predict
prioritise
prevent

TRESPASS

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

Deliverable D6.2.2

Final refinement of functional requirements

Project:	TRE$_S$PASS
Project Number:	ICT-318003
Deliverable:	D6.2.2
Title:	Final refinement of functional requirements
Version:	1.0
Confidentiality:	Public
Editor:	Roel Wieringa
Cont. Authors:	D. Ionita, W. Pieters, R.J. Wieringa, M. Martins, M. Ford, C.W. Probst
Date:	2015-10-30

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

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

Document History

Authors		
Partner	Name	Chapters
UT	Dan Ionita	1, 2, 3, 4, Appendices
UT	Roel J. Wieringa	4, Appendices
UT	Wolter Pieters	2, 3, 4, Appendices
ITR	Miguel Martins	3, 4, 5
CHYP	Margaret Ford	2
DTU	Christian W. Probst	2

Quality assurance		
Role	Name	Date
Editor	Roel Wieringa	2015-10-30
Reviewer	Lizzie Coles-Kemp	2015-09-30
Reviewer	Trajce Dimkov	2015-09-30
Task leader	Roel Wieringa	2015-10-30
WP leader	Miguel Martins	2015-10-30
Coordinator	Pieter Hartel	2015-10-30

Circulation	
Recipient	Date of submission
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 (TRESPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.*

Contents

List of Figures

List of Tables

Management Summary

Key takeaways:

- This document presents the final iteration of the requirements TRE$_S$PASS functional requirements elicitation process.
- A product perspective and functional decomposition of the TRE$_S$PASS toolkit are presented
- The final system architecture and supported components are described.
- Five Use Cases used to elicit required external functionality
- Interface Analysis used to elicit integration functionality

Established requirements engineering approaches are applied to the TRE$_S$PASS requirements elicitation process in order to better structure and describe both the tool architecture and the requirements themselves.

The envisioned TRE$_S$PASS toolkit is described from a product development perspective. The requirements for such a product focus on the availability of operational features, instead of standard software development requirements. As an internal perspective, the TRE$_S$PASS philosophy and general workflow, as well as the concept of attack navigators are described. From an external perspective, the intended scope and context are clarified. A functional decomposition of the project goal is used to generate a list of service descriptions. User characteristics, project-wide constraints, assumptions and dependencies are also discussed.

We describe five Use Cases, which were used to derive required an initial set of traceable requirements. Internal integration requirements elicited between Work Packages are added are added to this list .Further sources of requirements are a set of primary requirements, identified by the project leaders.

Furthermore, a process for eliciting a set of internal requirements intended to guarantee cohesion between the tools developed by various partners is described.

1 Introduction

Within the TRE$_S$PASS project, work package 6 deals with the integration of the tools. Task 6.2, refinement of functional requirements, aims at integrating all technical developments in the project, in order to support the development of the TRE$_S$PASS tools. To this end, the task will define the architecture, including databases and file formats, to allow interoperability of different components developed in other work packages. In order to achieve this, two different requirements elicitation approaches were employed:

1. At the start of the project, a *use-case based requirements engineering* approach was used: several use cases were envisioned for the TRE$_S$PASS toolkit and these were used as initial sources of requirements (D6.2.1). The resulting requirements were refined and allocated to Work Packages (i6.2.1).
2. After the first exploratory year, an *interface analysis* approach was undertaken: each Work Package was asked to elicit requirements for every other work package. The resulting requirements were centralized in a project-wide Central Requirements Repository.

The aim of this deliverable with regard to the Task 6.2 is to present the final system architecture and supported components as well as an up-to-date, integrated view of the requirements engineering effort in TRE$_S$PASS.

Throughout this document, a system of letters is used:

Fx Product Function x

Sx Product Service x

Ux.y Desired functionality y of Use Case x

Px Primary requirement x

Rx Requirement x

1.1 Choices Made

The TRE$_S$PASS methodology and toolkit are primarily intended for conducting Risk Assessments of existing implementation, or on future or prototype implementations with sufficient technical specification. However, it is not restricted to such systems and there are plans to support, for example, product certification based on established Information Security Standards, gap analysis, and even Risk Assessment during product design.

This does however impact the choices made during the Requirements Elicitation phase of the TRESPASS project: knowledge about the target of assessment's architecture and policies is assumed, information about the intended interaction between the system and its (especially social) context is required and the tools are geared toward producing reports describing risks and possible attacks.

1.2 Definitions

The following definitions represent the concepts used within the TRESPASS Project and their consensual interpretation at the time this document was written. The full list of TRESPASS concepts and their definitions can be found in the Glossary of Common Concepts (SVN: trunk/Documents/Terminology/). A snapshot of this Glossary is included here as the Glossary itself is not (part of) a project deliverable.

1.2.1 Socio-technical systems

The basic assumption in the modelling framework is that there are **systems** that can be represented in **models**, which have **states** representing properties that can change over time.

Socio-technical system
A *socio-technical system* is a system consisting of human behaviour, technology and the policies that influence human behaviour. The key properties in the *socio-technical system* are entities, interaction possibilities, and quantitative properties associated with interactions. The quantitative properties include difficulty, risk for attacker, rewards, visibility, excuses.

Socio-technical security model
A *socio-technical security model* is a model of a *socio-technical system* specifically focused on security risks in such systems, consisting of (1) a specification of objects, relations, and capabilities, (2) a specification of possible transformations, and (3) a specification of what constitutes an attack. The socio-technical security model is made up of several types of components:

Spatial components
This refers to the geometric representation of its shape in some coordinate space. This is referred to as its geometry

Social components
A human as an entity that interacts in the model. The human can change location between rooms and can have relations with entities. Humans are actors who can be malicious or not and can interact with each other.

Locations
Entities in the spatial component.

Object component
The set of all objects.

Objects
Entities that can be moved around between locations.

Digital component
This concerns all programs and data that are present in objects supporting digital data storage, processing and communication.

Entity
An *entity* or *element* is a part of the socio-technical system that is considered separate for the purpose of analysis.

Action
An *action* is a change to the state of the socio-technical system as represented in the socio-technical security model.

Actor
An *actor* is an (in)animate object that executes actions.

Asset
An *asset* is an object related to the organisation that, when unduly accessed, modified or made unavailable would cause harm to the organisation. According to The Open Group (The Open Group, 2009), it can be any data, device, object or other component of the environment that supports information-related activities, which can be illicitly accessed, used, disclosed, altered, destroyed, and/or stolen, resulting in loss.

Policy
A *policy* is a rule regulating access to assets.

Organisation
An *organisation* is a socio-technical system consisting of people, buildings, computers, and data, that needs its assets to achieve its goal. An organisation often formulates policies as operationalisations of the requirement of being able to achieve the organisation's goals. An attack will typically violate such policies.

Vulnerability
A *vulnerability* is a condition in a system, or in the procedures affecting the operation of the system, that makes it possible to perform an action that violates the explicit or implicit security (or survivability) policy of the system.

1.2.2 Risk

Threat
A *Threat* is anything that is capable of acting in a manner resulting in harm to an asset and/or the organization (The Open Group, 2009). It is a high level concept which can refer to any potentially malicious actor, object, or event. It can refer to

a single attacker, a group or community of attackers, a class of attacker profiles, a computer, a process, a set of instructions (code), environmental factor(s) or a collection of these.

Threat event
Threat event is a threat acting in a way that causes damage to an organization.

Threat event frequency
The *Threat event frequency* is the frequency with which certain Threat events occur.

Loss
Loss or *Damage* is any harm inflicted upon the organization. According to The Open Group (The Open Group, 2009), it can be of various forms:

Productivity: a reduction of the organization to effectively produce goods or services in order to generate value

Response: the resources spent while acting following an adverse event

Replacement: the cost to substitute/repair an affected asset

Fines and Judgements: the cost of the overall legal procedure deriving from the adverse event

Competitive Advantage: : missed opportunities due to the security incident

Reputation: : opportunities or sales due to the diminishing corporate image following the event

Loss event
A *loss event* is the occurrence of loss due to the occurrence of a threat event.

Probable Loss Magnitude
Probable loss magnitude (also called impact) is the damage that occurs when an attack scenario succeeds. A Threat Event may have *impact* either on one security dimension (confidentiality, integrity, availability) or on several dimensions and this impact may be complete, high, partial, low, etc. depending on the target (e.g. compromised asset, crashed service, etc.).

Risk
The *risk* associated with a type of threat event is the frequency of occurrence of loss events due to this threat, times the expected impact of a loss event.

1.2.3 Attacks

Attacker
An *attacker* or *adversary* is an actor with goals that, when achieved, would harm the organisation. An *attacker* may cause threat events by executing actions aimed at achieving his goal. An attacker can be described by the following attributes:

Attacker goal is expressed as utility functions, mapping possible outcomes of attacks to (e.g. monetary) value for the attacker. An *utility function* is the minimal data needed to express attacker goals. The *utility for the attacker* upon reaching a goal may or may not be different from the (negative) utility (i.e. value of the asset) to the organisation (impact).

Attacker resources are the total amounts of time and money available for the attacker to invest in attack scenarios.

Attacker investment is the amount of resources (time and money) that an attacker decides to apply to the execution of a specific action or attack scenario.

Attacker skill refers to the attacker's skills in relation to a high likelihood of successful attack. Characteristics are often described today in terms of "script kiddie" or "geek". The skill level is assumed to be constant over the course of the attack (analysis). A more skilled attacker has a higher likelihood of success with fewer resources. Different skills are required for human based (e.g. dumpster diving, impersonation, technical support, shoulder surfing, piggybacking, etc.) than for computer based activities.

Attacker strategy describes the decisions made by the attacker based on the expected utility of scenarios. A *stealthy strategy* is one in which an attacker chooses an attack approach that balances the likelihood of success with the likelihood of detection: attacks that have a reasonably high likelihood of success and at the same time an acceptable likelihood of detection.

Attacker profile An *attacker profile* includes attacker motivation/goals, strategy, capabilities, resources, knowledge of the system and initial access. The attacker profile may include a general attacker skill level, or different skill levels for different types of attack steps (e.g. technical or social).

Attack

An *attack* is a sequence of one or more attack steps intended to achieve the goal of an attacker.

Exploit

An *exploit* is an action consisting of a single-step (atomic) exploitation of a single vulnerability by an attacker.

Attack step

An *attack step* is an action (exploit or other activity), potentially available to an attacker in order to reach his goal, and considered atomic for the purposes of analysis. An attack step has an associated:

Attack step difficulty (in Open Group terminology: control strength) indicates the resources and time that a particular attacker would have to spend to achieve a certain likelihood of success. *Difficulty* thus defines how hard it is to do the action or exploit the vulnerability

Attack step execution is the execution of an attack step by a specific attacker or attacker type as part of a campaign toward the adversary's goal.

Likelihood of success of an attack step execution is a value between 0 and 1 indicating the expectation value of the outcome, where 1 is success (access acquired) and 0 is failure (in Open Group terms this is called “vulnerability (level)”). The likelihood of success is dependent on the difficulty of the attack step, the skill of the attacker, and the resources spent by the attacker.

Attack event

An *attack event* is an action that contributes to the adversary’s goal, but is not controlled by the attacker. An attack event has an associated *mean time to failure*, which indicates the average time an attacker would have to wait before the event occurs.

Attack scenario

An *attack scenario* or *(composite) attack* is a (partially ordered) collection of one or more attack steps and zero or more attack events, leading to an attacker’s goal, and constituting a threat event for the organisation. The likelihood of success of an attack scenario depends on the likelihood of success of the steps of which it is composed.

Attack execution is the execution of an attack scenario by a specific attacker or attacker type as part of a campaign toward his goal.

Threat Capability

The **threat capability** an attacker applies in an action is the combination of his skill and his investment.

Countermeasure

A *countermeasure* is a means to reduce the risk of attack, typically by decreasing the attacker’s expected utility. A countermeasure can be technical (for example, the use of encryption to protect a communications link) or procedural (for example, the implementation of dual controls) or physical (for example better locks on doors).

Reward

The *reward* for the attacker consists of the expected benefit when succeeding in getting access to a certain asset.

Utility

The *utility* of an attack scenario to the attacker is the expected value he obtains from executing it. This value depends on the expected likelihood of success, the expected reward upon success, the expected investment, probability of detection, etc. How these components are mapped to a utility value is dependent on the attacker profile (see above). This can be expresses as a *utility function* in the attacker profile. The utility for the attacker upon reaching a goal may or may not be different from the (negative) utility (i.e. value of the asset) to the organisation (impact).

Excuse

An *excuse* relates to the social environment in which attacks occur. Depending on the subcultures within an organisation, it may or may not be clear what is acceptable behaviour and what is not. Excuses may influence attacker strategy.

Provocation
Provocations may occur if the location and means of access of valuable assets are easily visible to the attacker. For example, laptops present on the ground floor are visible through a window. Such provocations may influence attacker strategy.

1.2.4 Navigator Maps and Attack Trees

Attack Tree
An *attack tree* is a hierarchical, graphical diagram for representing and analysing an attack scenario.

Attack navigator An *attack navigator* is a tool to support prediction, prioritisation and prevention of complex misuse scenarios.

Attack navigator map An *attack navigator map* is a graphical diagram which presents the relevant infrastructure in relation to the relevant assets. All possible attack scenarios that allow attackers to achieve their goals can be generated.

Visualisation
Visualisation is either a) forming a mental image of something or imaging it, or b) making something visible to the eye.

Expressivity
Expressivity refers to the power or potential to affect a viewer. Expressivity is an index not just of the power to evoke through visuality, but also it is an index of receptivity, an index of audience reactions.

1.3 Document structure

The overall document structure is based on established templates for requirements documents: IEEE Standard 830-1998 for Software requirements specification (IEEE., 1998) and the VOLERE Requirements Specification template (Robertson & Robertson, 1998).

Section 2 will provide a high-level description of the TRE$_S$PASS toolkit, including the underlying concept, the desired functionality and envisioned workflow, as well as the larger context is intended to operate in. Section 3 describes the Use Cases and one of the case studies used to extract required functionality. Section 4 will elicit and describe all the functional requirements identified thus far, both derived from Use Cases and from the interface analysis.

2 Overall description

2.1 Product perspective

This section aims to describe the TRE$_S$PASS approach and workflow, as well as its intended context. This serves to place the TRE$_S$PASS toolkit in the wider IT Management and Risk Management fields and describe its relation to other tools/systems, thus providing a high-level view of its intended use and added value.

2.1.1 The attack navigator concept

The main innovation of the TRE$_S$PASS 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 TRE$_S$PASS process (developed in Work Package WP5), an analytical approach to identifying attacks and evaluating their impact.

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 skill, 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.

Attack navigators are similar to attack trees, in the sense that they can describe and analyse attack paths. However, the following constitute innovative differences:

- Attack navigators start the analysis based on a map with multiple assets, rather than based on a tree with a single root goal. This provides a link between enterprise architecture and risk management, and enables analyses in which a single weakness puts multiple assets at risk;

- Attack navigators employ explicit attacker profiles as part of the analysis. This enables renewed analyses when the threat environment changes;
- Attack navigators consider visualisation as an integral part of the tools. It is possible to derive an attack tree from a navigator map plus an attacker goal. Attack trees will be used in this way in the early phases of the project, as an intermediate stage between the map and the analysis. It is foreseen that the analyses will work directly on the map in later stages.

2.1.2 Scope

As the term cyber security is open to interpretation, this section describes which parts of what is commonly understood as cyber security are addressed by TRE$_S$PASS, and which are not.

2.1.2.1 Target of analysis

The tools and processes are targeted towards organisations that could be a target of attacks, either because of critical availability of services, criminal business cases involving fraud, or espionage.

The *scope* of the models includes those aspects of organisations or information and communication services relevant to decision making in the context of cyber security risk assessment. This means that the scope includes only those risks that involve (a) malicious behaviour and (b) access to information systems (such as socio-technical systems). The scope of the models does not include attacks of sovereign states upon organisations.

The models are meant to cover a single organisation as target of analysis, but with support for external features and services provided across organizations such as the cloud services. It may be possible to extend the models to cover supply chains or value networks as well, but this is considered out of scope for this project. The precise system boundaries are determined during the process, either in a dialogue between the consultant / auditor and the organisation (heavyweight process and tools), or implicitly by the selection of the objects of interest (lightweight process and tools). This process should also identify the assets and security goals of interest, e.g. availability of a service, or confidentiality / integrity of certain data entities.

