predict
prioritise
prevent

TREsPASS

Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security

Deliverable D.6.3.1.

Prototype of the TREsPASS user interface

Project:	TREsPASS
Project Number:	ICT-318003
Deliverable:	D6.3.1.
Title:	Prototype of the TREsPASS user interface
Version:	1.0
Confidentiality:	Public
Editor:	J. Barendse
Cont. Authors:	F. Brodbeck, R. Smits
Date:	2015-10-30

Part of the Seventh Framework Programme
Funded by the EC-DG CONNECT

Document History

Authors		
Partner	**Name**	**Chapters**
LUST	Jeroen Barendse	1-5
LUST	Frederic Brodbeck	4
LUST	Robin Smits	4

Quality assurance		
Role	**Name**	**Date**
Advise	Wolter Pieters	2015-10-15
Advise	Christian W. Probst	2015-10-30
Editor	Frederic Brodbeck	2015-09-30
Reviewer	Olga Gadyatskaya	2015-10-15
Reviewer	Marieta Ivanova	2015-10-15
Coordinator	Pieter Hartel	2015-10-15
Task Leader	Jeroen Barendse	2015-10-30
WP leader	Miguel Martins	2015-10-30

Circulation	
Recipient	2015-10-30
Project partners	2015-10-30
European Commission	2015-10-30

Acknowledgement: *The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TREs-PASS). This publication reflects only the author's views and the Union is not liable for any use that may be made of the information contained herein.*

Members of the TREsPASS Consortium

1. University of Twente	UT	The Netherlands
2. Technical University of Denmark	DTU	Denmark
3. Cybernetica	CYB	Estonia
4. GMV Portugal	GMVP	Portugal
5. GMV Spain	GMVS	Spain
6. Royal Holloway University of London	RHUL	United Kingdom
7. itrust Consulting	ITR	Luxembourg
8. Goethe University Frankfurt	GUF	Germany
9. IBM Research	IBM	Switzerland
10. Delft University of Technology	TUD	The Netherlands
11. Hamburg University of Technology	TUHH	Germany
12. University of Luxembourg	UL	Luxembourg
13. Aalborg University	AAU	Denmark
14. Consult Hyperion	CHYP	United Kingdom
15. BizzDesign	BD	The Netherlands
16. Deloitte	DELO	The Netherlands
17. LUST	LUST	The Netherlands

List of Figures

Contents

Management Summary

This deliverable presents the prototype of the TREsPASS user interface (**Attack Navigator**) and shows how a user can access the various tools developed in the project. It gives special attention to the **Attack Navigator Map**, one of the core tools of the TREsPASS project that is part of the Attack Navigator.

Key takeaways

- The TREsPASS user interface focuses on developing a global interaction scheme for the Attack Navigator; an environment where all tools developed within the project can be viewed, accessed and connected.
- The Attack Navigator prototype takes into account the initial and final requirements as developed and described in D.6.2.1. and D.6.2.2. as well as ideas and wishes that arose from meetings and discussions. For the time being this prototype will be available next to the TREsPASS tool environment that has been online and accessible since M6 at `https://trespass.itrust.lu`, but will be integrated into one coherent system between M36 and M48.
- In the first iteration of the Attack Navigator Map, delivered in M24, the team explored how to create such maps in an intuitive but powerful manner, taking into account the underlying TREsPASS model as this was being developed simultaneously in WP1. This second iteration takes the whole chain of tools and data input and output into account, and allows for importing and exporting TREsPASS-XML.

1. Introduction

This report focuses on developing a global interaction scheme for the TREsPASS User Interface, the **Attack Navigator**; an environment where all tools developed within the project can be viewed, accessed and connected.

The deliverable describes two parts of the prototype:

1. The overall user interface, to be applied project-wide, and showcased in the **Attack Navigator (AN)**, including tools, tool chains and general settings and information, and;
2. The **Attack Navigator Map (ANM)**, one of the main tools in the Attack Navigator with which users can build attack navigator maps. The ANM functions as an interface between the various (analysis) tools that are developed within the project as well as the TREsPASS model.

The prototype, that constitute the most relevant output of this task, is available at `trespass.lustlab.net`, in a specially devised **demo environment**. Here the various demos can be tried out, demos can be commented on, and demos are explained by a read-me text. A log-in with password is provided in the text file login.txt which can be found at `https://www.trespass-project.eu/repositories/TREsPASS/Reviews/M36/login.txt`.

Once you are logged in you can browse all available prototypes through the demo interface (including older prototypes and visualisation demos). The demos related to deliverable D6.3.1. can be found online at `trespass.lustlab.net/proto/D6.3.1` or you can access the Attack Navigator directly via `trespass.lustlab.net/proto/an`, and the Attack Navigator Map via `trespass.lustlab.net/proto/anm`.

How the prototypes can be used, and what is currently implemented, is explained in detail in a walk-through video for the Attack Navigator and in a walk-through video for the Attack Navigator Map, both are accessible via `trespass.lustlab.net/proto/D6.3.1`. Early prototypes of the AN and ANM can also be found in the demo environment at `trespass.lustlab.net`.

1.2. How to read this deliverable

Chapter 2 defines the scope and objectives of the prototype, Chapter 3 looks into the specifications, design brief and requirements of the prototype. Chapter 4 is about the interpretation of the specifications and shows the prototype, as well the framework that enables this. Chapter 5 concludes the document and looks forward to future steps to be taken. Finally, the Appendix consists of a project summary including Figure A.2: Integration diagram for the TREsPASS project. The demos related to this deliverable can be found online at `trespass.lustlab.net/proto/D6.3.1` and has the following structure:

Interface

- Prototype Attack Navigator
- Prototype Attack Navigator Map
- Video Walk-through Attack Navigator Map
- D6.3.1. as PDF
- Navigation scheme for the wizard as PDF

Data

- Example of TREsPASS XML output from the ANM
- Data from the cloud scenario:
  - Input data in GraphML
  - GraphML converted to TREsPASS XML
  - Generated attack tree from TREsPASS XML

1.3. Foreground and background

The current prototypes of the TREsPASS user interface, namely the Attack Navigator and one of its main tools, the Attack Navigator Map, are foreground to the project. Early versions of some TREsPASS tools (ADTool (v1.1), DFTCalc, Approxtree etc.) are background of the project. The individual tools have been improved by their development teams and constitute foreground of the project. trespass.js is foreground to the project, other libraries are background to their respective developers (angular, react, bootstrap, keystone). Everything else in this document and on the prototype website is foreground of WP6.

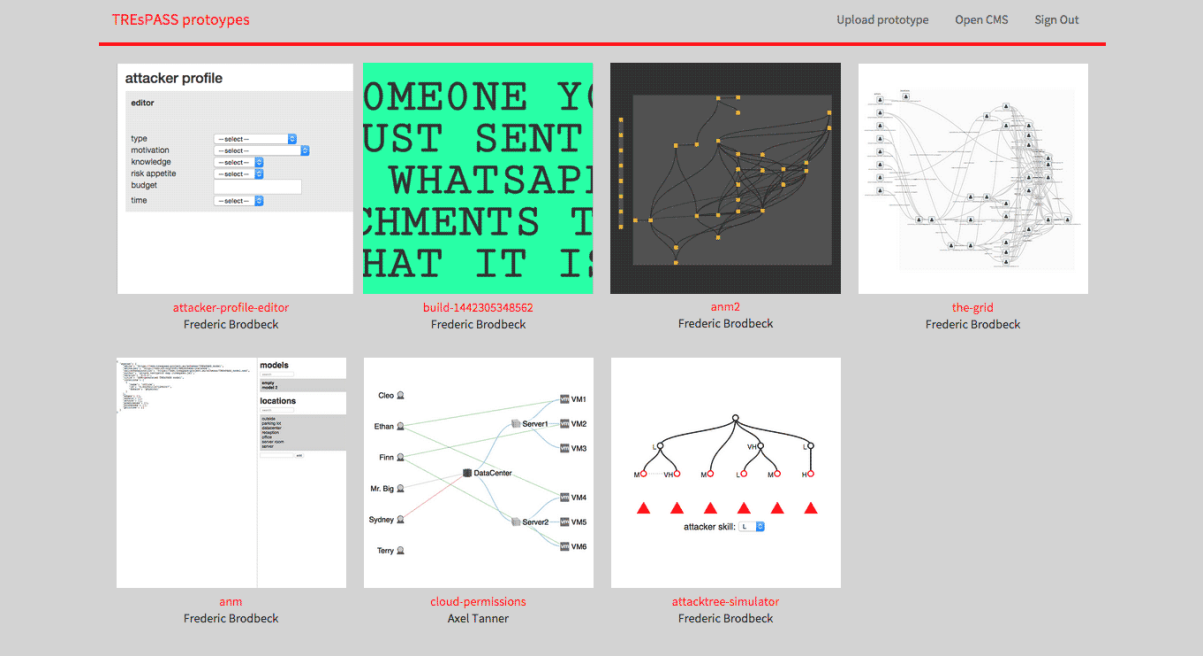


Figure 1.1: Screenshot of the demo environment for prototypes, available at `trespass.lustlab.net`. Users are allowed to see a limited set of demos depending on their access profile. Under each prototype, users can read the ReadMe text, as well as leave comments on the demo. The demo owner will be automatically notified when a comment is left.

2. Prototype scope and objectives

The TREsPASS user interface is focused on developing a global interaction scheme for the **Attack Navigator (AN)**; an environment where all tools developed within the project can be viewed, accessed, used and connected. The requirements, prototype specifications, and how they are interpreted can be found in Chapters 3 and 4. For a detailed description of the integrated TREsPASS process, please refer to D5.4.1., for the refinement of functional requirements, please refer to D6.2.2..

Additionally, a main part of the user interface is one of the major tools, the **Attack Navigator Map (ANM)**. As described in deliverable D6.2.2. "This is a tool that predicts and prioritises attack scenarios based on a model of the system or organisation concerned. It can also be used to judge the effect of countermeasures, by re-running the analysis with an adapted model. The model takes the form of a navigator map and a set of attacker profiles. The navigator map represents the system cartographically, displaying connections between the elements as potential steps that an attacker could take. These steps are annotated with relevant variables such as difficulty and cost. The attacker profile collects relevant characteristics of an attacker, such as skills, resources, motivations / goals, and initial access. The latter can be thought of as a starting point on the map. For a combination of a map and a profile, the system will calculate routes for the attacker across the map that provides utility to the attacker. Typically, this will involve gaining access to certain assets and compromising their confidentiality, integrity or availability, which may cause damage to the organisation. The routes with the highest utility for the attacker constitute the highest risk with respect to the selected attacker profile. Multiple profiles can be combined to provide an overall risk picture."

The initial prototypes delivered in M24, gave insight into how an Attack Navigator, including its tools, can function from a user interface perspective and its potential for developing such systems. This deliverable and the related prototypes focus on the practical aspects of the user interface (structure, functionalities, etc..), how a user can build maps, how the different tools, libraries and models are input for the ANM, what the ANM outputs, etc.. For the rationale behind the visualisation choices in the interface, we refer to D4.2.1..

3. Prototype specification

3.1. Overview

The fastest way to understand the relations between the different components in the TREsPASS user interface is a graphical representation. Figure 3.1. is a graphical representation of the relations of the various interface components, where the Attack Navigator is the global TREsPASS user interface that contains a number of tools, including the Attack Navigator Map. Next to the tools there will be a section that allows for building tool chains. There is also other functionality like a glossary of the terms used in the project, settings, tasks, log in/out and a help function.

