predict
prioritise
prevent

TRESPASS

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

Deliverable D1.3.2

Extensibility of Socio-Technical Security Models

Project:	TRE$_S$PASS
Project Number:	ICT-318003
Deliverable:	D1.3.2
Title:	Extensibility of Socio-Technical Security Models
Version:	1.0
Confidentiality:	Public
Editor:	Sven Übelacker
Cont. Authors:	S. Bleikertz, J.-W. Bullée, M. Ford, O. Gadyatskaya, D. Gollmann, R.R. Hansen, D. Ionita, H. Jonkers, L. Montoya, C.W. Probst, S. Saraiva, A. Tanner, S. Übelacker, L. Wolos, A.S. Yesuf
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
AAU	René Rydhof Hansen	3
BD	Henk Jonkers	2, 3
CYB	Margaret Ford	2
DTU	Christian W. Probst	all
GMVP	Sérgio Saraiva	2
GUF	Lars Wolos	2
GUF	Ahmed S. Yesuf	2
IBM	Sören Bleikertz	2
IBM	Axel Tanner	2
TUHH	Dieter Gollmann	1, Quality Assurance
TUHH	Sven Übelacker	all
UL	Olga Gadyatskaya	3
UT	Jan-Willem Bullée	2
UT	Dan Ionita	2
UT	Lorena Montoya	2

Quality assurance		
Role	Name	Date
Editor	Sven Übelacker	2015-10-30
Reviewer	Marianne Junger	2015-10-20
Reviewer	Sérgio Saraiva	2015-10-25
Task leader	Christian W. Probst	2015-10-30
WP leader	Christian W. Probst	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:

- The TRE$_S$PASS model can be extended to be adapted to support emerging requirements due to organisational changes or due to new attack methods.
- The dynamic extensions to the model can be performed ad-hoc as internal embeddings, as model extensions, or as tools.
- The dynamic extension mechanisms are demonstrated by and evaluated through the case studies.

The primary goal of the TRE$_S$PASS project is to develop tools that facilitate assessment and management of IT security-related risks in an organisation, spanning both technological and sociological issues. The risk assessment requires the modelling of the organisation under scrutiny, and it requires a modelling of external threats. These two topics are discussed in the two closely related deliverables: the current one (D1.3.2) and deliverable D1.3.3 (The TRE$_S$PASS Project, D1.3.3, 2015); where D1.3.2 discusses how to extend the model to address unforeseen requirements of the modelled organisation, D1.3.3 discusses how to evolve the model due to unforeseen threats and contextual changes, particularly temporal ones.

The TRE$_S$PASS socio-technical security model forms the basis for identifying possible attacks on an organisation. The technical aspects of this model are based on a combination of best practices for socio-technical security models found in literature and the evaluations throughout the TRE$_S$PASS project. Task T1.3 is concerned with developing model feature extensions to handle, for example, actor behaviour and detection controls. This task extends the model with support for specifying the dynamics of socio-technical security models; both in the sense that they must evolve when the organisation or its context evolves, but also in the sense that the availability of resources determines whether at a given time an attack scenario is actually feasible.

The current deliverable D1.3.2 describes how to develop socio-technical security models with support for requirements intrinsic to the organisation that cannot be handled by the model yet. We describe several approaches to this challenge, partly through internal embeddings into the existing language, partly through extensions of the language, and partly through specialised tools. The developments are based on requirements from the TRE$_S$PASS case studies, taking into account the results from the validation of the model in the case studies.

1. Introduction

This deliverable describes extension mechanism for socio-technical security models that have been explored in the TRE$_S$PASS project. Appendix A provides the context for this deliverable in the TRE$_S$PASS project. It describes the overall summary of the project and the TRE$_S$PASS workflow. Figure 1.1 shows the section of the TRE$_S$PASS workflow that is addressed by this deliverable.

The fundamental challenges in model-based risk analysis relate to the fact that two intrinsically different views of a system need to be captured and aligned: the operator's view and the attacker's view. The operator's view is framed by the intended use of a given system; it will include the features relevant for describing the operation of the system and may include defences against anticipated types of attacks.

For modelling the attacker's view, there are two approaches. On the one hand, the operator's view could be extended with attack points and lists of attacks possible at each point. Alignment of the two views is easy in this case, but there is a danger that the view on attacks is blinkered by too much familiarity with the intended use of the system. Attacks

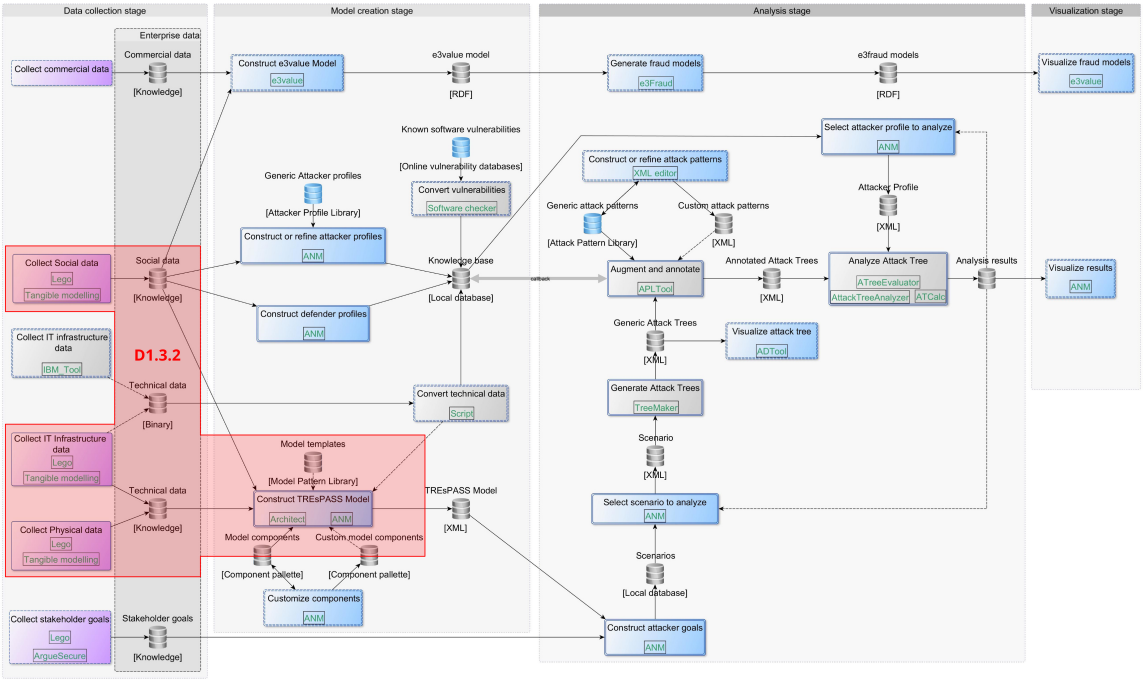


Figure 1.1.: Integration diagram for the TRE$_S$PASS project and the area that this deliverable addresses. Figure A.2 shows a larger version of the diagram.

that exploit features outside of the system model will be missed. An artful attacker, however, will search exactly for gaps between the operator's model and the actual system that could provide the levers for an attack.

On the other hand, the attacker's view could be created independently of the operator's view. Initial information about the system needs to be available; potential attacks identified at this stage may turn out to be infeasible because of specific features of the system under analysis. Here, alignment of the two views will in general be more challenging.

The TRE$_S$PASS model follows the first approach. It supports automated analysis for anticipated types of attacks. When a new type of attack is found that exploits features of the system not captured in the model, the model needs to be augmented. When a newly required feature cannot be modelled, the modelling capabilities need to be extended. This raises an engineering question: Should a given modelling tool be extended with new features, or should the risk analysis methodology be extended with a new tool?