The models should be capable of dealing with both preventive and detective controls. Preventive controls will increase the difficulty of attack scenarios (and thereby the effort required on the part of the attacker), whereas detective controls will increase the likelihood of detection (and thereby the risk for the attacker). Both are meant to reduce the frequency of successful attacks. Effects of both types of controls are relative to the attacker profile, as the decision to attack is in the end up to the attacker, and some types of attackers may not care that much about being detected, for example if they believe the activity cannot be traced to their physical identity.

TRESPASS focuses in particular on integrating technical and social aspects of cyber attacks and defences in a single model. In this context, social engineering refers to those attack steps where attackers interact with people inside the organisation being attacked, by using deception, manipulation and persuasion, or indirectly exploit the behaviour of the employees.

2.1.2.2 Threats

Risks involving only non-malicious behaviour are out of scope. However, it may be relevant to investigate the possibilities for integrating non-malicious events in the threat models, for example when someone forgets to lock a door.

In addition, even malicious cyber security threats sometimes behave like non-malicious threats. For example, viruses are not always targeted at a specific organisation; they roam the network and may hit a particular company. Such malicious threats behave like non-malicious threats, in the sense that the threat does not adapt to countermeasures taken, and does not attempt a different attack vector if the initial path is blocked. In this sense, there is a spectrum from malicious behaviour to non-malicious behaviour of cyber security threats, and the tools should take this into account.

Although the tools may be able to identify possible attack scenarios for terrorists, the tools and processes are *not* meant to defend against cyber warfare as a warfare domain nor against the actions of the state against organisations, and do not cover offensive capacities.

The attacker may be anything from a lone hacker to what is called an Advanced Persistent Threat. In the former case, the resources are limited, and the attacker may be diverted to another organisation by sufficient countermeasures. In the latter case, available resources are high, and it is unlikely that countermeasures will deter the attacker. The distinction between possible types of attackers is represented in attacker profiles, and different profiles will be relevant for different systems. Such attacker profiles may for example be based on the three-step classification of Abraham, Dolan, Double, and Stevens (1991), or similar ones.

Different types of criminals (e.g. criminals, gangs, insiders) can commit exactly the same legal offence but with very different goals in mind and hence different modus operandi (i.e. MO). The modus operandi involves the evaluation of the actions used by the attackers to execute the crime, prevent its detection and/or facilitate escape. The relevance of the MO for the TRESPASS project is that it can be used to determine links between attacks.

2.1.2.3 Analysis

The aim of the analysis is evaluating cyber security risk from the combination of a navigator map of an organisation or product, and a set of attacker profiles. This is done

by deriving possible attack scenarios and their associated risk. This analysis can be repeated for different versions of the model, thereby providing differences in risk induced by countermeasures.

The TRE$_S$PASS project does not develop new tools or processes for impact analysis (asset identification and valuation), but will rely on existing methods. Suitable methods will be identified in (The TRE$_S$PASS Project, D5.2.1, 2014).

In general, the imperfect predictive power of past results also applies to risk assessment. Therefore, any risk assessment method may fail to predict future events. In cases of adversarial risk assessment (Rios Insua, Rios, & Banks, 2009), overall uncertainty is increased due to uncertainty about the attacker profiles. It is therefore by no means implied that the TRE$_S$PASS tools would have failed if they do not predict the next attack. Rather, they are meant to give a reasonably accurate risk picture under a given organisational model and one or more attacker profiles.

When different types of analysis are available for the same purpose, the tool set should support triangulation: multiple analyses for the same issue. This allows the comparison between results from different but complementary methods addressing the same issues and can increase the user's confidence in the results.

For events beyond the attacker control (attack events), it is relevant how often such events occur (or what percentage of the time a particular state occurs, like a door being open), and how often the attacker will be present to exploit the event. For attacker-controlled events (attack steps), the likelihood of success may depend on a number of factors. These include organisational culture, the relation between the attacker and the victim, and the emotional involvement of the victim.

Therefore, social aspects of attackers, defenders, and their relation are relevant for the analysis. In particular, this requires the analysis tools to be able to deal with multiple variables describing attack steps. For example, an attacker may invest in authority to increase the success probability of social engineering, but authority is neither a necessary nor a sufficient condition for success. A distinction can be made between different types of social engineering:

- passive assistance of people in an attack, for example by not reporting suspicious activity, throwing confidential documents in a bin in a public area, allowing an attacker to shoulder-surf or tail-gate;
- active assistance of people in an attack, for example by opening a door for or providing a password to the attacker.

This distinction is related to the distinction between events controlled by the attacker (although the attacker may not be successful), and events not controlled by the attacker. For example, in case the attacker asks for a password over the phone, this event is under his control. However, if the attacker has to wait until someone accidentally leaves a door open, this event is not under his control.

Although the risk analysis methods support reasoning with probabilities, it is acknowledged that users may be unable to supply or interpret these. Therefore, queries and

(intermediate) results involving probabilities should be hidden from the user if possible. In addition, the tools should allow "graceful degradation": be able to provide the best possible results if certain inputs are unavailable or not amenable to quantification. The details will be investigated in task 2.5.

2.1.3 Workflow

The four main stages in the TRE$_S$PASS process are *data collection*, *modelling*, *analysis*, and *visualisation*. The stages may have various activities, not all of which are required for every risk assessment scenario. Some TRE$_S$PASS processes are manual, others are digital. These processes are described in more detail in Deliverable D5.4.1. Most activities are supported by dedicated tools within the TRE$_S$PASS tool-chain. The tools facilitating each activity are shown in the architecture diagram presented in Figure 2.

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. 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 organization including locations, buildings, rooms, doors, windows, etc.
- **Digital data collection** gathers information about the organization'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 **models** (WP1) developed in TRE$_S$PASS 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 TRE$_S$PASS model describes relevant aspects of the organisation and their connections. To explore contractual and commercial relationships, the e3value method has been adopted. The *model creation stage* handles the creation of the TRE$_S$PASS model and associated actor profiles. The e3value model creation process is complementary to the main TRE$_S$PASS model, for cases requiring a more specific financial focus:

- **TRE$_S$PASS model creation** is a key activity result in a system model that can be further extended and analysed.

  - **Components customization (optional)** takes place before or during the TRE$_S$-PASS model creation to create specialized custom model components.
  - **Attacker profile creation** creates the attacker profile that the TRE$_S$PASS 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* [1] to create business value models. These models structure the commercial information gathered in the data collection stage in a formal way.

The **analysis** methods (WP3) developed in TRE$_S$PASS 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. In the *analysis stage* different analyses are possible depending on the model chosen.

- The **analysis of the TRE$_S$PASS model** involves these steps:
  - In the **attacker profile selection**, the user selects the attacker profile to use in the analysis.
  - 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.
  - The **scenario selection** selects a scenario, consisting of a single pair of attacker and attacker goal, to run the TRE$_S$PASS analysis on.
  - 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.
  - **Attack generation** transforms the TRE$_S$PASS model to an attack tree.
  - **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.
  - 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 TRE$_S$PASS analysis and has only one step:

[1] http://e3value.few.vu.nl/tools/

  - 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.

The innovative **visualisations** (WP4) developed in TRE$_S$PASS 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. The *visualisation stage* can be used continuously to provide practitioners with feedback regarding the results of their activities:

- **Fraud model visualisation** shows the generated fraud scenarios as a ranked list of textual descriptions and displays charts showing the profitability for each actor.
- **Attack tree visualisation** shows the intermediary attack trees.
- **Attack tree analysis visualisation** visualises analysis results.

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.

In practice, the steps may not follow a linear (waterfall) 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. As such, five Use Cases have been defined to reflect this variety in goals and the impact it has on the workflow. This is discussed in more detail in the Use Cases in chapter 3. These cases are also used for the functional requirement elicitation.

2.1.4 External relationships/interfaces

Figure 2.1 shows a minimal set of stakeholders and systems that the TRE$_S$PASS toolkit is expected to interact with. The types of data that flow between this context and the TRE$_S$PASS toolkit is further detailed in Figure 2.2.

2.1.4.1 Integration into (IT/IS) Risk Management process

The context in which the TRE$_S$PASS tools operate is a risk management process, the details of which will be identified by Work Package 5. This context provides requirements in terms of the conceptual framework to be used, as well as the workflow in which the different functions and corresponding requirements should fit.

According to the European Network and Information Security Agency (ENISA), Risk Management (RM) is "a process aiming at an efficient balance between realizing opportunities for gains while minimizing vulnerabilities and losses" (ENISA Technical Department, 2006). Furthermore, it is itself part of the management practice. Information Security Risk

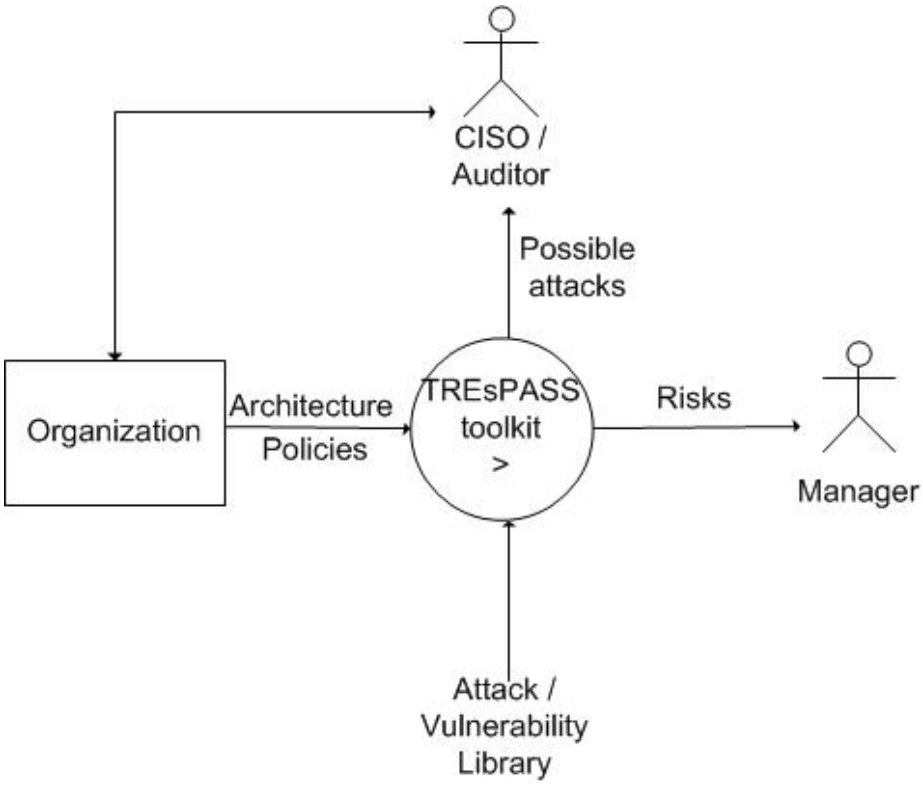

Figure 2.1: TREsPASS Context Diagram

Figure 2.2: TREsPASS Communication Diagram

Management, in particular, can either be part of the overall organisational Risk Management process, or can be implemented separately (Network & Agency, 2013).

Risk Assessment is a critical part of any Risk Management process as it involves obtaining a consistent overview of the current risk landscape. This usually implies identifying and analysing possible vulnerabilities of and threats to a given system, as well as the relative value of assets and possible damage resulting from their compromise. The TRESPASS process aims at achieving a similar result, so it will inevitably be integrated into a higher-level Risk Management process which might include such activities as: implementing appropriate policies and related controls, promoting awareness, as well as monitoring and evaluating policy and control effectiveness.

In particular, The TRESPASS Project, D5.1.1 (2013) has identified the Risk Taxonomy of The Open Group (2009), based on the Factor Analysis of Information Risk (FAIR) framework, as the basic conceptual framework for the risk management context. Definitions resulting from this framework will be presented in the next section.

Relevant input for an organization's Risk Management process that could be obtained by using the TRESPASS tool might be (Ionita, 2013):

- eliciting security requirements;
- aiding in the choice and specification of countermeasures;
- evaluating current Security Policies;
- assessing existing protection mechanisms or controls.

2.1.4.2 Compliance to standards

Current established Information Security standards and Risk Management methodologies were reviewed and compared to the TRESPASS approach, both conceptually and procedurally in D5.2.1. Since the TRESPASS toolkit should compete with current available methods, it is important that it at least provides comparable features. Furthermore, in order to improve adoption, it is crucial that controls and processes from relevant ISO standards, such as ISO 27001, 27005 and 13335 are taken into consideration when designing the TRESPASS workflow. However, the toolkit itself will not be explicitly compliant or show compliance to these standards. In other words, at this stage, compliance to standards is a desire-able property but not a requirement. Integration with/of ISO standards will be addressed in subsequent deliverables.

2.1.4.3 Integration into Security investment-making process

For informing enterprise security investment processes, the TRESPASS toolkit may be used to identify and rank a variety of desire-able security mechanisms:

- Social – e.g. policies / training / HR measures;
- Technical – e.g. architectural, implementation and operational;
- Physical – e.g. secure buildings, possibly with CCTV, segregated areas for particular purposes, strong access control, detailed record-keeping.

However, these recommendations are not specified at an implementation level. They might suggest types of access control or record keeping required but not particular devices or software to accomplish these. This is to prevent offering dated recommendations.

Furthermore, these recommended countermeasures should not be viewed as requirements but rather as suggestions towards achieving a higher level of security. In every case, the decision makers should also consider their own personal knowledge of the organisation when considering security investments. The reverse is also valid: managers

should have strong reasons to justify ignoring a strong recommendation or one that mitigates a risk that has been evaluated as high.

2.1.4.4 Capabilities for (Security) decision support

While the TRE$_S$PASS toolkit is not primarily intended towards decision support, it would be beneficial if the tools are able to offer a sufficient level of trace-ability as to also allow such usage. Not only will TRE$_S$PASS provide a range of reports, including impact estimation (+ve and -ve) and sensitivity analysis, but also means to change parameters and observe changes in the outcome. Types of decisions that might be supported by the TRE$_S$PASS analysis includes: implementation, investment, insurance or SLA clauses. The level of this capability has not yet been decided within the project but we expect to be able to provide at least some decision support at a management level. This is due to the fact that the TRE$_S$PASS process might be too cumbersome for operations and not be able to get appropriate input to being used at a technical level.

2.2 Product functions and services

A decomposition of the TRE$_S$PASS goal into functions (F) is depicted in Figure 2.3. Although it does not accurately represent the division into modules, it provides on overview of the core functions to be provided by the TRE$_S$PASS toolkit.

The services (S) corresponding to the functions (F) listed above are described following a standard Service Description notation (Wieringa, 2003) in the respective sub-sections below.

F1: Model development and maintenance

Users can develop and/or select navigator maps and attacker profiles. Model parameters can be entered manually, or by calls to data extraction tools. A first model development prototype was delivered in month 12 as part of D1.3.1. An overview of this module is shown in Figure 2.4.

S1.1: Model generation from ArchiMate

Description Based on a user selected ArchiMate model, a corresponding navigator map will be generated in the system. Where data or parameters are unavailable in the ArchiMate model, default values will be used. The generated navigator map can be modified, in the system, for instance to add features, parameters, and annotations specific to the navigator map. An example of the transformation from ArchiMate model into navigator map is shown in Figure 2.5.

Triggering event An ArchiMate enterprise architecture model is loaded.

