

The simulation tool allows simulating individual cyber-system components and networks of such components to enable the simulation of entire training scenarios defined in CTTP programs.

The emulation platform supports the generation of emulated cyber-system components, in the form of interconnected Virtual Machines (VMs), equipped with the appropriate software stack. The platform relies on well-established generic machine emulators, possibly open-source such as QEMU (Bellard 2005), VirtualBox (Oracle 2018), VMWare (VMWare

2018), OpenStack (Openstack 2019), Open Virtual Network (OVN 2019), to achieve the generation of the emulated components at different levels. Using these frameworks, it is possible to select and emulate different attack scenarios.

The gamification tool hosts various serious games, scenarios and training evaluation mechanisms, which enable trainees to develop skills in being resilient to and preventing social engineering attacks (e.g. phishing, impersonation attacks, etc.). The provided games are driven by the threats and assumptions specified in CTTP models (security assurance).

The training tool supports the definition of CTTP models and programs, the presentation of learning material/exercises of CTTP programs, enables trainee actions in response to cyber-threats, interactions with simulated and/or emulated cyber-system components, trainee performance evaluation, CTTP program evaluation and adaptation.

The visualization tool enables the graphical representation of simulations and emulations, the effect of training actions on simulated and emulated systems, as well as the status of the underlying components.

3.2 Platform Requirements Analysis

We will summarize the dependencies among the THREAT-ARREST platform tools in the context of their requirements, as documented in deliverable D1.2. Looking at requirements for dependencies among tools, it will help us to refine the high-level view of the initial THREAT-ARREST platform architecture in the context of data flow and communication among the tools. Further requirements analysis is expected to take place in the next stage of project activities with the aim to refine the platform's functionality and operations.

3.2.1 Tools' Dependencies

There are 73 architectural requirements, where 53 of them are mandatory (MUST) while 20 of them are optional (SHOULD). Among those, there are 14 (12 MUST / 2 SHOULD) requirements for the assurance tool, 13 (9/4) for the simulation tool, 8 (6/2) for the emulation tool, 13 (10/3) for the gamification tool, 6 (4/2) for the training tool, and 19 (12/7) for the visualization tool. All requirements are architectural and applicable to the whole spectrum of the THREAT-ARREST platform, representing the features which will validate the compliance with the designed functionality at the later stages of the project. For convenience, Annex I summarises all platform requirements from D1.2.

Table 1 presents the dependencies between the 6 main THREAT-ARREST components (assurance, simulation, emulation, gamification, training, and visualization) based on their requirements correlation. The visualization tool exhibits the highest degree of reliance from the other components (38 requirements) while the assurance tool is the most dative (demanding) one (expressing 25 requirements to other components).

Three main features are described in the table:

- The *diagonal* cells summarize the tools' own requirements. This reflects the tool's validation complexity.
- Then, each *row* represents the reliance of the relevant tool in the row from the rest components, as incoming dependencies. The final cell sums these incoming requirements and reflects the overall tool's reliance from the rest platform. The underlined cells highlight the most critical components that provide input to the tool.

- The *columns* detail the dependencies of the relevant tool in the column towards other components (outgoing dependencies). The last cell sums these outgoing requirements and reflects the tool's significance for the rest platform components.

Table 1 – THREAT-ARREST Platform Tools' Requirements Dependencies

Tools	Assurance	Simulation	Emulation	Gamification	Training	Visualization	Reliance from other tools
Assurance	***14***	**5**	**5**	3	0	1	14
Simulation	3	***13***	**8**	1	4	2	18
Emulation	2	**4**	***8***	1	3	3	13
Gamification	**10**	2	1	***13***	5	5	23
Training	**8**	2	2	4	***6***	0	16
Visualization	2	**10**	7	8	**11**	***12***	**38**
Dependencies towards other tools	**25**	23	23	17	23	11	

The requirements of the *assurance tool* mainly deal with the monitoring of the security properties of the training program and the recording of trainees' actions. It supervises the current CTTP models/programs and assesses the training procedures. The tool is in close cooperation with the rest components, as it collects input from the simulation, emulation, gamification, and training counterparts, and provides data to the visualization one.

The *simulation tool*'s requirements support the modelling of simulated network components and the evaluation of simulated networking scenarios as traces of real/synthetic events occuring at runtime. The requirements also oversee the synchronization of the simulation time with the emulated components and the training session progress.

The requirements regarding the *emulation tool* cover issues for determining the interaction of the emulated components with the users and the potential physical equipment. The emulations must be in concordance with the training process and the simulated scenarios, and the overall CTTP events and outcomes have to be reproducible.

The *gamification tool* requires to produce various types of games. This includes social engineering threats and training of non-security experts that are not easy to be modelled by the emulation/simulation tools. Moreover, the games should support multi-player scenarios and the CTTP evaluation of each individual user.

The *training tool* offers the means to adjust the training process based on the users' profiles and progress, updating accordingly the CTTP programs that are maintained by the assurance tool. It supports real-time interaction with the gamification components as well as synchronous/asynchronous communication with the emulation/simulation counterparts.

Finally, the *visualization tool* must collect information from all the aforementioned components and present the current status of the CTTP training. It is web-based and also supports a scenario definition language that mandates the operation of the other THREAT-ARREST components.

3.2.2 Platform Security Requirements

Given the stage of the project, it is early to design a proper security view. The overall structure of components and their location is crucial for such design. Generally speaking, the THREAT-ARREST platform must utilize current security best-practises.

Platform deployment security requirements. All Internet facing interfaces (e.g. Annex I requirement GT_R_12) must support authentication and encryption. This requirement regards the THREAT-ARREST platform general (own) connections to users and not the specific system being emulated/simulated (e.g. including its security mechanisms or controls). Nodes running such interfaces must reside on a separate network from the rest of the platform. That includes having firewall between those nodes and Internet as well as one between the nodes and the rest of the platform. Those firewalls must be configured in such a way that only necessary ports are opened. Software running on those nodes must be properly configured and maintained. That includes applying security patches and system updates. In case of user facing web interfaces, those interfaces must achieve at least A+ grade on Qualys SSL Labs[1].

Each component should be monitored as such mainly for its availability and utilization. That information must be logged in order to help discover what actually happened and when.

Users should be required to use strong passwords. Usage of second factor authentication should be considered. Users should be divided into several groups with different access rights and their action monitored. All personal data (if any) must be stored in an encrypted storage with access logs. Any user input (e.g. ST_R_10) must be sanitized and treated as possibly hostile (Fysarakis et al., 2014).

GDPR (GDPR, 2016) compliance should be sought for all user interactions and user data collection processes taking part of the THREAT-ARREST platform. During training sessions, user behaviour data from the training process is collected with high implication on user assessment and profiling. As such, the platform shall provide means for data protection by design and by default, and throughout the platform components data flow; and means for user consent and information sheet for the scope and purpose of the data collection taking part at the platform (Hatzivasilis et al., 2015; Papaefstathiou et al., 2015).

User Identity & Access Management (IAM) capability should be offered by the THREAT-ARREST platform. An IAM should offer scalable, to tools' deployment or operation needs, and central to the platform establishment and management unique authentication session, user ID, user Role (i.e. trainee or trainer), and other relevant user or session attributes. All such user session information should be available to all tools. A Single Sign-On realisation of user authentication should be considered to facilitate user access to different tools' interfaces.

In Table 2 we recall security requirements based upon D1.2, while in Annex I the complete set of platform's requirements.

Table 2 – THREAT-ARREST Platform Security Requirements from D1.2

Req-ID	Description	Req Level
AT_R_08	MUST be configurable and support user authentication and authorization	MUST
AT_R_10	Create and store a trace for each administration access to the tool and the associated actions (e.g. changes in settings, access of logs)	MUST

[1] SSL Lab Grading 2018: https://community.qualys.com/docs/DOC-6321-ssl-labs-grading-2018

AT_R_12	For each monitoring session, store the primitive monitoring events used for assurance with a clear record of their producers, contents and their time of occurrence; and the results of the checking of monitoring conditions of different types against these events (e.g. cyber system security monitoring rules, trainee actions monitoring rules)	MUST
AT_R_14	Provide audit functions to allow for the review of the assurance tool functions and configuration integrity checks	SHOULD
ET_R_04	Users can interact with the emulated components and their actions are saved in accessible logs. Enable defend and attack actions by individual users and user groups and the logging of these actions.	MUST
ET_R_06	Supply data on components status	MUST
GT_R_01	Authenticate each user before any action takes place	MUST
GT_R_02	Enforce proof of origin	MUST

3.3 Platform Architecture Data Flow View

We will present the data flow view of the THREAT-ARREST platform. It will give insights of what data flows that exist in the system and among the tools of the platform – from users' interactions with the platform for training and evaluation to simulation and emulation of system behaviour, and to security assurance assessment based on testing and monitoring. It will also reflect the flow of data from administrative platform access to CTTP models' storage and reports generation.

As initial architecture, the data flow view conceptually describes the flow of data within the THREAT-ARREST platform, i.e. among its tools, to perform its training functionality. All processes internal to the platform's tools regarding data transformation/processing are not shown in the figure. An implementation-specific view of the platform architecture will be addressed in the next stage of the project as part of WP6 activities.

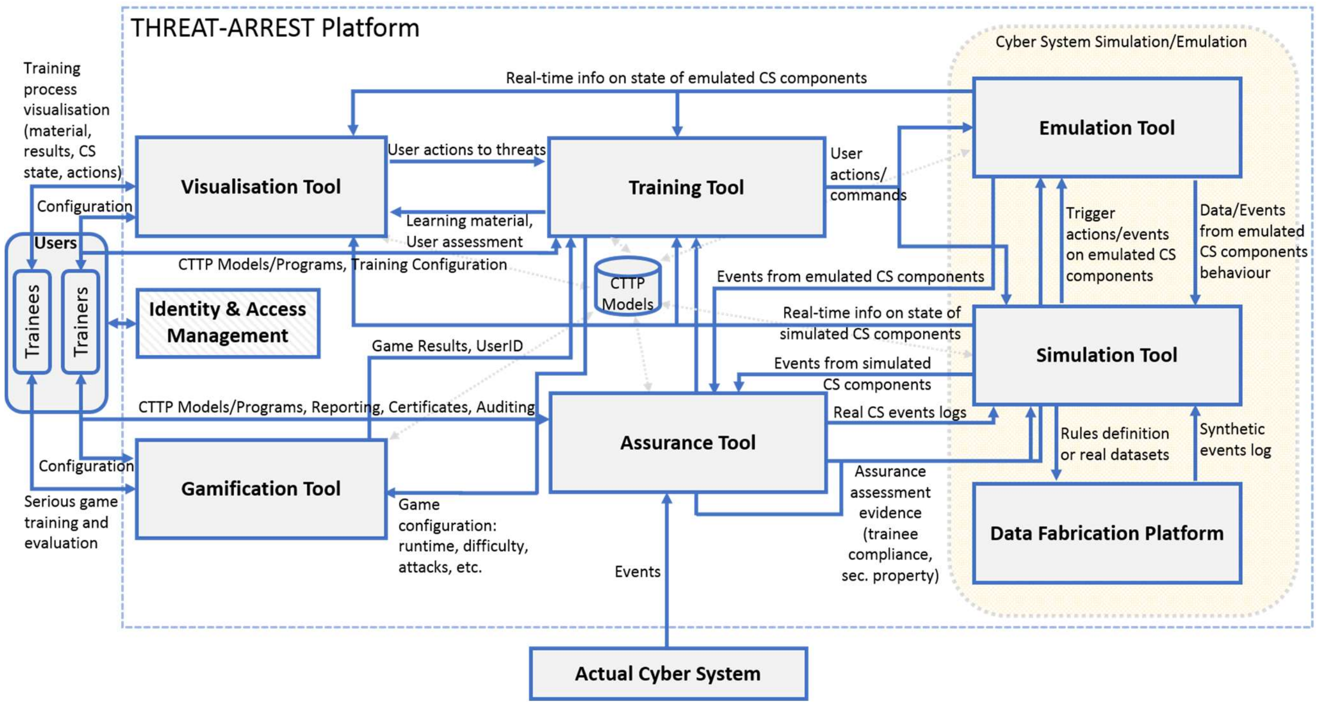

THREAT-ARREST Platform
Cyber System Simulation/Emulation
Training process visualisation (material, results, CS state, actions)
Real-time info on state of emulated CS components
Emulation Tool
User actions to threats
Visualisation Tool
Training Tool
User actions/ commands
Configuration
Learning material, User assessment
Users
Trainees
Trainers
CTTP Models/Programs, Training Configuration
Trigger actions/events on emulated CS components
Data/Events from emulated CS components behaviour
Events from emulated CS components
Identity & Access Management
CTTP Models
Real-time info on state of simulated CS components
Game Results, UserID
Simulation Tool
Events from simulated CS components
CTTP Models/Programs, Reporting, Certificates, Auditing
Real CS events logs
Assurance Tool
Rules definition or real datasets
Synthetic events log
Configuration
Gamification Tool
Serious game training and evaluation
Game configuration: runtime, difficulty, attacks, etc.
Assurance assessment evidence (trainee compliance, sec. property)
Data Fabrication Platform
Events
Actual Cyber System

Figure 4. THREAT-ARREST Platform Architecture High-level Data Flow View

Figure 4 shows the high-level (conceptual) data flow view of the THREAT-ARREST platform. CTTP models are important artefacts for the platform functionality and form an important input to all platform's tools for their proper operation. The architecture considers platform-wide access to CTTP models for the proper operation of the whole platform. In addition, two of the tools – the Assurance Tool and Training Tool – are considered for the specification of CTTP models. In the rest of the document, all tools are assumed to have accessed (obtained) relevant CTTP models.