To answer this question, the TRE$_S$PASS project has explored the scope and the limits of the TRE$_S$PASS modelling tools used in selected case studies. This deliverable summarises the findings from these case studies with respect to new features not yet contained in the TRE$_S$PASS model that emerged in these case studies, and the decisions made on whether to extend the TRE$_S$PASS model or to add a new item to the TRE$_S$PASS portfolio of analysis tools. Various extensions enable model parts to express a flexibility that covers upcoming feature requests. Other parts of the model should be regarded as fixed.

We define "extensibility" and "extension" as follows: **Extensibility** describes the capability of the TRE$_S$PASS model to be extended with elements that grant it the ability to widen the model applicability to newly identified aspects in a changing attack landscape. Extensibility itself can consist of various **extensions** like the refinement of model components, extending models by adding countermeasures or information on geolocation as well as demanding new elements, policies, etc. In general, we may want to add features because

- Novel *operational* aspects need to be considered, e.g., geolocation information in the ATM and cloud case studies, or new defences implemented in a system, or
- Newly identified *loopholes*, i.e., exploitable gaps between the model and the actual system, need to be closed.

We believe that the kind of modeller dilemma described above, and a similar model developer dilemma, are typical for modelling approaches. The findings and practical motivations for our approach should therefore be of interest to the wider modelling community.

1.1. Goals

The goal of this deliverable is to capture how the experiences gained in various case studies have influenced the development of the TRE$_S$PASS model. To this end brief sum-

maries of these case studies are given, examining whether the features of the original TRESPASS model were already sufficient for modelling the scenario investigated, and identifying any missing features relevant for that scenario if that should be the case.

The deliverable discusses, which features were implemented as extensions to the original TRESPASS model, and where the decision was made to capture a new feature by a new tool added to the TRESPASS toolbox. Adding features to an existing model by necessity adds some degree of complexity to the supporting tools. There is thus a general engineering challenge on how to strike the balance between having a comprehensive model and having tools that are easy to use by practitioners. The deliverable will explain the engineering decisions made in the TRESPASS project.

1.2. Choices Made

When new features were conceptually similar to features of the original TRESPASS model, they were implemented as extensions of the TRESPASS model. In many cases, these would be extensions of the operational model, e.g., the inclusion of geolocation information in the ATM and cloud case studies.

When the new feature was sufficiently distinct from those of the original TRESPASS model, e.g., the value flows in the Telco case study, a different modelling approach and analysis tools were developed. The rationale for the new features were attacks against the operators' business models, not so much attacks exploiting deficiencies in physical protection or in IT security. The decision was thus made not to overload the TRESPASS model with features for modelling business cases and value flows, but to cover these aspects in a separate tool designed for this purpose.

1.3. Foreground and Background

Of the findings presented in this deliverable, the ArchiMate language extension mechanisms and overlays discussed in Section 3.2.1 are background, and the remaining 90% are foreground.

1.4. Document Structure

The rest of this document is structured as follows. After a presentation of the case studies in Chapter 2, we discuss in Chapter 3 the different kinds of extensions developed in the TRESPASS project, ranging from internal embeddings over language extensions to new tool developments. Finally, Chapter 4 concludes this deliverable.

2. Case Studies

To motivate our approaches to extension described in Chapter 3, we now describe the case studies in the TRESPASS project and their requirements for extensibility of the model.

2.1. Customer Privacy Protection

The Customer Privacy Protection case study has developed over the life of the project to encompass two cases that both are focused around customer delivery in a financial services environment. Both cases have significant technical aspects, as well as social and, partly, geographical aspects.

The IPTV case study was the first to be developed and in terms of its technical architecture is relatively simple. In this respect, it did not require any significant extensions to the model and there are currently no requirements or plans to implement any in the future. The one concept which may have slightly extended the original plans for the model was the need for a concept of 'trust' between actors in order to model potential social engineering attacks on the user of the IPTV system.

While working with our case study partners, it was found that using the ArchiMate modelling language during early discussions provided a very useful way of gathering initial requirements. When checking these with the group on a subsequent meeting, they responded well to the ArchiMate diagrams, finding them a useful way to order the different elements of the scenario.

The ATM case study has been developed to test some of the methods developed within the project at a greater scale and to develop further the geographical capabilities of the TRESPASS modelling approach. It is still in its relatively earlier stages, although the scenario has been agreed and some initial data gathered.

The IPTV[1] and ATM[2] case studies are described in more detail below.

[1] The IPTV case study summary was provided by WP7 and is shared across public deliverables like The TRESPASS Project, D1.3.3 (2015). Section 2.1.1.2 was added by this deliverable.

[2] The summary of the ATM case study originates from The TRESPASS Project, D1.3.3 (2015) to offer the reader a comprehensive case study overview without reading other deliverables first. Section 2.1.2.1 and Section 2.1.2.2 were created by this deliverable.

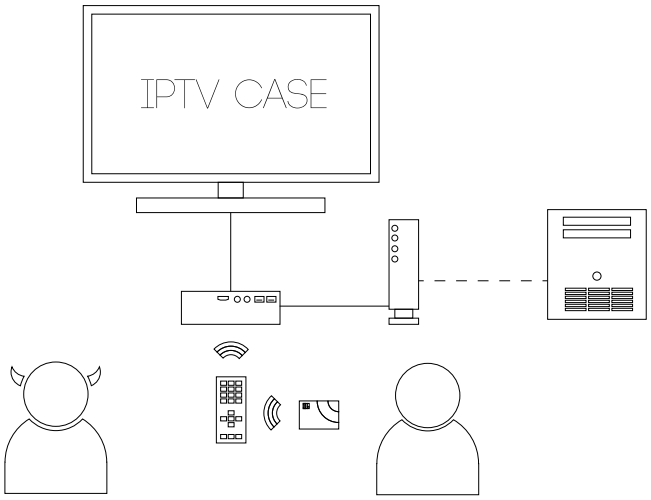


Figure 2.1.: IPTV case study

2.1.1. IPTV

In this section we describe the "IPTV case study", that has been used as a running example throughout the project[3]. The technical details as well as the people and companies involved are confidential and we therefore present an anonymised and slightly redacted version of the original case study. However, all the important features have been retained and the processes are fundamentally the same as in the original case study.

The case study concerns a system supporting primarily elderly and disabled people in performing online payments and managing their own money from their home. With the target demographic in mind, the system should be integrated into an existing device that is familiar and easy to use for the intended user groups, namely the television set. In practice this is accomplished by hooking up a small, dedicated computer to the TV and an enhanced remote control with a built-in card reader for authentication as illustrated in Figure 2.1. In this case study there are many different security aspects that may be considered: from the strictly technical, such as how information is protected while stored or transmitted, to the socio-technical, covering security issues arising from the use of and interaction with the technology. Within the project, we have explored the socio-technical features of the case study, as a means of validating the methods and tools which are being developed. In particular this case study has provided a context in which to explore the many possible approaches to handling social data.

2.1.1.1. Normal Usage and Context Assumptions

Figure 2.1 shows an overview of the IPTV case study: there are two primary actors: the attacker (represented by a devil in the figure) and the victim (the IPTV owner/user). Under normal operation, the user would first open a session on the IPTV, using a standard

[3]Here IPTV refers to television service(s) over the IP protocol.

password based authentication scheme. From this session, the user can then use different services, e.g., pay a bill or transfer money, by using a payment card with concomitant PIN code. The payment card is read by a card reader built into the IPTV remote control on which the PIN code is also entered.