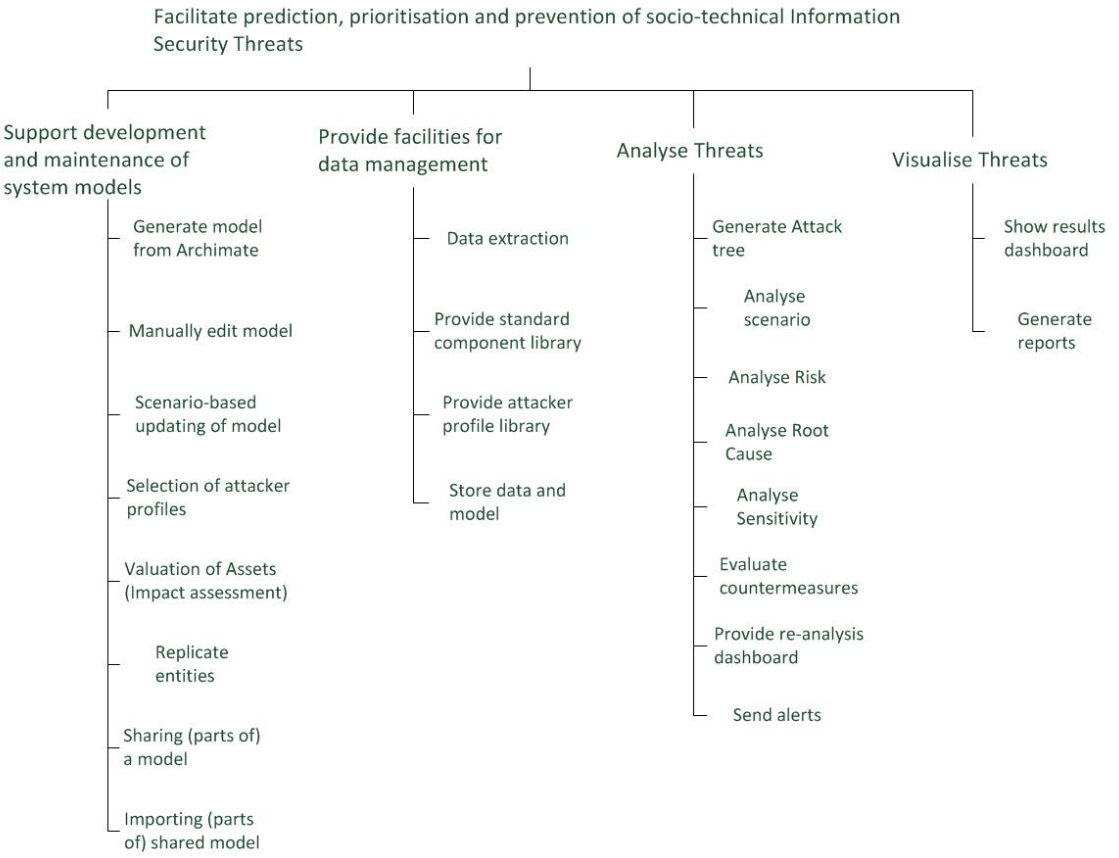

Figure 2.3: Function Refinement Tree

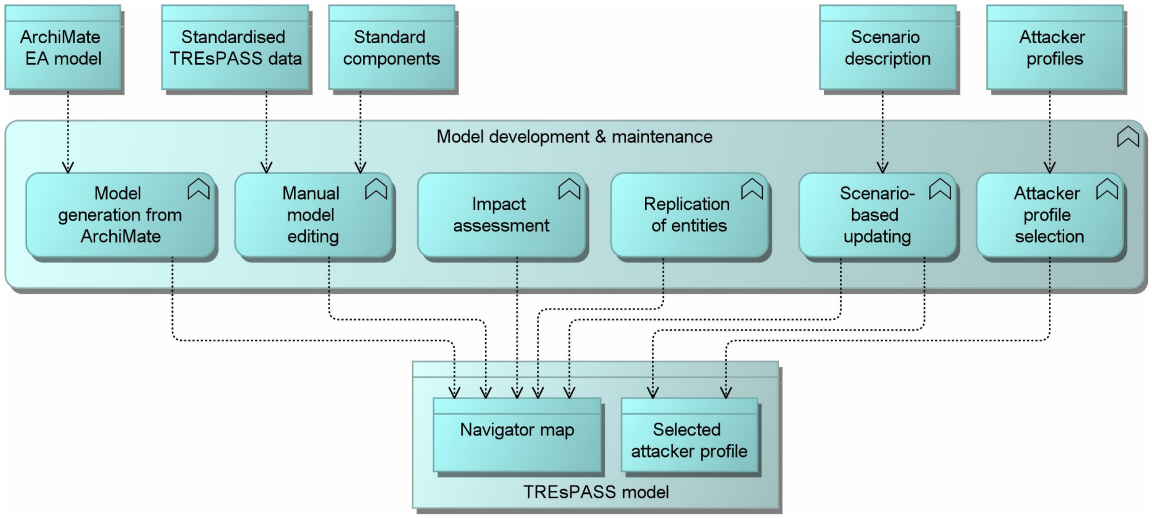

Figure 2.4: An overview of the model development and maintenance module and its functions.

Delivered service A TRE$_S$PASS navigator map is generated

Assumptions ArchiMate file is valid;

Responsible WP1

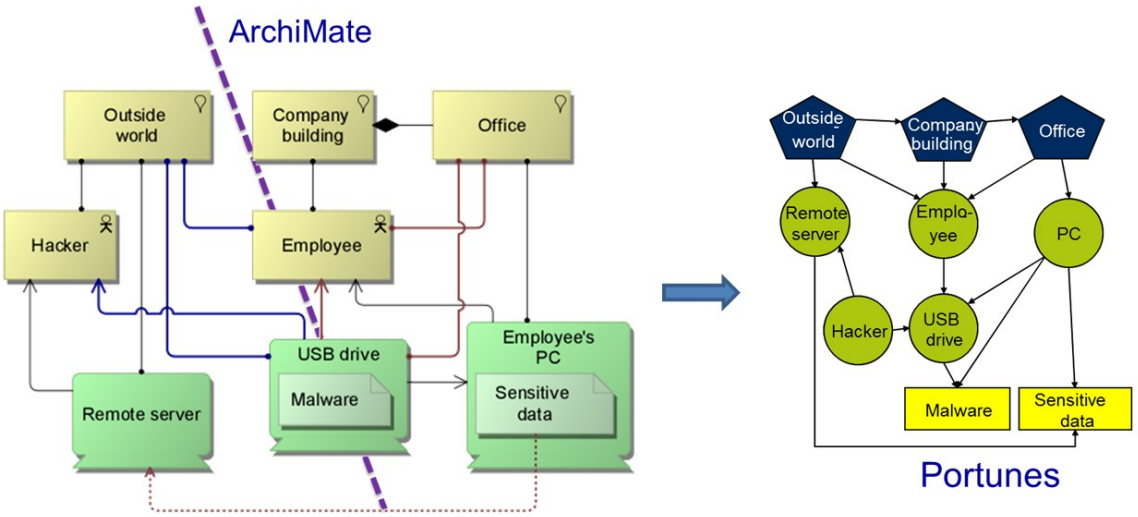

Figure 2.5: An example of model generation from ArchiMate.

S1.2 Manual model editing

Description The user can manually create and/or modify models through a graphical interface that allows the user to easily add various standard elements and entities into the model. The user can also click on elements to change their parameters.

Triggering event User actions in model editing interface

Delivered service Model specifications are updated accordingly

Assumptions Valid parameters are entered (according to TRE$_S$PASS specifications). Components are standard TRE$_S$PASS components.

Responsible WP1/WP5/WP6

S1.3 Scenario-based updating

Description The user inputs a scenario which has occurred; system proposes changes to the model that reflect the scenario. The user may then select a proposed change or update the model manually.

The user may also select scenarios from an analysis output that are *not* possible in practice. The system proposes then changes to the model that reflect this impossibility. The user may then select a proposed change or update the model manually.

Triggering event User inputs customised scenarios.

Delivered service Changes to the navigator map (and/or attacker profiles) are proposed such that the scenario is now possible (or impossible, depending on user preferences).

Assumptions The inputted scenario is compatible with TRE$_S$PASS specifications. The inputted scenario is marked accordingly (possible/impossible).

Responsible WP2/WP1

S1.4 Selection of attacker profiles

Description The user can add attacker profiles to the map, either by selecting them from a library or by building them from scratch.

Triggering event Attacker profile is selected or loaded.

Delivered service The TRE$_S$PASS model is updated to include the new profiles.

Assumptions Loaded profiles are valid with regard to TRE$_S$PASS specifications.

Responsible WP1/WP2

S1.5 Impact assessment

Description Through the system interface, the user can specify certain model entities as *assets*; these can be annotated with quantitative measures such as asset value, potential damage upon failure, etc. Existing methods may be used for the purpose of asset identification and valuation. Based on the specified assets and the concomitant quantitative measures a number of analyses can be performed, as selected by the user, resulting in an *impact assessment*, providing quantitative estimates of the potential impact following a successful attack (e.g., a breach of confidentiality, integrity, or availability) on the designated assets.

The tools should support tagging of assets with various properties or values, which could be used in combination with an attacker profile to identify, for example, the attractiveness (value) of the asset to the attacker.

Triggering event User annotates entities with quantitative values.

Delivered service Updated asset values in the TRE$_S$PASS model.

Assumptions Values are valid with regard to TRE$_S$PASS specifications.

Responsible WP1/WP2

S1.6 Replication of entities

Description 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.

Triggering event User requests duplication of an entity in the model.

Delivered service Entity is replicated as many times as requested, while maintaining any parameters.

Assumptions Entity to be replicated has been defined and parameterized correctly.

Responsible WP1

S1.7 Model sharing

Description The user can select areas on the map that are to be made available for use by other organisations. A meaningful label should be provided for the shared map area. The user can indicate which organisations are allowed to see the shared map area (by country, sector, etc.).

Triggering event User requests export of (a part of) the model to the Shared Model Library.

Delivered service A limited version (of the selected part) of the model is uploaded to the Shared Model Library.

Assumptions The model to be shared is a valid TRE$_S$PASS model; All potentially confidential elements in the model to be shared are marked accordingly. There exists a connection to the TRE$_S$PASS Shared Model Library or a local version of it.

Responsible WP1, WP5, WP6

S1.8 Importing shared model

Description When building a model, the user can include selected areas from previously shared models. The system should clearly distinguish between standard components and components shared by other users.

Triggering event User selects a model from the Shared Model Library.

Delivered service Selected areas of the model from the shared library are added to the current model.

Assumptions There exists a connection to the TRESPASS Shared Model Library or a local version of it. The model to be imported is a valid TRESPASS model.

Responsible WP1, WP5, WP6

F2. Data management

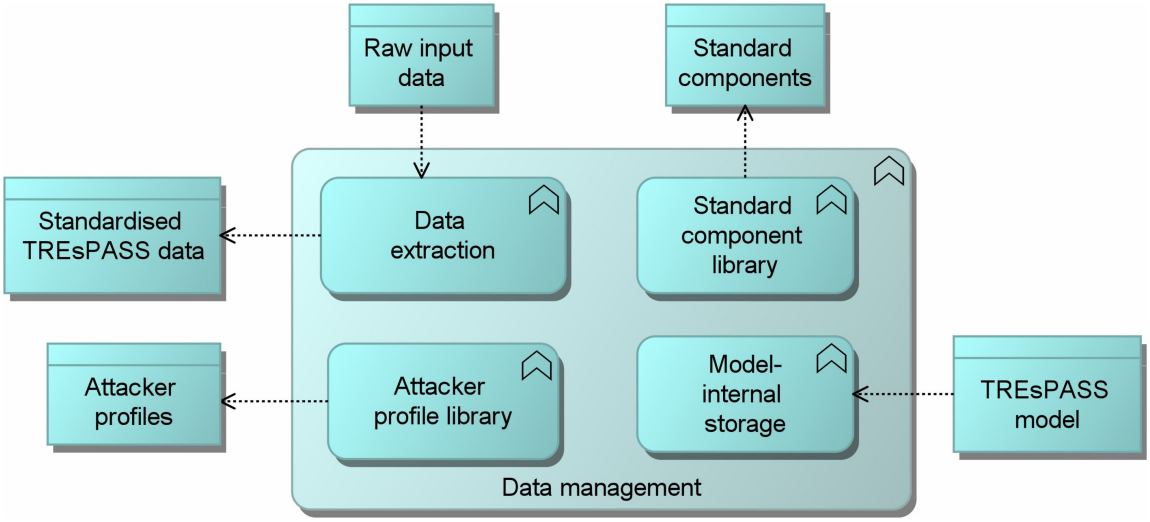

Figure 2.6: An overview of the data management module and its functions.

This section describes the functions for data extraction and data management. The formats for data representation are currently under development, but a overview of the variety of the *data formats* (used as input or produces as output by TRESPASS tools) is visible in the Architecture Diagram (Appendix 4.2).

S2.1 Data extraction

Description The data extraction function provides an interface to call relevant external tools to supply data, when needed in model development. In addition, the function can be configured to update data automatically with a specified time interval. The function also provides a stand-alone process to acquire public data for future use.

A first technical data prototype was delivered by month 12 as (The TRESPASS Project, D2.2.1, 2013). This prototype consists of a collection of suitable data extraction tools and a data storage format.

The data extraction module is supposed to provide an interface to the technical data extraction tools, including data format conversion. The module should be able to call data extraction tools upon request (“pull” mode), but it should also be able to feed the models with updates based on changes observed by the input tools (“push” mode).

Methods for extracting social data were be discussed in (The TRESPASS Project, D2.3.1, 2014). These data are not suitable for automatic extraction. Therefore, they will have to be supplied in the defined format, and no conversion is foreseen.

Triggering event User requests automated data extraction or scheduled data extraction time is reached or "push" notification is received from one of the available data extraction tools.

Delivered service All relevant automated data extraction tools are run and data is converted into standardised TRESPASS format.

Assumptions None

Responsible WP2

S2.2 Standard Component Library

Description In building a navigator map, the user can add standard components. These standard map components are associated with an attack tree. See the discussion on the Attack Pattern Library (APL) in (The TRESPASS Project, D5.3.1, 2013).

Triggering event User wants to include standard components in the model.

Delivered service A list of available components is displayed, from which the user can select one or more components for inclusion in the model.

Assumptions The computer is connected to the Internet or a local version of the Standard Component Library is available.

Responsible WP2/WP5/WP6

S2.3 Attacker Profile Library

Description After the development of a navigator map, the user can select attacker profiles. Typically, the relevant profiles will be selected from a standard library. The attacker profile library should be able to represent sets of attackers in a single profile, for example 1000 lone hackers with identical properties. The library should support both deterministic profiles (fixed attacker properties) and probabilistic profiles (probability distributions of attacker properties). Ratio and ordinal scales have to be supported for the properties.

Attacker profiles consist of static properties (for example attacker skill) and strategic properties. Strategic properties determine how an attacker will select scenarios to be executed, and how to invest his resources.

The user can add attacker profiles onto their initial position(s) on the map, to indicate to which areas or systems the attacker is assumed to have initial access. The user may also add other map components such as keys and passwords onto the attacker profile when placed on the map, to indicate the credentials the attacker has initial access to.

Triggering event User wants to load attacker profiles into the model at a certain location.

Delivered service A list of pre-defined profiles with configurable parameters is displayed from which the user can select and configure one or more such profiles to be included in the model at desired location.

Assumptions A valid TRE$_S$PASS model has been previously developed or loaded.

Responsible WP2/WP5/WP6

S2.4 Data and Model Storage

Description The system will provide storage of data concerning a single model / organisation in between calls to modules. This also forms the basis for dashboard functionality (Use Case 1). Model storage is intended as an intra-organisational process. This function should also keep track of versions of the model, the analyses that have been run, and the results of those analyses.

It is important that this storage is secure, as the information is typically considered confidential (see security requirements). Therefore, local storage is preferable over storage on a server, although storage on a server may be acceptable for the Use Cases involving SMEs.

Triggering event Request to save / load a model.

Delivered service Model is saved at (or loaded from) desired location (local/server).

Assumptions Location is accessible. When loading, the file must be valid according to TRE$_S$PASS specifications.

Responsible WP2/WP6

S2.5 Shared Model Library

Description The Shared Model Library will store (parts of) models that have been marked as share-able by other users or have been built to be used as samples or templates by TRE$_S$PASS designers. Users may upload (parts of) the models they have built, to be used by others when in the model creation process. As model-sharing is intended as an inter-organisational process, the Shared Model Library must only contain models from which all sensitive of potentially confidential information has been stripped. Special care has to be taken in order to guarantee this: users might not be thorough enough when uploading such models. As such, it might be desired that no component parameters are allowed to be stored in this library, only the components themselves and their architectural relationships.