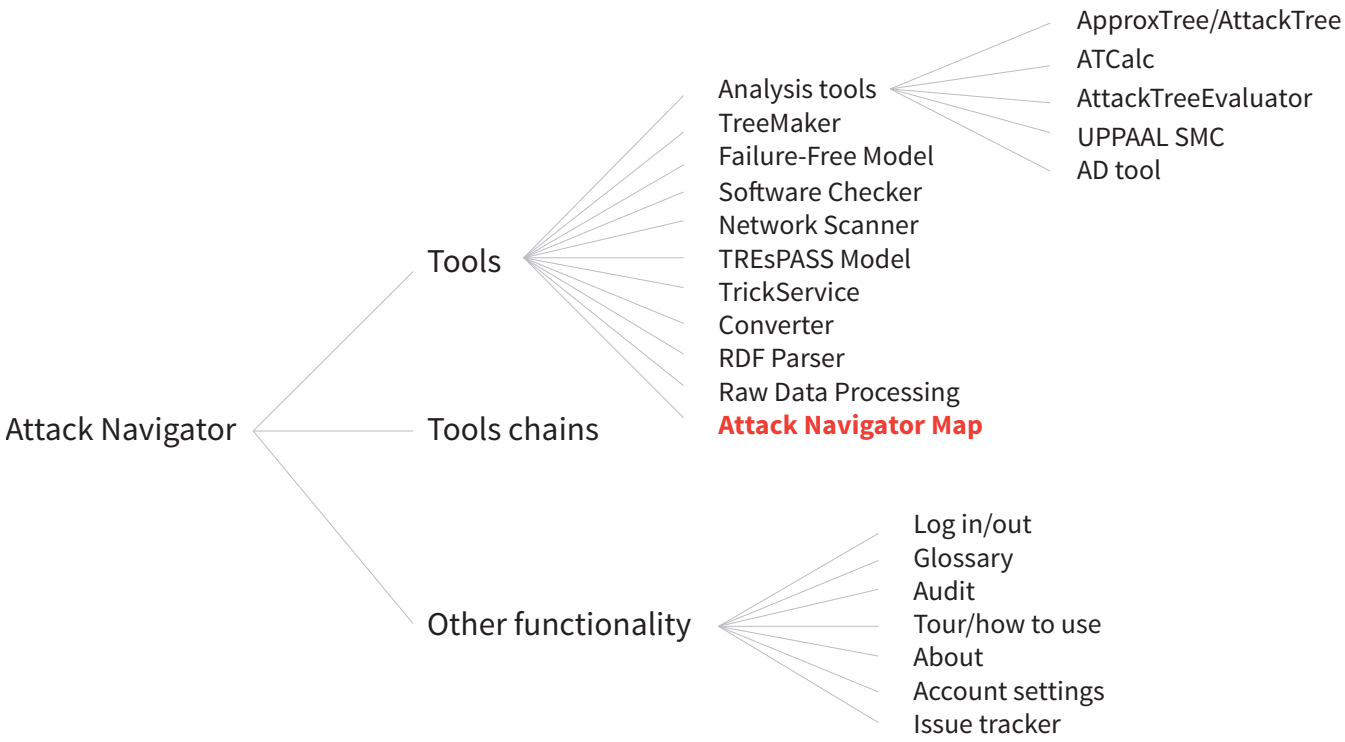


Figure 3.1: Graphical representation of the nested relations of the various interface components of the Attack Navigator. The Attack Navigator Map is highlighted among the different tools, as this is the main focus in this deliverable. The list presented here is not complete and only for showing types of functionality and how they are grouped.

Hierarchically the structure of the TREsPASS user interface can be described as follows:

TREsPASS user interface = Attack Navigator gives access to

- **Tools**, for example, but not limited to:
  - Attack Navigator Map
  - Software Checker
  - TrickService
  - Archimate
  - A/D annotator
  - Attack pattern
  - TREsPASS model
  - Converter
  - Analysis tools
    - ApproxTree
    - AD tool
    - ATCalc
    - AttackTreeEvaluator
    - UPPAAL SMC
  - Raw Data Processing
  - Visualisation tools
    - Lego
    - Visualisation Atlas
    - Expressivity tool
    - Attack tree visualiser
    - Password visualiser
  - etc.
- **Tool chains**
  - Tool chain 1
  - Tool chain 2
  - etc.
- **Functionalities**
  - **Tour/how to use**: gives first time users a visual introduction on how to use the interface.
  - **Glossary**: an alphabetical list of terms and definitions from the TREsPASS-domain of knowledge.
  - **Account Settings**: system wide, each tool can also have its own preferences if needed
  - **Issue tracker:** opens in a new window, all issues can be reported here and dealt with.
  - **Audit**: links to a page that gives information about audits
  - **Tasks**
  - **Log in/out**

3.2. Design brief and requirements for the user interface

The design brief for the TREsPASS user interface is partly based on the functional requirements laid forward in various deliverables (D6.1.1., D6.2.2.). In addition to these formal requirements there are also other notions that have been formed through discussions, meetings and brainstorms.

The prototype of the TREsPASS tools user interface builds on loosely coupled solutions, using a centralised integration component and database. The modules coming from the different work packages in the scope of the prototype exchange data through the central component. The TREsPASS user interface (Attack Navigator) builds further on this loosely coupled solution, integrating different tools without aiming to develop one large and therefore complicated system.

The user interface should allow for all the requirements that are currently to be found in deliverable D6.2.2. Refinement of Functional Requirements. The user interface that will be developed should be able to deal with the visualisation of data input, navigator maps, attacker profiles and attack tree scenarios. The user interface should be flexible enough to allow using existing and future tools, as well as connect several tools into one or more tool chains.

3.2.1. General requirements

- [R06] ”Apps style" user interface
- [R68] The TREsPASS tools enable the user to store a TREsPASS model, including navigator map, attacker profiles, and history of analyses
- [R10] Interface between analysis tools and visualisation tools
- [R69] The TREsPASS tools enable the user to load a previously stored TREsPASS model, including navigator map, attacker profiles, and history of analyses.
- [R81] Visualisation of social and technical data, maps, scenarios, countermeasures.
- [R82] TREsPASS tools should be accessible through a web interface
- [R83] The TREsPASS integrated component should use loosely coupled tools that are also available individually
- [R86] Users can select base models from a Model Template Library.
- [R40] Support for attack tree visualisation (via ADTool)

3.2.2. Interface requirements

- [R52] Users can drag and drop standard elements onto the map.
- [R53] Users can add connections between elements on the map.
- [R54] Users can change parameters of map elements by clicking and entering new values
- [R55] Users can change parameters of map elements by clicking and requesting information from available data extraction tools.
- [R57] The user can select attacker profiles from the library.
- [R58] The user can build attacker profiles by editing their properties.
- [R59] The user can assign asset values and properties to elements on the map.
- [R60] The user should be able to replicate an entity in the model by clicking on it, selecting the replicate option, and indicate the number of replications and associated parameters.

Full version/ Light version

Two types of users are foreseen for the TREsPASS tools which results in two additional requirements for the user interface.

1. Firstly, security practitioners (either security officers within an enterprise or external consultants) are expected to be able to use the tools in security risk management processes and audits. In particular, practitioners can use the tools and processes to provide decision support on security measures to be implemented. In case of audits, they can use the tools and processes to judge whether security measures are adequate with respect to the targets of the audit.
2. Secondly, SMEs are expected to be able to use a “lightweight” version of the tools in relatively quick scans of their security posture.

All the above requirements have been taken into account while developing the user interface.

3.3. Requirements for the Attack Navigator Map

Within the Attack Navigator, the Attack Navigator Map has a special status. It is one of the tools, but its also a tool that connects and works together with almost all the other tools that are available in the interface. In that sense, the Attack Navigator Map is a central part in a chain of tools, including the TREsPASS model. Figure 3.2. shows all instances where the Attack Navigator Map plays a role, where data and tools function as input (as TREsPASS-XML), and where it exports to (as TREsPASS-XML).

For the TREsPASS model a graph-based format was chosen (D1.1.2. Final Specifications and Requirements for Socio-Technical Security Models) that is internally represented as XML. This enables wide tool support, easy transformation to different formats, and relatively pain free understanding by humans. In the XML format, type attributes are used to represent hierarchies of elements, and to associate elements with properties, and all elements are required to have a unique id that is enforced by XML parsers. The TREsPASS workflow as described in detail in D5.4.1. and in Chapter 1.1.2. goes through four stages and the interface of the ANM should take this into account. The stages may have various activities, not all of which are required for every risk assessment scenario. The main stages are the **data collection** stage, the **model creation** stage, the **analysis** stage, and the **visualisation** stage.

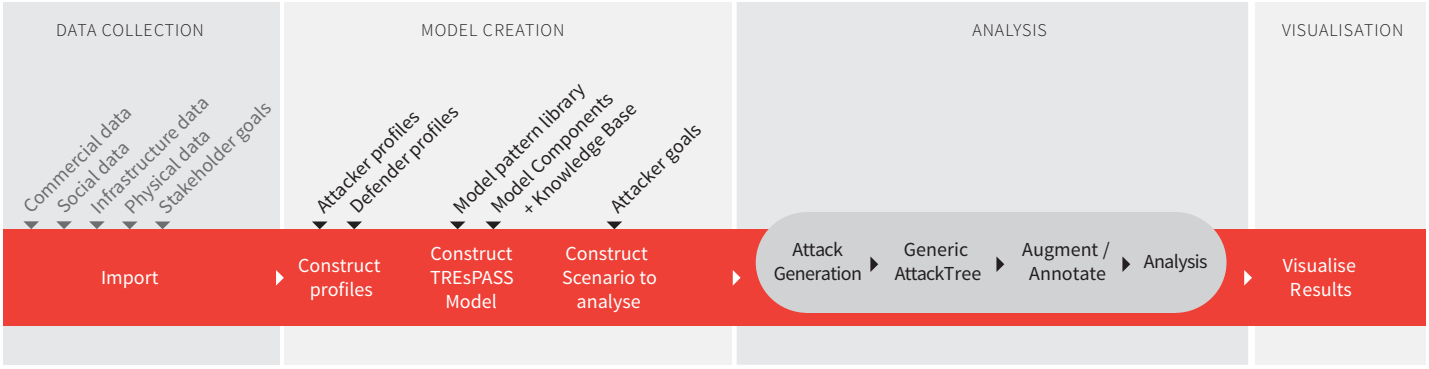


Fig. 3.2: Graphical representation of the various functions and places where the Attack Navigator Map (in red) is used within the TREsPASS workflow of four sequential stages (data collection, model creation, analysis, visualisation). The grey area in the analysis stage on top of the ANM workflow indicates that those parts do not take place in the ANM, but in the separate TREsPASS tools. The output of those results are fed back into the ANM in order to visualise results. A typical workflow would consist of import, construct and refine, analyse, and visualise. This graph is based on the integration diagram that can be found in the Appendix as Figure A.2.