The design is intended to be simple and offer the means for people of all ages and abilities to be able to access the services they require. It is intended to complement other means of delivery, not to replace them. In particular, it offers the opportunity for people who are not familiar or comfortable with mobile technology to receive the benefits of the ever-increasing range of mobile services in their homes via their television screen (Egelman, Brush, & Inkpen, 2008). Although this system could offer great convenience, it also has the potential to expose the account holder to significant social risks, particularly those stemming from the involvement of both professional carers and family members. These carers could be considered as knowledge insiders, with the potential to act as malicious insiders.

We have made a number of assumptions about the context for the case study:

1. The card-holder has a functional IPTV in his/her house prior to the attack.
2. The IPTV security configuration ensures security for the communication of data between the different physical devices.
3. One Internet Service Provider (ISP) is used for all Internet access.
4. The source code of the software of the IPTV system is not freely available.
5. Firmware updates are not cryptographically encoded.
6. The IPTV set-top box uses a standard API.
7. The user can log on and off the IPTV system at will.

While these assumptions help delineate the scope of the case study, they are not critical and can be relaxed or modified to better capture a specific system.

2.1.1.2. Modelling with ArchiMate

During the work with the case study owners, it became quickly clear that a computer-based modelling approach would be beneficial. Since this case study started very early in the TRESPASS project, no such tool approach was available yet. Instead, we opted to support modelling by adopting a tool for the ArchiMate language to extend the support for modelling. This development is described in Section 3.2.1.

2.1.2. ATM

ATM machines are composed of a money safe and a computer that controls the ATM's devices (screen, keyboard, printer, network interfaces, money safe mechanisms, etc.) through software programs. Most ATM computers are composed of outdated hardware running legacy operating systems (i.e., Windows NT), which are not supported by the vendors or by the anti-virus providers. The installation of the ATM machines varies as 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. Examples include:

- Physical attacks where the attacker physically steals the ATM to open the safe and take the money.
- Digital attacks where the attacker installs a malware agent into the operating system to take control of the devices including the ability to withdraw money from the safe through the device' s interfaces.

In order to perform a proper risk assessment of the ATM network as a whole, three types of data must be considered and combined into a single unified model, suitable to be processed on a consolidated data schema:

- Technical data (about the machines)
- Environmental data (about the territory)
- Historical data (about past occurrences)

Attacks do not happen randomly in time and are neither equally distributed over the territory. Machines installed on certain places are more likely to be attacked than others, thus spatio-temporal hotspots exist. This case study therefore adds both the temporal and a geographic dimension to the project.

The ATM case study builds an analysis process at the macro-level based on data from the ATM network and surrounding area, and outputs the vulnerability level of each ATM. More specifically, this case study illustrates how to perform an analysis at a macro level for identifying priority areas in need of detailed analysis using the TRE$_S$PASS model and tools.

2.1.2.1. ATM Geolocation Extension

It is widely acknowledged in the field of criminology (refer to the Crime Pattern Theory (Brantingham, 2010) Rational Choice Theory (Clarke, 1997) and Routine Activity Approach (Cohen & Felson, 1979)), that given target objects of equal benefit, risk and cost, the offender will choose the one which is physically closer to him. In other words, a crime distance decay effect exists (O'Leary, 2011). For this reason the TRE$_S$PASS model could be extended to include distances. Distances could be measured in x/y coordinates or

alternatively they could be also operationalised in more complex ways which include 'friction' aspects and that could be modelled as, e.g., time. In this scenario we consider the operationalisation of distance as x/y coordinates.

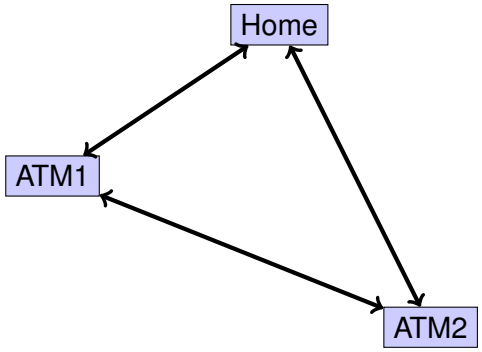


Figure 2.2.: Scenario

```
<?xml version="1.0" encoding="UTF-8"?>
<system
    xmlns="..."
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="127.0.0.1"
    author="J.H. Bullee"
    date="27-05-2015"
    version="alpha">
  <title>Tiny-Bank</title>
  <locations>
    <location id="Home" domain="physical" />
    <location id="ATM1" domain="physical" />
    <location id="ATM2" domain="physical" />
  </locations>
  <edge>
    <source>ATM1</source>
    <target>ATM2</target>
  </edge>
  <edge>
    <source>ATM1</source>
    <target>Home</target>
  </edge>
  <edge>
    <source>ATM2</source>
    <target>Home</target>
  </edge>
</system>
```

Listing 2.1: Full TRESPASS model

```
  <locations>
    <location id="Home" domain="physical" location="(6,6)"/>
    <location id="ATM1" domain="physical" location="(3,4)"/>
    <location id="ATM2" domain="physical" location="(8,2)"/>
  </locations>
```

Listing 2.2: Extension

The scenario involves a snapshot of a financial organisation with multiple ATMs and customers. The location of each ATM is known as well as the home address of each customer. The scenario consists of 3 locations, 1 customer's home and 2 ATMs (refer to Figure 2.2). A textual version of the model is provided as well in Listing 2.1.

In the current version of the TRESPASS model, the positioning of the objects relative to each other has no meaning other than that provided by the edge, which connects two `locations` indicating that it is possible to travel from one `location` to another `location`. The direction of the edge indicates the travel direction.

An addition to this model therefore consists of annotating `locations` with x/y coordinates. This extension has the following benefits:
i) `Locations` can be bound to geographic locations, ii) Knowledge of the geographical locations allows distance calculations and iii) This extension can be an initial step leading to further extensions, e.g. a map overlay for adding more environmental/context information.

In the TRESPASS file format, the `locations` can be easily extended by adding a location-tag. This tag includes an x and y coordinate related to a position on the map, e.g. `location="(6,6)"`. An example of this extension is provided in Listing 2.2. A refinement could include a z coordinate to denote floor level.

The equation to calculate the distance between two `locations` is based on the Pythagorean theorem. Let $\mathbf{p}$ and $\mathbf{q}$ each be a Cartesian coordinate (x, y). Then distance $\mathbf{d}$ is calculated by: $\mathrm{d}(\mathbf{p}, \mathbf{q}) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$. An example illustrates the distance between 'Home' and 'ATM1', which with Home situated at $\mathbf{p}(6, 6)$ and ATM1 situated at $\mathbf{q}(3, 4)$, computes as: $\mathrm{d}(\mathbf{p}, \mathbf{q}) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} = 3.61$.

2.1.2.2. Geolocation Application: ATM Heat Map

TRESPASS partner GMV Portugal created for the Lisbon region an ATM heat map combining several public resources. The geolocation extension is one essential factor that made these risk assessment heat maps possible, cf. Figure 2.3 and Figure 2.4.

The heat maps represent the likelihood of an ATM to be attacked (high level risk evaluation or statistical level) calculated by evaluating several layers of technical, social, environmental and historical data. This geographic evaluation consists on a high level analysis (statistical level) and aims to provide hints regarding the ATM network. These hints enable to understand hotspots for possible attacks knowing that risk is not uniform across the territory: in some zones, ATM attacks are more likely to happen. This statistical level is performed taking several dimensions of analysis into consideration: attacker profile (amateur, professional, others), attack type (physical, malware, others), season (winter, summer, etc.), day period (night, day, etc.), and similar. Each one of these dimensions can produce very different results. Providing these hints to a detailed analysis process (TRESPASS attack tree based model), that should be performed for each machine individually, is one objective of the ATM case study.