To facilitate presentation, we will describe the platform data flow view by 4 specific data flows.

Data Flow: Simulation/Emulation of Cyber Systems. The THREAT-ARREST platform's Assurance Tool collects and monitors events coming from the real cyber system subject of simulation and emulation. There is a dedicated component, part of the assurance tool framework, which will be present in the real cyber system to ensure all relevant events are properly captured and communicated to the Assurance Tool.

The Assurance Tool will not only monitor the events from the real cyber system but will also feed those to the Simulation Tool. The Simulation Tool will perform statistical profiling of the real cyber system based on the events received and use these event logs as input to the Data Fabrication Platform. The Data Fabrication Platform will, in turn, extract relevant metadata from the input datasets to initiate the generation of synthetic events which are streamed or input to the Simulation Tool for realist cyber system simulation runs.

As part of realistic cyber system simulation runs, the Simulation Tool may require triggering of specific actions or events on emulated components of the cyber system and receive data/events from the emulated components behaviour, respectively. As such, the Simulation Tool interacts with the Emulation Tool on specific actions or events. In turn, the Emulation Tool will provide back relevant events/data as response to the triggered actions/events.

The Emulation Tool is in charge of emulating system components as identified in the CTTP models.
Both, the Simulation Tool and the Emulation Tool are in charge of feeding back to the Assurance Tool all events triggered by the simulated or emulated components behaviour, respectively. The Assurance Tool, based on the events being received (both from the real cyber system and from the simulated/emulated components of the cyber system) will perform security assurance assessment and provide relevant trainee assessment evidence results to the Training Tool. We note that the assessment evidence (of trainee performance or security property compliance) is also fed back to the Simulation and Emulation Tools for their corrective behaviour of simulation/emulation processes, respectively.

Data Flow: User Training and Evaluation. Users of type trainees interact with the Visualisation Tool as the central platform means for visualisation of training processes. The training process visualisation is interactive and includes, among other things, learning material presentation, results of evaluation assessment of trainees, real time cyber system status visualisation, and actions to be performed on the system as part of the training process. The Visualisation Tool receives data from: i) the Emulation Tool on real-time status of emulated cyber system components; ii) the Simulation Tool on real time status of simulated cyber system components; and iii) the Training Tool on the learning material and trainee assessment reports; All user actions on the visualisation interface of the tool will be fed back to the Training Tool for the corresponding trainee assessment process.

The Training Tool, upon receiving the trainee actions, and as part of evaluation of the trainee performance, will trigger the corresponding actions or commands to simulated/emulated cyber system components. To do so, the Training Tool interacts with the Emulation Tool and/or the Simulation Tool with input the desired actions/commands.

Based on the emulated/simulated components behaviour (events generated in response to the user actions/commands), the Emulation and Simulation Tools will provide real time information on the state of the emulated/simulated components back to the Training Tool, to the Visualisation Tool and to the Assurance Tool, while the Assurance Tool will provide to the Training Tool trainee assessment (compliance) evidence. All that information is used by the Training Tool for the trainee performance evaluation.

Trainees also interact with the Gamification Tools through dedicated to the tool serious games interfaces. The trainees receive extended and specialised training and evaluation by playing the games. As the initial architecture, the Training Tool is envisaged to provide relevant game configuration data (specific to the training process/program) to the Gamification Tool. The Gamification Tool will provide results of the games played by the user (trainee) back to the Training Tool for trainee performance evaluation.

Data Flow: Platform Administrative Access. There are a number of administrative interfaces access to the Visualisation Tool, the Training Tool, the Assurance Tool and the Gamification Tool. These administrative interfaces are restricted to users of type trainers. Trainers specify CTTP Models and Programmes (through a dedicated CTTP Model Editor), configuration parameters for the training process and reporting (e.g. for the Training and Gamification Tools), and configuration settings for the visualisation of the training processes (the Visualisation Tool).

In addition to the trainers' administrative access, there are interfaces envisaged to offer access to dedicated operations/results of the tools. For instance, the Assurance Tool offers dedicated interfaces for Certification Authorities on the certificate generation and issuance process as a result of the assurance assessment performed.

Data Flow: Identity and Access Management. User authentication and access management are considered as integral THREAT-ARREST platform functionalities. The IAM component provides central to the platform establishment and management of unique authentication session, user ID, user Role (trainee/trainer), etc. All platform tools, and in particular the Visualisation Tool, the Gamification Tool, the Training Tool and the Assurance Tool, will request such user session data from the IAM upon user interactions with the tools' interfaces (both functional and administrative). The exact data flow between the tools and the IAM component, and between the IAM component and the users will be detailed in the next stage of project activities. Such data flow will also depend on the adoption of specific off-the-shelf solution for IAM.

3.4 Platform Architecture Component Communication View

We have discussed the high-level data flow view of the THREAT-ARREST platform, with a focus on the data flow among the tools. As initial architecture, it has been also recognised the need of providing more details on the functional components of each tool and identify what type of information are these components exchanging. Such component communication view is an essential step towards a finer-grained specification of the platform and its integration (subject of WP6), as well as the I/O interconnections among the tools (subject of tasks T2.4, T4.6, and T5.4).

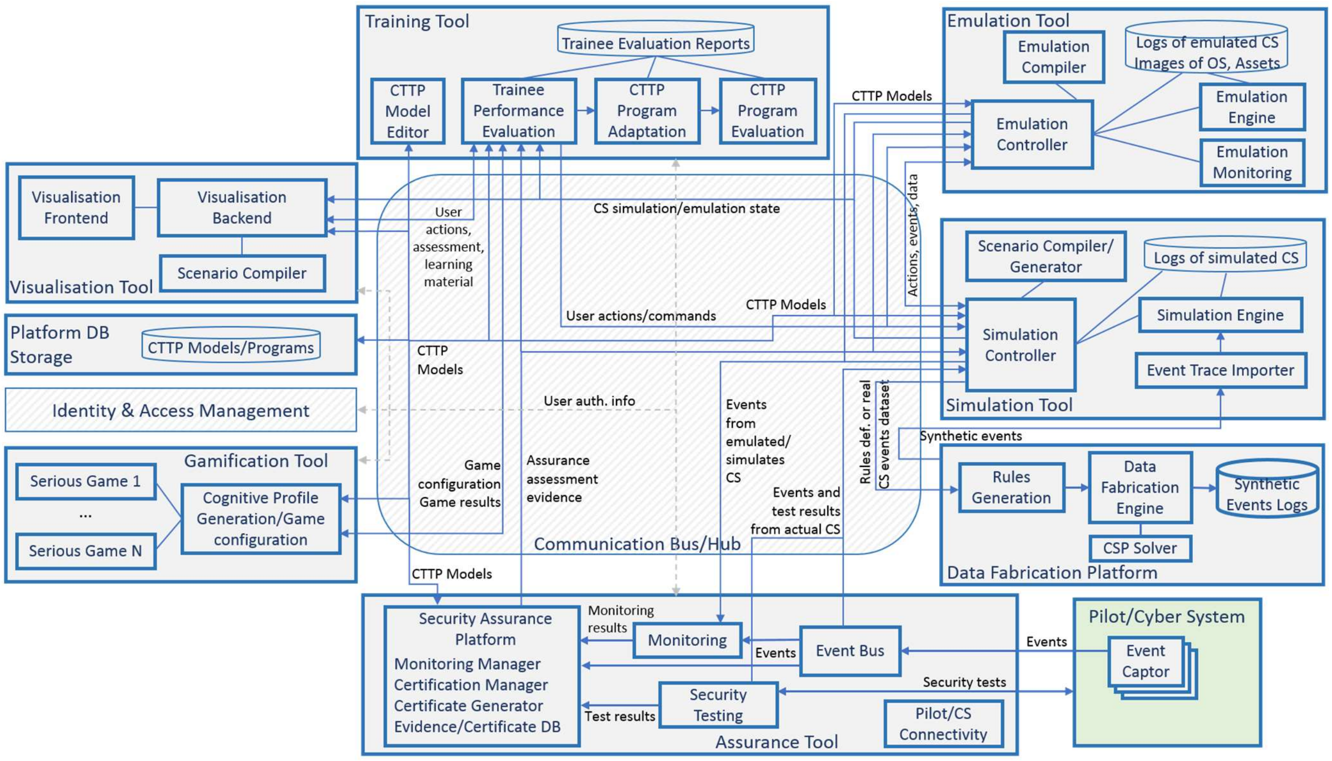


Figure 5. THREAT-ARREST Platform Architecture Component Communication View

Figure 5 shows the component communication view of the THREAT-ARREST platform. It is a communication-centric view of the platform tools and components. In the following, we will describe each tool's components communications with other components of the platform.

Communication Bus. The Communication Bus is a central component of the platform in charge of facilitating and establishing transparent and efficient data communication among all tools. The Communication Bus realisation will be dependent on the underlying platform used to implement the THREAT-ARREST capabilities, for instance the adoption of an open source cyber range solution, and will also depend on I/O specification and interconnection needs of individual tools in tasks T2.4, T4.6, and T5.4. The Communication Bus will be further specified and detailed in the next stage of the project.

Identity & Access Management. The IAM component offers unified to the platform user authentication and identity management solution. The component allows all tools to have access to relevant attributes of user authentication session and roles the user is assigned. The user identifier (user ID) from the authentication session is unique across all tools in the platform. An off the shelf solution for IAM will be used suitable to the needs of the platform. As of the initial architecture, the tools depending on the IAM component are the Visualisation Tool, the Gamification Tool, the Training Tool and the Assurance Tool. All these tools offer functional and administrative interface access restricted to specific types of users.

Platform DB Storage. The platform storage serves the need of THREAT-ARREST for platform-wide access to and management of CTTP models. The availability, integrity and dependency on such storage component are recognised as important properties for a proper platform operation, given the fact that the whole platform functionality is driven by CTTP models. The platform storage can also serve the needs of individual tools to access and manage data (beyond CTTP models) of importance to other platform tools or components.

An important component related to the platform storage is the *CTTP Model Editor*. The CTTP Model Editor assists users, through a dedicated UI, to specify, store and update CTTP Models in the platform storage.

It is recognised the need of a CTTP Model Editor functionality in the Assurance Tool allowing (through administrative access) dedicated users (trainers) specify CTTP models (and programmes) for the different aspects of training, evaluation and assurance assessment, respectively.

Assurance Tool. The Assurance Tool interconnects with a target (pilot) cyber system through a dedicated *Connectivity component* in charge of incapsulating means of connectivity specific for a given cyber system (settings). Given established connectivity, the Assurance Tool receives events from the target cyber system by means of an Event Captor component(s) deployed in the system. Particularly, each *Event Captor* sends events to the *Event Bus* component of the Assurance Tool. The Event Bus component transmits all received events to the *Monitoring component* for monitoring results; to the *Security Assurance Platform* for evidence collection; and to the Simulation Tool for statistical profiling of the cyber system and synthetic event logs generation.

The Monitoring component of the Assurance Tool receives events also from the simulated and emulated components of the cyber system. The Emulation and Simulation Tools are in charge of communicating such events to the Monitoring component.

The Security Assurance Platform, a central component of the Assurance Tool, is in charge of performing assurance assessment and certification on trainees' activities according to a CTTP model's specification, and based on the results of monitoring performed on both the simulated/emulated cyber system components and of the actual cyber system events, and results of security testing performed on the actual cyber system.

The *Security Testing* component performs dynamic security testing on the target cyber system (e.g. (Hatzivasilis et al., 2014) and feeds results of performed tests to both the Security Assurance Platform for security assessment, and the Simulation Tool for more realistic simulations (e.g. regarding the actual effectiveness of security controls). We refer to Section 4 for details on the architecture and components' specification of the Assurance Tool.

Simulation Tool. The Simulation Tool's main component in charge of communicating with the rest of the platform's tools is the *Simulation Controller*. The Simulation Controller makes the simulation functionality of THREAT-ARREST accessible by other components of the platform such as the Training Tool for triggering user actions on simulated cyber system components, or by the Emulation Tool for sending events (data) from the emulated components' behaviour to the simulation runs.

The Simulation Controller also offers results of simulation runs to other components of the platform such as to the Monitoring component of the Assurance Tool on events generated from the simulated cyber system components' behaviour; or to the Emulation Tool for triggering events/actions on emulated cyber system components given the needs of the simulation run; or to the Training and Visualisation Tools for supplying real-time information on the status of simulated cyber system components.

Based on the events received from the actual cyber system operation (through the Event Bus component), the Simulation Controller will perform statistical profiling of the actual cyber system, and communicate with the *Data Fabrication Platform* for the generation of synthetic events' logs needed for the proper execution of the simulation runs.

We refer to Section 5 for details on the architecture and components' specification of the Simulation Tool.

Data Fabrication Platform. The Data Fabrication Platform component plays an essential role for the generation of synthetic events enabling realisation of realistic simulations of the target cyber system. It accepts two main types of input – user defined rules or dataset of real data collection. Based on the selected input, the *Data Fabrication Engine* will produce synthetic data (events). Outcome of the engine will be communicated to the *Event Trace Importer* of the Simulation Tool. The exact means of data transfer between the Simulation Tool and the Data Fabrication Platform will be defined in the next stage of the project. For instance, the Data Fabrication Platform supports, among other formats, data streaming for the outcome of synthetic data generation, which seems more suitable when real time performance of simulation runs is deemed necessary. We refer to Section 10 for details on the architecture and components' specification of the Data Fabrication Platform.