Functions of the Attack Navigator Map

The following functions of the Attack Navigator Map are not from the list of requirements that is described in D6.2.2., but are taken from the integration diagram that can be found in the Appendix as Figure A.2. These functions are ordered according to the stage each function belongs to:

- **Data collection stage**
  - Import data from the various sources identified in the data collection stage
- **Model creation**
  - Construct and refine profiles
    - Import of attacker profiles (from attack pattern library)
    - Import of defender profiles
    - Import of model pattern library
    - Export to knowledge base
  - Construct the TREsPASS model
    - Import of Model templates
    - Import of Model components
    - Export to TREsPASS model
  - Construct attacker goals
    - Export to scenarios
- **Analyse stage**
  - Select attacker profile to analyse
    - Export to attack generation (TreeMaker)
  - Select scenario to analyse
  - Construct attack patterns (ADTool)
    - Import generic attack patterns
    - Import attack patterns through attack pattern library
    - Export to generic attack patterns
    - Export to custom attack patterns
- **Visualisation stage**
  - Visualise results

4. Interpretation of the specifications and requirements

The design brief and requirements have been interpreted as follows:

The existing loosely coupled solution of tools bears a lot of similarities to a collection of apps that can be found in many modern operating systems. Instead of trying to make programs that do “everything”, there is a tendency in IT and user interfaces to move to an environment with the least possible amount of settings. Programs are more and more designed to do one task, and are only optimised for that task, similar to the Unix philosophy[1]. Apps are not about adding feature on feature (featuritis), but should focus on its key task. The different tools in the TREsPASS project are optimised for their task, and will have certain user interface elements that will be treated the same for all tools. Also gestures and behaviours will be standardised as much as possible throughout all interfaces. The tool chains are envisioned to allow for connecting the different tools.

A general guideline in terms of information overload is that information and visual clues should be minimised when possible. Next to a dashboard-like view, the system should preferably interpret the information and give actionable information: *There is a vulnerability **here** and **here** in the system, and you should act on this in **this** way.*

A number of initial guidelines for the interface were formulated based on the requirements and the design brief. The general user interface should:

- support the tool and the user
- “disappear” when not needed
- show context-dependent information as much as possible
- be easily accessible when needed
- offer multiple ways of accessing the tool
- be consistent in its functioning
- allow for building toolchains
- avoid information overload
- present as much as possible actionable information

1 http://en.wikipedia.org/wiki/Unix_philosophy

To tie the variety of tools together (existing and in development) a number of standardised visual elements should be defined. These are:

- a dedicated, uniform color palette
- a typeface and typographic rules
- location of main menu and submenu
- a set of transitions and animations
- a standardised legend
- uniform gestures and placement of interface elements

4.1. Prototype user interface

In the following sections the prototype of the user interface for the TREsPASS project is presented. First we present the main interaction scheme for the **Attack Navigator**, and consequently the prototype for the **Attack Navigator Map**. The Attack Navigator runs on a standard web server and is built in HTML5 and works on all modern web browsers without having to download plug-ins. Some tools run online, some tools can be downloaded and can be executed from a user's computer.

4.1.1. Attack Navigator

The user interface of the **Attack Navigator** is based on the principles that interface elements should be accessible at all times, leave as much room for the individual tools as possible, and 'disappear' when not needed. This is manifested in interface elements that appear when hovered over with a mouse or clicked and disappear on hover-out. When something is selected, the interface should be very consistent in its placement of menu items.

The main user interface, where all main menu items are accessible, can be accessed when one hovers over the TREsPASS logo on the left top of the window. Here all available tools can be launched and explored, tools can be connected in tool chains, and all other functionalities can be accessed. If needed, secondary, tool specific interface elements can be accessed on the right side of the browser window. These secondary interface elements can expand to allow for more information when needed. Similar to the functionality of the main menu, showing more contextual information, as libraries, zoom functionality, and undo. Additionally, the user can access the same menu items by using a right click with the mouse which opens a contextual menu window. The design is toned down so that the user can focus as much as possible on what should be done, instead of the interface itself.

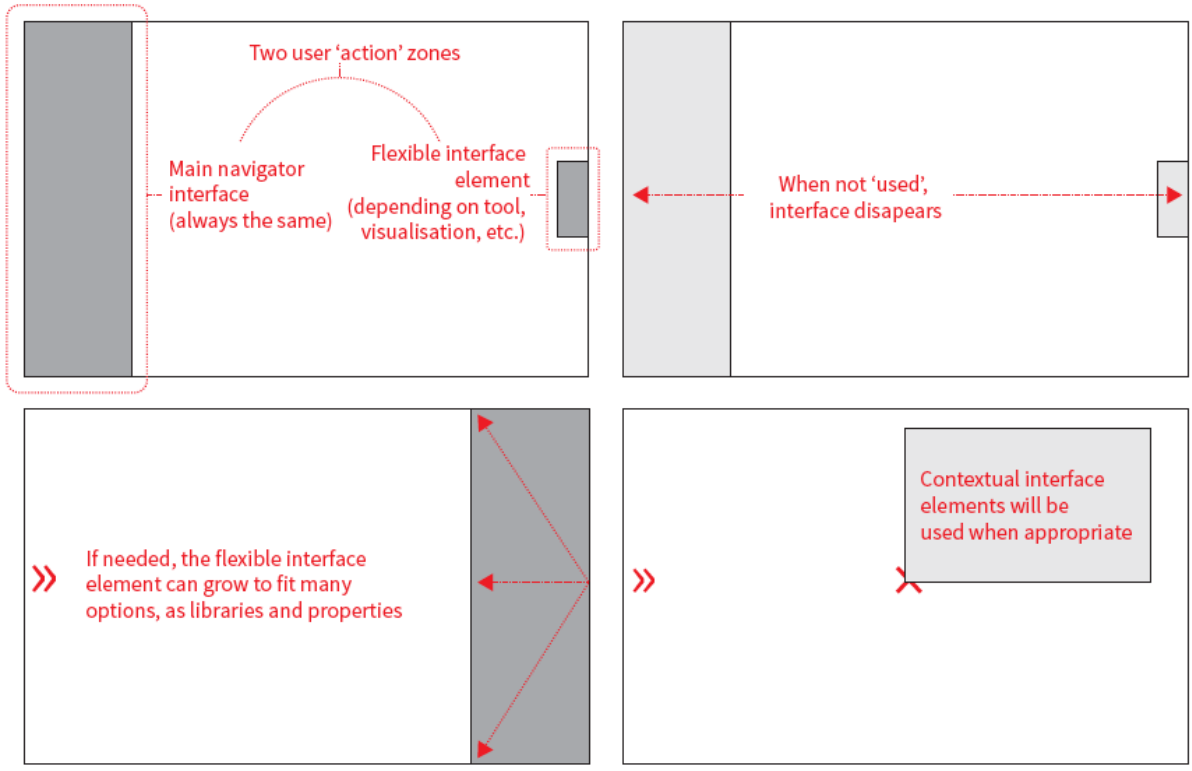


Figure 4.1: The main Attack Navigator functions are accessible from a panel on the left, which is globally present. On the right side of the view is a toolbar specific to the currently open tool—this is different from tool to tool. It contains buttons to execute actions or open a panel with additional options. When they are not needed, panels automatically move out of the way. Contextual menus, dialogs and palettes are used within the tool, when appropriate / needed.

The best way to explore the Attack Navigator user interface and its workings is going through the prototype. This prototype can be found on `trespass.lustlab.net/proto/an`, as well as a video walk-through that explains step-by-step how the interface functions. For an explanation on how to log in, please see Chapter 1, Introduction.

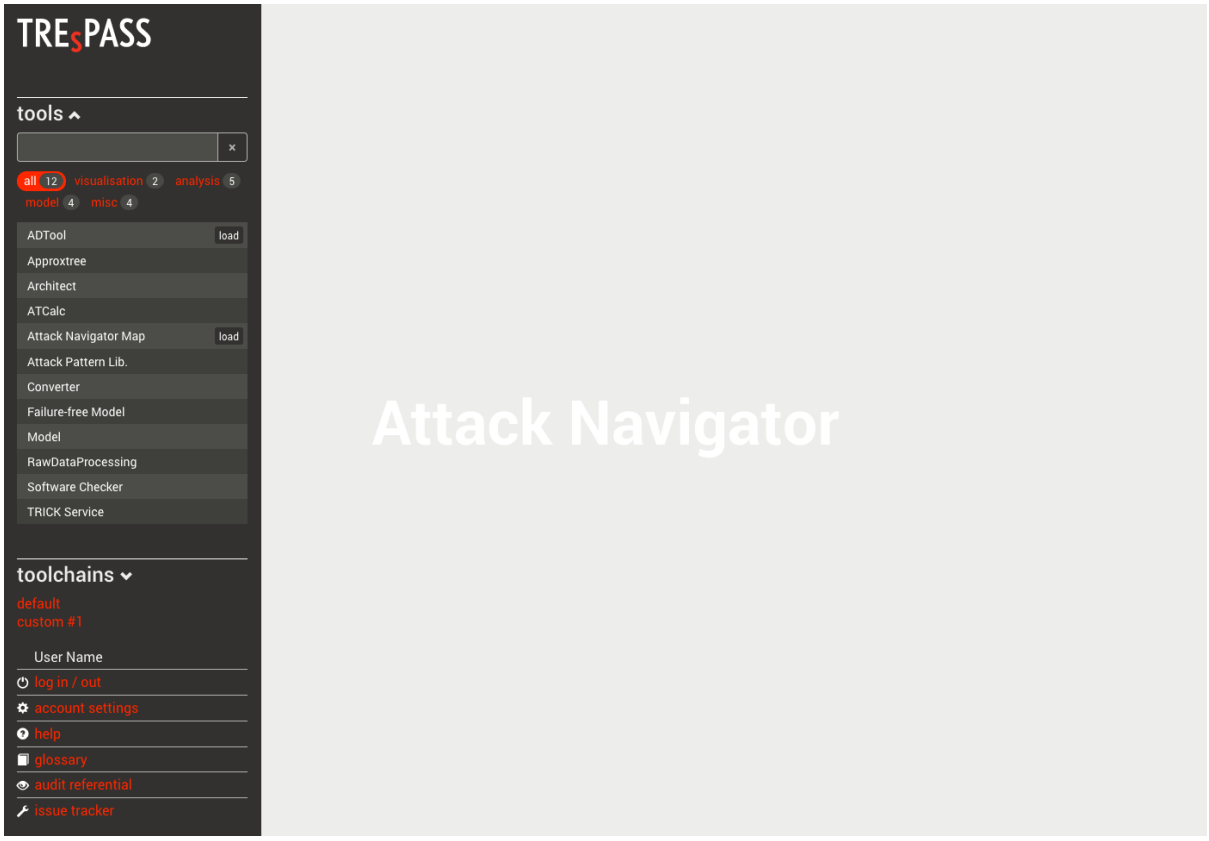


Figure 4.2.: Example of an Attack Navigator interface page. In this figure, the user has hovered over the TREsPASS logo in the left top corner, which opens the main functions of the interface. By default the tools section opens. Each tool will load in the area that is currently grey. When a user has opened a specific tool, the interface will disappear, but will always be accessible by hovering over the logo.

4.1.2. Tool: Attack Navigator Map

The user interface of the Attack Navigator Map (ANM) functions within the framework of the TREsPASS Attack Navigator (AN), as one of the tools. The first prototype allowed for simple modelling tasks as adding nodes (from a library), nesting nodes (for instance locations) and defining relationships between nodes by creating labelled edges. The second prototype is focussed on the integration of the various tools.

The Attack Navigator Map is the central tool in the **model creation** stage of the TREsPASS tool chain. It is where data from different sources (data collection stage) come together, to be incorporated in the model. The model itself is graphically represented as a 'map', which visualises the threat landscape of the system, properties of its components, and the relations between them.

Data from the **data collection** stage is either imported or added manually. Most technical data (such as data from automated network discovery processes, for instance) is converted if needed, and can then be loaded via an import function. Social data – knowledge, in many cases – requires manual editing of the model.

The basic building blocks for constructing a model come from libraries of single components, or of prefabricated model fragments (groups of components with relations), such as the model pattern library. These libraries will contain commonly used patterns, that can be used as templates to rapidly build the basic structure, which can then be refined and tweaked.