2.2. Cloud Computing

Cloud computing[4] has gained remarkable popularity in recent years due to the economic and technical advantages of this new way of delivering computing resources. Customers benefit from rapid provisioning and seemingly infinite scalability, while only being charged on a pay-per-use basis. Computing resources can be provided on different abstraction layers (Mell & Grance, 2009b), where the lowest one provides basic resources (servers, network, and storage), and higher ones provide applications to the end-users (e.g., Google's GMail, Salesforce.com). In this case-study we are focusing on the lowest abstraction layer, that is *Infrastructure-as-a-Service* or *Infrastructure Clouds*, since it is the most generic layer and higher ones often build upon this layer.

Although the benefits of cloud computing are evident and users demand cloud services, security is a major inhibitor (Mell & Grance, 2009a). An analysis of risks and threats in cloud computing has been conducted in Cloud Security Alliance (2010) and ENISA (2009). In particular, both reports agree that insider attacks and malicious insiders are a major risk and are among the top 10 threats. The risk is amplified due to the disappearance of physical boundaries that makes it very challenging to define a security perimeter that divides insiders from outsiders (Hay, Nance, & Bishop, 2011; Pieters, 2011).

Figure 2.5 provides an overview of the entities and components involved in a model of an infrastructure cloud. As compared to a typical IT department within an organisation, the

[4]This summary of the cloud case study originates from The TRESPASS Project, D1.3.3 (2015) to offer the reader a comprehensive case study overview without reading other deliverables first. The section "Extensibility Outlook" (Section 2.2.1) was created by this deliverable.