Emulation Tool. The Emulation Tool's central component in charge of communicating with the rest of the platform's tools is the Emulation Controller. The *Emulation Controller* makes the emulation functionality of THREAT-ARREST accessible by other components of the platform such as the Training Tool for triggering user actions on emulated cyber system components, or by the Simulation Tool for triggering events (actions) on the emulated cyber system.

The Emulation Controller also offers results of emulation executions to other components of the platform: i) to the Monitoring component of the Assurance Tool on events generated from the emulated cyber system components' behaviour; ii) to the Simulation Tool on events/data in response to the triggered events/actions (by the Simulation Tool) on the emulated cyber system components; and iii) to the Training and Visualisation Tools for supplying real-time information on the status of emulated cyber system components.

We refer to Section 6 for details on the architecture and components' specification of the Emulation Tool.

Training Tool. Two major functionalities of the Training Tool are the support of CTTP models and programs specification, and the provision of interactive and customised training and performance evaluation.

A CTTP Model Editor is the component in charge of offering suitable GUI to trainers to support the specification of CTTP models, structured content visualisation and verification of conformance to a CTTP model schema. It is important that any CTTP model instance is a well formed CTTP model. An off-the-shelf editor solution will be sought and extended as necessary to allow for proper CTTP model specification and verification. The CTTP Model Editor has the primary role to access the platform storage for storing or modifying CTTP models.

A Trainee Performance Evaluation component is in charge of offering a high level of interactivity with the trainees to deliver training scenarios, enable trainees' actions in response to threats, and offer learning material for the training process. To do so, it communicates with Visualisation Tool for effective visualisation and interaction with the trainees, and with the Gamification Tool for customised serious-game-based training and education on specific attacks such as those based on social engineering. Based on trainees' interactions with the platform's Visualisation and Gamification tools the trainees' performance is evaluated.

The Trainee Performance Evaluation component is also in charge of triggering appropriate commands or actions to the simulated or emulated cyber system components according to the trainees' actions and progress evaluation, and continuously receiving real time status of the simulated or emulated cyber system behaviour in response to trainees' actions/commands and their effects on the system. To do so, it has strong dependencies not only on communications with the Simulation and Emulation Controllers, but also on the Security Assurance Platform for receiving continuous assessment of the trainees' compliance to target system security properties and assumptions.

The Training Tool is also responsible for the adaption and evaluation of CTTP programmes based on trainees' performance assessment and evaluation reports to better support the needs and capabilities of the training process. We refer to Section 8 for details on the architecture and components' specification of the Training Tool.

Visualisation Tool. The Visualisation Tool has two main components addressing different types of communications – the *Visualisation Frontend* and the *Visualization Backend*. The former addresses all visualisation and interactions with users (trainees) while the latter all communications with the platform's tools on training process visualisation and real time status of the simulated/emulated cyber system. Particularly, as of the initial architecture, the Visualisation Backend offers dedicated services for data source visualisation to the Simulation, Emulation and Training Tools for the provision of data visualised to users. These services will allow relevant platform tools to continuously feed data for visualisation to trainees.

Another relevant communication aspect of the Visualisation Tool is the provisioning of a data source service of the tool itself. This service will enable other tools such as the Training Tool to receive notifications on user interactions performed in the Visualization Frontend such as specific user actions in response to threats. We refer to Section 9 for details on the architecture and components' specification of the Visualisation Tool.

Gamification Tool. The Gamification Tool offers serious games for educating people and improve their awareness on critically dependent on human factors decisions regarding cyber-attacks, threat elicitation and organizational defences. The tool will offer dedicated and specific for each game UI to enable trainees engage actively in the training. As of the initial architecture, the UI of each game will come with its own visualization support, and further synergy with the Visualization Tool will be studied to offer unified and integrated visualisation experience to trainees.

The Gamification Tool receives specific game configuration data from the Training Tool such as runtime, level of difficulty, attacks and quest formulation, etc. When a game is finished, the tool reports the user ID and game results back to the Training Tool. The specific data configuration input necessary for the serious games will be subject of further definition with respect to the CTTP model specification developed. We refer to Section 7 for details on the architecture and components' specification of the Gamification Tool.

In the following sections, we will present the specification of each main tool of the platform in terms of its architecture and internal components definition and functionality. We note that as an initial architecture, each tool specification will be further analysed and refined along with the overall architecture as necessary based on further project activities and developments.

4 Assurance Tool Specification

The Assurance Tool is an integrated framework of models, processes, and tools to enable the certification of security properties of services. It uses different types of evidence to demonstrate the support for the required properties and award the corresponding certificate. The types of evidence, which have been envisaged for our framework, include monitoring data, testing data, and the combination of these two. Within the THREAT-ARREST platform, the Assurance Tool will enable the continuous assessment of the security of cyber systems through the combination of runtime monitoring and dynamic testing, in order to provide information about the status of the actual cyber systems. It will also collect runtime system events and generate alerts that provide the basis for setting up realistic simulations.

The role of the Assurance Tool will be to specify: (i) potential attacks, (ii) security controls of cyber systems against those attacks, and (iii) tools that may be used to assess the effectiveness of these security controls.

For the scope of THREAT-ARREST project, this tool will be extended to:

a) Enable continuous monitoring and dynamic testing required for assurance as required by CTTP models and incorporate anomaly detection capabilities and statistical profiling of logged events;

b) Support the generation of synthetic event logs as specified by CTTP models;

c) Support the range of simulations required by CTTP models and interaction with the emulated mechanisms of the THREAT-ARREST platform; and

d) Accommodate advanced scenarios of cyber threats' mitigation and new visualization components.

4.1 Assurance Tool Architecture and Message Flow

The architecture of the Assurance Tool and typical message flows are depicted in the figure below.

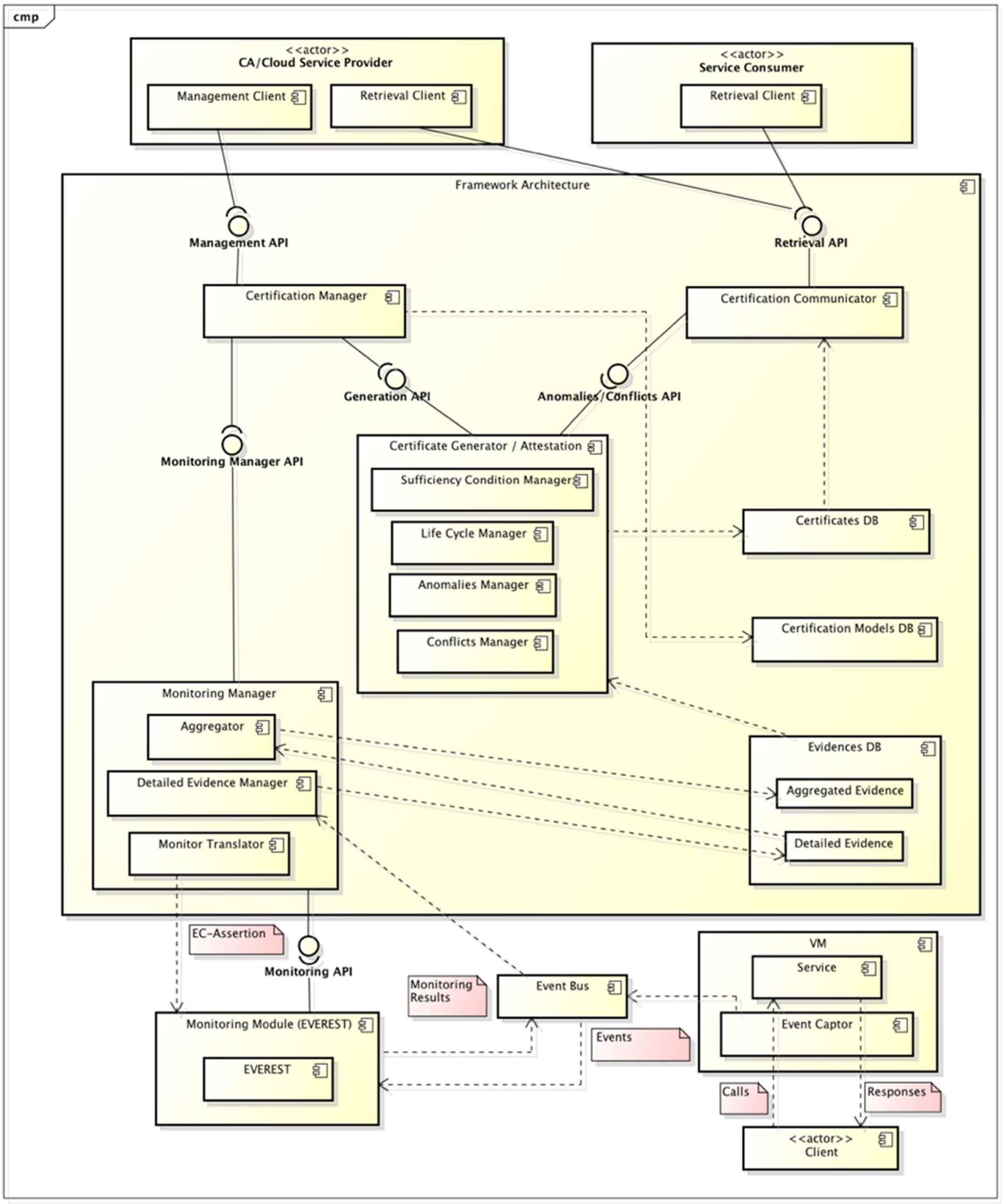


Figure 6. Assurance Tool Architecture and Message Flow

The main components of the Assurance Tool are the following:

- *Monitoring Manager* is the component responsible for initiating, coordinating and reporting the results of the monitoring process.
- *Certification Generator* is the core of the assurance tool and provides management capabilities for its operation, such as sufficiency condition, anomalies, conflicts and life cycle management.

- *Certification Manager's* main functionality is to begin the certification process, receiving requests from the application GUI/dashboard and selecting the appropriate underlying tools to carry out the certification.
- *Certification Communicator* offers the means for getting the generated certification, via the Retrieval API, for both the service consumer and the CA/Cloud Service Provider.
- *Certificates DB***,** that holds the certificates generated via the Certificate Generator.
- *Certification Models DB*, that keeps all the generated Certification/CTTP Models.
- *Evidence DB***,** that holds evidence and detailed evidence aggregated by the Monitoring Manager and used to generate the Certification/CTTP Model.

External components utilized by the Assurance Tool are the following:

- *Monitoring Module (EVEREST)***,** is a runtime monitoring engine built in Java that offers an API for establishing monitoring rules to be checked. The module is made of three submodules: a monitor manager, a monitor and an event collector. The role of the module is to forward the runtime events from application's monitored properties and finally obtain the monitoring results.
- *VMs and other cyber infrastructures*, to enable the continuous assessment of the security of cyber systems through the combination of runtime monitoring and dynamic testing, in order to provide information about the status of the actual cyber systems and support the range of simulations required by CTTP models and the interaction with the emulated mechanisms of the THREAT-ARREST platform.

As input the Assurance tool requires a *Certification Model (CM)/CTTP Model*; this is basically an XML file that contains information regarding the service that needs to be assessed, the security properties and the condition that need to be satisfied, as well as the lift cycle of the assessment process. Additionally, supplementary input from other modules attached to the THREAT-ARREST platform *Communication Bus* for the continuous runtime assessment of the aspects of the target cyber system that are important for CTTP training programme. These include the actual cyber system, and potentially *simulated/emulated parts* of the cyber system as well, depending on the specific deployment scenario analysed.

For output the primary dependence of the Assurance Tool is with the *Simulation Tool*, as the collected monitoring events and testing outcomes form the operational system evidence that is passed over to simulation component to enable statistical profiling and thereby the generation of realistic simulations. Nevertheless, the results and assumptions generated from the assurance tool do affect other aspects of the operation of the THREAT-ARREST platform as well (i.e. all of its key modules) in a direct or indirect way.

Particularly for end-users, the THREAT ARREST dashboard/backend GUI will have access to the Assurance Tool through the Management API, provided by the Certification Manager that can be utilized to initiate the certificate generation via the Generation API and manage the monitoring process via the Monitoring Manager API. Additionally, the provider may use the Retrieval API exposed by the Certification Communicator, to obtain already produced Certificates.

4.2 Assurance Tool Interfaces

Five main interfaces are foreseen for the Assurance Tool; these are: Access Control API, Retrieval API, Management API, Notification API and, potentially, Auditing API. Some details for these interfaces are presented in the table below.

Table 3 – Assurance Tool Main Interfaces

Interface	Type	Description
Access Control API	Offered	It will allow the authentication and authorization of users.
Retrieval API	Offered	It will enable sending requests for certified components, checking the validity of certificates, and getting runtime-related information.
Notification API	Offered	It will deal with subscriptions and notifications for receiving certificates that fulfil the specified requirements at runtime.
Management API	Offered	It will allow to carry out framework management tasks, like adding and updating certification models in the database, as well as requesting to issue a certificate for a specific service.
Audit API	Offered	It will allow the THREAT ARREST auditor to review the certification process supported by the Assurance Tool.

The Management API, as exposed by the Certification Manager module of the Assurance Tool, is the main means through which the Assurance Tool will be interfaced to (and managed from) the THREAT-ARREST platform dashboard. As such, below we focus on some key functions this will support.

Table 4 – Assurance Tool Management Interfaces

Interface/Operation Name	Input Data	Output Data	Description.
getPropertyAndTOCs	-	-	This method returns the Targets of Monitoring and Security Properties that can be certified through the Certification Models present in the Database.
getCertificationModels	TOC identifier (String), The security property identifier (String).	All the CM/CTTP instances that satisfy the request (String).	This method returns all the available Certification Models/CTTP that are related to a specific Security Property and cyber system.
getCertificatesAndModels	-	A list of XML pair, one for the certificate and one for the CM/CTTP instance.	This method returns a list of all Certificates with their related CM/CTTP instances.
getCertificate_Monitoring	Certification identifier (Integer).	The whole certificate document represented in XML (String).	This method returns the xml of a specific certificate.