Triggering event User opens Shared Model Library.

Delivered service The user is shown a list of (sub-)models available in the Shared Model Library

Assumptions A valid TRE$_S$PASS model has been previously developed or loaded.

Responsible WP1/WP2/WP5

F3. Analysis

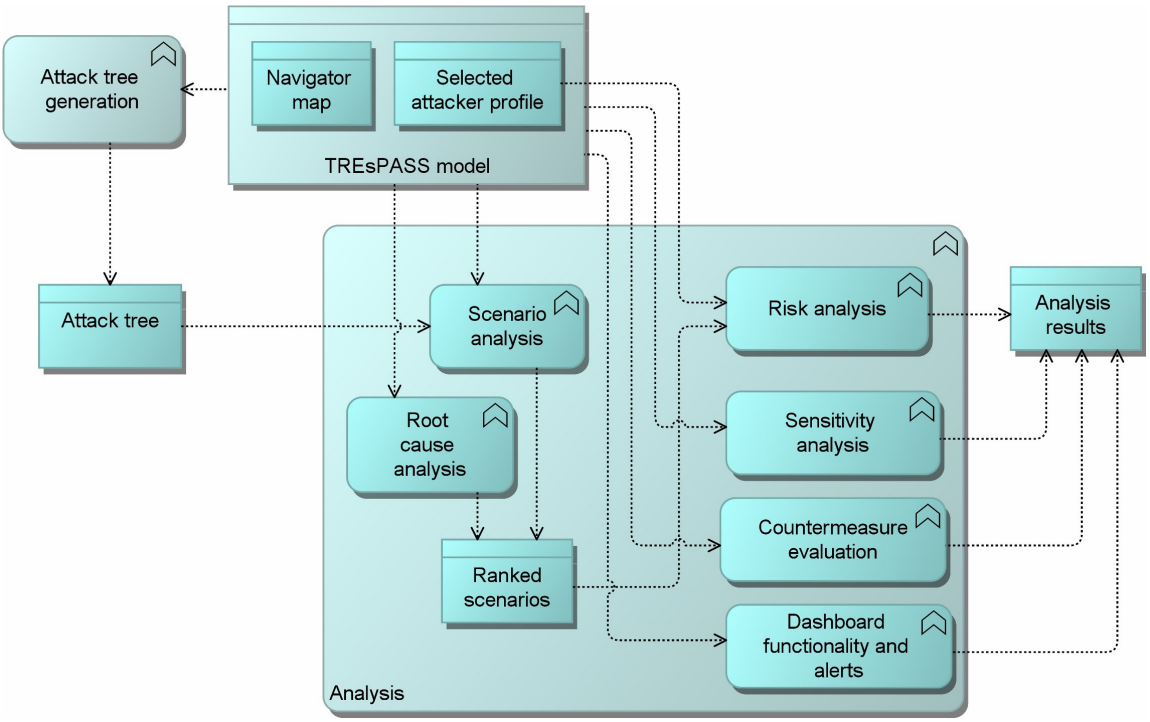

Figure 2.7: An overview of the analysis module and its functions.

From the interface, the user can select a number of possible analyses on the model, or on the attack trees generated from the model, depending on the modelling formalism used and the parameters available in the model. An analysis may also be triggered by automatic updates of the model. Not all analyses may work with all formalisms. The analysis typically runs in a separate tool, but its outputs will be available within the interface. An overview of the analysis module and its functions is shown in Figure 2.7.

S3.1 Attack tree generation

Description Based on a navigator map and an attacker profile, a tree will be generated representing the possible paths the attacker can take to reach the goal. The top goal in the tree can be abstract (e.g. generate attacker utility, or cause damage), but subtrees will represent more concrete goals (e.g. steal secret plans).

Triggering event User requests attack tree generation based on a certain navigator map, attacker profile (including location), and target asset.

Delivered service An attack tree representing all possible attack paths on the selected navigator map that the selected attacker profile might take in order to reach the target asset.

Assumptions Selected navigator map is valid and correctly defined. Attacker profile allows at least one attack path to target asset.

Responsible WP1

2.2.0.5 S3.2 Scenario analysis

Description The core analysis for the risk assessment will receive as inputs (a) the navigator map or attack tree, (b) a set of attacker profiles, and (c) the parameter to be calculated. It will output a ranked list of scenarios with respect to the requested parameter (e.g. cost or likelihood of success). In particular, the system should support as target parameter a function from attacker investment (time, money) to expected attacker gain (probability of success times gain upon success). For example, the output may indicate that if an attacker invests 4 hours and € 10,000 in a particular scenario, then his expected gain is € 15,000.

Triggering event User requests scenario analysis on a certain navigator map or attack tree, plus set of attacker profiles, with desired target parameter.

Delivered service Desired scenarios will be simulated and ranked according to target parameter.

Assumptions Selected navigator map or attack tree are valid. Selected attacker profiles are compatible (capable) with regard to the desired scenarios.

Responsible WP3

S3.3 Risk analysis

Description A second step of the analysis will get the ranked scenarios as input, and derive a risk value as annual loss expectancy from these figures. This requires an estimation of how frequently the selected attacker would execute the different scenarios, depending on his preferences and available resources. In the simplest case, the attacker would only choose the highest ranked scenario, and as many times as is feasible. In a more sophisticated analysis, the attacker would select scenarios on a probabilistic basis, with the optimal scenario having the highest probability of being selected.

Triggering event Scenario analysis is complete and user requires risk analysis to be ran on the results.

Delivered service System outputs the risk associated with the scenarios as annual loss expectancy.

Assumptions Scenario analysis has been ran at least once. Scenario analysis has generated at least one scenario.

Responsible WP3/WP5

S3.4 Root cause analysis

Description In case a particular asset has been compromised, the user can request an analysis on the most likely scenarios that led to the compromise.

This analysis should include the possibility for the user to input multiple in-between steps that have been registered, for example that an attacker accessed the building through a particular door.

Triggering event User requests scenarios that compromise a certain asset and (optionally) contain certain actions.

Delivered service System generates a list of most likely scenarios, ranked by probability.

Assumptions Selected actions are possible w.r.t the model. Selected asset is properly parametrised. System model is consistent.

Responsible WP3

S3.5 Sensitivity analysis

Description The TRE$_S$PASS tools should be able to calculate the effect of differences in the model on the risk values.

Triggering event User makes simulated modifications to the model or some of its parameters.

Delivered service System outputs the difference in the outcome of the risk analysis that simulated changes would bring (can also be a function from parameter values to risk values).

Assumptions Simulated changes are valid;

Responsible WP3, WP6

S3.6 Countermeasure evaluation

Description From the menu associated with a countermeasure on the map, a user can run an analysis of the cost-effectiveness of this countermeasure.

Triggering event User requests evaluation of a certain countermeasure, given its cost function, with respect to the current navigator map and a selected set of attacker profiles.

Delivered service System outputs cost-effectiveness in Euro (prevented risk minus countermeasure costs)

Assumptions At least one countermeasure has been properly defined on the navigator map.

Responsible WP3, WP6

S3.7 Dynamic (re-)analysis

Description In addition to the user-triggered analysis, the dashboard functionality requires analyses to be triggered by changes in the data input for the models. For example, if there is a change in the network, this may trigger a re-run of the basic risk analysis. The user can define which analyses to re-run under which conditions, but a default setting is provided.

Triggering event Updates in the model or its parameters.

Delivered service Relevant analyses are (re-)run and results are updated accordingly.

Assumptions Changes made invalidate previous results; Dynamic (re-)analysis has been activated and properly configured; At least one analysis has been marked as dynamic.

Responsible WP3, WP6

S3.8 Alerts

Description The system raises alers when changes in the underlying data lead to significant changes in risk.

Triggering event Re-analysis is ran and output is significantly different than previous analysis.

Delivered service Send alerts to selected users.

Assumptions Dynamic re-analysis is enabled for at least one analysis type;

Responsible WP6

F4. Visualization

S4.1 Visualization dashboard

Description LUST is, with support from WP4 members, responsible for producing the design briefs for each version of the prototype's interface and supporting visualisations. LUST will have overall control of the design of the Navigator tool interface. The responsible parties for producing different facets of the interface will be specified within the design brief. Currently it is envisaged that the final prototype will contain the following visual areas: visualisation of data input, visualisation of navigator maps, visualisation of attacker profiles and visualisation of attack tree scenarios. The list of visual areas and the content of the visual areas will be determined during the development of each design brief.

Triggering event The user requests results and/or graphical representations to be displayed (of data, navigator maps, attacker profiles, attack trees or analysis results).

Delivered service Visualisations of the above-mentioned items are generated and displayed.

Assumptions A valid model is available and parametrized. Data has been loaded into the model. Relevant analyses have been run at least once (for result visualisation).

Responsible WP4, WP6

S4.2 Generate Reports

Description Various (regular and one-time) reports should be generated in order to provide summarised overviews for relevant stakeholders. These might include charts, graphs, and other visual representations of specific analysis results, management or audit reports and other custom (aggregating) reports. These reports should allow comparisson of subsequent assessments, providing an overview of the risk landscape and risk levels.

Triggering event Timed (scheduled reports) or user request.

Delivered service Relevant report is generated and (optionally) sent to desired recipients.

Assumptions The data needed for the report is available in the model;

Responsible WP3, WP4, WP6

2.3 User characteristics

Two classes of users are foreseen for the TRESPASS tools. 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. Secondly, SMEs are expected to be able to use a "lightweight" version of the tools in relatively quick scans of their security posture. This lightweight version should be usable by people with decent security awareness but no specialised training in neither the TRESPASS tools nor other Risk Management / Risk Assessment methodologies or standards and should be supported by relevant manuals and step-by-step tutorials.

A key decision in relation to the TRESPASS system users is which tasks should be performed by the system, and which by the user. Our initial position is that decisions about relevance must be left to individuals. We do not aim to build software that can judge relevance by artificial intelligence or similar methods. By contrast, reasoning through *all* the consequences of a proposition or a change is best left to machines, which is what the formal methods employed in Work Package 3 can do. In particular, the tools are meant to assist the user in creating navigator maps and attacker profiles, to explore the possible attack paths by systematic reasoning, and to assist the user in interpreting the results in the context of risk assessment.

Figure 2 indicates which tasks (and respective tools) require user interaction.

2.4 General Constraints

The main types of constraints that the TRESPASS approach will face are related to data availabiltiy. Since not all types of relevant data might be available or even required for all types of users, the toolkit should provide flexibility in terms of data requirements and analysis. This is achieved by providing alternative data extraction methods and analysis tools. In order to support *data collection*, *modelling*, *analysis*, and *visualisation* across various application scenarios (some of which are listed in Section 3) the TRESPASS toolkit has to be usable in a modular fashion. Figure 2 provides an overview of the large variety of modules that make up the TRESPASS toolkit, as well as the data exchange formats used. In practice, stages may not follow a linear order.

2.5 Assumptions and dependencies

The project partners should share a common vision of the TRESPASS approach and a mutually accepted domain description of the Information Security field. The partners should have access to a necessary and sufficient number of professionals possessing

the necessary and sufficient knowledge about respective parts of desired requirements (Reinhartz-Berger, Sturm, & Wand, 2005). Furthermore, necessary and sufficient contact with practice should be maintained in order to maximise the utility, usability and practical relevance.

Due to the distributed nature of the research and development process, the TRE$_S$PASS toolkit consists of a collection of loosely coupled modules, designed by various partners. As some tools require specific input or produce custom output, there exists a large number of dependencies in terms of data exchange formats. An overview of the modules, respective tools and data exchange dependencies is presented in Figure 2.

3 Use Cases and Case Studies

An important resource for requirements elicitation is the intended scope of the toolkit. Throughout the project, several Case Studies owned by industry partners have been used as initial sources of requirements, but mostly as means of validating the TRE$_S$PASS process and tools.

Other important sources of (functional) requirements are of course the goals of the intended users. A set of Use Cases can be used to specify the different ways to use the system, and thus defines all behaviour required of the system, therefore bounding the scope of the system (Malan & Bredemeyer, 1999). In this section, we attempt to describe the main application scenarios envisioned for the TRE$_S$PASS methodology and tools in terms of specific Use Cases that can then be used as a source for the elicitation of an initial set of requirements. Furthermore, required functionality for each Use Case (U) is distilled.

3.1 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 TRE$_S$PASS 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 TRE$_S$PASS 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 TRE$_S$PASS 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.

These Case Studies and their respective requirements are described in detail in D7.1.2 (The TRESPASS Project, D7.1.2, 2015).

3.2 Use Case 1: Security Investment

Typical user

The user is a consultant or security officer with specific training on the use of the tools and processes. In particular, the user has received training on the creation of TRESPASS models, selection and representation of countermeasures, working with a TRESPASS project structure, as well as the execution of analysis on such models.

Functional goals/needs

In this Use Case, a large company uses the TRESPASS tools to decide on investments in information security. This can be either because there is a limited security budget and the company wishes to select the optimal countermeasures, or because the company is interested in investigating how much money would be needed to implement the most cost-effective set of countermeasures. Existing enterprise architecture models (ArchiMate) can be used as the starting point for the Use Case.

The Use Case may be a one-time effort, but typically the use of the models for security investment decisions is part of a long-term strategy in which the TRESPASS tools and processes are implemented in an organisation to support various cyber security decisions over a longer period of time.

This Use Case links to security economics, in the sense that the questions asked are about cost-effectiveness or return on security investment. This means that the primary variables of the analysis are economic. The basic underlying analysis is a *minimax* analysis, in which the defender's investment is optimised under the assumption that the selected adversaries (attacker profiles) will maximise their own utility given the defender's choices (Cox, 2009).

Description (workflow)

1. The user opens the interface, which shows an empty map of the organisation. The user drags and drops several standard elements (buildings, offices, people, servers, smartphones, etc.) onto the map and draws the relevant connections.
2. If an ArchiMate model already exists, this may be used as a basis instead.
3. The user clicks on the items to select the corresponding properties. This may consist of asset values, access control mechanisms (door locks, passwords, etc.), and an estimation of their strength. Default values can be used if information is unavailable.
4. The user answers a number of general questions about the organisation in survey style (e.g. organisational culture). The answers are used to set properties of the items of the map, in particular the people / employees.
5. The user answers a number of general questions about the threat environment. Based on the answers, the system suggests relevant attacker profiles. The user can select the attacker profiles of interest, and adapt these if necessary.
6. The user starts the analysis. A prioritised list of scenarios / attack trees is presented.
7. *(optional)* The user can select items on the map for sensitivity analysis. The system determines to what extent the result depends on changes in that particular item.
8. *(optional)* The model can be used for continuous monitoring. Updates are needed under the following conditions:
   a) Changes in the enterprise architecture;
   b) Changes in known vulnerabilities;
   c) Changes in asset values;
   d) Changes in attacker profiles;
   e) Attacks or attack attempts occurring in the organisation;
   f) Attacks or attack attempts occurring in similar organisations;
   g) Changes in the organisational culture;
   h) Changes in the standard map components used in the model.

   Updates should be automatically processed in the tools whenever possible. Manual edits should also be supported, for example when the user decides that new (manual) data requests should be issued as part of the risk assessment process.
9. Alerts will be issued to the user when important updates have been made, or when user action is required to assist in the updates.
10. *(optional)* The organisation can share parts of the model, for example with other organisations in the same sector.

11. *(optional)* After the organisation discovers that a particular secret has been leaked, the user starts an analysis for finding the most likely scenarios that led to that leakage. The system outputs a ranked list of the most likely scenarios.

(Average) Expected time investment

consultant / security officer: 120 man hours for setup
organisation: 40 man hours for setup
consultant / security officer: 16 man hours per month for maintenance
When an external consultant is hired for setup, the organisation may choose to train an internal officer for maintenance.