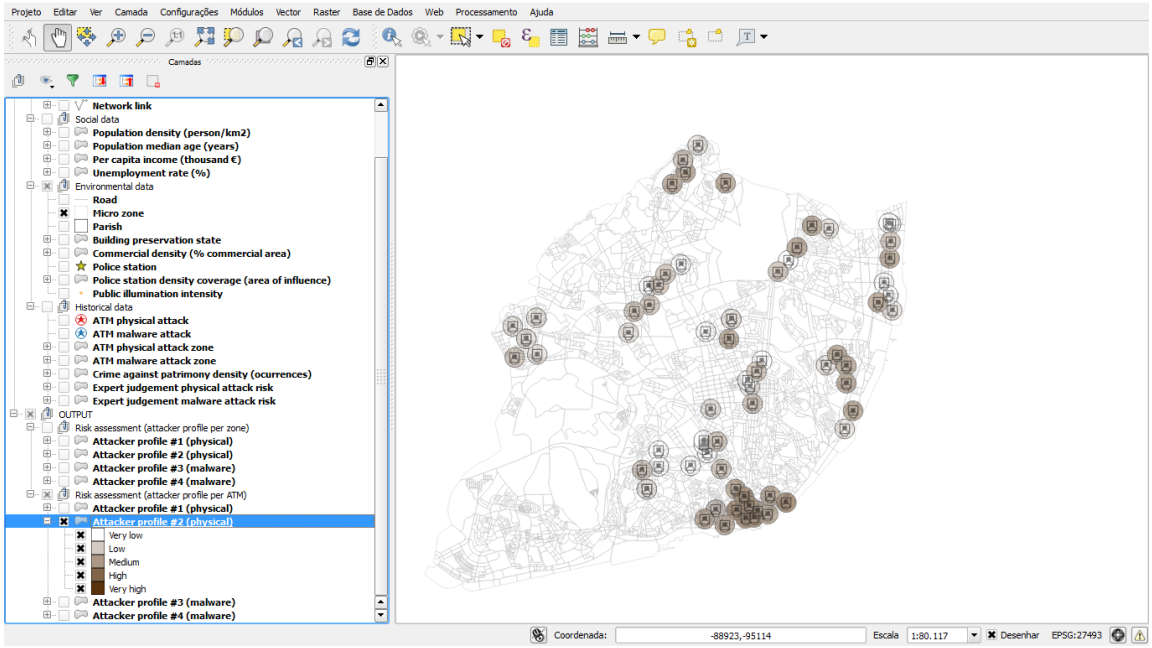


Figure 2.3.: ATM heat map of Lisbon for attacker profile per ATM

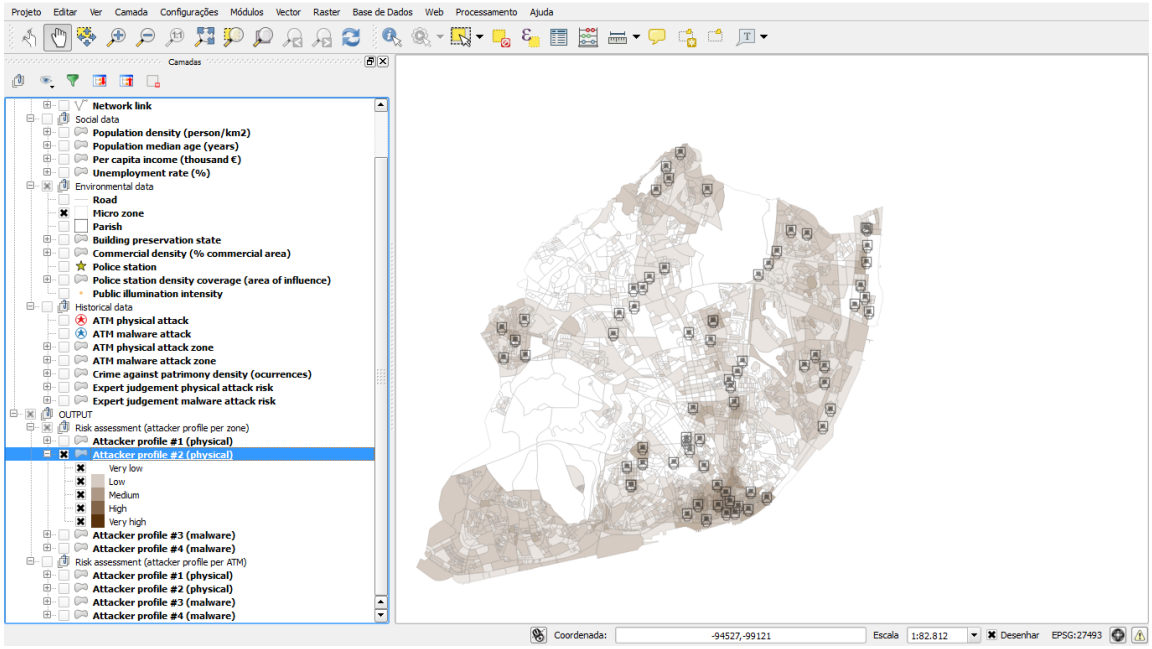


Figure 2.4.: ATM heat map of Lisbon for attacker profile per zone

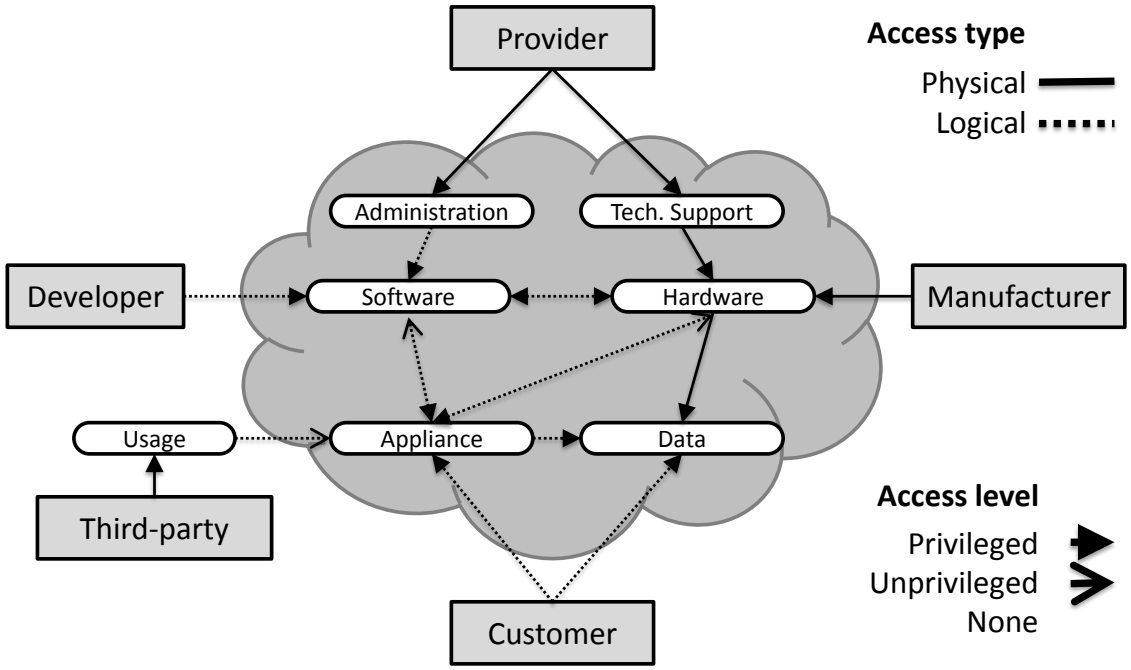


Figure 2.5.: Overview of entities and components in an infrastructure cloud model.

provider of the cloud services in such a shared infrastructure cloud is a new additional and powerful party. The multi-tenancy (different customers, including competing organisations, using the same cloud services) leads also to an unprecedented sharing of computing and

infrastructure resources. The cloud provider therefore becomes an important additional factor (and risk), as it has full physical and logical access to all resources across the different consumers. In addition, the infrastructure itself is dynamic and flexible in response to the requirements of the customers.

For cloud infrastructures, the TRE$_S$PASS model distinguishes components at a level of abstraction that corresponds well to security-relevant control points in these domains, enabling the discovery and analysis of potential attacks that exploit their connectivity. Using the model, one can formalise typical components in cloud infrastructures and their interrelationships. These include network components like switches, routers, firewalls; virtual and physical servers; actors, including administrators, users, and attackers; location details that represent rooms, doors, and other physical consideration. Because these component models show how actions on one element influence other elements, they can be combined with the connectivity relations to form an implicit search-space of all possible activity paths in the system.

2.2.1. Extensibility Outlook

The cloud case study does not require to add any extensions in the form of language extensions to the existing model currently. The major requirement to be able to model the network connectivity between nodes in a cloud infrastructure can be added through internal embeddings (Nidd, Ivanova, Probst, & Tanner, 2015).

Possible extensions probably on the language or tool level could express the need to include jurisdictional information into the model, for example, based on geolocation. The location information could help to clarify compliance issues for privacy and data protection as required by national, state, or in other laws and regulations, for example, on the European level.

In an international context, a customer of cloud computing services could require compliance with certain geolocation policies in the contract or SLA. Policies could cover data protection or security rules to ensure, e.g., *1)* location of virtual machines including the data processed within the virtual machines, *2)* storage location of persistent data (e.g., backups, archiving or timestamped remote logging), *3)* location of network equipment (physical) as well as *4)* its configured routing (digital), or *5)* office location from where all components are supervised by the service provider.

2.3. Telco

Telecommunications products[5] and services are functional in a complex environment involving a multitude of different interconnected networks, service providers and network

[5]This summary of the Telco case study originates from The TRE$_S$PASS Project, D1.3.3 (2015) to offer the reader a comprehensive case study overview without reading other deliverables first. The section "e3fraud" (Section 2.3.1) was created by this deliverable.

operators with opposed financial interests acting in highly competitive markets offering complex products and services.

Due to the market structure indicated above, new customers are not easily available, but normally need to be lured away from competitors. As the providers try to escape the pricing pressure resulting from replaceability of the communication goods, e.g. new tariffs more and more become a mixture of free (flat rate) components being heavily advertised and, less noticeable, much more expensive components for compensation and revenue generation. New products need to be launched under significant time pressure resulting from strong competition in the market, leaving little time and space to account for potential misuse of the product. Apart from that, often the misuse resulting from product design flaws is a learning process which takes place once the respective product or service has been launched into the market.

The above tendency, in contrast, encourages cherry-picking among customers of Telco companies. This is especially true for so-called knowledge insiders that know the market very well, trying to make as much use of (or monetary gain from) the products offered as possible.

An example of the sort of fraud addressed by the project involves using insider knowledge to exploit flat rate tariffs for call termination, which despite the terms and conditions of the respective network operators, provides a profit for the fraudster (refer to Figure 2.6). An example includes fraud involving the false pretence of being willing and able to pay for calls.

2.3.1. e3fraud

Telecommunication services are complex product packages that rely on a large and complex technical infrastructure. However, fraudulent use of such telecommunication services rarely exploits hardware vulnerabilities. Instead, most common exploits operate at a business level, capitalising on the unexpected interaction between various product packages from multiple providers.

An investigation of several known telecommunication fraud scenarios revealed that, contrary to more technical attacks, they are best described in terms value exchanges (such as money or services) amongst profit/loss responsible actors (The TRE$_S$PASS Project, D7.3.1, 2014).

It became quickly clear that fraud scenarios are fundamentally different from the kind of attacks investigated in socio-technical security models such as the ones considered originally in TRE$_S$PASS. The lack of support for fraud in the model is however so substantial, that it was decided to apply a separate modelling approach and tool, described in Section 3.2.3.

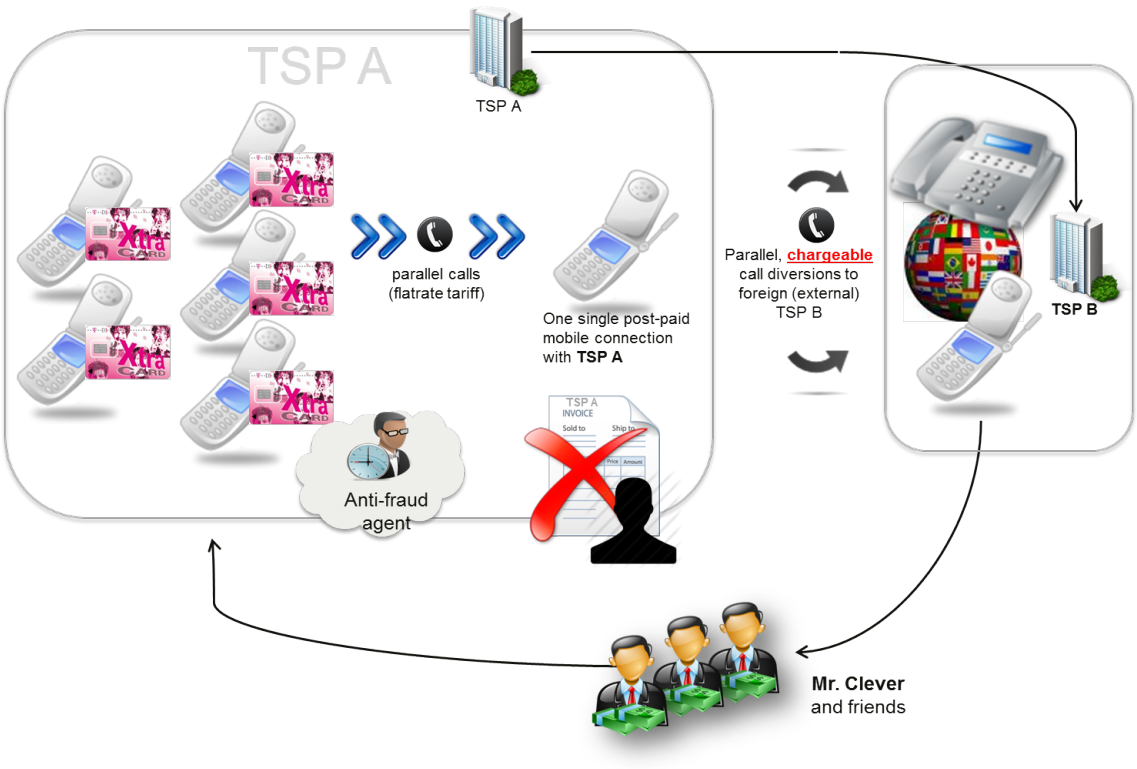


Figure 2.6.: Fraud scenario – fraud involving the false pretence of being willing and able to pay.