DeleteCm_Monitoring	Identifier of a CM/CTTP instance (String)	Related message to the delete operation (String), additionally, True if the CM/CTTP was deleted, False otherwise (Boolean)	This method deletes a CM/CTTP instance from the Database.
addCertificationModel	An XML document (String)	New CM/CTTP identifier or an error message (String), True if the CM/CTTP Instance was added, False otherwise (Boolean)	This Method allows to add a new certification model.
submitCertificationModel	CM/CTTP instance identifier (String)	Certificate identifier (Integer)	This method allows to start the monitoring process accordingly to the given CM/CTTP, in order to produce a certificate.
SendMonitoringResults	The results gathered from the monitor for a specific assertion (String).	-	This method is used by the monitor to provide periodically the monitoring results.
startMonitoring	The monitoring assertion that needs to be checked (String)	-	This method allows to start the monitoring process accordingly to the given assertions. The results of the process are going to be provided asynchronously to the caller.

4.3 Technology Supporting Assurance Tool Realisation

The interfacing with the Assurance Tool (i.e. management and retrieval) can be achieved through standard REST APIs.

The main additional module that will be introduced along with the Assurance Tool will be the EVEREST monitoring module (Spanoudakis et al., 2009); a runtime monitoring engine built in

Java that can reason for events against rules. It offers an API for submitting monitoring rules to it for checking. The monitor will forward the runtime events from the cyber systems that are being monitored and obtains the monitoring results, which in turn will be sent to the Assurance Tool (through the Monitoring API defined above). In terms of protocols, the current version of the core system utilizes AMQP (AMQP, 2019), an open standard application layer protocol for asynchronously passing messages between applications, offering its reliability and interoperability, as it provides a wide range of features including publish/subscribe messaging. More specifically, the system uses the RabbitMQ message broker (RabbitMQ, 2019), which supports AMQP, in order to receive (subscribe) or send (publish) events.

5 Simulation Tool Specification

Within the project, discrete event simulation will be used to model the network components and actors in a training scenario. During the simulation run the behaviour of the components in response to actions performed by simulated actors/components as well as users (trainee) will be simulated, assessing the results of these actions on the simulated network.

Based on a scenario description file, the simulation model can be configured to be usable, e.g., by the serious gaming tools.

It will offer means to exchange messages with emulated components during a training session. Furthermore, event traces from the Data Fabrication Platform will be includable in a simulation run. The tool has to send frequent status updates to the visualization component and also has to be able to receive user input/be notified of user actions.

5.1 Simulation Tool Architecture and Interfaces

The intended division of the Simulation Tool in sub-components is shown in Figure 7. The central component of the Simulation Tool is the Simulation Controller. The four components shown in the upper part of the Simulation Tool (Simulation Controller, Scenario Compiler/Generator, Simulation Engine, Event Trace Importer) will typically run within the same process on a dedicated computer.

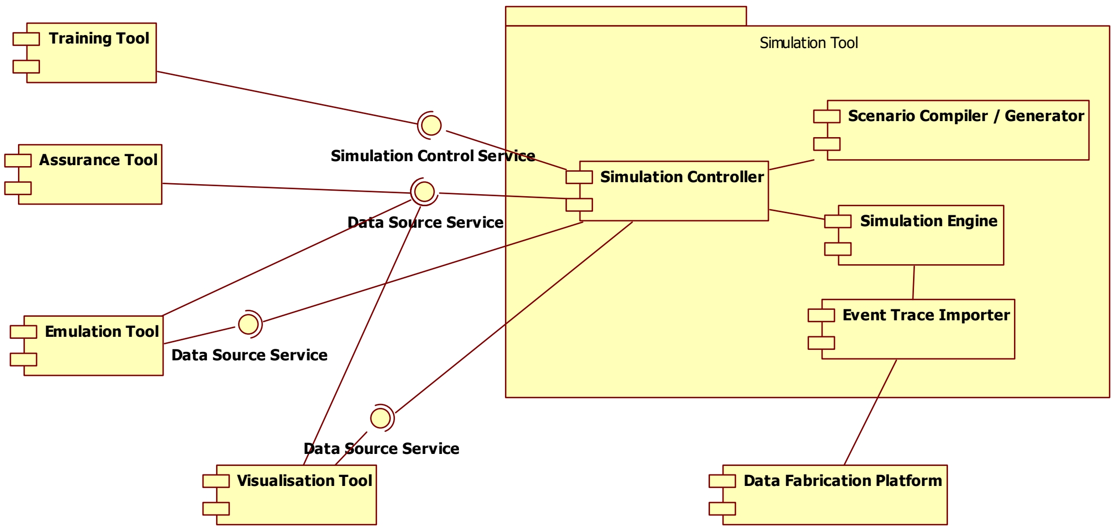


Figure 7. Simulation Tool Architecture and external connections

The *Simulation Controller* provides the central component to make the simulation functionality accessible to the other components of the THREAT-ARREST platform. It is responsible for creating and configuring simulation runs (based on a scenario definition), starting, pausing/continuing and stopping a simulation run via the *Simulation Control Service*. The Simulation Control Service will be used by the training tool.

Via the component *Data Source Service*, it allows other components of the THREAT-ARREST platform to query state variables in the simulation and to be notified when certain simulated events occur. This can be achieved either via a pull mechanism (an external component frequently queries the state it is interested in), or a push mechanism. The push mechanism will use a publish/subscribe mechanism so external components will be notified whenever the value of the variables of interest within the simulation change. This service will be used by the visualization, assurance and training tools.

Similar to the Data Source Service to export information to the other platform components that require it, the Simulation Controller also connects to data sources required as input to the simulation. This will include information from the visualization tool (mostly events triggered by end-users) and events created by emulated components.

Internally, the Simulation Controller uses the *Scenario Compiler/Generator*. This component interprets the scenario description file and creates/configures a simulation instance based on this so it is ready to be executed by the *Simulation Engine*. The Simulation Engine (using SimPlan's Java-based simulation library "jasima" (Jasima, 2019)) is responsible for model execution. It creates the data that is exposed externally by the Simulation Controller. The *Event Trace Importer* is responsible for including real and artificial event traces provided by the *Data Fabrication Platform*, see Section 10, in the simulation run.

5.2 Simulation Tool I/O Dependencies

The data provided and received by the Simulation Tool is summarized in Table 5 and Table 6.

Table 5 – Simulation Tool Input Dependencies

Input dependence	Description
Events from emulated components	Emulation Module
Real and generated event traces	Data Fabrication Platform
User Interactions	Visualization Module or other Platform Components (e.g. from Serious Gaming Module)
Scenario Description File	Part of a central component used to parameterize a training session

Table 6 – Simulation Tool Output Dependencies

Output dependence	Description
Simulation State Messages	Used by e.g. the visualization module to update visualization views
Messages to Emulation	Messages to emulation to trigger emulation events based on simulation component's actions

5.3 Technology Supporting Simulation Tool Realisation

As the simulation engine, the simulation library "jasima" (Jasima 2019) will be used. jasima is a discrete event simulation software library written in Java (its name is an abbreviation of "Java Simulator for Manufacturing and Logistics" indicating its origin). While having some GUI components, its main purpose is to offer a high-performance simulation software library that is fully extendable and easy to integrate in other software. In the THREAT-ARREST project the functionality of jasima will be adjusted to the requirements of a CTTP simulation. It will be accessible to the other components of the system as a back-end service via the Simulation Controller.

The current architecture concept is web-based with loosely coupled components interacting via web protocols. We need synchronous/pull (via REST-API) and asynchronous/push communication (e.g. via WebSockets (RFC 6455, 2011)) to access the simulation service and update/synchronize information in the GUI and the other THREAT-ARREST components. At least for the non-UI components a message queue system like ActiveMQ (ActiveMQ, 2019) could be an option as well. The data exchanged should use JSON as the data format.

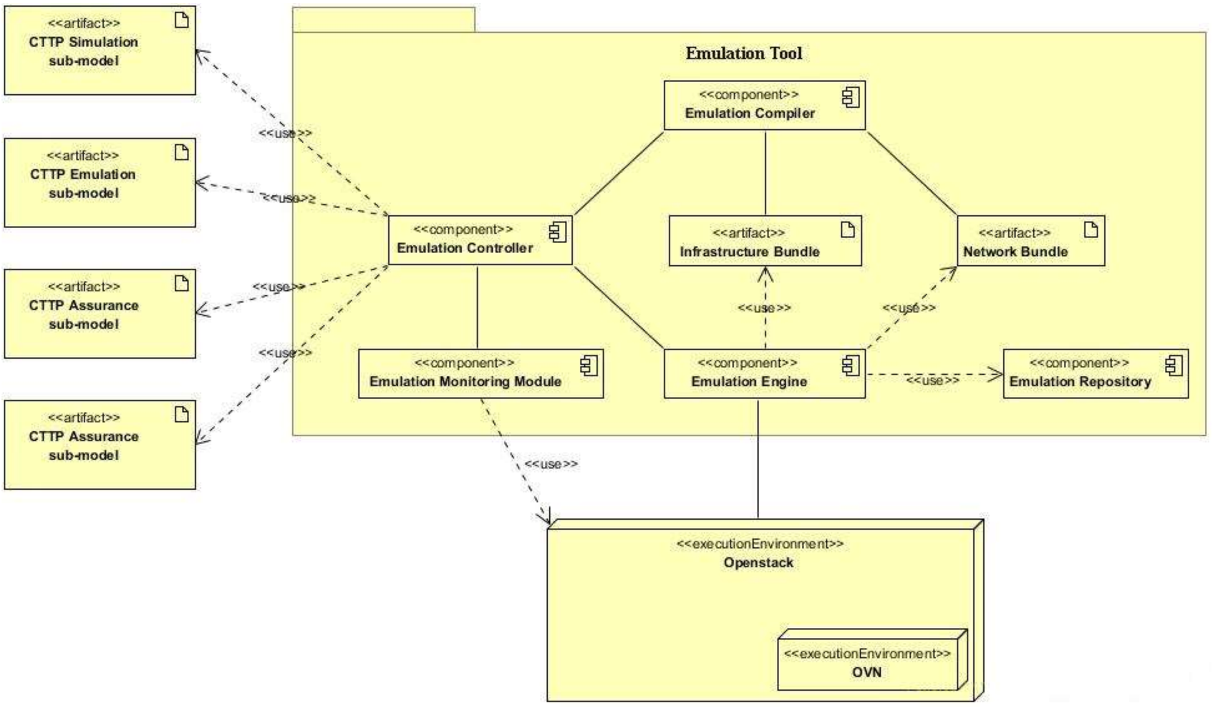
6 Emulation Tool Specification

The tool will support the generation of the emulated environment composed by:

- The VMs that emulate the physical machines that will be the subjects and the actors of the attack scenarios;
- The virtual network components that emulate the gateways, routers, and switches enabling the communication within the emulated environment, replicating the network configuration of the pilots or the actual training system;
- Possible connections with the actual environment through virtual gateway.

6.1 Emulation Tool Architecture and Message Flow

The description of the real-world environment to emulate, in terms of VMs and network nodes and configurations, will be included in the CTTP emulation sub-model that represents the main input of the tool. Figure 8 shows the Emulation Tool architecture and its input dependencies.



<<artifact>> CTTP Simulation sub-model
<<artifact>> CTTP Emulation sub-model
<<artifact>> CTTP Assurance sub-model
<<artifact>> CTTP Assurance sub-model
<<use>>
Emulation Tool
<<component>> Emulation Compiler
<<component>> Emulation Controller
<<artifact>> Infrastructure Bundle
<<artifact>> Network Bundle
<<component>> Emulation Monitoring Module
<<component>> Emulation Engine
<<component>> Emulation Repository
<<executionEnvironment>> Openstack
<<executionEnvironment>> OVN

Figure 8. Emulation Tool Architecture

The CTTP Emulation sub-model will be then processed by the *Emulation Controller*, which acts as the entry point and common interface for the other tools of the system, and orchestrator of the activities of the Emulation Tool. In turn, the Controller will call the *Emulation Compiler* giving as input the received CTTP sub-models, and returning the following artefacts:

- *Infrastructure Bundle*, containing the translation of the description of the VMs in a set of scripts that allow the instantiation of the system in the target environment;
- *Network Bundle*, containing the translation of the description of the network components in the set of scripts needed to virtualize the network infrastructure.

It is important to note that the Emulation Compiler will be specific of the CTTP model language and the emulation/network framework selected for the overall system.

Finally, the Controller will call the *Emulation Engine* giving as input the Infrastructure and Network Bundle, which will be executed in order to instantiate the whole target environment.

In particular, the Emulation Engine will be designed to operate with respect to the following open source frameworks:

- *OpenStack* (OpenStack, 2019), that will manage the creation, execution, and monitoring of the VMs;
- *OVN* (OVN, 2019), the Open Virtual Network for Open vSwitch, which will allow the emulation of the actual network infrastructure.

The architecture includes also the Emulation Repository, that contains the images of generic operating systems, as well as the images of pilot assets to be emulated and specific simulation machine used to execute the attacks. The Emulation Engine will have the task to instance the VMs image as indicated in the CTTP model.

Furthermore, The Emulation Controller will coordinate the execution of the *Emulation Monitoring Module*, that collects data from the status of the overall system and provide other tools with the expected data.

Finally, the Emulation Compiler can take as input the CTTP Simulation and Training sub-models, to integrate in the environment specific simulated components and training configurations and nodes, i.e. VMs specific for the generation of network traffic or misconfigurations of the network components to simulate attacks.