The underlying data structure is a directed graph of nodes (components with properties) and edges (relations between those components). Future work on the ANM will concentrate on refining the visual language of the map, to make it look less like a network graph and more like an actual landscape. We envision a more abstract representation with a look and feel that is more specific to component domains and properties. Furthermore, it will be based on the principle of showing details only selectively, when they are relevant to the user.

The ANM is one of the core tools in TREsPASS – that does not mean though, that models have to be created in the ANM exclusively. Diversity in the choice of tools is desirable. Therefore, if a user prefers a different modeling tool or does not want to change an existing workflow, that should not be a problem, as long as the model is or can be converted to the TREsPASS-XML format before importing it into the TREsPASS tool chain. Other tools that can generate compatible model files are (at the moment):

- ArchiMate Architect, developed by BizzDesign: a full-fledged model editor
- Graphml-to-stm-converter, developed by LUST and IBM: a command-line tool to convert Graphml files (created with a graph editor like yEd) to TREsPASS XML.

Once done editing, a scenario (the model + the attacker goal) is constructed, and passed on – along with the attacker profile(s) – to the next step: attack generation.

Once the analysis finishes, its results are visualised, with the possibility for the user to rank and filter the attacks. In a split screen dashboard, there may be multiple other views besides the list of attacks. Overview might include showing the attack traces in the context of the intermediate attack tree, or highlight the relevant parts of the model/ANM that play a role in the attack(s).

The Attack Navigator Map is implemented as a single-page web application, using react.[2]

2 reactjs.net/

Creating a map

The Attack Navigator Map offers two modes in which a user can build a map:

1. **Advanced mode**
   Lets the user model in a free-form environment, for users who already have knowledge of the concepts of TREsPASS and how to use the map and model.
2. **Wizard mode**
   In order to make the creation of navigator maps more easy and intuitive, the interface also provides a wizard function. This wizard leads the user step-by-step in a linear way through the various stages needed to create and complete a map with locations, actors and assets. The wizard also gives access to reusable library items, for example standard room configurations, as well as reusable common patterns, for instance a common hierarchical pattern in a company (department > role > actor) as a basis, where properties can be adjusted. The wizard takes the user by the hand in the entire model creation phase, including the creation of an attacker profile. The wizard ends with running the analysis.

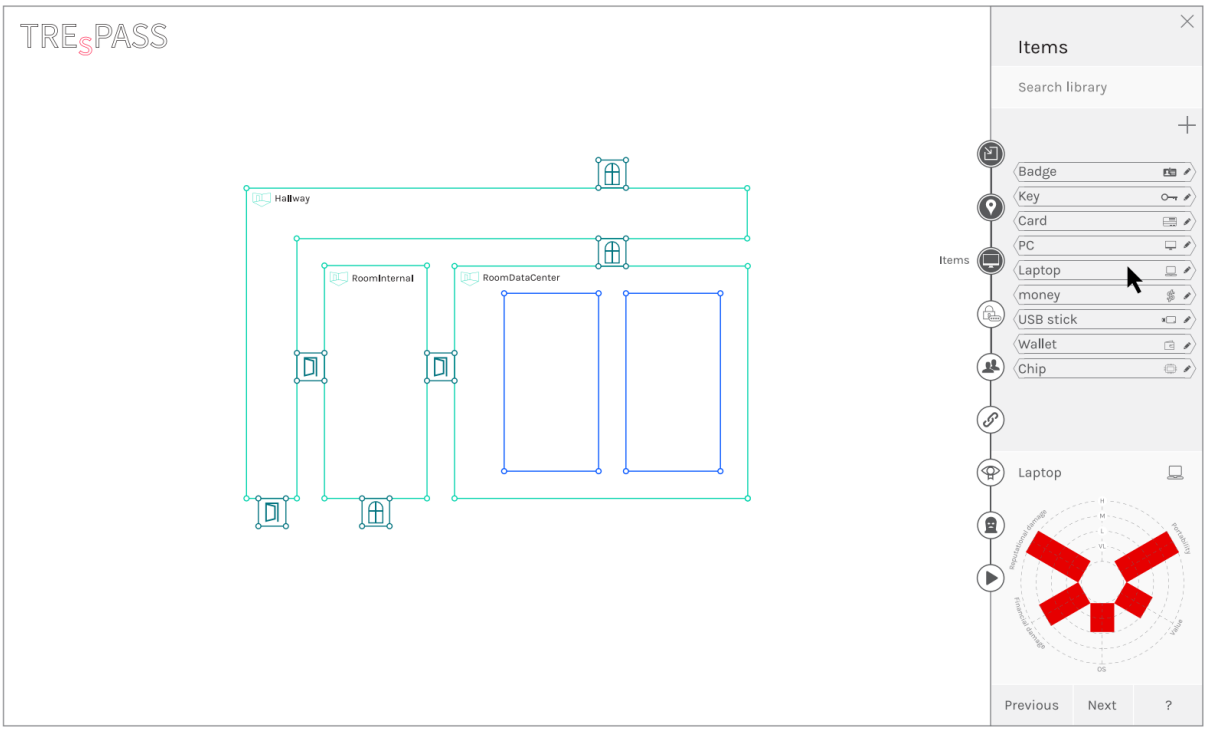


Figure 4.3.: Example of a step in the wizard mode. The panel on the right (light grey background) shows all available steps, which can be followed in a chronological order (top-to-bottom). This figure shows the Items library, where the item 'laptop' is selected. The graph on the bottom right displays the properties of the laptop, which can be edited there directly. Note that the wizard will be implemented in M36-M48 as it is not part of the requirements

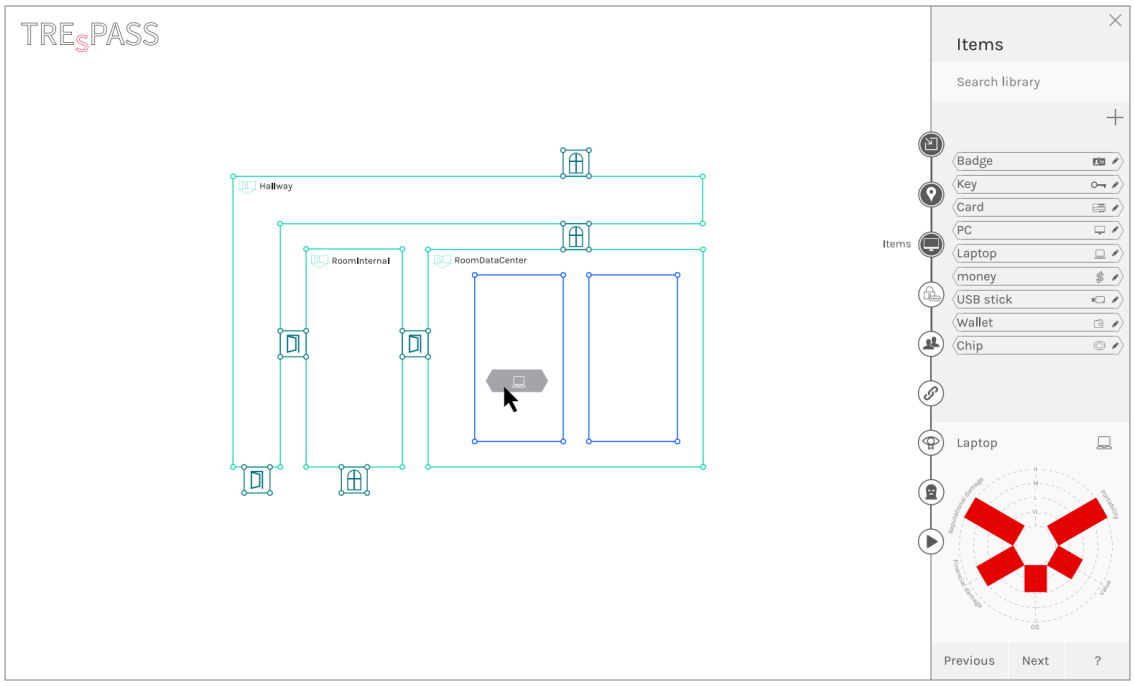


Figure 4.4: A user drags the laptop item on the map and transform while dragging to the icon that will represent the laptop on the map.

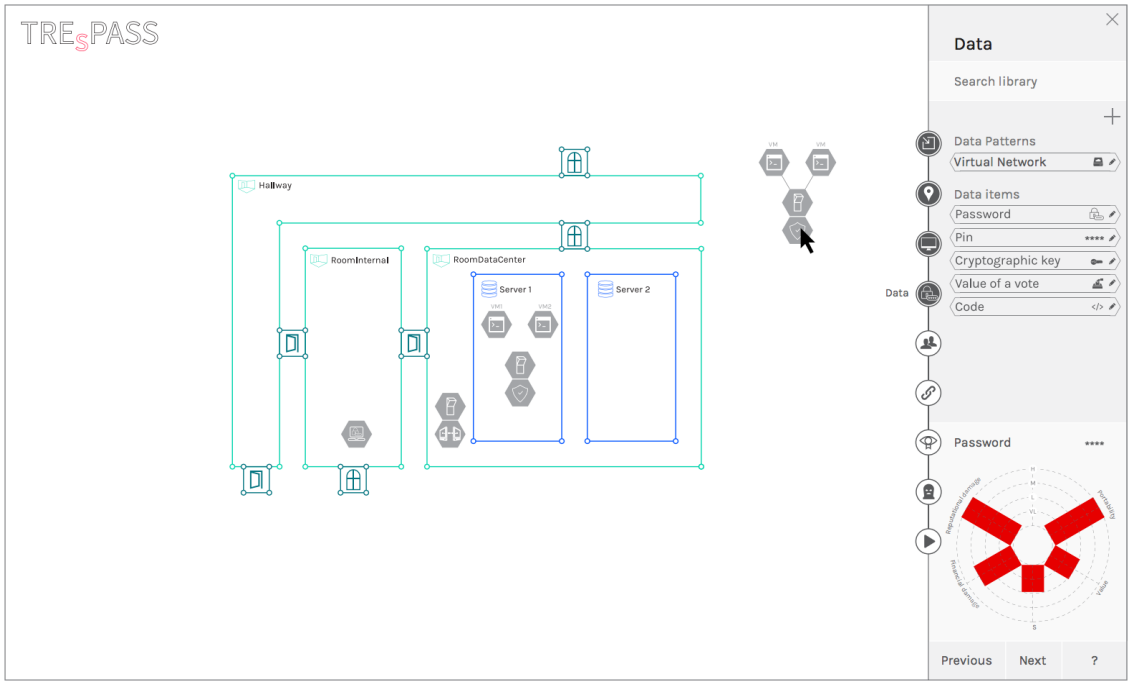


Figure 4.5: Users can also drag precomposed patterns on the map, in this case a 'data pattern' for a virtual network. The virtual network consists of 2 VMs, a switch and a firewall. These patterns make it easier for users to create maps. The properties for each node can be still changed to reflect the user's preference.