3. TRESPASS Model: Continuity through Extensibility

In this section we now present the different approaches used in the TRESPASS project for extending the modelling capabilities based on requirements intrinsic to the organisation. As discussed above, we have explored three avenues, which all have their drawbacks and advantages: internal embeddings, extensions to the language, and tool support.

3.1. Extending the Model Language

The TRESPASS model is at the core of the processes and tools developed in the project: it constitutes the interface between the attack navigator map, analyses, and visualisations, and the organisation under scrutiny. As with all models and abstractions, the expressivity of the modelling language constrains what can be put in the model, i.e., what elements of the organisation can be represented in the model. While the vanilla modelling language (The TRESPASS Project, D1.3.1, 2013) supports a wide range of elements that can be expected to be in models, it clearly only covers a subset what modellers will consider necessary.

The TRESPASS modelling language can therefore be extended with new elements, or existing elements can be refined (Probst & Hansen, 2008). In this section we list some examples for such extensions.

The issue with model extensions is their embedding into the TRESPASS model. In principle, two approaches are possible:

- Low-level embeddings translate the model extension to elements that exist in the original modelling language, while
- High-level embeddings actually extend the language with new constructs.

These different kinds of extensions come each at a price: for low-level embeddings, the translation into elements of the original language must be defined and checked, and the translation must be performed before the model is accessed in the TRESPASS processes and tools. For high-level embeddings, on the other hand, the necessary changes are more involved:

- The representation of models must be enhanced with the new elements;
- The model API must be extended to support the new elements;

- The semantic rules for the new elements must be specified; and
- The analyses and visualisations must be extended to be able to deal with the new elements.

While the necessary steps for high-level embeddings make it seem very unattractive to use them, they have one huge benefit over their low-level siblings: the analyses and visualisations *know* about them, and can treat them accordingly. For low-level encodings, the model does not preserve information on, what the original extension was; even if it did, the analyses and visualisations would not be able to deal with them as they are forgotten in the embedding.

In the remainder of this section we discuss several examples for extensions to the model.

3.1.1. Policies

The TRE$_S$PASS policy language consists of two components – the policy itself and one or more processes, which might be triggered if a policy is enabled by an actor. This separation has proven to be very powerful in simulating other policy languages (The TRE$_S$PASS Project, D1.2.2, 2015). At the same time, it is an obvious hook to extend the modelling language through processes that enable certain actions in the modelled organisation.

Another extension of policies is based on replacing the credentials that in TRE$_S$PASS policies are used to enable actions. In this case, almost any policy language can be used that can be translated to DataLog queries, the approach used internally in attack generation and policy validation.

Both these extensions are embedded as internal embeddings, which means that the knowledge about the external policy language or the connection between policy and process is lost. However, this loss of information is deemed acceptable, since the information is (currently) only used in the attack generation to identify which credentials an attacker needs to perform a certain action.

3.1.2. Language Extension with New Actions

TRE$_S$PASS-like socio-technical models can be extended with additional actor actions, such as *Destroy*, and explicit object properties, such as *stealable* and *lockable*.

To support new properties, we can extend the model with predicates. For example, the property that a door can be locked or unlocked can be abstracted by a predicate *locked*: it is true when the door is locked, and false when the door is unlocked. Not-lockable doors are always unlocked. For objects we may also add predicates *movable* and *destroyable*. To enable actors to perform the actions implied by predicates, we then can add new actions *Destroy*, *UnLock*, and *Lock*. Actors in the extended model can explicitly manipulate the objects directly.

These properties and actions are difficult to model implicitly as internal embeddings, partly since they require rewriting policies in the model, partly because properties do not have direct relations to actions in the TRESPASS model. We have therefore chosen to add these extensions on the language level.

As discussed above, this extension approach implies a significant amount of work, since all components of the process must be adapted, from the semantics of the underlying formalism, over analyses and visualisations, to tools and implementations. The extended model has been implemented and then analysed with the probabilistic model checker PRISM (Lenzini, Mauw, & Ouchani, 2015). This exercise has given us more insights about finding the right balance between the expressivity and the overhead in modelling complex scenarios with socio-technical model languages.

3.2. Extensions through Tools

In the previous section we discussed internal and external extensions of the modelling language. As mentioned before, this is way of extending the modelling language is not always feasible; in some situations, the benefit of using a completely different modelling language and tool outweighs the benefit of staying in the establishing modelling approach.

In this section we present two such tools: One approach based on ArchiMate, which in uses a different modelling language and the associated tools, the other approach based on e3fraud, a modelling approach that addresses business value models to identify fraud. The two tools represent techniques resembling the two approaches mentioned before:

- The ArchiMate-based language and tools combine internal and external embeddings. The internal embedding is a translation of the internal model to a TRESPASS model, the external embedding is the analysis of the ArchiMate language with its tools, and mapping the results back.
- The e3fraud model uses an external embedding approach. The e3fraud models are currently not easily transformed into TRESPASS models; instead, the analysis results can be mapped back just like for ArchiMate.

3.2.1. Extensions in the ArchiMate Language

Within some of the TRESPASS case studies, the ArchiMate enterprise architecture modelling standard is used as a front-end to construct socio-technical security models. The ArchiMate standard (The Open Group, 2013) describes two main mechanisms that organisations or end-users can apply to extend the language: (1) by means of *specialisation* of language concepts (elements or relationships), similar to the use of stereotypes in UML, and (2) by means of adding (sets of) *attributes* to language concepts.

In a white paper published by The Open Group it is shown how the ArchiMate language can be extended for modelling risk and security aspects. This includes both concepts

to support risk analysis and concepts to support security deployment, i.e., for specifying control objectives and (requirements for) control measures. Three options to do this are identified, in decreasing order of preference:

- The use of ArchiMate concepts unmodified, as specified in the standard.
- The use of the extension mechanisms as specified in the standard to define additional attributes or specialisations of existing ArchiMate concepts.
- The use of additional concepts that do not yet exist in the ArchiMate standard and cannot be directly linked to existing concepts.

The white paper also proposes a “risk and security overlay” for the ArchiMate language, defining specialisations of both ArchiMate core elements and elements from the Motivation extension (the latter mainly for risk analysis), using the first two options listed above. Figure 3.1 shows the structure of this overlay. The red elements are specialisations of core and motivation elements to model the results of a risk assessment, while the green elements are specialisations of motivation elements to model control objectives, principles and control measures. Table 3.1 summarises the elements, their definition and possible attributes (based on, e.g., the FAIR standard (Josey et al., 2014), or a security profile that specifies security attributes such as confidentiality, integrity, and availability).