6.2 Emulation Tool I/O Dependencies

High-level description of the input and output dependencies of the Emulation Tool are summarized in Table 7 and Table 8 below.

Table 7 – Emulation Tool Input Dependencies

Input dependence	Description
Description of the nodes to emulate as VM	CTTP Emulation sub-model
Description of the network infrastructure to emulate	CTTP Emulation sub-model
Specific simulation configurations and nodes	CTTP Simulation sub-model
Specific training configurations	CTTP Training sub-model

Table 8 – Emulation Tool Output Dependencies

Output dependence	Description
Simulation Tool	The Simulation Tool will rely on the full instanced emulated environment to simulate. Furthermore, the Tool can require additional configuration and nodes to operate.
Training Tool	The Training Tool will exploit the emulated environment to execute training activities.
Assurance Tool	The training tool will rely on the CTTP Model provided by the Assurance Tool and stored in the system data source.
Visualisation Tool	The Visualisation Toll will use Emulation Tool monitoring data to show the current status of the emulated environment

6.3 Technology Supporting Emulation Tool Realisation

The main technological frameworks underpinning the operation of the Emulation Tool are:

- *Openstack* managing the creation, execution, and monitoring of VMs; and
- *OVN* allowing the emulation of the actual network infrastructure.

7 Gamification Tool Specification

The gamification tool contains two serious games for educating people of the dangers of social engineering attacks. The tool receives exercise configuration data from the THREAT-ARREST platform to:

- Configurations for the run time of a game;
- Data about attacks to be integrated in the games;
- Difficulty level of a game;
- Define further settings of a game.

Note that our games do not store any user information after a game is finished. When a game is finished, the game reports user ID and game result back to the training tool.

We have the following assumptions regarding the THREAT-ARREST platform:

- Identification and Access Management is provided by the THREAT-ARREST platform;
- GDPR (GDPR, 2016) compliance for all user data, e.g. consent of players to process their data, is provided by the THREAT-ARREST platform;
- Storage of persistent user scores and high scores is provided by the THREAT-ARREST platform.

7.1 Gamification Tool Architecture

Figure 9 shows the Gamification Tool architecture high-level view. Two serious games are offered as part of the Gamification Tool component.

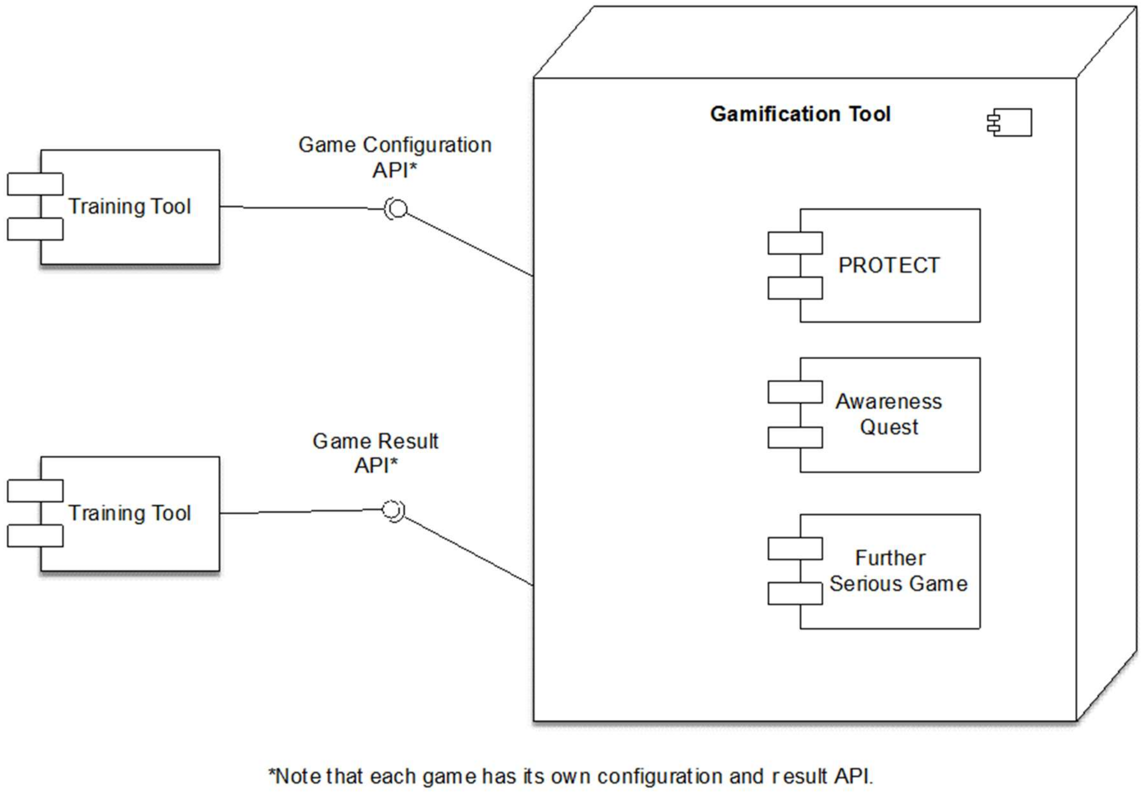


Figure 9. Gamification Tool Architecture and Message Flow

The serious games are started by a user via their own GUI. They are developed using Angular 6 (Angular, 2019) and use REST for all software interfaces. The players can play them on their browsers on PC or on a mobile device. The games support automatic scaling to the type of screen size they are played on. The games have each their own interface, because the data needed to configure them highly depends on the game itself and is almost not generalizable.

Currently, the games PROTECT and Awareness Quest are under development. Both games have their own interfaces. If a decision to develop a further serious game, it would have the same two interfaces.

7.2 Gamification Tool Interfaces

Conceptually, there are two APIs provided/required: one for configuring games e.g. run time, difficulty, user ID, etc. by the Training Tool and one for transmitting the results and the user ID to the Training Tool.

Table 9 – Gamification Tool – Protect Interfaces

Interface/Operation Name	Input Data	Output Data	Description
GameSetting.Protect.playerID	Integer	-	Unique ID of player
GameSetting.Protect.playerName	String	-	Name of the player
GameSetting.Protect.gameTime	Integer	-	Time the game is running
GameConfiguration.Protect.difficulty	Integer	-	Difficulty level
GameConfiguration.Protect.jokers	Integer	-	Number of jokers in the game
GameConfiguration.Protect.attacks	String	-	New attack and defense description
GameResult.Protect.score	-	Integer	Points scored in the game
GameResult.Protect.highscore	-	String	Entire highscore list of all players
GameResult.Protect.playerID	-	Integer	Unique ID of player
GameResult.Protect.playerName	-	String	Name of the player

Table 10 – Gamification Tool – Awareness Quest Interfaces

Interface/Operation Name	Input Data	Output Data	Description
GameSetting.AwarenessQuest.playerID	Integer	-	Unique ID of player
GameSetting.AwarenessQuest.playerName	String	-	Name of the player
GameSetting.AwarenessQuest.gameTime	Integer	-	Time the player can answer questions
GameConfiguration.AwarenessQuest.challenge	Integer	-	Complexity level of the questions
GameConfiguration.AwarenessQuest.Questions	String	-	Add new questions and anserws to the game
GameResult.AwarenessQuest.Correct	-	Integer	The number of correct answers a player provided
GameResult.AwarenessQuest.playerID	-	Integer	Unique ID of player
GameResult.AwarenessQuest.playerName	-	String	Name of the player

We specified the interfaces of the serious games PROTECT and Awareness Quest in Table 9 and Table 10, respectively. We use REST services for the data exchange communicating in a JSON data format.

7.3 Technology Supporting Gamification Tool Realisation

The games PROTECT and Awareness Quest are programmed in Angular 6 and are communication over REST Web services using messages in the JSON format. All configurations and player data can be set and retrieved using REST and JSON. We selected common technologies that are used widespread to ensure compatibility with numerous end user devices and common standards for data exchange in web applications.

8 Training Tool Specification

The purpose of the training tool is to support the definition of CTTP models and programmes, the presentation of learning materials/exercises of CTTP programmes, enable trainee actions in response to cyber threats, interactions with simulated and/or emulated cyber system components, trainee performance evaluation, CTTP programme evaluation and adaptation.

Beyond supporting the definition of CTTP models and programmes, the training tool will also ensure a high level of interactivity with the trainees and deliver the training scenarios, enabling them to respond, sending the appropriate commands to the emulated and simulated components. Also, it will continuously receive information about the status of the emulation and simulation, evaluating in real time the state of progress based on user's responses and their effects on the components and will determine the overall performance of the trainees. The tool will also be responsible for validating the assumptions of the assurance model based on the trainees' responses to the training scenarios and generate warnings in case these assumptions are violated. It will also be able to assess the performance of trainees and evaluate and adapt CTTP programmes. Finally, the tool will collaborate with the visualization tool for the effective delivery of training.

8.1 Training Tool Architecture and Message Flow

The Training Tool will be based on a number of high-level components, as depicted in Figure 10. Based on input provided by the CTTP Model Editor (WP3, STS), and the Components Interaction Hub (in the framework of WP3-WP4-WP5), all necessary data will be aggregated from other components (visualisation, gamification, simulation and emulation) to be exploited towards the real-time assessment of the trainees' performance.

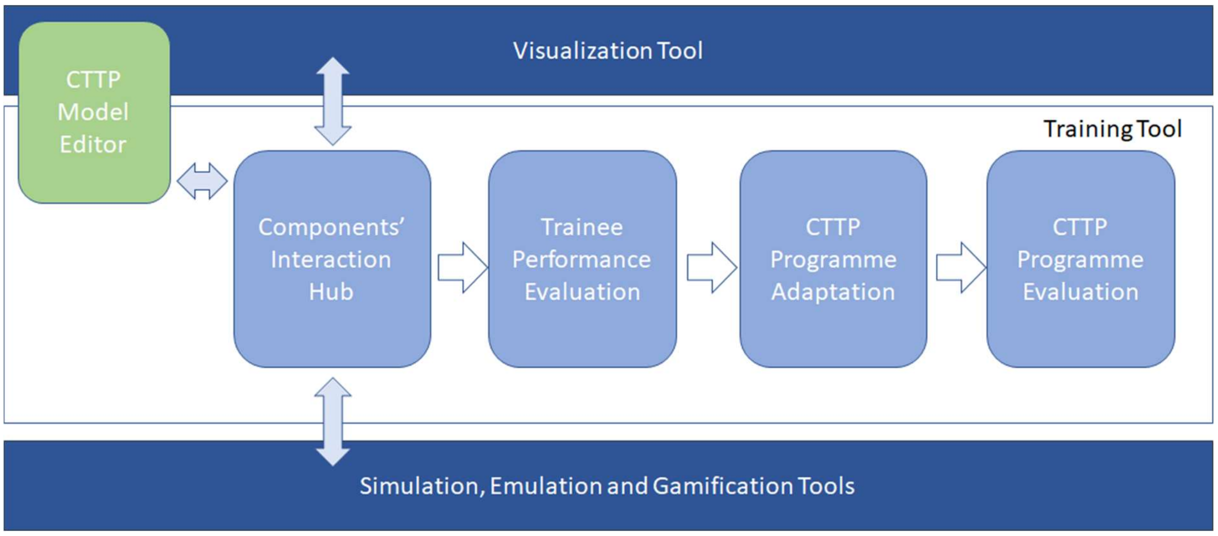

Visualization Tool
CTTP Model Editor
Training Tool
Components' Interaction Hub
Trainee Performance Evaluation
CTTP Programme Adaptation
CTTP Programme Evaluation
Simulation, Emulation and Gamification Tools

Figure 10. Training Tool Architecture and Message Flow

The main components related to the THREAT-ARREST Training Tool can be summarized as follows:

- Components' Interaction Hub (CIH): This module acts as a mediator/proxy between the user/visualization tool and the Simulation, Emulation and Gamification Tools. User interactions are stored for real-time and offline assessment. CIH will also include Authentication & Authorisation services.
- Trainees' performance evaluator: The User Evaluation Component evaluates the performance of users based on their interactions with the interfaces (visualisation and

gamification modules), the pre-defined CTTP models, and the simulation and emulation environments. It also provides real-time assessment outcomes to the trainees.

- CTTP programme adaptation & CTTP programme evaluation: Based on the real-time trainees' assessment as well as the trainees' evaluation reports the CTTP programmes may be adjusted real-time to support the needs and the capabilities of the training process; based on the level at which such needs occur during the training process, thorough evaluations of the CTTP programmes will also become available.

The data flow within the training module is more clearly depicted in Figure 11. CTTP models are going to be stored in a DB focused specifically for the training tool; this DB will provide the necessary details for the CTTP models to the CIH, that will also be fed by the users through the visualisation tool (to capture the progress of the simulation and/or emulation-based training in real time). User evaluation / assessment will be carried out based on both the CTTP models and the real-time information of the trainees' interaction with the system, providing a report to feed the program adaptation process. Based on the continuous programme adaptation, reported and stored to the training tool DB, programmes will be also evaluated.