Required functionality

U1.1 support for automatic updates and alerts (“dashboard”)

U1.2 visualisations to show the impact one investment has on the environment and the impact that different combinations of investments have upon these scenarios;

U1.3 creation of an empty model;

U1.4 conversion of model from ArchiMate;

U1.5 drag and drop of model elements (from library);

U1.6 properties menu associated with model elements / entities (available upon click);

U1.7 TRE$_S$PASS project structure (like a software development project in a programming environment) to support model development over time;

U1.8 survey mode or menu for general project properties (e.g. organisational culture) and attacker properties;

U1.9 selection of attacker profiles (based on survey);

U1.10 menu for adaptation of attacker profiles;

U1.11 ranking of attack scenarios;

U1.12 sensitivity analysis (changes in output based on changes in a particular parameter);

U1.13 support for automatic updates (events feed);

U1.14 support for manual updates;

U1.15 support for alerts based on (manual or automatic) updates;

U1.16 support for sharing (parts of) the model;

U1.17 support for backward analysis (if these goals were reached, what is the most likely course of events?)

3.3 Use Case 2: Audit

Typical user

The user is an auditor with specific training on the use of the tools and processes. In particular, the auditor will have received specific training on validating existing models, in case the target organisation has these in place for internal risk management (Use Case 1).

Functional goals/needs

The objective of the audit is assessing whether a target organisation complies with a certain security level, defined in a referential. This referential can be internal to the target organisation, an international standard or part of the auditor's methodology.

In order to assess the level of compliance, the auditor may use models that have already been developed for the organisation (see Use Case 1), or, depending on the audit requirements, an independent model can be developed.

Description (workflow)

1. The auditor selects a template from a set of standard models, based on the similarity to the target organisation, or creates a new model.
2. The auditor asks a number of general questions about the organisation (e.g. organisational culture). The answers are used to populate the model with some default quantitative values and set properties of the items of the map, in particular the people / employees.
3. The auditor opens the interface, which shows an empty map of the organisation. The auditor drags and drops several standard assets (buildings, offices, people, servers, smartphones, etc.) onto the map and draws the relevant connections. The standard components are filled with default quantitative values, based on the answers to the questions.
4. *(optional)* The auditor can adapt the model where needed, by moving, deleting, or adding (drag-and-drop) elements and changing properties.
5. The auditor clicks on the items to select the corresponding properties. This may consist of asset values, access control mechanisms (door locks, passwords, etc.), and an estimation of their strength. Default values can be used if information is unavailable.
6. The auditor can select one or more of the following results:
   a) A list of possible attack scenarios, ordered by their risk;
   b) A list of possible countermeasures, ordered by their cost-effectiveness;

c) A visualisation of the map, showing the “weakest links”, for example by increasing their size, or changing their colour.

7. The auditor runs the risk analysis to verify that risk levels for relevant assets are in line with the thresholds defined in the risk assessment.
8. The system shows to the auditor which risks need to be reduced to meet the target level (gap analysis).
9. In interaction with the target organisation, the auditor verifies that the countermeasures that reduce the risk below the thresholds are being properly implemented. This can be shown by re-running the risk analysis with the countermeasures and its expected effects added to the model.
10. *(optional)* The auditor finds that certain attack scenarios are unrealistic and adapts the model accordingly.
11. *(optional)* The auditor finds that certain known attack scenarios do not show up in the results and adapts the model accordingly.
12. *(optional)* The auditor re-runs the analysis and adapts the model until satisfied with the output scenarios.
13. *(optional)* The auditor drags and drops the selected countermeasures onto the map and re-runs the analysis as desired. The system keeps track of the mapping of countermeasure selection to risk results.
14. *(optional)* The auditor can also make other changes in the model in order to identify changes in the risk profile based on changes in the enterprise architecture.
15. *(optional)* The model can be saved for future adaptations.

(Average) Expected time investment

auditor: 120 man hours
organisation: 40 man hours

Required functionality

U2.1 Support for replication of items (say, 1000 identical servers, or slight variations)

U2.2 Library of Model Templates with descriptions

U2.3 Preliminary audit questionnaire to determine default quantitative and qualitative values

U2.4 Model editing window: move, delete, and change properties of components

U2.5 Standard Component Library with drag-and-drop functionality

U2.6 Result dashboard

U2.7 The audit referential

U2.8 Generation of attack scenarios

U2.9 Ranking of attack scenarios

U2.10 Suggestion of countermeasures

U2.11 Ranking of counter-measures sorted by Return on (Security) Investment

U2.12 Drag-and-drop countermeasures to add them to the model

U2.13 Navigator map visualization (that can reflect analysis results)

U2.14 (re-)run analysis button

U2.15 Functionality to export and/or save model

3.4 Use Case 3: Innovation

Typical user

The user is an SME employee with medium security awareness, in possession of a short tutorial on the tools.

Functional goals/needs

In this Use Case, a small and innovative enterprise (SME) wishes to design a new product and/or service. In order to take security into account in the design stage, a TRE$_S$PASS model is built to identify potential security weaknesses in the product / service provided. This Use Case is linked to the notions of security-by-design / privacy-by-design (Schaar, 2010) or value-sensitive design (Friedman, Kahn Jr, & Borning, 2006; Van den Hoven, 2007). The goal is providing early awareness to the designers about potential security and privacy issues with their product / service.

Description (workflow)

1. The user collects the relevant documents on the design of the new product / service.
2. The user opens the interface. The user can select an empty map or several types of default innovation maps to start with. The user then drags and drops additional standard elements (buildings, offices, people, servers, smartphones, etc.) onto the map (or deletes existing ones) and draws the relevant connections.
3. The user clicks on the items to select the corresponding properties. This may consist of asset values, access control mechanisms (door locks, passwords, etc.), and an estimation of their strength. Default values can be used if information is unavailable.

4. The user answers a number of general questions about the innovation (e.g. context of use). The answers are used to set properties of the items of the map, in particular the people.
5. The user clicks a button to start the analysis. A prioritised list of scenarios / attack trees is presented.
6. Based on the output, the user can investigate problems and solutions, and make changes in the map;
7. The user can re-run the analysis to identify the effect of investment in countermeasures, or combinations of countermeasures;
8. The system keeps track of the different versions of the map, the changes (diff) between them, and the results of the analysis in terms of risk. The user can open a menu to select previously used maps / analyses, and select them to view the stored results.
9. The user can annotate the different versions with costs, as to identify the cost-effectiveness of countermeasures.

(Average) Expected time investment

organisation: 40 man hours
consultant (optional): 40 man hours

Required functionality

U3.1 Support for analysis without data on the actual use of a system (there is only a system-to-be in this Use Case)

U3.2 Support for modelling and analysis with ordinal scale values (low, medium, high, very high)

3.5 Use Case 4: Product-service system

Typical user

The user is a consultant or internal security expert in the organisation. The user has received specific training on the use of the tools and processes.

Functional goals/needs

The overall objective of this Use Case is to perform an evaluation of the potential impact a proposed new product/service may have on security for the combined product/service portfolio of an enterprise.

In order to perform this evaluation, the user must be able to *(1)* easily re-use, extend, and modify existing models; *(2)* maintain a set of variants/versions of the extended models; and *(3)* extend existing analysis results to take the modified model(s) into account.

The context for this Use Case is that of a large enterprise that is designing a new product and related services. Here, the term product-service system refers to a combination of product(s) and service(s) offered, such as the combination of a cell phone and a subscription. As part of the design process, the enterprise wishes to perform a security evaluation to assess whether the proposed product/service introduces unacceptable risks into the overall product/service portfolio and also if it presents new or increased opportunities for attackers, e.g., fraudsters.

A comprehensive security evaluation must, by necessity, take into account aspects that are external to the new product/service and the enterprise designing it, including properties of other similar or relevant product/service systems from other vendors, factors determined by the market for the new product/service, and the intended ecosystem of the product/service.

This Use Case is similar to Use Case 3 in the sense that the assessment is about a product/service offered rather than the internal organisation, but the focus lies on the entire portfolio, and it concerns a larger organisation. In this Use Case, it is assumed that a suitable model for the existing portfolio already exists. If this is not the case, such a model will need to be developed first. To identify the effects of the new product/service, the new offering is added to the model. The user is a person within the enterprise with suitable training on the tools and processes, or an external consultant.

Description (workflow)

1. The user opens the existing model of the product/service portfolio of the enterprise as well as models of all the relevant context, e.g., models of similar products/services from other vendors.
2. The user selects the option to add a new product/service and proceeds to specify the product/service as a *business process*, incorporating relevant properties such as price points, specifications, what is required of the customer, what is to be delivered.
    a) *(optional)* The user can make use of a library of templates designed for the target enterprise/business type (drag-and-drop).
    b) *(optional)* The user can adapt existing models for similar product(s)/service(s) by modifying elements and properties.

c) *(optional)* The user can explore various design directions by saving named/versioned variations of the underlying model(s).

For example, in case of a telephone service: the customer provides identification (with a certain assurance level), and a credit check may be performed. A telephone and SIM-card are sent to the customer, which can then be used as a means to access (and seize) network resources, e.g., make calls or change configuration parameters. A price scheme is specified for the services, and the customer either pre-pays for the services, or a bill is sent to the customer afterwards.

3. *(optional)* The user selects a pre-defined analysis profile. The relevant set of analyses may vary from organisation to organisation; having one or more predefined analysis profiles, e.g., with specific defaults, can save time and effort.
4. The user selects the option to analyse the new product/service in the context of the portfolio and any relevant external context. The risk analysis identifies possibilities for an attacker to combine different services in such a way that a business case for fraud results.
5. Identified potential attack vectors are presented to the user in combination with the expected utility for an adversary. The various attack vectors can be explored using the attack navigator for deeper understanding of an attack.
6. The user can adjust the specification of the new product/service, e.g., by adding countermeasures against the most likely identified attacks, and re-run the analysis to identify changes in the risks.

   a) *(optional)* A library of standard of often used countermeasures, either enterprise or business type specific, can be used to quickly explore possible defenses.

(Average) Expected time investment

consultant / security officer: 120 man hours
organisation: 40 man hours

Required functionality

U4.1 Support for business processes as architecture components

U4.2 Template library (both for business processes and countermeasures)

U4.3 Version control of models

U4.4 Risk analysis (with expected utility)

U4.5 Analysis profiles (defining which analyses to use and their parameters)

U4.6 Navigator map visualisation (reflecting analysis results)

3.6 Use Case 5: Quick Scan

Typical user

The user is an SME employee with medium security awareness, in possession of a short tutorial on the tools. They are not specialised in IT security and are not experienced with Risk Assessment.

Functional goals/needs

In this Use Case, an SME wants to get a quick picture of its security posture.

There is no money and/or time for a full-blown risk assessment or risk management process. As such, the TRE$_S$PASS toolkit need to provide basic, out-of-the-box functionality that in-experienced and/or less specialised users can benefit from it.

Thus, the tool needs to provide standard templates that can be populated semi-automatically (by giving only a broad overview of the organisation). Of course, fine-tuning should also be possible, but optional. Certain basic analyses should be runnable on such limited models, that should provide enough information to support management-level security investment decisions by exposing critical vulnerabilities.

Description (workflow)

The workflow below is estimated to necessitate between 16 and 40 man hours to complete (depending on experience and depth of analysis)

1. The user selects a template from a set of standard models, based on the similarity to the target organisation.
2. The user answers a number of general questions about the organisation (e.g. organisational culture). The answers are used to populate the model with some default quantitative values and set properties of the items of the map, in particular the people / employees.
3. *(optional)* The user can adapt the model where needed, by moving, deleting, or adding (drag-and-drop) elements and changing properties.
4. If an ArchiMate model already exists, this may be used as a basis instead.
5. The user clicks a button to start the quick scan.
6. The user can select one or more of the following results:
    a) A list of possible attack scenarios, ordered by their risk;
    b) A list of possible countermeasures, ordered by their cost-effectiveness;

   c) A visualisation of the map, showing the “weakest links”, for example by increasing their size, or changing their colour.

7. *(optional)* The user finds that certain attack scenarios are unrealistic and adapts the model accordingly.
8. *(optional)* The user finds that certain known attack scenarios do not show up in the results and adapts the model accordingly.
9. *(optional)* The user re-runs the analysis and adapts the model until satisfied with the output scenarios.
10. *(optional)* The user drags and drops the selected countermeasures onto the map and re-runs the analysis as desired. The system keeps track of the mapping of countermeasure selection to risk results.
11. *(optional)* The user can also make other changes in the model in order to identify changes in the risk profile based on changes in the enterprise architecture.
12. *(optional)* The model can be saved for future adaptations.

(Average) Expected time investment

organisation: 16-40 man hours, depending on experience and depth of analysis

Required functionality

U5.1 Granularity of results presentation, number of presented scenarios;

U5.2 A library of standard map components, with associated attack trees;

U5.3 A library of standard countermeasures, with associated map components and local effect on these components.

U5.4 Library of Model Templates with descriptions

U5.5 Questionnaire to determine default quantitative and qualitative values

U5.6 Model editing window: move, delete, and change properties of components

U5.7 Standard Component Library with drag-and-drop functionality

U5.8 ArchiMate model import function

U5.9 Result dashboard

U5.10 Generation of attack scenarios

U5.11 Ranking of attack scenarios

U5.12 Suggestion of countermeasure

U5.13 Ranking of counter-measures

U5.14 Drag-and-drop countermeasures to add them to the model

U5.15 Navigator map visualization (that can reflect analysis results)

U5.16 (re-)run analysis button

U5.17 Functionality to export and/or save model

4 Requirements

This section aggregates and classifies all the requirements elicited thus far within the TRESPASS project: Primary requirements (P) and specific requirements (R).

4.1 Primary requirements

The primary functional requirements of the TRESPASS toolkit, as described in D6.2.1, are:

P1 TRESPASS tools should assist the user in building a navigator map;

P2 TRESPASS tools should assist the user in acquiring the data needed for the map;

P3 TRESPASS tools should be able to derive attack scenarios from the navigator map;

P4 TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation;

P5 TRESPASS tools should be able to calculate the effect of the uncertainty of the input values and the sensitivity of the prioritisation;

P6 TRESPASS tools should be able to calculate the cost-effectiveness of a proposed countermeasure;

P7 TRESPASS tools should be able to visualise maps, scenarios, and countermeasures.

4.2 Specific requirements

In order to refine the initial set of high-level primary requirements described in Section 4.1, a Use Case based requirements engineering approach was applied. Specifically, the previously described set of common Use Cases for the TRESPASS toolkit are used to inform the requirements elicitation/refinement process. This allows us to (1) guarantee that the required functionality of expected Use Cases is covered by the elicited (functional) requirements; and (2) provide trace-ability from each such requirement to a particular Use Case or application scenario.

However, as the project advanced, and due to the large network of partners working collaborating on it's development, a new type of requirements became more important: integration requirements (sometimes referred to as *derived requirements*). These may be subsystem requirements (that are imposed on particular subsystems but do not necessarily provide a direct benefit to the user) or interface requirements (that describe how

the subsystems need to communicate with one another to achieve the desired result). In our case, these usually take the form of requirements coming from a TRESPASS Work Package that are targeted towards other specific Work Packages. The goal of these internal requirements is to not only ensure that the tools being developed can work together and function as a whole but also to avoid functional overlap, redundant work and missing functionality, while promoting a consistent, project-wide collaborative view on the research and development process.

Below, the template used to describe each requirement is presented, with an indication towards the interpretations of each field: These integration requirements are defined as follows:

- Identifier – Unique identifier assigned to requirements. Once used it is never reused.
- Requirement – Description of a desired property of the system. Ideally stated as:
  - Behaviour: something the system should be able to do, e.g. produce attack tree from a model;
  - Property: some observable property of system behaviour that should be achieved, e.g. speed, usability, etc.
- Source WP – WP that originates this requirement, i.e. desires to see it implemented.
- Owner – Name of person/partner responsible for stating and clarifying the requirement, and for finding an implementer. If anyone has a problem understanding the requirement, this is the person to ask. The owner is responsible for advancing the state of the requirement. This field is omitted from this document due to confidentiality issues.
- Target WP – WP responsible for implementing this requirement.
- Implementor — Name of person/partner responsible for advancing the status of the requirement. This includes responsibility for implementing it. This field is omitted from this document due to confidentiality issues.
- Goals – Explanation of why the requirement is desirable to the owner. What goals will be achieved by it?
- Acceptance criteria – Which observations must be done to decide whether the requirement is completed? Ideally, some tests should be specified here.
- Status – Notes or comments regarding the requirement, its stage of implementation or potential issues. One of the following:
  - Proposed (Number, Requirement, Source and Owner have a value).
  - Consulted (Implementer found, and is consulting with Owner to clarify the requirement)
  - Agreed (Stated requirement will be implemented by the implementer, any dependencies have been mapped, acceptance criteria have been agreed on)

  - Shelved (Stated requirement will not be implemented)
  - Tracked (Implementation has started)
  - Completed. (Owner and Implementer agree that the implementation is finished successfully)
- Dependencies – Which other requirements need to be implemented first, before this one can be implemented? In addition, which other requirements assume that this one is implemented first?