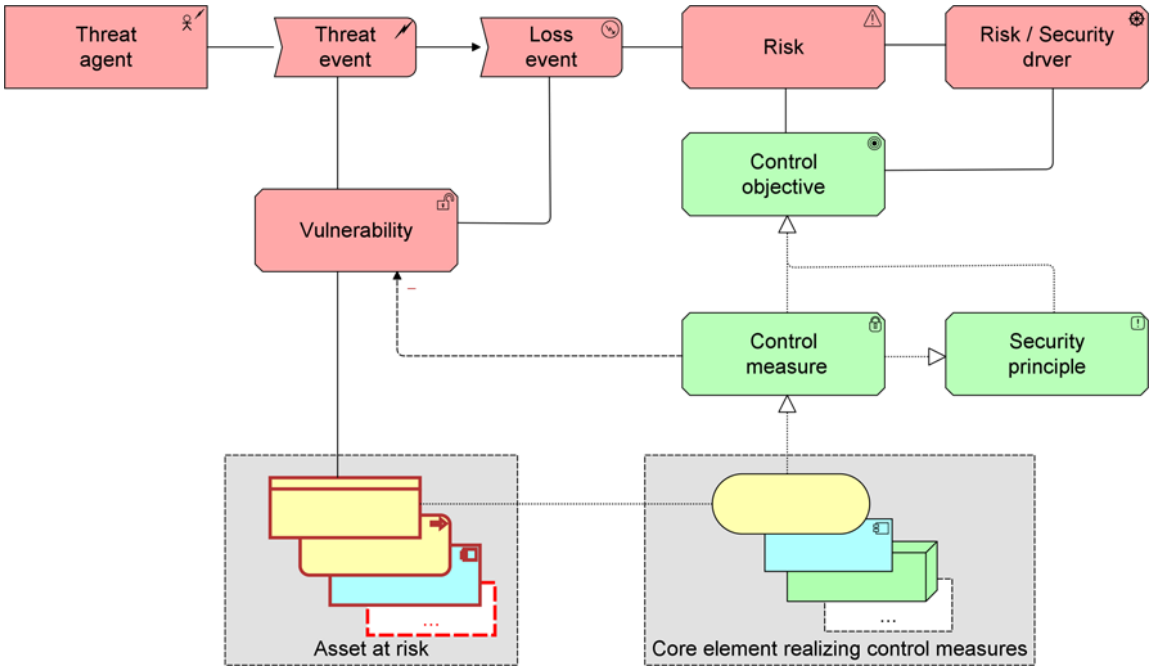


Figure 3.1.: ArchiMate risk and security overlay

Table 3.1.: Specialised ArchiMate element in the risk and security overlay

Element	Specialisation of	Definition	Attributes
Asset	Any core element	Anything tangible or intangible that is capable of being owned or controlled to produce value	Value
Vulnerability	Assessment	The probability that an asset will be unable to resist the actions of a threat agent	Vulnerability level
Threat agent	Business actor	Anything — for example, an object, substance, individual, or group — that is capable of acting against an asset in a manner that can result in harm	
Threat event	Business event	Event with the potential to adversely impact an asset	Threat event frequency; threat capability
Loss event	Business event	Any circumstance that causes a loss or damage to an asset	Loss event frequency; probable loss magnitude
Risk	Assessment	The probable frequency and probable magnitude of future loss	Risk level
Control objective	Goal	Aim or purpose of specified controls measures which address the risks that these control measures are intended to mitigate	Security profile
Security principle	Principle	A principle that guides the design or implementation of control measures	
Control measure	Requirement	An action, device, procedure, or technique that reduces a threat, a vulnerability, or an attack by eliminating or preventing it, by minimising the harm it can cause, or by discovering and reporting it so that corrective action can be taken	Control strength
Risk domain	Grouping	A domain consisting of entities that share one or more characteristics relevant to risk management or security	Security profile

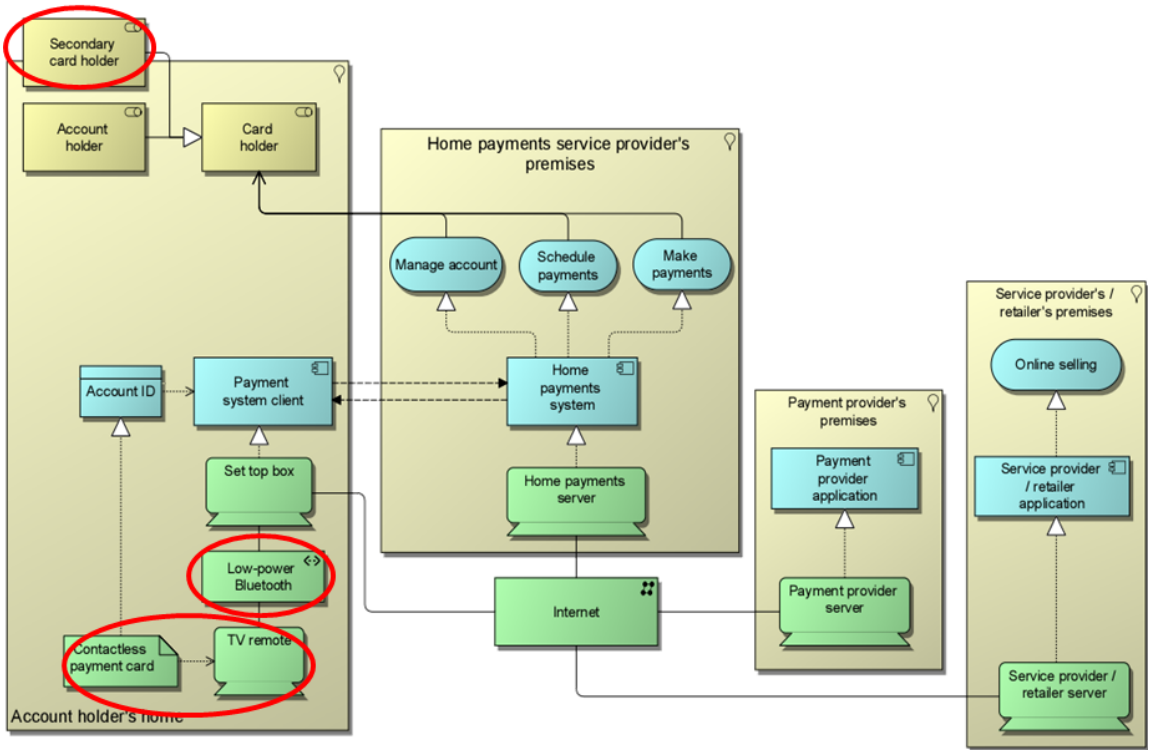


Figure 3.2.: IPTV infrastructure modelled in the ArchiMate language

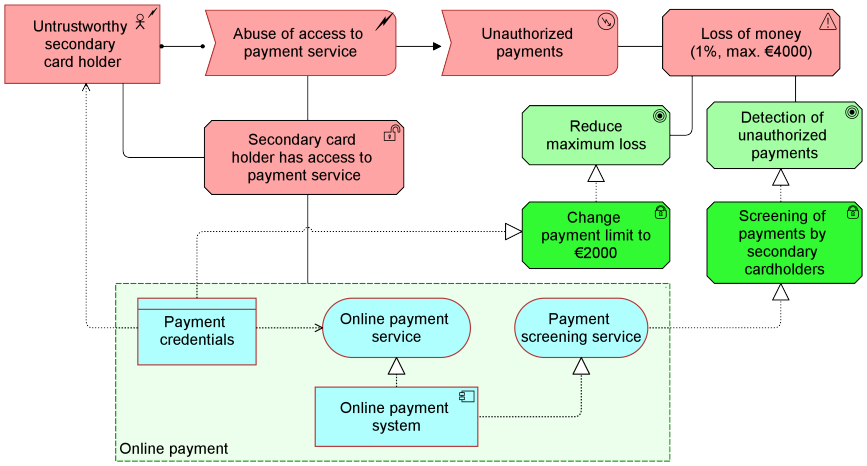


Figure 3.3.: Example ArchiMate risk analysis model for the IPTV case

3.2.2. Extended ArchiMate Modelling in the IPTV Case

The ArchiMate language, extended with the risk and security overlay as described in Section 3.2.1, has been applied in the IPTV case study. The IPTV infrastructure, describing the locations, technical infrastructure, application components and services, and users, is modelled using standard ArchiMate core elements (see Figure 3.2). This infrastructure model may serve a number of purposes:

- For modellers who already are familiar with ArchiMate, it can be used as a user-friendly "front-end" from which a TRE$_S$PASS socio-technical security model can be derived, either manually or (partly) automated.
- It can serve as a basis for vulnerability and risk assessment.