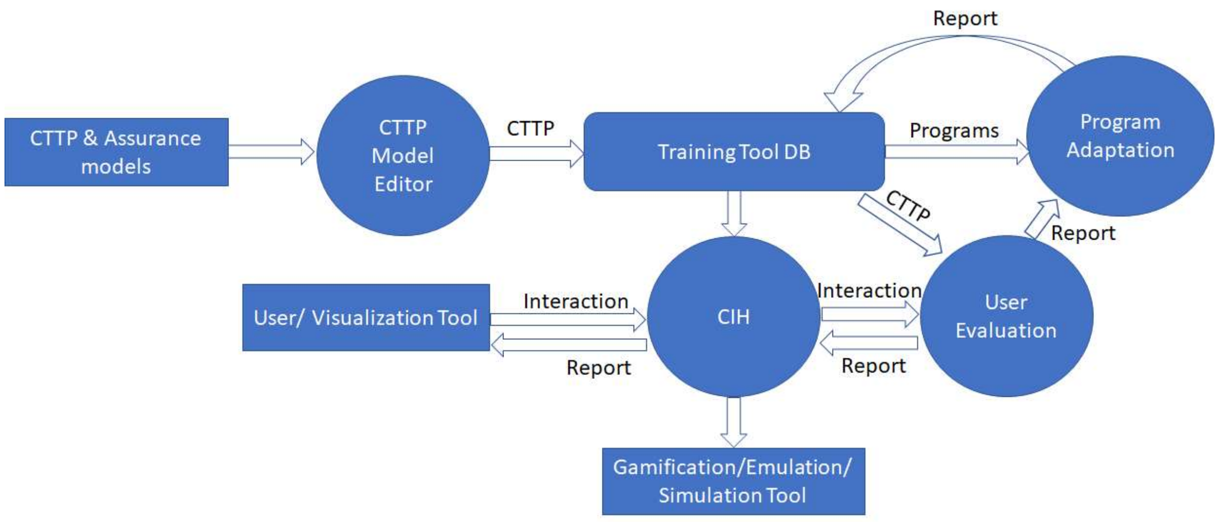


Figure 11 Dataflow Diagram

8.2 Training Tool Interfaces

The data provided and received by the training tool is summarized in Table 11 and Table 12.

Table 11 – Training Tool Input Dependencies

Input dependence	Description
CTTP models	Training models from Assurance tool
Trainees' interactions with Visualization tool	The real-time interactions of the trainees with the platform's interface
Trainees' interactions with Gamification tool	The real-time interactions of the trainees with the platform's gamification module

Table 12 – Training Tool Output Dependencies

Output dependence	Description
Reports on trainees' performance	Used by e.g. the visualization module to provide feedback to the trainees
Reports on training programs' evaluation	Used to deploy reports for suggestions for CTTP models' adaptations

CTTP models' adaptation suggestions	Reports for the continuous updates of the CTTP models to the CTTP Model Editor

In the above, it is crucial to include the CIH: a component that acts as a mediator/proxy between the user/visualization tool and the Simulation, Emulation and Gamification Tools.

8.3 Technology Supporting Training Tool Realisation

The core of the Training Tool implementation is the student and CTTP program evaluation. These components will be inspired by security training programme evaluations and assessment methods found in the literature. Examples are listed below:

- Shimon Y. Nof et al. (2015) approach introduces the basic ideas and main problems that characterize the e-learning network environments from a knowledge management standpoint, focusing on the concept of knowledge and specifically concerns the competencies of those working for organizations. Real-time communications can be used in the project to emulate face-to-face interaction that occurs in learning environments.
- Another research work by Buchmann and Jecan (2008) promotes a non-repudiation system for student evaluation in an e-learning environment based on web services, AJAX frameworks (Garrett 2005) and PEAR packages (PEAR, 2019). Their system implemented XML security standards, provide improved user experience, asynchronous data exchange and message authentication for on-line test papers.
- Samra et al. (2017) formalized a theoretical framework for an interactive e-training system. They took into consideration e-training system requirements, focusing on applying cloud technologies to ensure the dynamic scalability of services and computing power while maintaining QoS and security.

The Training Tool will be based on open technologies, utilizing RESTful APIs and JSON for the asynchronous interchange of data with the other components & layers of the THREAT-ARREST architecture. Moreover, for both the needs of synchronous (i.e. with the Visualization & Gamification components) and asynchronous communication, TLS will be implemented so as to ensure privacy and data security.

For the use of the Training Tool's specific database, a NoSQL DB (probably MongoDB (MongoDB, 2019)) will be used, so as to take advantage of its flexible document data model. The protocols and technologies will be further clarified and aligned with the rest of the architecture, upon finalization of the rest of the interconnected components.

9 Visualisation Tool Specification

The Visualization Tool will be responsible for the representation of the status of the simulated and emulated components and the effects of the training actions on them. It will enable the trainees to have a clear view of the cyber-system status and attacks mounted against it, which will be updated in real-time. It will also provide information about their assessment status and enable real-time interaction.

The visualization mechanisms will cover all the layers in the implementation stack of a cyber-system, use appropriate visualization metaphors for different types of attacks and system components, enable zoom-in and zoom-out views over the system, and be interactively controlled by the user. In summary, the Visualization Tool will be able to parse and make use of visualization scenarios contained in or referenced by the CTTP models. The Visualization Tool will be able to display textual data, images, tabular data as well as diagram types such as line/bar charts, Sankey diagrams, and Gantt charts.

9.1 Visualisation Tool Architecture and Interfaces

The Visualization Tool consists of a backend and frontend part. Its main components and connections to other parts of the THREAT-ARREST platform are shown as a UML component diagram in Figure 12.

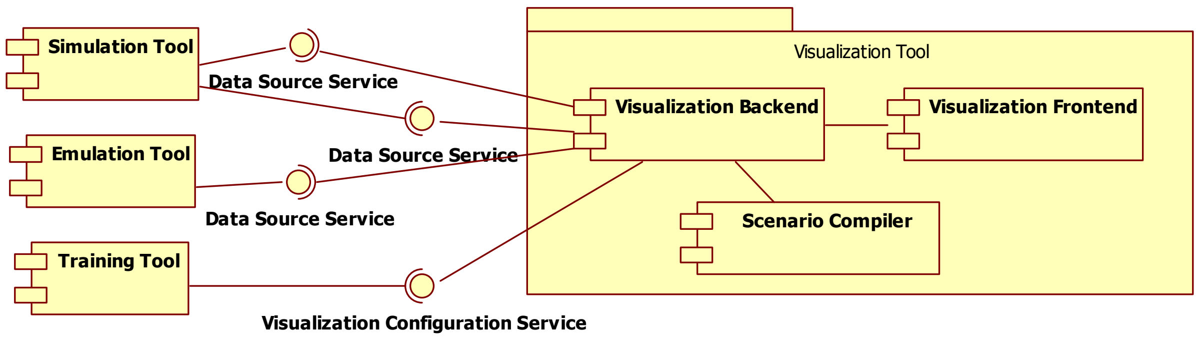


Figure 12. Visualisation Tool Architecture and Interfaces

The backend runs as a (Web-) service and is responsible for configuring visualization views for a training session defined in a scenario description file via the “Visualization Configuration Service”. This will create the necessary view elements and connect them with the respective data sources. Parsing and generating the implementing view elements will be performed internally by the “Scenario Compiler”.

The Visualization Tool also provides a data source itself, so other components can be notified of user interactions performed in the views.

The graphical representation of the user interface is provided by the visualization frontend. This will run in a web browser (using HTML5 (HTML5, 2019), JavaScript) on the computer or mobile device of a user and allows embedding a visualization view in web-based user interfaces of other components as frames or Web Components (WebComponents, 2019).

The backend is responsible for connecting and frequently pulling the data sources used in a visualization view. If the data source offers a push mechanism, then the backend will subscribe to the data source to be notified whenever relevant changes occur. Once updated data is received, this will be propagated to the front-end so associated GUI components can be updated.

If technically possible, the process of accessing the data sources might also be performed by the front-end directly bypassing the back-end, so the additional step of passing data through the back-end might be avoidable.

9.2 Visualisation Tool I/O Dependencies

The data provided and received by the Visualisation Tool is summarized in Table 13 and Table 14.

Table 13 – Visualisation Tool Input Dependencies

Input dependence	Description
Events from connected data sources	Messages to update the view, i.e., the information displayed.
User Interactions	User interactions (clicks, text input) will be received and routed to target components.
Visualization Scenario Description File	Part of/delivered by a central component used to configure the visualization. It is used to define hierarchies of visualization views, their elements and the data sources to connect with and to link these view elements with.

Table 14 – Visualisation Tool Output Dependencies

Output dependence	Description
Data Source subscriptions/requests	Used to connect to other components to get the information to be displayed. This information can be requested in a kind of pull scheme but is preferably sent by the other components via a push message (subscription mechanism) to achieve an up-to-date visualization also for dynamic scenes.
User interaction notifications	Used to notify components of user interactions in the Visualization Tool.

9.3 Technology Supporting Visualisation Tool Realisation

The architecture envisaged is web-based with loosely coupled components interacting via web protocols. We need synchronous (via a REST API) and asynchronous communication (e.g. via WebSockets (RFC 6455, 2011)) to access the simulation service and update/synchronize information in the GUI. At least for the non-UI components a message queue system like ActiveMQ could be an option as well. The messages should use JSON.

The front-end of the visualization tool will use HTML5 and JavaScript (e.g. JavaScript-based charting libraries like D3.js (D3.js, 2019)) and encapsulate visualizations as WebComponents.

10 Data Fabrication Platform Specification

The IBM Data Fabrication Platform is an officially supported IBM product with established customer base.

The THREAT-ARREST Simulation Tool can leverage the IBM Data Fabrication Platform in two innovative ways:

- Fabricate a rich and consistent set of high-fidelity system log files and other artefacts mimicking real cyber-attack events and actions.
- Fabricate a rich and consistent set of parameters and properties to guide and control an external attack simulation tool.

10.1 Data Fabrication Platform Architecture and Message Flow

The methodology used is termed "rule guided fabrication". In rule guided fabrication, the user:

1) Either specifies the desired data structure or lets the tool extract metadata from a target database;
2) Specifies rules, reflecting the desired data properties and relations.

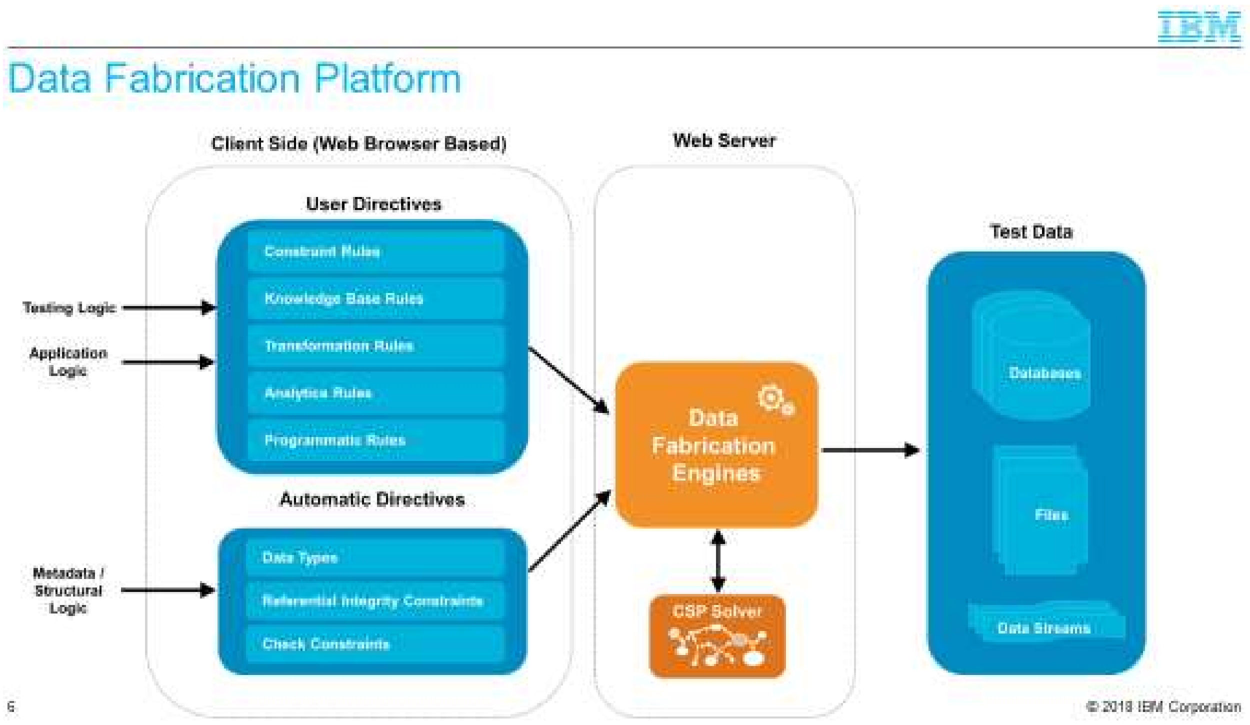


Figure 13. Data Fabrication Platform Architecture High-level View

Figure 13 shows a high-level view of the Data Fabrication Platform architecture. In brief, the Data Fabrication Platform is a 3-tier Web application. A user operates the tool via client-side graphic user interface, which is rendered by a Web browser. User commands and directives are served by the tool's business logic – Data Fabrication Engine – deployed on a Web Server. The Engine leverages the Constraint Satisfaction Problem (CSP) Solver as a back-bone problem solving technology. Finally, the fabricated data is stored at the data layer – as a set of data records populated into a database or a data file – or streamed out over a data stream.