The complete set of final TRESPASS functional requirements, available at the date this deliverable was published can be found in Appendix 4.2.

References

Abraham, D. G., Dolan, G. M., Double, G., & Stevens, J. V. (1991). Transaction security system. *IBM Systems Journal*, *30*(2), 206-229. doi: 10.1147/sj.302.0206

Cox, L. A. T., Jr. (2009). Game theory and risk analysis. *Risk Analysis*, *29*(8), 1062–1068. Retrieved from http://dx.doi.org/10.1111/j.1539-6924.2009.01247.x doi: 10.1111/j.1539-6924.2009.01247.x

ENISA Technical Department. (2006, june). *Risk management: Implementation principles and inventories for risk management/risk assessment methods and tools* (Tech. Rep.). ENISA.

Friedman, B., Kahn Jr, P. H., & Borning, A. (2006). Value sensitive design and information systems. *Human-computer interaction in management information systems: Foundations*, *5*, 348–372.

Gordijn, J., Akkermans, H., Koks, A., & Schildwacht, J. (2004, April). *User manual e3-value editor.* http://e3value.few.vu.nl/docs/misc/manual_version2.pdf. Vrije Universiteit Amsterdam.

IEEE. (1998). *IEEE Recommended Practice for Software Requirements Specifications* (Tech. Rep.). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=720574

Ionita, D. (2013, July). *Current established risk assessment methodologies and tools.* Retrieved from http://essay.utwente.nl/63830/

Janssen, W., van Buuren, R., & Gordijn, J. (2005). Business case modelling for e-services. In D. R. Vogel, P. Walden, J. Gricar, & G. Lenart (Eds.), *Proceedings of the 18th bled conference (e-integration in action)* (p. cdrom,). Maribor, SL: University of Maribor.

Malan, R., & Bredemeyer, D. (1999, june). *Functional requirements and use cases.* Retrieved from http://agile.csc.ncsu.edu/SEMaterials/UseCaseRequirements.pdf

Network, E., & Agency, I. S. (2013, February). *Introduction (risk management).* http://www.enisa.europa.eu/activities/risk-management/current-risk/risk-management-inventory/introduction.

Reinhartz-Berger, I., Sturm, A., & Wand, Y. (2005). Tutorial 3: Domain engineering – using domain concepts to guide software design. *Perspectives in Conceptual Modeling*, 461–463. Retrieved from http://dx.doi.org/10.1007/11568346_50

Rios Insua, D., Rios, J., & Banks, D. (2009). Adversarial risk analysis. *Journal of the American Statistical Association*, *104*(486), 841-854. Retrieved from http://www.tandfonline.com/doi/abs/10.1198/jasa.2009.0155 doi: 10.1198/jasa.2009.0155

Robertson, J., & Robertson, S. (1998). *Volere. Requirements Specification Template. Edition 6.0* (Tech. Rep.). Atlantic Systems Guild.

Schaar, P. (2010). Privacy by design. *Identity in the Information Society*, *3*(2), 267–274.

The Open Group. (2009). *Risk taxonomy* (Tech. Rep. No. C081). The Open Group. Retrieved from www.opengroup.org/pubs/catalog/c081.htm

The TRESPASS Project, D2.2.1. (2013). *Technical data extraction prototype.* (Deliverable D2.2.1)

The TRESPASS Project, D2.3.1. (2014). *Social data and policy extraction prototype.* (Deliverable D2.3.1)

The TRESPASS Project, D5.1.1. (2013). *Initial requirements for process integration.* (Deliverable D5.1.1)

The TRESPASS Project, D5.1.2. (2015). *Final requirements for process integration.* (Deliverable D5.1.2)

The TRESPASS Project, D5.2.1. (2014). *Currently established risk-assessment methods.* (Deliverable D5.2.1)

The TRESPASS Project, D5.3.1. (2013). *Abstraction levels for model sharing.* (Deliverable D5.3.1)

The TRESPASS Project, D7.1.2. (2015). *Final requirements for implementation of case studies.* (Deliverable D7.1.2)

Van den Hoven, J. (2007). ICT and value sensitive design. In *The information society: Innovation, legitimacy, ethics and democracy in honor of professor jacques berleur sj* (pp. 67–72). Springer.

Wieringa, R. (2003). *Design methods for reactive systems: Yourdan, statemate, and the UML*. Boston: Morgan Kaufmann Publishers.

Appendix A – TRESPASS functional requirements

Requirement R01

Requirement : Common XML shared model

Source WP: WP6

Target WP: WP2

Goals: simplify data exchanges between tools

Acceptance criteria: Model available for current toolset

Status: Agreed

Dependencies: None

Requirement R02

Requirement : Central database structure

Source WP: WP6

Target WP: WP2

Goals: store data needed by several tools

Acceptance criteria: Structure available for current toolset

Status: Agreed

Dependencies: None

Requirement R03

Requirement : Availability of an attack pattern library

Source WP: WP6

Target WP: WP5

Goals: reuse of sub-attack-trees

Acceptance criteria: One example per case study

Status: Agreed

Dependencies: None

Requirement R04

Requirement : Integration based on loose coupling and on a central integration component

Source WP: WP6

Target WP: WP6

Goals: Nature of data exchanges

Acceptance criteria: General integration approach, with flexibility to adapt to the tools being integrated

Status: Agreed

Dependencies: None

Requirement R05.1

Requirement : Support for asynchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: Nature of tools from other WPs

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: Partly conflicts R04

Requirement R05.2

Requirement : Support for synchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: At some point, there may be tools colecting and processing real time data. It will be implemented if needed

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: None

Requirement R06

Requirement : "Apps style" user interface

Source WP: WP6

Target WP: WP6

Goals: Modern user-friendly style

Acceptance criteria: ?Apps? should work stand alone as well as in a tool chain (whenever appllicable)

Status: Agreed

Dependencies: None

Requirement R07

Requirement : Ability to generate an attack tree

Source WP: WP5

Target WP: WP3

Goals: Needed to proceed with Task 5.3

Acceptance criteria: The tool exists and runs correctly

Status: Agreed

Dependencies: None

Requirement R08

Requirement : Parameter values for the trees in APL

Source WP: WP5

Target WP: WP2

Goals: Usability of APL depends on it. Needed to produce meaningful analysis results.

Acceptance criteria: Parameter values are provided and look reasonable

Status: Agreed

Dependencies: None

Requirement R09

Requirement : Classification of data types

Source WP: WP2

Target WP: WP4

Goals: Needed to produce visualisation toolkit

Acceptance criteria: Visualisation toolkit supports XML model data types

Status: Agreed

Dependencies: Trespass XML model to be defined. The ability to accurately define the data types will depend on the quality of data that comes through from WP2.

Requirement R10

Requirement : Interface between analysis tools and visualisation tools

Source WP: WP6

Target WP: WP4,6

Goals: Needed to generate visualisations

Acceptance criteria: The output of the analysis tools is available for visualisations

Status: Agreed

Dependencies: None

Requirement R11

Requirement : There must exist a clear description of the models.

Source WP: WP3

Target WP: WP1

Goals: Needed to produce visualisation toolkit

Acceptance criteria: A well-defined language for the socio-technical model.

Status: Agreed

Dependencies: None

Requirement R12

Requirement : Develop a visual language

Source WP: WP4

Target WP: WP4,6

Goals: Needed for development of interface

Acceptance criteria: Documented language accepted by reviewers in Deliverable 4.1.1 (achieved) Accepted interpretation by practitioner panel and data analysis designers (WP) in Attack Navigator Map (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R13

Requirement : Define a visualisation process

Source WP: WP4

Target WP: WP4,5,6

Goals: Needed for development of interface

Acceptance criteria: Documented process accepted by reviewers in Deliverable 4.1.1 (achieved) Accepted interpretation by practitioner panel and data analysis designers (WP) in Attack Navigator Map (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R14

Requirement : Develop visual thinking tools

Source WP: WP4

Target WP: WP2,4,5

Goals: Needed to produce meaningful analysis results

Acceptance criteria: Documented thinking tools accepted by academic publication peer reviewers (achieved), documented thinking tools accepted by project reviewers in deliverable 2.3.2 (on-going) tools accepted by practitioner panels (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R15

Requirement : Define methods of visualisaiton evaluation

Source WP: WP4

Target WP: WP2,4

Goals: Needed to ensure robust tool

Acceptance criteria: Documented in deliverable 4.2.1 and reviewed by project reviewers (achieved) visualisations accepted by practitioner panel (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R16

Requirement : Provide model content such as infrastructure, policies, etc

Source WP: WP1

Target WP: WP2

Goals: Needed for model

Acceptance criteria: For the case study scenarios, the relevant elements and their properties are provided as text documents, and enable a modeller to create a matching WP1 model.

Status: Agreed

Dependencies: None

Requirement R17

Requirement : Map analysis results back to model

Source WP: WP1

Target WP: WP3

Goals: Needed to communicate analysis result back to TREsPASS tools

Acceptance criteria: For a non-trivial attack, the model elements that are involved in the attack can be identified through API calls.

Status: Agreed

Dependencies: None

Requirement R18

Requirement : Programmatic interface in other components in the TREsPASS tool

Source WP: WP1

Target WP: WP6

Goals: Needed to provide access to model

Acceptance criteria: The model is successfully integrated in a workflow ANM-model-treemaker-analyses-visualisations.

Status: Agreed

Dependencies: None

Requirement R19

Requirement : Socio-technical security models must provide the necessary abstraction to represent infrastructure and other components.

Source WP: WP2,6

Target WP: WP1

Goals: Needed to use the model

Acceptance criteria: XML Format defined

Status: Agreed

Dependencies: None

Requirement R20

Requirement : The model should support macros to ease modelling of, e.g., complex repeated properties and domain-specific extensions.

Source WP: WP5-7

Target WP: WP1

Goals: Needed for case studies

Acceptance criteria: All the reasonable system components that I can think of can be modeled using the model extension framework.

Status: Agreed

Dependencies: None

Requirement R21

Requirement : The model needs to be able to manage a list of relevant properties of the entities (products and services) involved.

Source WP: WP7

Target WP: WP1

Goals: Needed in order to incorporate telco scenarios (The model should be able to represent actors, assets, policies, etc. from the real world involved in the provisioning of telecommunications products and services.)

Acceptance criteria: The model should explicitly show entities and their properties in a telco scenario.

Status: Agreed

Dependencies: None

Requirement R22

Requirement : The model should represent relations between different entities

Source WP: WP7

Target WP: WP1

Goals: Needed in order to incorporate telco scenarios

Acceptance criteria: The model should show the relations between entities in a telco scenario.

Status: Shelved

Dependencies: None

Requirement R23

Requirement : The TREsPASS approach needs to account for (partial) risks that are induced through the properties of the entities (products and services) involved.

Source WP: WP7

Target WP: WP3

Goals: Needed in order to incorporate telco scenarios

Acceptance criteria: For sufficiently large models the analysis should be faster than manual analysis.

Status: Agreed

Dependencies: None

Requirement R24

Requirement : Risks and relations should include contractional agreements and jurisdictional requirements.

Source WP: WP7

Target WP: WP1,3

Goals: Needed in order to incorporate telco scenarios

Acceptance criteria: For WP1: Should model the contractual agreements and jurisdictional requirements between entities. For WP3: generation of attacks (risks) considering the contractual agreements and jurisdictional requirements between entities.

Status: Completed

Dependencies: None

Requirement R25

Requirement : The TREsPASS approach should handle complex correlations of partial risks

Source WP: WP7

Target WP: WP3

Goals: Needed for case studies

Acceptance criteria: Should work when tested with telco scenarios

Status: Shelved

Dependencies: Reason for shelving: we don?t know how to do this mathematically, so we will not be able to implement this in the an alysis tools.

Requirement R26

Requirement : It should be possible to generate attack scenarios from the model.

Source WP: WP7

Target WP: WP3,5

Goals: Needed for case studies

Acceptance criteria: Generation of attack scenarios from the telco models

Status: Agreed

Dependencies: R26

Requirement R27

Requirement : The TREsPASS tool should provide solutions or mitigation strategies for attack scenarios.

Source WP: WP7

Target WP: WP3,5

Goals: Needed for case studies

Acceptance criteria: Mitigation strategy for the identified attack scenario

Status: Shelved

Dependencies: Reason for shelving: in line with the discussion that occurred in WP5 (as put forward by Jan Willem during the MT meeting), and the points raised by the advisory board, that the scope of our work is not ?impact analysis?.

Requirement R28

Requirement : TRESPASS tool will need to be handed over to one or several industry partners for testing and application.

Source WP: WP7

Target WP: WP2

Goals: Needed for the models, in relation to compatibility of data formats

Acceptance criteria: The tool should consider compatiblity with different platforms.

Status: Agreed

Dependencies: None

Requirement R29

Requirement : TRESPASS tool needs to provide performance scalability in order to (potentially) be able to cope with massive amounts of data, e.g. billions of call data records (CDRs).

Source WP: WP7

Target WP: WP2

Goals: Needed for the models

Acceptance criteria: The tool should be scalable.

Status: Agreed

Dependencies: None

Requirement R30

Requirement : The data formats used in the tool should be flexible as data sources in the domain of the respective industry partners could be different.

Source WP: WP7

Target WP: WP2

Goals: Needed for case studies

Acceptance criteria: The TREsPASS tool should accept known data fomats (XML, JSON, ?)

Status: Agreed

Dependencies: None

Requirement R31

Requirement : The analysis tool should provide for the possibility to generate attack scenarios based on the structural components described in the requirements to WP1.

Source WP: WP7

Target WP: WP3

Goals: Needed for case studies

Acceptance criteria: Identical to R24: generation of attack scenarios from the models

Status: Agreed

Dependencies: None

Requirement R32

Requirement : The analysis tool should stipulate the provisioning of solutions or mitigation strategies for attack scenarios.

Source WP: WP7

Target WP: WP3

Goals: Needed for case studies

Acceptance criteria: Identical to R24: For WP1: Should model the contractual agreements and jurisdictional requirements between entities. For WP3: generation of attacks (risks) considering the contractual agreements and jurisdictional requirements between entities.

Status: Shelved

Dependencies: Reason for shelving: in line with the discussion that occurred in WP5 (as put forward by Jan Willem during the MT meeting), and the points raised by the advisory board, that the scope of our work is not ?impact analysis?.

Requirement R33

Requirement : The tool needs to provide adequate visualisation for technical correlations within different misuse scenarios

Source WP: WP7

Target WP: WP4

Goals: Helps discussion with practitioners. Task 7.3 dependes on this - from a WP4 perspective this depends on data coming through from WP2 and the analysis engines provided by WP3

Acceptance criteria: Visualisation tools delivered as part of the attack navigator map for models, analysis and identified risks

Status: Agreed

Dependencies: Agreed subject to data being passed in a timely manner from WP2, analysis capability clearly defined by WP3 and WP7 being able to articulate the visualisation requirements. This is also agreed subject to agreement being reached as to the visualisation that can be developed in the time available

Requirement R34

Requirement : The tool needs to provide adequate visualisation for risks related to features of the involved products and services of the telecommunications company

Source WP: WP7

Target WP: WP4

Goals: Needed for case studies. Fron a WP4 perspective this depends on the data from WP2 and the analysis from WP3.

Acceptance criteria: Visualisation tools delivered as part of the attack navigator map for models, analysis and identified risks

Status: Agreed

Dependencies: Agreed subject to data being passed in a timely manner from WP2 and analysis capability clearly defined by WP3. This is also agreed subject to agreement being reached as to the visualisation that can be developed in the time available

Requirement R35

Requirement : The tool needs to provide adequate visualisation for evaluation of risks

Source WP: WP7

Target WP: WP4

Goals: Needed for case studies. From a WP4 perspective, this depends on the data coming from WP2 and the analysis from WP3.