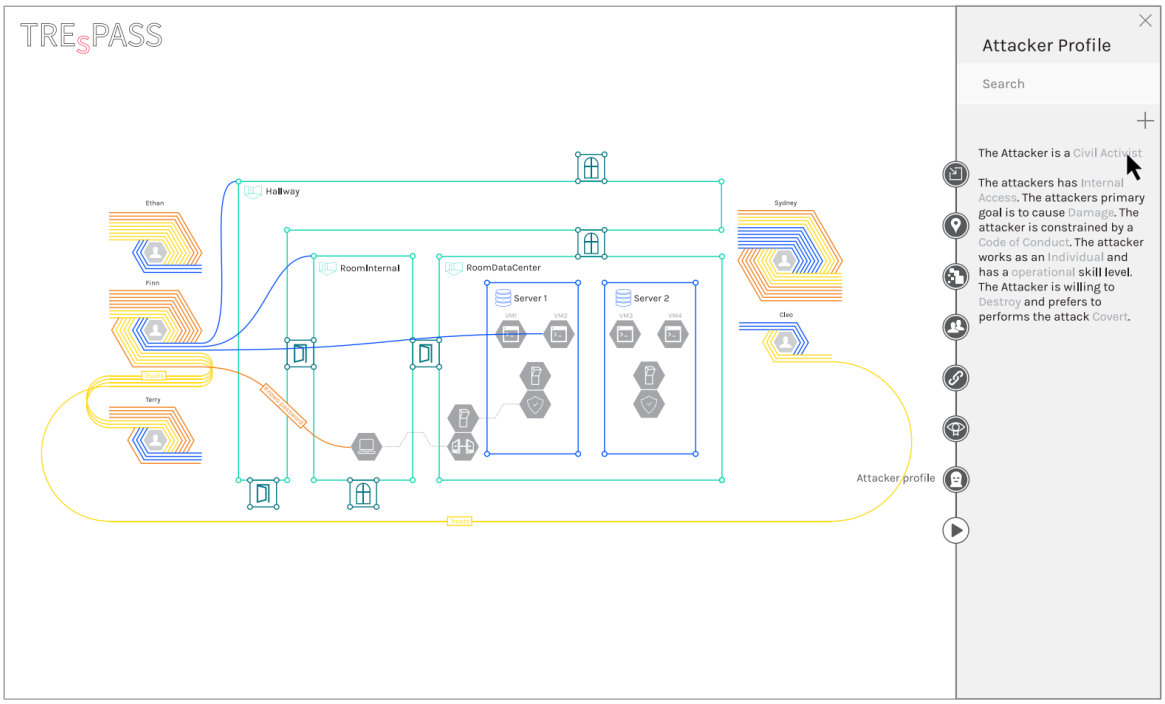


Figure 4.6.: Here, actors are added to the map and the user is editing the attacker profile. The attacker profile is represented as text where each grey word can be edited to hand craft the profile.

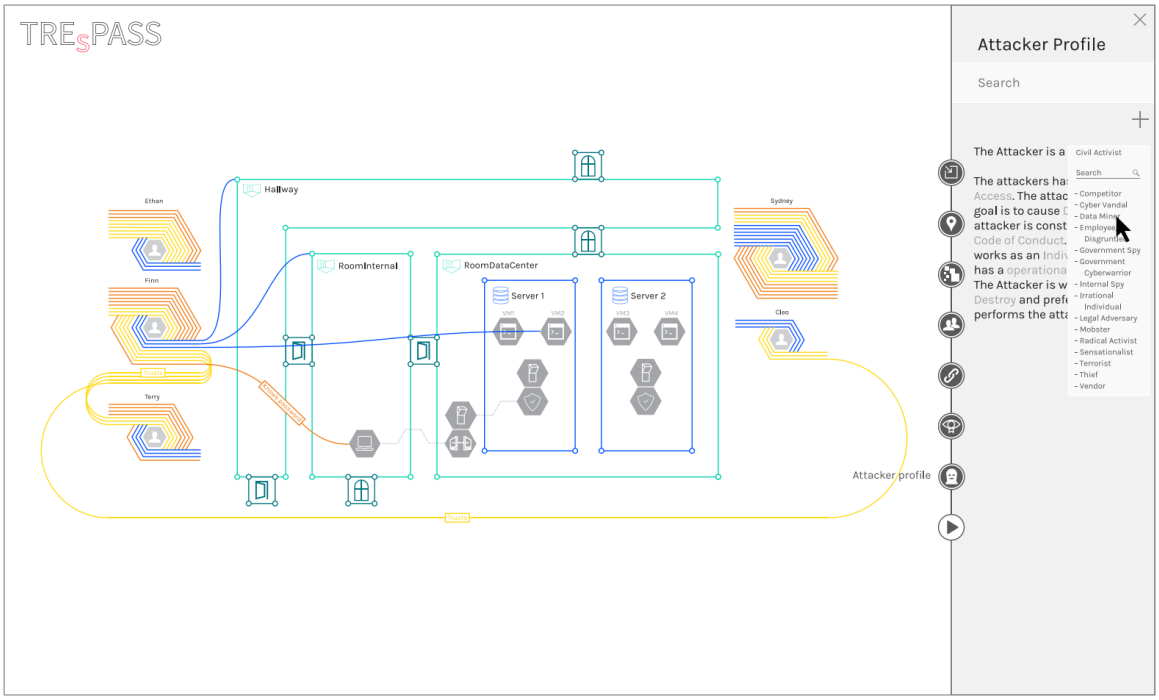


Figure 4.7.: The attacker profile consists of 22 predefined threat agent profiles as competitor, cyber vandal, mobster (refer to D5.3.2. to see a more detailed description of threat agents).

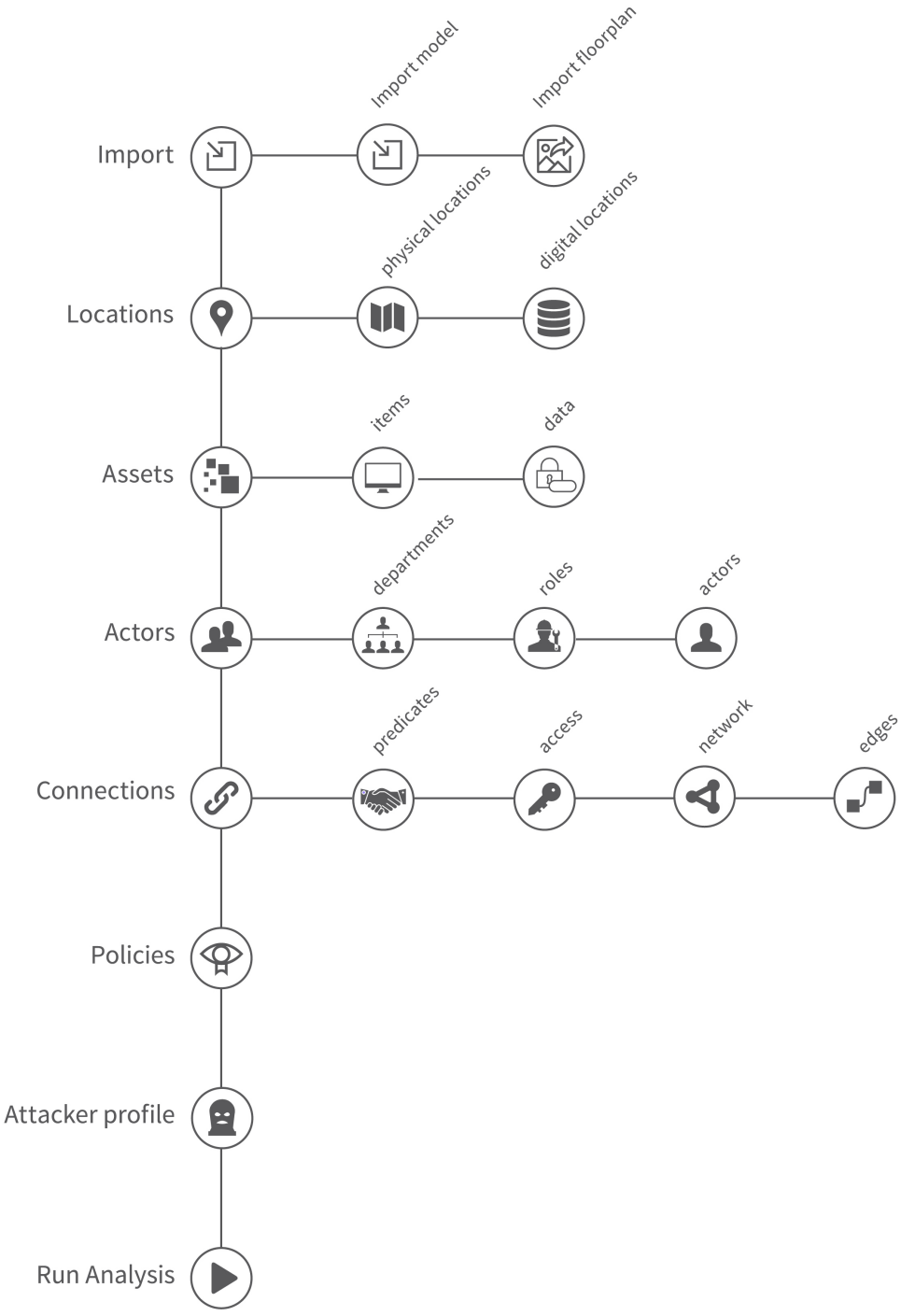


Figure 4.8.: List of steps in the wizard that a user needs to go through in order to create a complete attack navigator map. Each step consists of a series of sub steps that the user needs to fulfill, or can choose from.

In the next four pages we show an example of how various programs and editing tools can work together, and -via export and import of TREsPASS-XML- forms a solid basis for a scenario that can be built in the Attack Navigator Map.

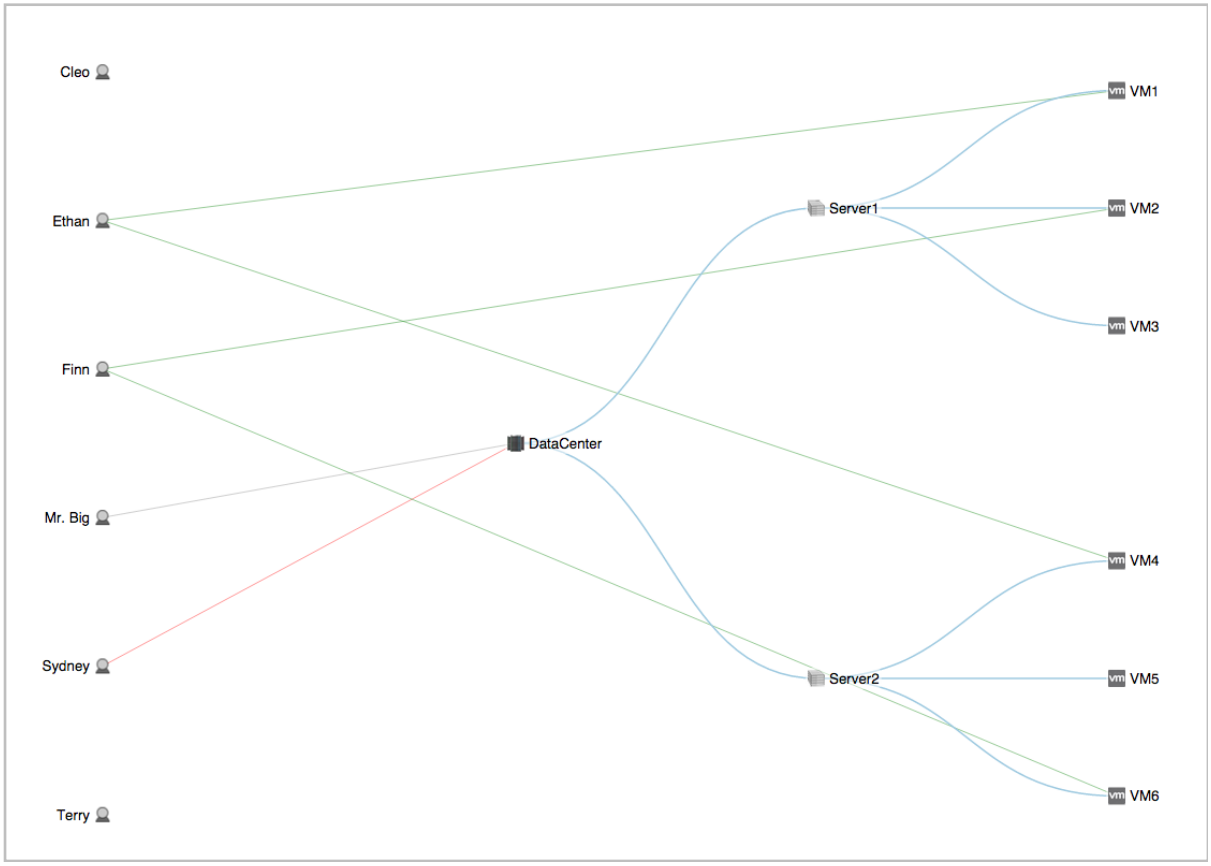


Figure 4.9.: Example of an interactive visualisation for access control of various actors for a small data center, including virtual machines. A live demo can be found at the demo environment at `trespass.lustlab.net`. Image courtesy of IBM.