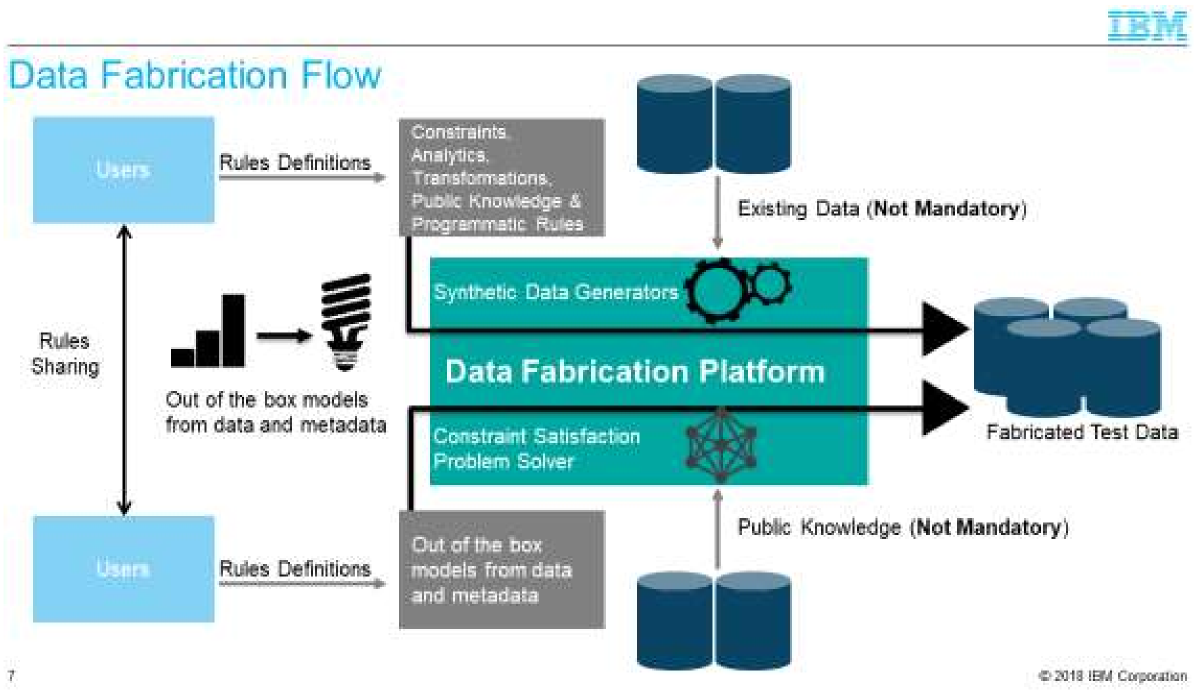
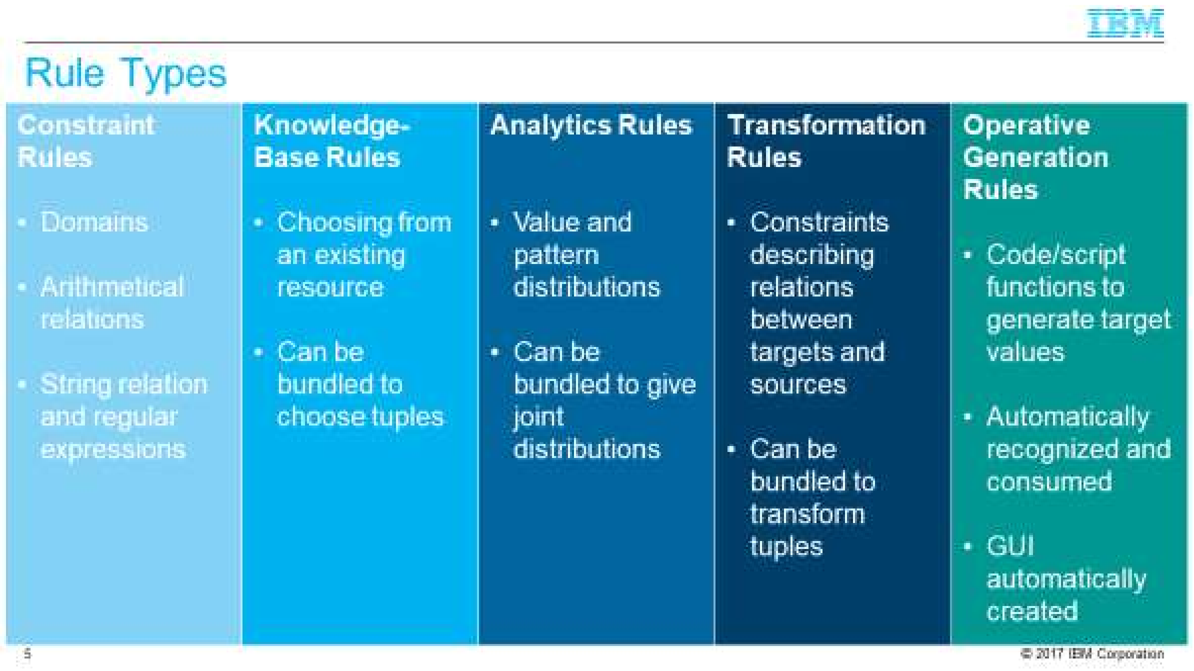

IBM
Data Fabrication Flow
Users
Rules Definitions
Constraints, Analytics, Transformations, Public Knowledge & Programmatic Rules
Existing Data (Not Mandatory)
Rules Sharing
Out of the box models from data and metadata
Synthetic Data Generators
Data Fabrication Platform
Constraint Satisfaction Problem Solver
Fabricated Test Data
Public Knowledge (Not Mandatory)
Users
Rules Definitions
Out of the box models from data and metadata
7
© 2018 IBM Corporation

Figure 14. Data Fabrication Platform Workflow

Figure 14 shows the workflow of data fabrication. Once the user requests the generation of a certain amount of data the Platform internally ensures that the generated data satisfies the specified structure, database consistency and other constraints as well as the user-provided rules. User rules and other directives are sent over to the Data Fabrication Engine. The Engine translates a user problem into a mathematical model –CSP – and sends it down to the CSP Solver to solve. The problem solution essentially is a consistent set of data values assigned to the problem variables, while the values satisfy all the specified constraints. Finally, the Engine inserts the fabricated data into a target database, writes it into a target file or sends it over to a data stream.



IBM

Rule Types

Constraint Rules	Knowledge-Base Rules	Analytics Rules	Transformation Rules	Operative Generation Rules
• Domains • Arithmetical relations • String relation and regular expressions	• Choosing from an existing resource • Can be bundled to choose tuples	• Value and pattern distributions • Can be bundled to give joint distributions	• Constraints describing relations between targets and sources • Can be bundled to transform tuples	• Code/script functions to generate target values • Automatically recognized and consumed • GUI automatically created

5 © 2017 IBM Corporation

Figure 12. Data Fabrication Platform Rule Types

Figure 12 shows rule types that user can use to guide the fabrication process to gain desired data properties and/or data field relationships.

10.2 Data Fabrication Platform I/O Dependencies

IBM Data Fabrication Platform is an interactive stand-alone web application. It requires manual definition of data structures along with user directives (rules) to perform its operations.

IBM Data Fabrication Platform is able to populate the output data into relational databases or files. Data-dependent applications, such as the THREAT-ARREST Simulation Tool, may consume the output data accessing the database and files directly with no interaction with the Platform.

IBM Data Fabrication Platform has a clear 3-tier architecture. The core technology is implemented in Java programming language. Both the specification and fabrication functionalities are available via a set of Java and REST APIs, so an external client application can leverage the tool's capabilities directly via these APIs. In addition, the Platform's data writer module may be easily extended to directly drive external data consumers.

The tool interfaces, external dependencies and means of integration within the THREAT-ARREST Simulation tool architecture will be defined further in the project activities.

10.3 Technology Supporting Data Fabrication Platform

IBM Data Fabrication Platform is deployed as a distributed web-based application, running under Apache Tomcat web server. An underlying CSP Solver is deployed as a stand-alone Linux application. The Platform supports major relational databases (e.g. DB2, Oracle, MS SQL Server, PostgeSQL, and SQLite) and multiple file formats.

11 Conclusions and Next Steps

We have presented the THREAT-ARREST initial reference architecture. Particularly, we have extended the architecture identified in the DoA by introducing two specific views – the Data Flow View and the Component Communication View. The data flow view provided insights on data flows in the system and data dependencies among the tools of the platform, while the component communication view provided fine-grained details on functional components of each tool and what these components interact for with the other tools.

Following the overall architecture, we have detailed the specification of each main tool of the platform, discussed its architecture, the components forming part of the architecture and their role, and interfaces or I/O dependences on other tools of the platform.

Next steps related to architecture activities will focus on:

- Identity and Access Management component specification, available off-the-shelf solutions, and data flow with other tools of the platform.
- Communication Bus (hub) specification addressing requirements of the tools on communications and message/data exchange.
- I/O mechanisms and interfaces specification addressing platform tools' interoperation needs.
- Platform security view specification addressing security requirements of the tools.

The steps above will lead to a finer-grained specification of the platform and its component integration which will be subject of WP6.

References

ActiveMQ, 2019. Apache ActiveMQ. http://activemq.apache.org/ [26 February 2019]

AMQP, 2019. Advanced Message Queuing Protocol (AMQP). http://www.amqp.org/ [26 February 2019]

Angular, 2019. https://angular.io/ [26 February 2019]

Bellard F, 2005. QEMU, a Fast and Portable Dynamic Translator, USENIX Annual Technical Conference, FREENIX Track. Vol. 41. Available from: https://www.qemu.org

Buchmann R. A. and Jecan S., 2008. An Arbitration Web Service for E-learning based on XML Security Standards.

D3.js, 2019. https://d3js.org/ [26 February 2019]

Fysarakis, K., et al., 2014. Embedded systems security challenges. Measurable security for Embedded Computing and Communication Systems (MeSeCCS 2014), within the 4th International Conference on Pervasive and Embedded Computing and Communication Systems (PECCS 2014), 7-9 January 2014, Lisbon, Portugal, pp. 1-10.

Garrett J. J., 2005. "Ajax: A New Approach to Web Applications". http://adaptivepath.org/ideas/ajax-new-approach-web-applications/ [26 February 2019]

GDPR, 2016. European Parliament, Council of The European Union: Regulation (EU), 2016/679 General Data Protection Regulation (GDPR). http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679 [26 February 2019]

G. Spanoudakis, C. Kloukinas, and K. Mahbub, 2009. “The SERENITY Runtime Monitoring Framework”, In Security and Dependability for Ambient Intelligence, Springer, pp. 213-238.

Hatzivasilis, G., et al., 2014. A reasoning system for composition verification and security validation. 6th IFIP International Conference on New Technologies, Mobility & Security (NTMS 2014), IFIP, 30 March – 2 April 2014, Zayed University, Dubai, UAE, pp. 1-4.

Hatzivasilis, G., et al., 2015. Lightweight password hashing scheme for embedded systems. 9th WG 11.2 International Conference on Information Security Theory and Practice (WISTP), IFIP, Heraklion, Crete, Greece, 24-25 August 2015, Springer, LNCS, 9311, pp. 249-259.

HTML5, 2019. https://www.w3.org/TR/html5/ [26 February 2019]

Jasima, 2019. Jasima: Java Simulator for Manufacturing and Logistics. SimPlan AG. https://www.simplan.de/en/software/jasima/ [26 February 2019]

MongoDB, 2019. MongoDB: Open Source Document Database. https://www.mongodb.com/ [26 February 2019]

Nof S. Y. et al., 2015. Revolutionizing Collaboration through e-Work, e-Business, and e-Service, e-Learning and e-Training pp. 357-390.

OpenStack, 2019. https://www.openstack.org/ [26 February 2019]

OVN, 2019. Open vSwitch, http://openvswitch.org/ [26 February 2019]

Papaefstathiou, I., et al., 2014. International Conference on Advanced Technology & Sciences (ICAT 2014), 12-15 August 2014, Antalya, Turkey, pp. 1-9.

PEAR, 2019. PEAR Packages – PHP. https://pear.php.net/packages.php [26 February 2019]

RabbitMQ, 2019. RabbitMQ: An open source messaging broker. https://www.rabbitmq.com/ [26 February 2019]

RFC 6455, 2011. RFC 6455 - The WebSocket Protocol. Internet Engineering Task Force (IETF), https://tools.ietf.org/html/rfc6455 [26 February 2019]

Samra H. E., Li A. S., Soh B., and AlZain M. A., 2017. "A Cloud-Based Architecture for Interactive E-Training," *International Journal of Knowledge Society Research (IJKSR) 8 (2017)*: 1, [26 February 2019], doi:10.4018/IJKSR.2017010104 Available at https://www.igi-global.com/article/a-cloud-based-architecture-for-interactive-e-training/181262

SRIA-cPPP, 2019. European Cyber-Security Strategic Research and Innovation Agenda (SRIA) for contractual Public-Private Partnership (cPPP). Available from: https://www.ecs-org.eu/documents/ecs-cppp-sria.pdf [26 February 2019]

THREAT-ARREST DoA, 2018. THREAT-ARREST Grant Agreement Annex I – "Description of Action" (DoA).

THREAT-ARREST D1.1, 2018. The pilots' requirements analysis report. THREAT-ARREST Project deliverable D1.1, https://www.threat-arrest.eu

THREAT-ARREST D1.2, 2018. The platform's system requirements analysis report. THREAT-ARREST Project deliverable D1.2, https://www.threat-arrest.eu

VirtualBox, 2019. VM VirtualBox. Available from: https://www.virtualbox.org/ [26 February 2019]

VMware, 2019. Server Virtualization Software – vSphere. Available from: https://www.vmware.com/products/vsphere.html [26 February 2019]

WebComponents, 2019. https://www.webcomponents.org/ [26 February 2019]

Annex I: THREAT-ARREST Platform Requirements

The annex summarizes the platform requirements as they were documented in the related deliverable D1.2. The requirement level (i.e. MUST/SHOULD) is in accordance with the RFC 2119 – IETF standard convention[2]. In total, there are **73 architectural requirements**, where 53 of them are mandatory (MUST) while 20 of them are optional (SHOULD). The next table details these requirements.