Acceptance criteria: Visualisation tools as part of the attack navigator map for models, analysis and identified risks

Status: Agreed

Dependencies: Agreed subject to data being passed in a timely manner from WP2 and analysis capability clearly defined by WP3. This is also agreed subject to agreement being reached as to the visualisation that can be developed in the time available

Requirement R36

Requirement : Due to a persistent increase in product complexity and diversity on the feature side and margin pressure on mass products, risks cannot be eliminated, but need to be assumed and accepted instead, whilst minimising their possible impact. Hence, the focus should be the question of calculability.

Source WP: WP7

Target WP: WP5

Goals: Needed for case studies

Acceptance criteria: A tool/technique for prioritizing the risks/attack scenarios

Status: Agreed

Dependencies: None

Requirement R37

Requirement : Identification of required analysis measures and quantities

Source WP: WP3

Target WP: WP5

Goals: In order to develop dedicated analysis methods, we need a clearer idea of the kind of properties to be analysed. This is needed for task 3.3

Acceptance criteria: Agreed-upon list of measures/quantities

Status: Completed

Dependencies: Task 3.3 depends on this.

Requirement R38

Requirement : Attack tree generation from socio-technical model

Source WP: WP3

Target WP: WP3

Goals: This is an indispensible part of the intended tool chain. This is needed for task 3.4

Acceptance criteria: Working tool

Status: Agreed

Dependencies: Task 3.4 depends on this.

Requirement R39

Requirement : Realistic data with regard to structure, frequency and costs of common attacks for each case study.

Source WP: WP3

Target WP: WP7

Goals: For realistic output, we need realistic input

Acceptance criteria: Data that is as realistic as possible (given the constraints of company-confidentiality), in a format that can be processed by the tools'

Status: Agreed

Dependencies: None

Requirement R40

Requirement : Support for attack tree visualisation

Source WP: WP3

Target WP: WP4,6

Goals: Improved presentation of tool output

Acceptance criteria: Visual inspection

Status: Agreed

Dependencies: None

Requirement R41

Requirement : Data categorisation needed

Source WP: WP2

Target WP: WP2,7

Goals: Require categorisation of data types and capabiltiies

Acceptance criteria: Trespass Model and Attack Pattern Library data requirements specified

Status: Agreed

Dependencies: All data types from the different tools chains are clarified

Requirement R42

Requirement : Data analysis model relevant to case studies

Source WP: WP3

Target WP: WP4

Goals: Require capabilities to sufficiently analyse data for case study (WP7) purposes

Acceptance criteria: ?A means to visualise the analysis results from WP3 tools, understood by consultants"

Status: Agreed

Dependencies: Agreed subject to data being passed in a timely manner from WP2 and analysis capability clearly defined by WP3. This is also agreed subject to agreement being reached as to the visualisation that can be developed in the time available

Requirement R43

Requirement : The model and data extraction must be able to cope with a dynamic system, i.e., changes in the system need to be detected and represented in the model. And the model should have rules to change the model itself.

Source WP: WP7

Target WP: WP1,2

Goals: Needed for cloud case study, since the systems are highly dynamic.

Acceptance criteria: The data extraction tools need to detect and obtain change events in the cloud infrastructure. Those change events need to be translated into a change for the model and applied to the model.

Status: Agreed

Dependencies: None

Requirement R44

Requirement : Actions in processes should accept wild cards instead of concrete parameters.

Source WP: WP7

Target WP: WP1

Goals: Allowing masked IP addresses in processes was necessary to simulate packet forwarding. This operation is indicated by the operator tilde.

Acceptance criteria: Model needs to be able to express wildcards such as in IP addresses. Tools need to able to parse it.

Status: Agreed

Dependencies: None

Requirement R45

Requirement : There must be a non-destructive read action.

Source WP: WP7

Target WP: WP1

Goals: In order to implement knowledge assets on systems, a non-destructive read operation is very useful. For example, FTP or HTTP operations can be encoded as simple read calls to the tuplespace, rather than including separate processes for every available asset. In addition to brevity, this representation also allows for assets to be moved or deleted.

Acceptance criteria: The model needs to provide syntactic sugar to express non-destructive read operations in a consice way.

Status: Agreed

Dependencies: None

Requirement R46

Requirement : Policies should contain variables to bind parameters in the enabled actions to concrete values in credentials, or to enforce certain values.

Source WP: WP7

Target WP: WP1

Goals: Policies can use free variables, as in the following example: policies = { [X, contains(friends,X)] : {out(""request"", fileX, X)}} processes = { in(""request"", !F, !src).out(F)@src } The left hand side of this policy is consistent with previous descriptions of this format, allowing that if a request originates from actor X, and X is in the set friends, then the operation is permitted. Restricting the contents of the out operation to start with ""request"", fileX was also supported. The extension is to allow the free variable X also to appear on the right hand side.

Acceptance criteria: The model and tools needs to be able to handle free variables, for instance, as in the given example.

Status: Agreed

Dependencies: None

Requirement R47

Requirement : Global Keywords to provide metainfo in policies or processes.

Source WP: WP7

Target WP: WP1

Goals: A number of policy left hand side elements are implicit, such as a plain reference to an actor or location requiring that the request be originated by that actor or in that location. Rather than extend this to functions, such as using knows(info) to indicate that the actor that originates the operation must know asset info, the global keyword ACTOR has been Introduced to refer to the actor that originated an operation. Using this keyword, the above example becomes knows(ACTOR, info), which is consistent with other uses of the knows function.

Acceptance criteria: The model and tools needs to be able to handle global keywords, for instance, as in the given example.

Status: Agreed

Dependencies: None

Requirement R48

Requirement : It must be possible to add comments to model representations.

Source WP: WP7

Target WP: WP1

Goals: Comments can be used to quickly disable parts of the model but also to add explanations. Especially with automatically-generated files, explanations are often helpful.

Acceptance criteria: The model and tools need to be able to handle comments, at least line comments.

Status: Completed

Dependencies: None

Requirement R49

Requirement : End user visualisation requirements

Source WP: WP4

Target WP: WP7

Goals: A prioritised list of visualisation requirements from the case studies would be useful additional input during Year 3 as WP4 concentrates on end-user requirements.

Acceptance criteria:

Status: Completed

Dependencies: None

Requirement R51

Requirement : Navigator maps can be automatically generated from an ArchiMate model.

Source WP: MT

Target WP: WP1

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment) and Use-case 2 (Audit), specifically U1.4 , U4.1 and U5.8.

Acceptance criteria: 80perc. of these users accept the risk results derived from ArchiMate based TRESPASS models.

Status: Agreed

Dependencies: None

Requirement R52

Requirement : Users can drag and drop standard elements onto the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment) and Use-case 2 (Audit), specifically U1.5, U2.4, U2.5, U4.1, U4.2, U5.6 and U5.7

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training. Standard elements must include business processes.

Status: Agreed

Dependencies: None

Requirement R53

Requirement : Users can add connections between elements on the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.4 and U5.6.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Potentially conflicts R07, R38

Requirement R54

Requirement : Users can change parameters of map elements by clicking and entering new values

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.12, U2.4, U5.6.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Potentially conflicts R07, R38

Requirement R55

Requirement : Users can change parameters of map elements by clicking and requesting information from available data extraction tools.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.4 and U5.6

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to respond effectively to the suggested updates.

Status: Agreed

Dependencies: None

Requirement R56

Requirement : The system can suggest updates to the TRESPASS model based on scenarios and associated parameters.

Source WP: MT

Target WP: WP3,5,6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.12, U1.13, U2.4 U5.6.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to respond effectively to the suggested updates

Status: Shelved

Dependencies: Reason for shelving: adaptations to the sociotechnical model will have to be done manually, not automagically.

Requirement R57

Requirement : The user can select attacker profiles from the library.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.9.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: None

Requirement R58

Requirement : The user can build attacker profiles by editing their properties

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.6, U1.8 and U1.10

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: None

Requirement R59

Requirement : The user can assign asset values and properties to elements on the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.6, U2.4 and U5.6.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: None

Requirement R60

Requirement : 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.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.1, U2.4 and U5.6.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: None

Requirement R61

Requirement : User can select parts of the model to share with a specified group of other users .

Source WP: MT

Target WP: WP5 (backend).

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.16, U2.15 and U5.17

Acceptance criteria: Only explicitly selected parts are made available.

Status: Agreed

Dependencies: None

Requirement R62

Requirement : Users can add parts of shared models to their own model.

Source WP: MT

Target WP: WP5

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Partly supports Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U2.2, U4.2 and U5.4.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

Requirement R63

Requirement : The system should be able to supply data on hardware, software, version, configuration, as well as social aspects, upon request of the user when building or editing a model.

Source WP: MT

Target WP: WP2

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.14.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: R5

Requirement R64

Requirement : The system should be able to handle push messages from external tools in order to generate push messages (updates) to users/models

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.1 and U1.13.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

Requirement R65

Requirement : The system should be able to autonomously collect public data for future use in models, by updating component / attacker libraries.

Source WP: MT

Target WP: WP2

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment) and Use-case 4 (Product-service system), specifically U1.13 and U4.2.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

Requirement R66

Requirement : The system can retrieve standard model components by name.

Source WP: MT

Target WP: WP1

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.5, U5.2 and U5.7

Acceptance criteria: In 80perc. of the retrievals, the user is satisfied with the contents of the selected component

Status: Agreed

Dependencies: None

Requirement R67

Requirement : The system can retrieve standard attacker profiles by name.

Source WP: MT

Target WP: WP5

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 5 (Quick scan), specifically U5.3

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: Assumes the attack pattern library exists

Requirement R68

Requirement : The TRESPASS tools enable the user to store a TRESPASS model, including navigator map, attacker profiles, and history of analyses

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.18, U2.15, U4.3 and U5.17.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: R7

Requirement R69

Requirement : The TRESPASS tools enable the user to load a previously stored TRESPASS model, including navigator map, attacker profiles, and history of analyses.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 4 (Product-service system), specifically U4.3

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: R7

Requirement R70

Requirement : The system should be able to generate an attack tree from a map, an attacker profile, a target asset, and an attacker position.

Source WP: MT

Target WP: WP3

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U2.8, U4.4 and U5.10.

Acceptance criteria: All paths in the generated attack tree correspond to rational at- tack paths according to map definitions and attacker utility. All rational paths appear in the attack tree. Quantitative values of actions correspond to the values on the map.

Status: Agreed

Dependencies: None

Requirement R71

Requirement : The system should be able to generate an attack tree from a map, an attacker profile, and an attacker position, based on attacker utility.

Source WP: MT

Target WP: WP3

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U2.8, U4.4 and U5.10

Acceptance criteria: All paths in the generated attack tree correspond to rational at- tack paths according to map definitions and attacker utility. All rational paths appear in the attack tree. Quantitative values of actions in the tree correspond to the values on the map.

Status: Agreed

Dependencies: None

Requirement R72

Requirement : The user should be able to run scenario analysis on a navigator map with associated attacker profile.

Source WP: MT

Target WP: WP3

Goals: Derived from P4 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation). Supports Use-case 4 (Product-service system), specifically U4.4.

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: None

Requirement R73

Requirement : The user should be able to run scenario analysis on an attack tree with associated attacker profile.

Source WP: MT

Target WP: WP3

Goals: Derived from P4 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation).

Acceptance criteria: 80perc. of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: None

Requirement R74

Requirement : The system should present at most 7 scenarios to the user at once.

Source WP: MT

Target WP: WP3

Goals: Supports Use-case 5 (Quick scan), specifically U5.1

Acceptance criteria: 80perc. of the users specified in the Use Cases indicate that they can understand the default view of ranked scenarios.

Status: Agreed

Dependencies: None

Requirement R75

Requirement : The tools can calculate the total risk associated with a ranked set of scenarios and a set of attacker profiles.

Source WP: MT

Target WP: WP3,5

Goals: Derived from P4 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.11, U2.9, U4.4 and U5.11.

Acceptance criteria: The output is consistent with the mathematical definition.

Status: Completed

Dependencies: None

Requirement R76

Requirement : The output is consistent with the mathematical definition.

Source WP: MT

Target WP: WP3

Goals: Derived from P4 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.11, U2.9, U4.4 and U5.11.

Acceptance criteria: The output is consistent with the mathematical definition.

Status: Agreed

Dependencies: None

Requirement R77

Requirement : The system should enable an analysis of the most likely scenarios that led to compromise of a particular asset

Source WP: MT

Target WP: WP3

Goals: Derived from P4 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.11, U1.17, U2.9 and U5.11.

Acceptance criteria: The output is consistent with the mathematical definition.

Status: Agreed

Dependencies: None

Requirement R78

Requirement : The TRESPASS tools should be able to calculate the effect of differences in the model on the risk values

Source WP: MT

Target WP: WP3

Goals: Derived from P5 (TRESPASS tools should be able to rank attack scenarios in a risk-based prioritisation). Supports Use-case 1 (Security Investment) and Use-case 4 (Product-service system), specifically U1.2 and U4.4

Acceptance criteria: The output is consistent with the mathematical definition.

Status: Agreed

Dependencies: R05

Requirement R79

Requirement : The TRESPASS tools can calculate the effectiveness of a countermeasure in a navigator map from the map, a set of attacker profiles, and a cost function of the countermeasure.

Source WP: MT

Target WP: WP3

Goals: Derived from P6 (TRESPASS tools should be able to calculate the cost-effectiveness of a proposed countermeasure). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.2 U2.11, U4.2 and U5.13

Acceptance criteria: The output is consistent with the mathematical definition.

Status: Agreed

Dependencies: None

Requirement R80

Requirement : The TRESPASS tools can start analyses automatically after updates of the corresponding models.

Source WP: MT

Target WP: WP6

Goals: Supports Use-case 1 (Security Investment), specifically U1.1

Acceptance criteria: Results of the chosen analyses are available after the corresponding events.

Status: Shelved

Dependencies: None

Requirement R81

Requirement : Visualisation of social and technical data, maps, scenarios, countermeasures.

Source WP: MT

Target WP: WP4,6

Goals: Derived from P7 (TRESPASS tools should be able to visualise maps, scenarios, and countermeasures). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.2, U2.6, U2.13, U4.6, U5.9 and U5.15.

Acceptance criteria: The visualisations are sufficiently expressive.

Status: Agreed

Dependencies: R03, R06

Requirement R82

Requirement : TRESPASS tools should be accessible through a web interface

Source WP: MT

Target WP: WP6

Goals:

Acceptance criteria: The web interface functions correctly on the most common platforms (Android, iOS).

Status: Agreed

Dependencies: R02, R06

Requirement R83

Requirement : The TRESPASS integrated component should use loosely coupled tools that are also available individually

Source WP: MT

Target WP: WP6

Goals:

Acceptance criteria: 80perc. of the users specified in the Use Cases are satisfied upon use of each individual tool separately.

Status: Agreed

Dependencies: R01

Requirement R84

Requirement : The TRESPASS analysis should suggest a ranked list of countermeasures, with associated map components which can be dragged-and-dropped into the model and affect the analysis accordingly. The countermeasures should be available in a library.

Source WP: MT

Target WP: WP1,WP2,WP3,WP4

Goals: Supports Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U2.10, U2.12, U.4.2, U5.3, U5.13, U5.14.

Acceptance criteria: 80perc. of the users specified in the Use Cases are satisfied with the suggested countermeasures. 80perc. of users are able to successfully include counter- measures them into the model.

Status: Shelved

Dependencies: None

Requirement R85

Requirement : There must be a method to create an empty model.

Source WP: MT

Target WP: WP1

Goals: Supports Use-case 1 (Security Investment) and Use-case 3 (Innovation), specifically U1.3 and U3.1

Acceptance criteria: There must be a method to create an empty model in the API.

Status: Completed

Dependencies: None

Requirement R86

Requirement : Users can select base models from a Model Template Library.