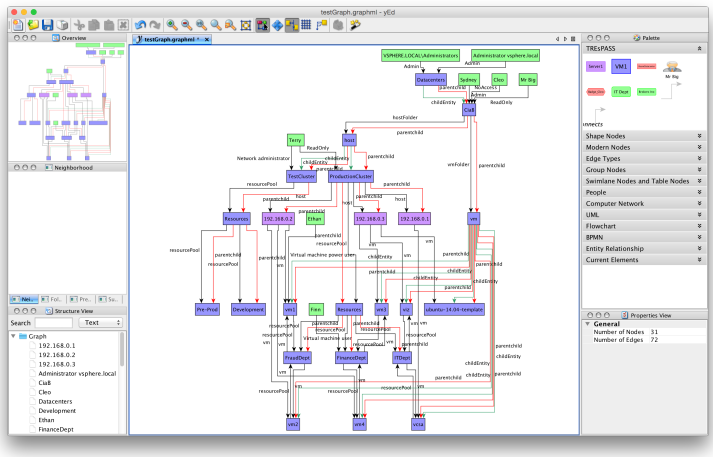


Figure 4.10.: The same scenario as Figure 4.9. modelled in yED (`www.yworks.com/en/products/yfiles/yed/`), a Graphml editor. This can be exported to TREsPASS- XML, which can then be imported in the Attack Navigator Map to be edited, and built upon.

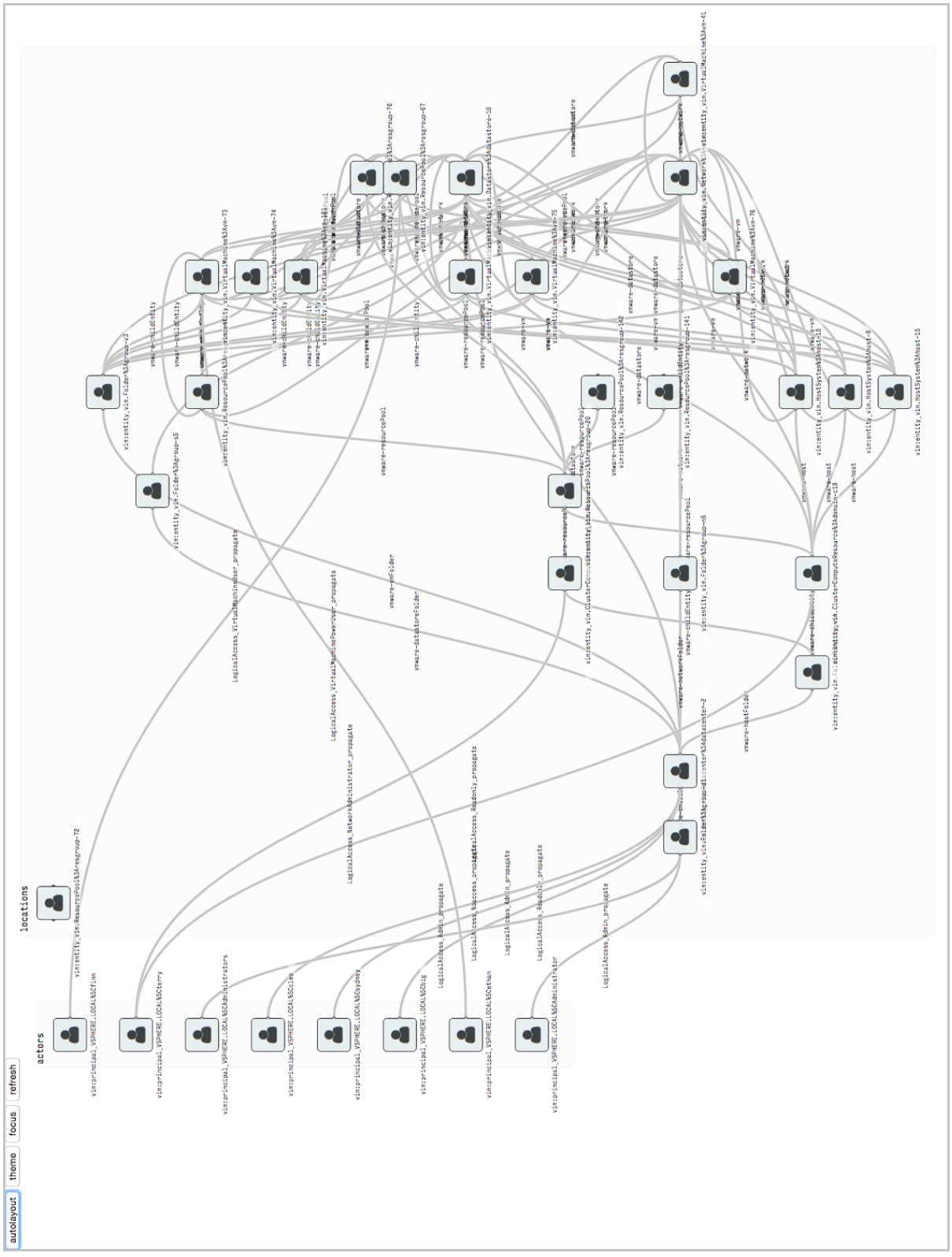


Figure 4.11.: Visualisation of an import of the TREsPASS-XML that was exported from the yED program. Note that all nodes are treated the same.

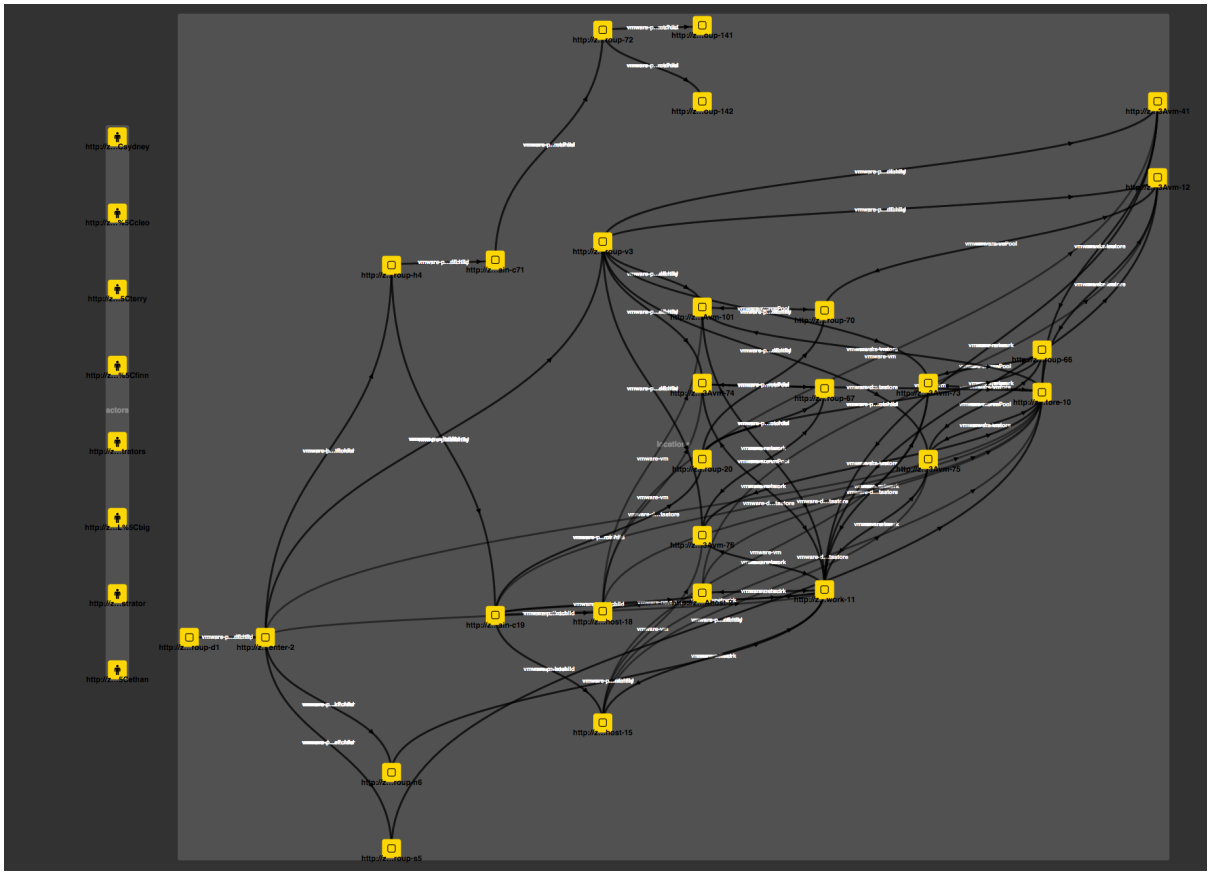


Figure 4.12.: Result of the import of the TREsPASS XML from Figure 4.9. and 4.10 in the ANM, using the auto lay-out feature. In this version, different categories of nodes are recognised, and are bundled in groups. The actors are located in the row on the left side, the servers and various virtual machines are bundled on the main lighter grey square. Each yellow square is a node. Nodes can be actors, locations, and assets. Assets can be items or data.