Table 15 – Architectural requirements

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
Assurance Tool			
AT_R_01	Provide support for the monitoring of all security properties of the target cyber-system and the emulated/simulated versions of it used in CTTP training programs, as long as the latter can be monitored	MUST	Simulation Tool (ST_R_06), Emulation Tool (ET_R_06), Training Tool (TT_R_06), AT_R_02
AT_R_02	Provide support for the monitoring of actions of trainees, who are also users of the target cyber-system, that are related to security properties of the target actual cyber system (e.g. compliance to security restrictions)	MUST	Piloting (actual cyber system) environment connectivity
AT_R_03	Provide support for monitoring security-related actions of trainees against the target cyber-system before and after the training to enable an evaluation of the effectiveness of the training	MUST	AT_R_02
AT_R_04	Provide support for monitoring conditions related to assessing the level of compliance of the trainee actions to expectations set by the security assurance sub-model of the CTTP model, as extracted by the CTTP model translation	MUST	Simulation Tool (ST_R_01), Emulation Tool (ET_R_01), Training Tool (TT_R_02), Gamification Tool (GT_R_03)
AT_R_05	Provide support for security properties assessment from both the actual targeted cyber system and the simulated/emulated versions of it used in training	MUST	Simulation Tool (ST_R_02), Emulation Tool (ET_R_02), Gamification Tool (GT_R_04), Training Tool (TT_R_02), AT_R_06

[2] RFC 2119: https://www.ietf.org/rfc/rfc2119.txt

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
AT_R_06	Support the collection of assurance assessment evidence and make it available to other layers of the THREAT-ARREST platform	MUST	Simulation Tool (ST_R_06), Emulation Tool (ET_R_06), Gamification Tool (GT_R_04), Training Tool (TT_R_06), Visualisation Tool (VT_R_02), AT_R_02
AT_R_07	Support the monitoring of conditions involving events collected from different layers of the THREAT-ARREST platform	SHOULD	AT_R_06
AT_R_08	MUST be configurable and support user authentication and authorization	MUST	-
AT_R_09	Provide a set of assurance assessment support administration functions, including the retrieval of the collected assurance assessment evidence (i.e., events) and the specification of rules to be used for security assurance assessment	MUST	-
AT_R_10	Create and store a trace for each administration access to the tool and the associated actions (e.g. changes in settings, access of logs)	MUST	AT_R_08, AT_R_09
AT_R_11	Provide the following assurance assessment functions: specification of the target cyber system to be assessed, specification of the monitoring and testing interfaces that may be used for assurance assessment, specification of conditions regarding trainee actions to that need to be monitored, specification of restrictions regarding the accessing of evidence collected through the assessment process	MUST	Simulation Tool (ST_R_01), Emulation Tool (ET_R_01), Training Tool (TT_R_02), AT_R_06
AT_R_12	For each monitoring session, store the primitive monitoring events used for assurance with a clear record of their producers, contents and their time of occurrence; and the results of the checking of monitoring conditions of different types against these events (e.g. cyber system security monitoring rules, trainee actions monitoring rules)	MUST	AT_R_06, AT_R_07, AT_R_08, AT_R_11
AT_R_13	Produce auditable assurance assessment results, including digital certificates (where appropriate), based on the evidence collected	MUST	AT_R_12
AT_R_14	Provide audit functions to allow for the review of the assurance tool functions and configuration integrity checks	SHOULD	AT_R_10, AT_R_13
Simulation Tool			

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
ST_R_01	Allow the definition of simulation scenarios consisting of relevant network components by parameterizing scenario templates predefined for training	MUST	ST_R_04, Emulation tool (ET_R_03)
ST_R_02	Offer a library of simulated network components (modelling their structure and required behaviour)	MUST	-
ST_R_03	Components in the component library should include actors in a training scenario (attacker, defender, user) as well as relevant communication network/IT components; their behaviour will be specified primarily by rules describing their reactions to relevant input events	SHOULD	Assurance tool (AT_R_01, AT_R_02), Gamification tool (GT_R_01), Emulation tool (ET_R_03, ET_R_04)
ST_R_04	Allow scenario templates to be defined using, connecting and parameterizing components from the simulation library	MUST	ST_R_02
ST_R_05	Allow the creation of a simulation run given a simulation scenario definition	MUST	ST_R_01
ST_R_06	Allow triggering actions/events in the emulation component	SHOULD	Emulation tool (ET_R_04, ET_R_07)
ST_R_07	Receive and act upon events received from emulation	SHOULD	Emulation tool (ET_R_04, ET_R_07)
ST_R_08	Import and use synthetic and real event logs	MUST	Data Fabrication Platform
ST_R_09	Provide real-time information to users of the system about the current state of the simulation (usually displayed via the visualization component)	MUST	Visualization tool (VT_R_01, VT_R_07)
ST_R_10	Receive and process user input (interactive simulation)	MUST	Training tool (TT_R_05, TT_R_06)
ST_R_11	Alter the behaviour of simulated components/networks based on user input	MUST	Assurance tool (AT_R_02), ST_R_10
ST_R_12	Synchronize simulation time with emulated components and training session progress	SHOULD	Emulation tool (ET_R_07), Training tool (TT_R_02, TT_R_05)
ST_R_13	Ensure repeatability and randomness. Every execution of a scenario, using basic configuration with the same input, should produce the same results. At the same time, some randomness should be ensured by modifying the initial configuration/input, in order the results not be identical	MUST	ST_R_05
Emulation Tool			
ET_R_01	Emulation sub-model of CTTP model will drive the definition of the emulated network and components	MUST	Simulation tool (ST_R_01)

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
ET_R_02	Align the training process with operational cyber-system security assurance mechanisms	SHOULD	Assurance tool (AT_R_01)
ET_R_03	The emulation tool will be enable to install software and communicate with external physical components as defined in the Emulation sub-model	MUST	Assurance tool (AT_R_02), Simulation tool (ST_R_03)
ET_R_04	Users can interact with the emulated components and their actions are saved in accessible logs. Enable defend and attack actions by individual users and user groups and the logging of these actions.	MUST	Training tool (TT_R_02)
ET_R_05	Support the interaction with trainees of the CTTP program	SHOULD	Gamification tool (GT_R_13), Training tool (TT_R_05)
ET_R_06	Supply data on components status	MUST	Training tool (TT_R_02), Visualization tool (VT_R_01, VT_R_02, VT_R_08)
ET_R_07	Support the propagation of data and other stimuli generated by emulated components to other (simulated or emulated) parts of a cyber-system	MUST	Simulation tool (ST_R_06)
ET_R_08	Ensure reproducibility. The same configuration with the same input and emulated components should have the same behaviour.	MUST	Simulation tool (ST_R_13)
Gamification Tool			
GT_R_01	Authenticate each user before any action takes place	MUST	-
GT_R_02	Enforce proof of origin	MUST	-
GT_R_03	Provide games that are driven by the threats/assumptions which are specified in the CTTP models	MUST	Assurance tool (AT_R_01, AT_R_04)
GT_R_04	Evaluate the trainee's performance and provide related input to the emulation/simulation components in order to adjust the training process	MUST	Assurance tool (AT_R_03, AT_R_04, AT_R_06) Simulation tool (ST_R_01, ST_R_03) Emulation tool (ET_R_05)
GT_R_05	Deploy visualization techniques and cooperate with the visualization tool	MUST	Visualization tool (VT_R_01, VT_R_05, VT_R_11)
GT_R_06	Support a cognitive profiling of trainees and measure their familiarity with different security concepts	MUST	Assurance tool (AT_R_11)
GT_R_07	Adjust the type and the level of difficulty of the training process based on the user's profile	MUST	Assurance tool (AT_R_11)

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
GT_R_08	Support post-training assessments of trainee awareness which are useful in tailoring other forms of CTTP training	MUST	Assurance tool (AT_R_02, AT_R_03, AT_R_11) Training tool (TT_R_01)
GT_R_09	Host several serious games, scenarios and training evaluation mechanisms	MUST	Training tool (TT_R_01, TT_R_05, TT_R_06)
GT_R_10	Develop specific games that are focused on social engineering aspects	MUST	-
GT_R_11	Offer games and training suitable for non-security experts	SHOULD	Training tool (TT_R_01)
GT_R_12	Implement web/mobile application interfaces	SHOULD	Virtualization tool (VT_R_03, VT_R_04)
GT_R_13	Service many users in parallel	SHOULD	-
Training Tool			
TT_R_01	Provide means to allow continuous collaboration with the serious gaming tool	MUST	Gamification tool (GT_R_04, GT_R_08)
TT_R_02	Offer a mechanism for real-time performance assessment of the trainees, whilst they undertake CTTP programs	MUST	Assurance tool (AT_R_01)
TT_R_03	Provide CTTP program evaluation functionalities, through mechanisms enabling the evaluation of the effectiveness of CTTP programs to inform and enable the continuous improvement of such programs	MUST	Assurance tool (AT_R_01, AT_R_03, AT_R_04, AT_R_05)
TT_R_04	Support and facilitate the dynamic adaptation of CTTP programs, through systematic procedures enabling: (a) dynamic tailoring of CTTP programs to the needs of individual trainees, and (b) continuous improvement of CTTP programs	MUST	TT_R_03, Assurance tool (AT_R_04)
TT_R_05	Support synchronous and asynchronous communication between the other THREAT–ARREST components	SHOULD	Assurance tool (AT_R_06), Simulation tool (SR_R_11), Gamification tool (GT_R_04), Emulation tool (ET_R_05)
TT_R_06	Provide means for efficient interconnection with the Assurance, Simulation and Emulation modules	SHOULD	Assurance tool (AT_R_06), Simulation tool (SR_R_11), Gamification tool (GT_R_04), Emulation tool (ET_R_05)
Visualization Tool			
VT_R_01	Offer means to connect data sources (simulation, emulation, etc.) to the visual elements and cover all	MUST	Emulation tool (ET_R_07),

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
	the layers in the implementation stack of the overall THREAT-ARREST platform		Simulation tool (ST_R_09), Gamification tool (GT_R_05), Training tool (TT_R_06), Assurance Tool (AT_R_06)
VT_R_02	Cover the state of the real and the simulated/emulated cyber system components, the attacks upon them and the effects of user actions	MUST	Emulation tool (ET_R_05 & 06), Simulation tool (ST_R_09), Gamification tool (GT_R_05), Training tool (TT_R_06), Assurance Tool (AT_R_06)
VT_R_03	Support a web-browser as the primary user interface, while being compatible with many platforms (Windows Client, Web, Mobile Device)	MUST	User Interface / Platform OS
VT_R_04	Be “integratable” with web-based user-interfaces of other platform components	MUST	Gamification tool (GT_R_12), User Interface / Platform OS
VT_R_05	Provide means to allow real-time bi-directional communication between platform components (both front-end and back-end) in a cloud/web-based environment	MUST	Emulation tool (ET_R_07), Simulation tool (ST_R_09), Gamification tool (GT_R_05), Training tool (TT_R_06), Platform OS
VT_R_06	Offer elements to navigate to GUI components of the other platform components	MUST	Emulation tool (ET_R_04), Simulation tool (ST_R_09), Gamification tool (GT_R_05), Training tool (TT_R_06)
VT_R_07	Offer real-time updating of visualization elements in response to changes in the connected data sources	MUST	Simulation tool (ST_R_09)
VT_R_08	Support synchronous and asynchronous communication between components	SHOULD	Emulation tool (ET_R_07), Simulation tool (ST_R_09),

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
			Gamification tool (GT_R_05), Training tool (TT_R_05)
VT_R_09	Be compatible with SIMPLAN's "Jasima" simulation tool[3]	MUST	Simulation tool (Jasima) (ST_R_01)
VT_R_10	Offer a scenario definition language to describe visualization scenarios usable by simulation and other components	MUST	Emulation tool (ET_R_07), Simulation tool (ST_R_01), Gamification tool (GT_R_05), Training tool
VT_R_11	Include Serious Gaming elements in order to increase learning motivation for small and medium groups	MUST	Gamification tool (GT_R_09), Training tool (TT_R_01), CTTP program adaptor (TT_R_04)
VT_R_12	Implement basic visualization principles (expressiveness/ effectiveness/ congruence/ apprehension) and optimize a balance between adequate context and complexity	SHOULD	-
VT_R_13	Use appropriate visualization metaphors for different types of attacks and platform/simulated components	SHOULD	Emulation tool (ET_R_02), Simulation tool (ST_R_02)
VT_R_14	Offer visualizations that can consist of various textual (tables, labels) and graphical elements (various 2D charts; 3D layout views – symbolic visualization of simulation events)	SHOULD	-
VT_R_15	Handle big and dynamic datasets and effectively support data abstraction over large numbers of data objects	SHOULD	-
VT_R_16	Offer elements to allow user interaction and provide means to define scenarios and training sessions	MUST	Training tool (TT_R_04), CTTP program adaptor (TT_R_04)
VT_R_17	Offer hierarchical modelling of visualization views (each containing various visualization elements); a user should be able to navigate in this hierarchy (drill down/zoom up)	MUST	User Interface
VT_R_18	Utilize real-time comparative performance measures, scenarios' reconfiguration and parameters' adjustment	SHOULD	CTTP modelling & CTTP program adaptor (TT_R_04)

[3] Jasima Simulator: https://www.simplan.de/en/software/jasima/

Req-ID	Description	Req Level (MUST/ SHOULD)	Dependencies
VT_R_19	Be capable of post-process animation of simulation events	SHOULD	Simulation tool (ST_R_13), CTTP program adaptor (TT_R_04)