Source WP: MT

Target WP: WP4,5,6

Goals: Supports Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U4.2, U5.4

Acceptance criteria: 80perc. specified in the case studies are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Assumes the Model Template Library exists

Requirement R87

Requirement : Users can (re-) run analyses manually or at scheduled times/intervals

Source WP: MT

Target WP: WP6

Goals: Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.14, U5.16

Acceptance criteria: 80perc. specified in the case studies are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

Requirement R88

Requirement : The representation of socio-technical security models must provide the necessary abstraction to make the model accessible to both tools and human readers.

Source WP: WP1

Target WP: WP1

Goals: Needed to support both tools and enable simple manual checks.

Acceptance criteria: The TREsPASS tools are able to work with the model, and the users in the case studies can summarize the structure defined by a given text model.

Status: Completed

Dependencies: None

Requirement R89

Requirement : The representation of the socio-technical security model must provide structured access for formal analysis methods.

Source WP: WP1

Target WP: WP1

Goals: Needed to support the interaction between analysis and model.

Acceptance criteria: The analyses developed in WP3 are able to obtain the necessary input.

Status: Agreed

Dependencies: None

Requirement R90

Requirement : The model should support hierarchies and types of artefacts to ease modelling.

Source WP: WP1

Target WP: WP1

Goals: Needed to model some aspects of containment.

Acceptance criteria: 80perc of the users specified in the case studies use hierarchies when modelling their case study where applicable.

Status: Agreed

Dependencies: None

Requirement R91

Requirement : Each element in the model should have a unique identifier.

Source WP: WP1

Target WP: WP1

Goals: Needed to retrieve elements and to ease refering to unique model elements.

Acceptance criteria: Model elements can be accessed by a unique identifier

Status: Completed

Dependencies: None

Requirement R92

Requirement : A reader not intimately familiar with the modelling language should be able to gain a high-level understanding of the model with minimum explanations.

Source WP: WP1

Target WP: WP1

Goals: The modelling language should be selfexplaining to enable quick sanity checks of model files.

Acceptance criteria: 80perc of the users specified in the case studies can summarize the structure defined by a given text model.

Status: Completed

Dependencies: None

Requirement R93

Requirement : The modelling language should support domain experts, with little or no specific training, by allowing them to quickly develop initial high-level models.

Source WP: WP1

Target WP: WP1

Goals: The lower the initial learning curve, the faster the uptake of TREsPASS will be.

Acceptance criteria: 80perc of the users specified in the case studies can after some training use the TREsPASS tools to define a small, high-level model.

Status: Agreed

Dependencies: None

Requirement R94

Requirement : The model must be able to represent locations in the infrastructure and (directed and undirected) connections between them.

Source WP: WP1

Target WP: WP1

Goals: Infrastructure is an essential component of socio-technical systems.

Acceptance criteria: Locations that are present in the case studies can be represented in the according model.

Status: Completed

Dependencies: None

Requirement R95

Requirement : To support formal methods and improve their precision, the model should support different domains for locations.

Source WP: WP1

Target WP: WP1

Goals: Domains allow to avoid impossible analysis artefacts, for example, a human actor entering a computer.

Acceptance criteria: Locations that are in different domains can be assigned to such when modelling the scenario.

Status: Completed

Dependencies: None

Requirement R96

Requirement : The model should support creation and destruction of locations in some domains.

Source WP: WP1

Target WP: WP1

Goals: For certain domains such as cloud infrastrcutures, model elements must be created from insider the model, eg, for virtual machine creation.

Acceptance criteria: For relevant case studies, the creation and desctruction of locations can be modelled, for example, for the cloud case study.

Status: Completed

Dependencies: None

Requirement R97

Requirement : Socio-technical security models must provide the necessary abstraction to represent assets.

Source WP: WP1

Target WP: WP1

Goals: Assets are an essential component of socio-technical systems.

Acceptance criteria: Relevant assets that occur in the case studies can be represented in the model.

Status: Completed

Dependencies: None

Requirement R98

Requirement : Assets must be able to be located at locations in the infrastructure, at actors, or at other assets.

Source WP: WP1

Target WP: WP1

Goals: The model must be able to represent where assets are located to make them accessible to analyses.

Acceptance criteria: The location of relevant assets can be represented in the model.

Status: Completed

Dependencies: None

Requirement R99

Requirement : The model should support tangible and intangible assets (items and data).

Source WP: WP1

Target WP: WP1

Goals: Not all assets are tangible but available as knowledge or data. This is so large a group, that it deserves a categorie of its own.

Acceptance criteria: Relevant data and items that occur in the case studies can be represented in the model.

Status: Completed

Dependencies: None

Requirement R100

Requirement : The model must be able to express containment between assets.

Source WP: WP1

Target WP: WP1

Goals: Containment enables us to model data in a hard disk in a compute.

Acceptance criteria: Assets that are located at assets can be represented in the model.

Status: Agreed

Dependencies: None

Requirement R101

Requirement : Socio-technical security models must provide the necessary abstraction to represent processes and their actions.

Source WP: WP1

Target WP: WP1

Goals: Processes and actions are an essential component of socio-technical systems.

Acceptance criteria: Relevant processes that occur in the case studies can be represented in the model.

Status: Agreed

Dependencies: None

Requirement R102

Requirement : Processes must be able to access intangible assets.

Source WP: WP1

Target WP: WP1

Goals: The model must be able to represent processes handling data.

Acceptance criteria: Relevant processes that work with data can be represented in the model.

Status: Agreed

Dependencies: None

Requirement R103

Requirement : Processes must be able to move in the location domain they belong to.

Source WP: WP1

Target WP: WP1

Goals: Domains reduce analysis artefacts by avoiding processes that enter building locations, for example.

Acceptance criteria: Moving processes are restricted to their domain.

Status: Agreed

Dependencies: None

Requirement R104

Requirement : The assets associated with a process must move with them.

Source WP: WP1

Target WP: WP1

Goals: Processes do not forget data when moving.

Acceptance criteria: For relevant processes the data they "know" is still available once they moved.

Status: Agreed

Dependencies: None

Requirement R105

Requirement : Processes should be able to contain a series of actions that represent the process.

Source WP: WP1

Target WP: WP1

Goals: Actions define processes, so these are essential.

Acceptance criteria: Actions define processes, so these are essential.

Status: Agreed

Dependencies: None

Requirement R106

Requirement : Socio-technical security models must provide the necessary abstraction to represent actors and their behaviour.

Source WP: WP1

Target WP: WP1

Goals: Actors are an essential component of socio-technical systems.

Acceptance criteria: Relevant actors and their behaviour can be represented in the model

Status: Agreed

Dependencies: None

Requirement R107

Requirement : Actors must be able to access tangible and intangible assets.

Source WP: WP1

Target WP: WP1

Goals: The model must be able to represent actors handling assets.

Acceptance criteria: Relevant actors can access the data and assets they should be able to access.

Status: Agreed

Dependencies: None

Requirement R108

Requirement : Actors must be able to move in the location domain they belong to.

Source WP: WP1

Target WP: WP1

Goals: Domains reduce analysis artefacts by avoiding actors that enter computers, for example.

Acceptance criteria: For relevant actors in the case studies it is possible to specify a domain that they are bound to.

Status: Agreed

Dependencies: None

Requirement R109

Requirement : The assets associated with an actor must move with them.

Source WP: WP1

Target WP: WP1

Goals: Actors do not forget data or loose assets when moving.

Acceptance criteria: For relevant actors, the data and items they "know" is still available once they move.

Status: Agreed

Dependencies: None

Requirement R110

Requirement : Socio-technical security models must provide the necessary abstraction to represent policies.

Source WP: WP1

Target WP: WP1

Goals: Policies are an essential component of socio-technical systems.

Acceptance criteria: Relevant policies in the case studies can be represented in the model.

Status: Agreed

Dependencies: None

Requirement R111

Requirement : Policies must require a set of credentials and enable a set of actions, both possibly complete or empty.

Source WP: WP1

Target WP: WP1

Goals: To be able to decide whether a policy is fulfilled, we need to be able to check whether all credentials are provided, and which actions are enabled.

Acceptance criteria: For policies from the case studies, the enabling credential and the enabled actions can be represented.

Status: Agreed

Dependencies: None

Requirement R112

Requirement : Policies should be associated with data and/or locations.

Source WP: WP1

Target WP: WP1

Goals: Both data and locations can access control policies, so the policies must be stored there.

Acceptance criteria: Policies that are associated with data or locations can be stored in the model at the locations that represent them.

Status: Agreed

Dependencies: None

Requirement R113

Requirement : Socio-technical security models must support the storage of data for elements in the model.

Source WP: WP1

Target WP: WP1

Goals: Data represents properties of elements in the model, or analysis results. The model establishes the relation between elements and the data.

Acceptance criteria: For all elements in the model, arbitrary data can be stored at them and retreived.

Status: Agreed

Dependencies: None

Requirement R114

Requirement : The data must be retrievable and settable, based on the element, its type, or a certain action to be performed on it.

Source WP: WP1

Target WP: WP1

Goals: Some of the data is available from the beginning, other will be computed, so it must be accessible.

Acceptance criteria: There exist API functions to retrieve and set data based on certain attributes.

Status: Agreed

Dependencies: None

Requirement R115

Requirement : Socio-technical security models must support the storage of analysis results in the model.

Source WP: WP1

Target WP: WP1

Goals: Analysis results must be stored in the model either for elements or parts of the model.

Acceptance criteria: For all elements in the model, analysis data can be stored at them and retreived.

Status: Agreed

Dependencies: None

Requirement R116

Requirement : The model must support to associate model elements with attacks and vulnerabilities.

Source WP: WP1

Target WP: WP1

Goals: When considering changes to a part of the model, it must be clear which attacks this part is identified in.

Acceptance criteria: For all elements in the model, attacks and vulnerabilites can be stored at them and retreived.

Status: Agreed

Dependencies: None

Requirement R117

Requirement : The model must support locations that represent countermeasures.

Source WP: WP1

Target WP: WP1

Goals: The TREsPASS process not only identifies attacks, but also adds countermeasures. These must be storeable in the model.

Acceptance criteria: For relevant case studies, identified countermeasures can be represented in the model.

Status: Agreed

Dependencies: None

Requirement R118

Requirement : There should exist an API to the model that supports access to model elements and data associated with these elements.

Source WP: WP1

Target WP: WP1

Goals: When developing model tools, there must be an API to create the model and to access the data stored in the model.

Acceptance criteria: The TREsPASS tools access the model through the APIs, not directly.

Status: Agreed

Dependencies: None

Requirement R119

Requirement : Model elements must be accessible by their identifier.

Source WP: WP1

Target WP: WP1

Goals: Model parts can contain other parts, that are represented by their identifiers. The API must be able to obtain the model part with this identifier.

Acceptance criteria: The API provides functionality to obtain model elements.

Status: Agreed

Dependencies: None

Appendix B – System architecture and data flow @ 23.06.2015

Each stage (grey container) involves several activities – represented as boxes – not all of which are required for every risk assessment scenario. A dotted border indicates an optional activity, while a solid border means the activity is required in order to advance to the next stage. Most activities are digital and either fully automated (grey boxes) or require user interaction (blue boxes), but some are carried out without the help of a computer (pink boxes).

Each activity is supported by one or more dedicated tool(s), as indicated in green. Of course, most tools assume input (incoming arrow) or produce output (outgoing arrow). The nature of this input and/or output is indicated using data store symbols: grey to indicate organization-specific content and blue to indicate generic data. The underlying text, contained between square brackets indicated the format in which the data is stored or exchanged. Finally, a solid incoming line means the input is required for the tool and its respective activity to function, while a dashed arrow means it is not optional.

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 (see D5.1.2. (The TRESPASS Project, D5.1.2, 2015)).

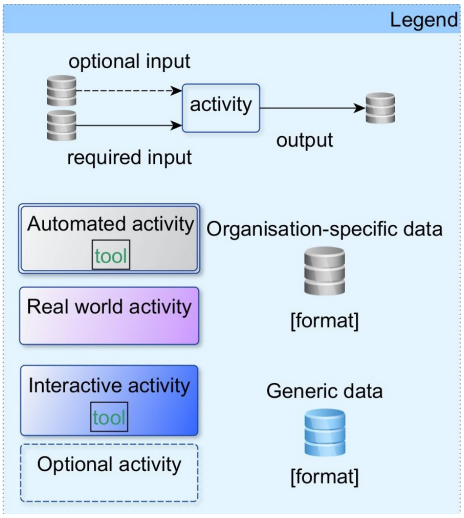

Figure 1: System architecture diagram legend

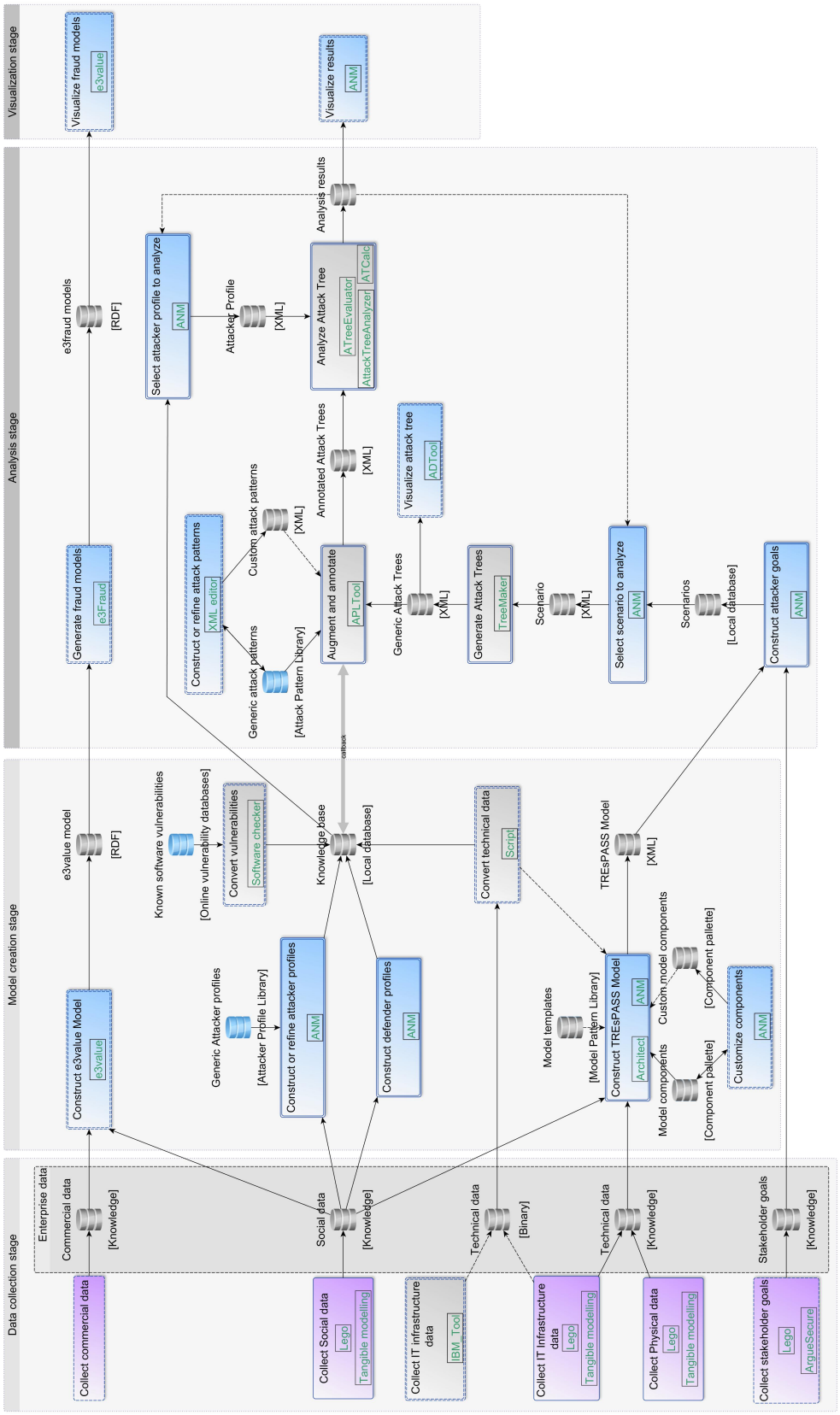

Figure 2: TRESPASS system architecture diagram