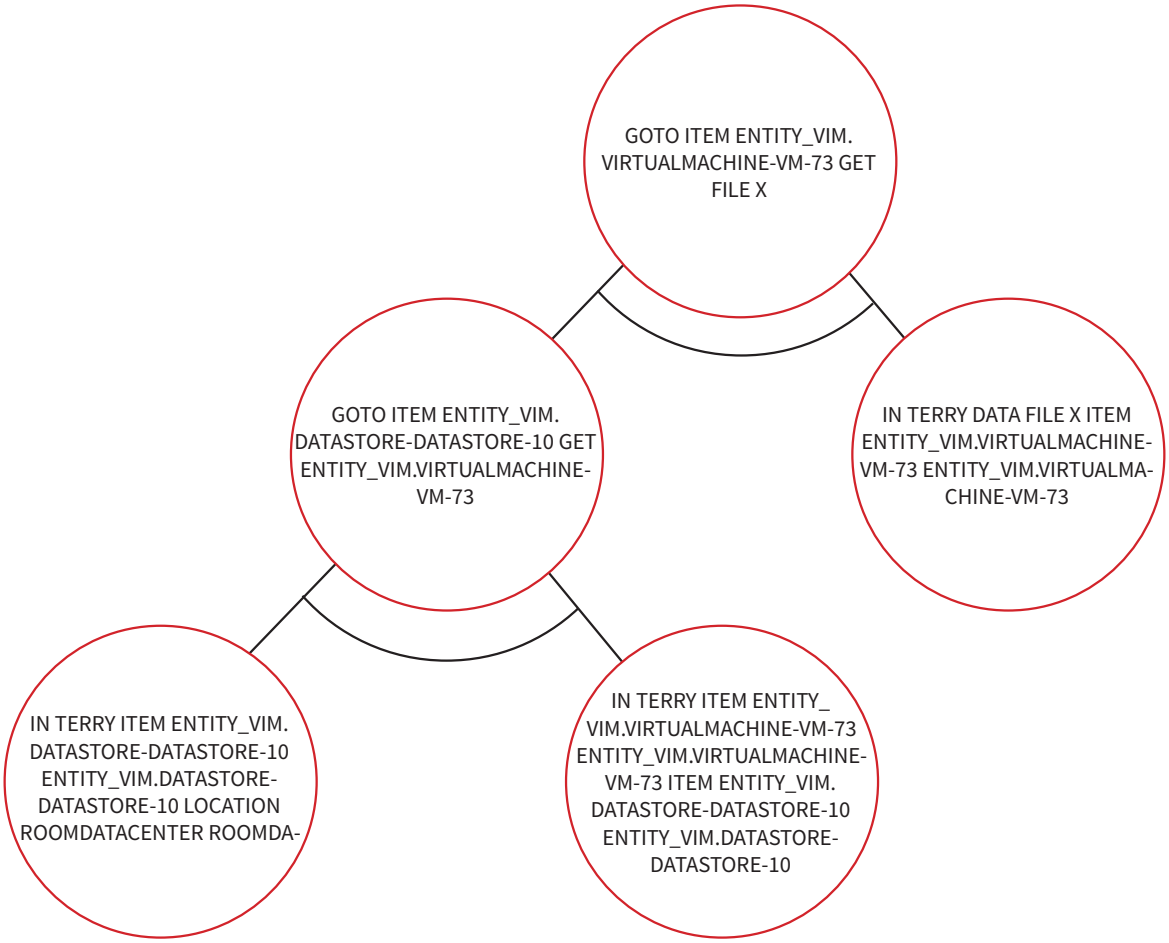


Figure 4.13.: A (redrawn) attack tree generated by TreeMaker for one actor. The scenario from Figure 4.12. was exported to the TREsPASS Model in the TREsPASS XML-format. The TREsPASS Model could then generate attack trees from this, which can be augmented and annotated. Vulnerabilities can be visualised in a dashboard-like manner to get an insight in where a user should focus and spend resources on. Policies are not yet included in this example, but will be added in the final instantiation of the Attack Navigator Map.

Demo

The best way to explore the Attack Navigator Map user interface and its workings is the prototype. This prototype is a web application and can be found on `trespass.lustlab.net/proto/anm`, as well as a video walk-through that explains the map creation process step-by-step.

4.2. Used languages and libraries

An overview of the programming languages, libraries and frame-works we currently use, or have the intention to use in future iterations.

trespass.js
trespass.js is the core library used in the Javascript-based (node.js and web-based) tools in the TREsPASS project. It provides reusable methods that facilitate working with input data like models or attack trees, as well as utility functions for developing visualisations and prototypes, without having to reinvent the wheel every time.

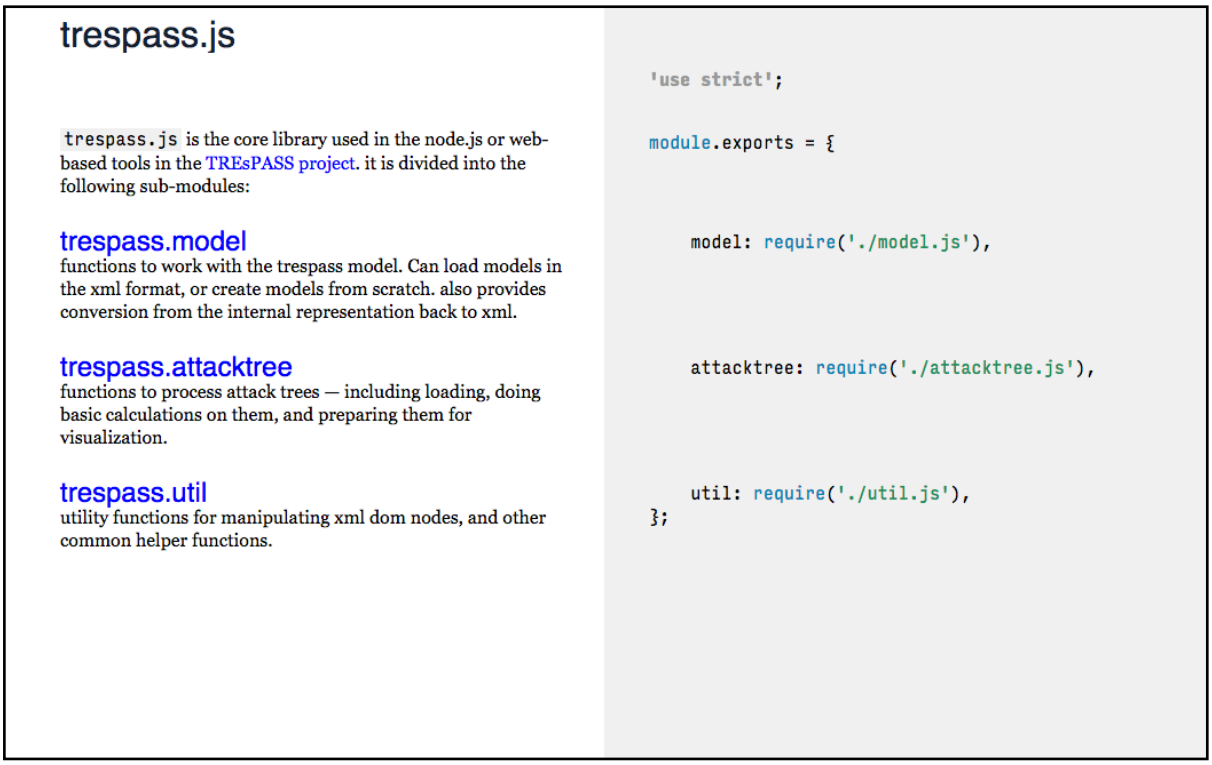
The library is divided into different sub-modules, serving different purposes:

- **trespass.model:** Functions to work with the TREsPASS model. Can load models in the TREsPASS XML format, or create models from scratch. Also provides conversion from the internal representation back to XML.
- **trespass.attacktree:** Functions to process attack trees including loading, doing basic calculations on them, and preparing them for visualisation.
- **trespass.util:** Utility functions for manipulating XML DOM nodes, and other common helper functions.

Both command line tools like *graphml-to-stm-converter,* and web-based applications like the Attack Navigator Map have trespass.js as a dependency.

Other libraries used:

- **react:** A web application framework (https://facebook.github.io/react/)
- angular: A web application framework (https://angularjs.org/)
- **bootstrap:** An HTML, CSS, and JS framework (http://getbootstrap.com/)
- **docco:** a Javascript documentation generator (https://jashkenas.github.io/docco/)
- **keystone:** node.js CMS & Web Application Platform. The open source framework for developing database-driven websites, applications and APIs in Node.js. Built on Express and MongoDB. (http://keystonejs.com/)



trespass.js

`trespass.js` is the core library used in the node.js or web-based tools in the TREsPASS project. it is divided into the following sub-modules:

trespass.model
functions to work with the trespass model. Can load models in the xml format, or create models from scratch. also provides conversion from the internal representation back to xml.

trespass.attacktree
functions to process attack trees — including loading, doing basic calculations on them, and preparing them for visualization.

trespass.util
utility functions for manipulating xml dom nodes, and other common helper functions.

```
'use strict';

module.exports = {

    model: require('./model.js'),

    attacktree: require('./attacktree.js'),

    util: require('./util.js'),
};
```

Figure 4.14.: Description of trespass.js as it is documented using docco[3].

3 http://jashkenas.github.io/docco/)

5. Conclusions and plans

The current version of the prototype is the result of task T6.3 and is based on the gathering and interpretation of the requirements, the formulating of the design brief and brainstorms and discussions with WP6, and other work packages, in the last 36 months.

The first prototype reflected the process of getting grip on the various elements that needed to be accounted for and connected already many elements. It showed the main principles and how they could potentially be applied to various instances within the interface. Except for visual changes, the main user interface interaction scheme did not change very much in the second iteration. Most effort was spent in Y3 on the Attack Navigator Map, as this is the most complicated tool in this task that should work together with other tools via tool chains. The integration and streamlining of the input and output of the various tools was therefor one of the most important focus points, that does not stop in M36, but will continue till M48. This meant close cooperation with other work packages, mostly with WP1 for the TREsPASS model language, and with WP3 for the analysis tools. The Attack Navigator will also function as a data gathering tool (WP2).

Another important part of the current prototype is the way a user can create and build maps in the Attack Navigator Map, from scratch or from other tools, edit these maps and improve them, analyse and visualise threats and vulnerabilities on them and present different views for interpretation and action.

5.1. Plans for M36–M48

Progress on the final iteration of the prototype is underway. Year 3 was mainly spend with integration between the various tools and components and how they work together. At the end of Y3 the implementation of the user interface for the Attack Navigator and tools as the Attack Navigator Map are finished, and the interface is up and running. Y4 will be spend on user testing, deploying, debugging, further integration of all tools and making sure the underlying model is integrated in the most optimum manner.

Up till now the tool has been mainly tested with the Cloud Center case study. This is a good example since this case study concerns physical and virtual spaces, as well as physical and virtual infrastructure. The scalability of the map and the scenarios it can handle is one of the major focus points in Y4. In the design, large scenarios are foreseen, but handling thousands of nodes in a fluid manner can be a challenge and is a focus point that is crucial.

As the model (WP1) and various tools (mainly WP3) are still being developed and improved, also the ANM will be worked on, tested, and refined. We will try to make the user's workflow as fluid as possible. Also in Y4 we expect to refine the design and visualisation elements since development in Y3 was largely technical, trying to understand all tools, and communicate how those tools can work together. Last but not least, also a user manual will be developed for the ANM, to provide an easy understanding of the user interface, and the TREsPASS conceptual framework where the user interface and user experience are based upon.

References

The TREsPASS Project, D1.1.2. (2015). *Final specifications and requirements for socio-technical security models* (Deliverable D1.1.2.)

The TREsPASS Project, D4.1.2. (2015). *Final requirements for visualisation processes and tools* (Deliverable D4.1.2.)

The TREsPASS Project, D5.3.2. (2015). *Best practices for model creation and sharing* (Deliverable D5.3.2.)

The TREsPASS Project, D5.4.1. (2015). *First report on the integrated TREsPASS process* (Deliverable D5.4.1.)

The TREsPASS Project, D6.1.1. (2013). *Initial requirements for tool integration.* (Deliverable D6.1.1.)

The TREsPASS Project, D6.2.1. (2013). *Initial refinement of functional requirements.* (Deliverable D6.2.1.)

The TREsPASS Project, D6.2.2. (2015). *Refinement of functional requirements* (Deliverable D6.2.2.)

Appendix

A.1 Project Summary

This chapter gives an overview of the TREsPASS project and its use cases. The section is shared by the public deliverables to provide the necessary background and to put the current deliverable in context.

Information security threats to organisations have changed completely over the last decade, due to the complexity and dynamic nature of infrastructures and attacks. Attacks like StuxNet involve technical and human factors, and they damage physical infrastructure. The recent attack on a German steel mill[4] was a combination of both targeted phishing emails and social engineering attacks. The phishing helped the hackers extract information they used to gain access to the plant's office network and then its production systems. As a result, the technical infrastructure of the mill suffered severe damage.

The attack on the German steel mill illustrates that we need to integrate the social and technical aspects of systems in assessing their security - and we need to do so today. Socio-technical systems pose new challenges by combining parts for which we often understand the security issues; the combined system is however much more complex due to interactions between these parts.

The main innovation of the TREsPASS project is the attack navigator, a tool and metaphor that enables defenders to predict and preventing attacks on socio-technical systems. The attack navigator supports current risk-assessment techniques with the TREsPASS process (developed in Work Package WP5), an analytical approach to identifying attacks and evaluating their impact.

The four main stages in the TREsPASS process are *data collection*, *modelling*, *analysis*, and *visualisation*. Data collection WP2) is vital to understanding the nature of a scenario and providing input to subsequent tasks of modelling, analysis and visualisation. Within the project, the focus has been on collection and analysis of social, technical and physical data and the ways in which these relate to one another. Within each of these domains, different approaches have been taken to provide different viewpoints on the nature of the organisation being investigated.

4 BBC News, *Hack attack causes 'massive damage' at steel works*, `http://www.bbc.com/news/technology-30575104` last visited October 31, 2015

The models (WP1) developed in TREsPASS can be adapted to the application scenario. We have developed physical modelling techniques in order to understand where further investigation may usefully be targeted. The TREsPASS model describes relevant aspects of the organisation and their connections. To explore contractual and commercial relationships, the e3value method has been adopted.

The analysis methods (WP3) developed in TREsPASS identify attacks in models and identify the most effective controls to prohibit these attacks. The analyses are supported by tools and together they provide the defender with a comprehensive understanding of properties attacks, *e.g.*, cost for the attacker, required skills, or required time.

The innovative visualisations (WP4) developed in TREsPASS focus focus particularly on visualising elements of the analysis, as this is key to the overall project goal of providing decision support to practitioners. However, visualisations contribute also to model development and data gathering.

Practitioners can access the TREsPASS toolkit via the attack navigator map interface, which provides an intuitive means of selecting appropriate tools (WP6) for data gathering, modelling, analysis and visualisation. These can be used, individually or in combination, to strengthen operational and strategic decision-making.

A.1.1. Case Studies

The TREsPASS process and tools are validated by means of case studies (WP7) in the area of cloud infrastructure, telecommunications infrastructure, ATM infrastructure, and an organisation processing privacy sensitive data.

A cloud infrastructure shares infrastructure within or across organisations, giving the cloud services provider and its employees full physical and logical access to all resources across the different consumers. In TREsPASS we formalise typical components in cloud infrastructures as well as human actors and their interrelationships, to identify their contribution to attacks on the organisation.

In telco infrastructure new products need to be launched under significant time pressure, often opening up loopholes for so-called knowledge insiders who know the market very well, trying to make as much monetary gain from the new products as possible. In TREsPASS we model both the infrastructure and contractual relationships to identify physical and monetary attacks.

The ATM infrastructure connects machines that are composed of a money safe and a computer that controls the ATM's devices. There are well protected ATMs installed inside bank branches, while others are deployed in the street and some are not even

embedded in a wall. ATM attacks are common and include classic physical attacks and emerging digital attacks. In TREsPASS we model ATM installations, and identify attack likelihoods using geospatial data.

The organisation processing privacy sensitive data develops a system supporting primarily elderly and disabled people in performing online payments and managing their own money from their home. This case study involves from strictly technical security aspects, such as how information is protected while stored or transmitted, to socio-technical security aspects covering security issues arising from the use of and interaction with the technology. In TREsPASS we identify social-engineering and trust-based attacks on such systems.

A.1.2. Overview of TREsPASS Integration

The TREsPASS workflow involves several stages with various activities, some of which are optional. Figure A.2 shows the architecture diagram and Figure A.1 shows a visual description of the notation used. In practice, stages may not follow a linear order. For example, depending on the goal of the risk assessment, new data requests may be issued later in the process, or automatic updates of data may be supported.

The **Data collection stage** prepares for analysis and modelling steps, and may require the gathering of one or more of the following kinds of data.

Physical data collection provides knowledge about the physical layout of the organisation including locations, buildings, rooms, doors, windows, etc.

Digital data collection gathers information about the organisation's IT infrastructure.

Social data collection focuses on organisational and individual data, and results in actor profiles containing**,** *e.g.*, attributes of employees, stakeholders, or potential attackers.

Commercial data collection gathers information required for *e3fraud* analyses, which focus on potential fraud.

Stakeholder goal collection identifies assets and policies the protection of which is critical to one or more stakeholders.

The **model creation stage** handles the creation of the TREsPASS model and associated actor profiles. The e3value model creation process is complementary to the main TREsPASS model, for cases requiring a more specific financial focus:

TREsPASS model creation is a key activity result in a system model that can be further extended and analysed.

Components customisation (optional) takes place before or during the TREsPASS model creation to create specialised custom model components.

Attacker profile creation creates the attacker profile that the TREsPASS model analysis should consider, based on ready-made attacker profiles.

Defender/target profile creation creates similar profiles for the other actors in the model based on the social data gathered in the social data collection activity.

e3value model creation This interactive activity involves using the *e3value* toolkit[5] to create business value models. These models structure the commercial information gathered in the data collection stage in a formal way.

In the **analysis stage** different analyses are possible depending on the model chosen. The analysis of the TREsPASS model involves these steps:

3. In the **attacker profile selection**, the user selects the attacker profile to use in the analysis.
4. The **attacker goals creation** provides the attack generation with the attacker goals. These can be derived by hand from the stakeholder goals or deduced automatically from the selected attacker profiles.
5. The **scenario selection** selects a scenario, consisting of a single pair of attacker and attacker goal, to run the TREsPASS analysis on.
6. To extend attack trees, **attack pattern creation and sharing** provides libraries with known attack steps. The attack tree generation can only reach a certain level of abstraction, which may not be sufficient for quantitative analyses.
7. **Attack generation** transforms the TREsPASS model to an attack tree.
8. **Attack tree annotation & augmentation** then extends the attack tree with attack patterns and decorates leaf nodes with parameter values from the data collection stage for quantitative analysis.
9. The **attack tree analyses** compute quantitative properties of attacks, *e.g.*, utility for the attacker or success probability of the attack.

The analysis of the *e3value model* is complementary to the core TREsPASS analysis and has only one step:

1. For the **fraud model generation**, the user needs select an attacker and an interval of expected occurrence rates of the commercial transactions specified by the e3value model. The e3fraud tool then identifies all possible violations of contracts, the loss for actors, and the delta in profit for the other actors.

5 http://e3value.few.vu.nl/tools/

The **visualisation stage** can be used continuously to provide practitioners with feedback regarding the results of their activities:

1. **Fraud model visualisation** shows the generated attacks as a ranked list of textual descriptions of the attack steps and displays charts showing the profitability for each actor.
2. **Attack tree visualisation** shows the intermediary attack trees.
3. **Attack tree analysis visualisation** visualises analysis results.

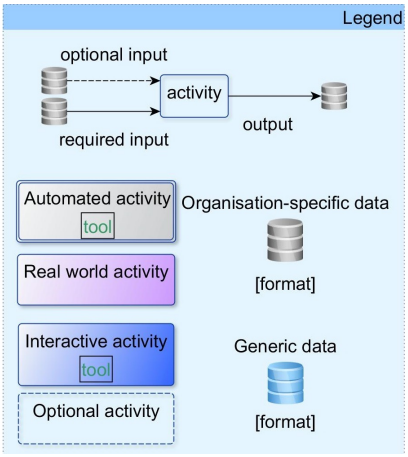


Figure A.1: Legend for the Integration diagram in Figure A.2.

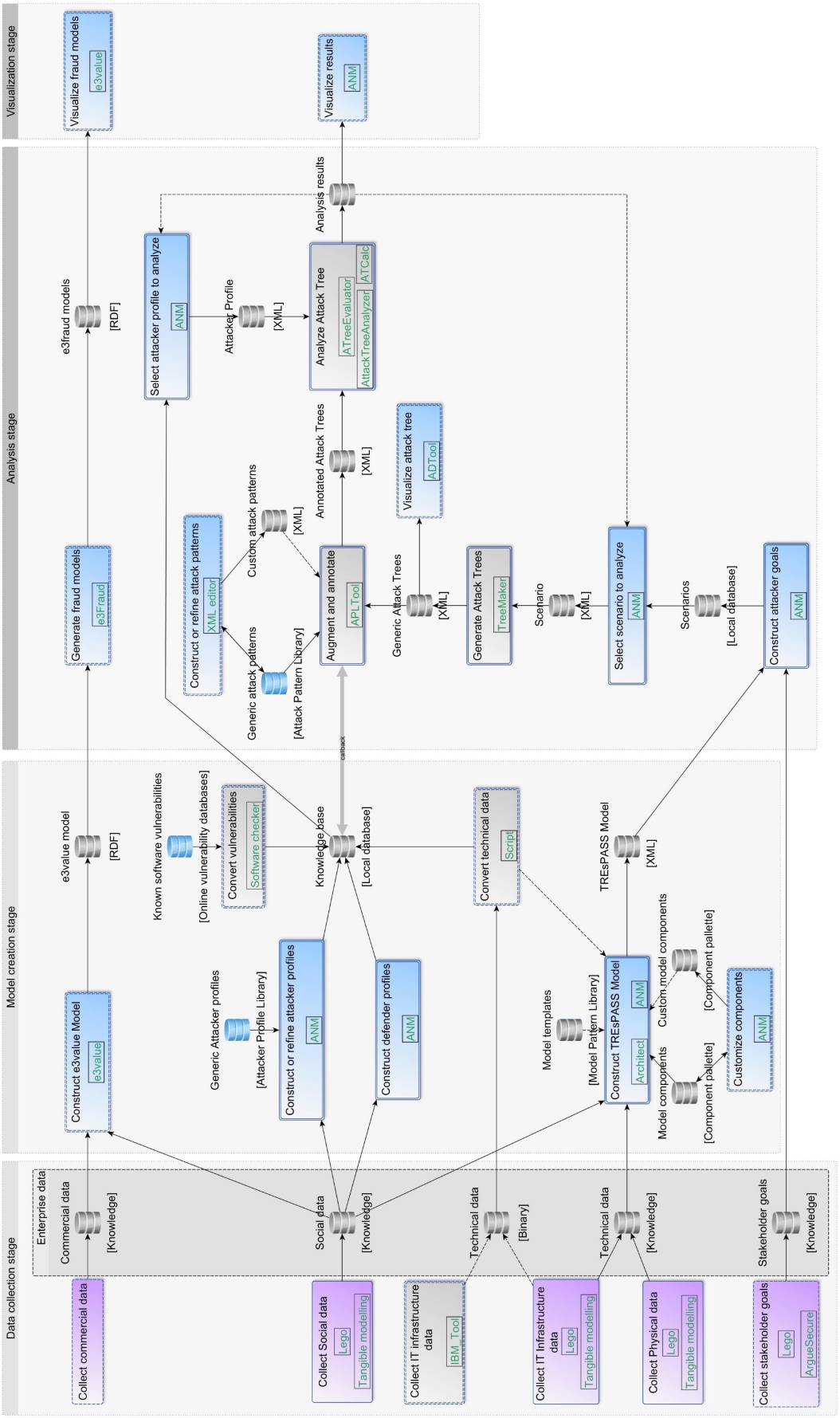


Figure A.2: Integration diagram for the TREsPASS project.