The red ellipses in Figure 3.2 indicate potential vulnerabilities in the IPTV setup, which may be of a technical nature (e.g., the bluetooth connection could be susceptible to eavesdropping), but also of a social nature: the secondary cardholder has access to the same payment functions as the account holder, which is a risk in case of an untrustworthy secondary card holder.

Figure 3.3 shows the details of a risk assessment of the latter vulnerability, a potentially untrustworthy secondary card holder that has access to the payment service. This model uses specialised ArchiMate elements from the risk and security overlay. A possible threat event based on this vulnerability is abuse of access to the payment services, by making unauthorised payments (e.g., to the personal account of the secondary card holder). The risk element provides a quantification of this, in terms of likelihood or frequency (abuse is estimated to occur in 1% of the cases) and probable loss magnitude (the payment service has a limit of €4000 per month).

This risk cannot be completely prevented (assuming that we want to keep the option of a secondary card holder), but a number of options are available to reduce the risk, which are also shown in the model. A first possible control objective is to detect the occurrence of unauthorised payments, so that they can be rolled back (and the payment card of the secondary card holder can be revoked); a control measure to achieve this is by adding a service that screens the payments of secondary card holders. A second possible control objective is a reduction of the maximum loss, which can be achieved by the control measure of reducing the monthly payment limit for secondary card holders.

3.2.3. Modelling and Analysing Fraud with e3fraud

This section describes e3fraud, an extension of the TRE$_S$PASS toolkit that uses business value models for the quantification of risks associated with e-service packages (such as telecommunication plans). The e3fraud models are based on the e3value framework, which models a network of end users and enterprises who exchange things of economic value with each other. While e3value is designed for mutually beneficial value models, we have extended it to accommodate the scenarios in questions (see Section 3.2.3.1) and provide meaningful output (see Section 3.2.3.1), respectively.

3.2.3.1. The e3fraud Approach

The e3fraud (Ionita, Wieringa, Wolos, Gordijn, & Pieters, 2015) methodology consists of two core elements: a modelling language and an analysis tool, briefly described below.

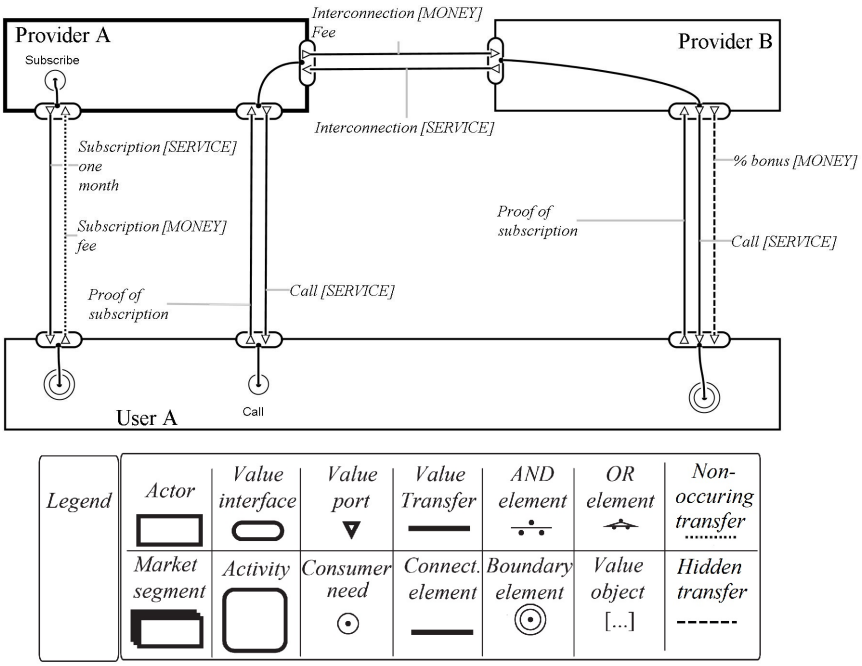


Figure 3.4.: Sub-ideal model: User A calls himself and earns money

The e3fraud Ontology: A normal e3value model assumes economic reciprocity, which means that value provisions from actor A to B are paired with reciprocal in such a way that in an ideal model, both actors are better off. In e3fraud, we extend this to allow the representation of sub-ideal value models, in which some transactions that are present in the ideal model may not occur, and other transactions that are absent in the ideal model, may occur. This also allows us to model collisions, where actors thought to be different in the ideal model, are identical in a sub-ideal model. In an e3fraud model, the business under attack is called the Target of Assessment (ToA). The ToA and the attacker are designated nodes. The ideal model essentially represents the ToA's view of the value constellation; the sub-ideal models are the attackers' views.

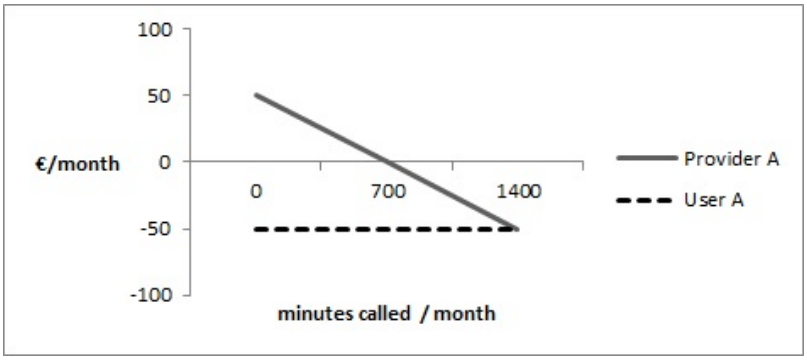
Figure 3.4 show an e3fraud model describing an instance of Revenue Sharing Fraud. Revenue Share Fraud (RSF) involves setting up a revenue sharing agreement with one provider, and a flat-rate (unlimited) subscription with another, and then calling yourself. This triggers the payment of interconnection fees from one provider to the other, thus resulting in a transfer of economic value between the providers. In this example, the ToA is Provider A and the malicious actor is User A. This end-user has a flat-rate monthly subscription with Provider A, which allows him/her to place a unlimited number of calls for free. However, user A *also* has access to a telephone hosted by provider B. The contract between user A and provider B states that for *received* call, user B gets part of the interconnection fee obtained by provider B in the form of a *bonus*. Again, since we take the point of view of Provider A, we have no information on how User A obtained a contract with provider B or what the structure of their agreement is. For the fraud analysis, it is sufficient to assume such a bonus is being paid. Furthermore, since the bonus pay-out

is hidden to Provider A (the ToA), it is represented using a *dashed* line. Note that user A only uses provider B to *receive* calls.

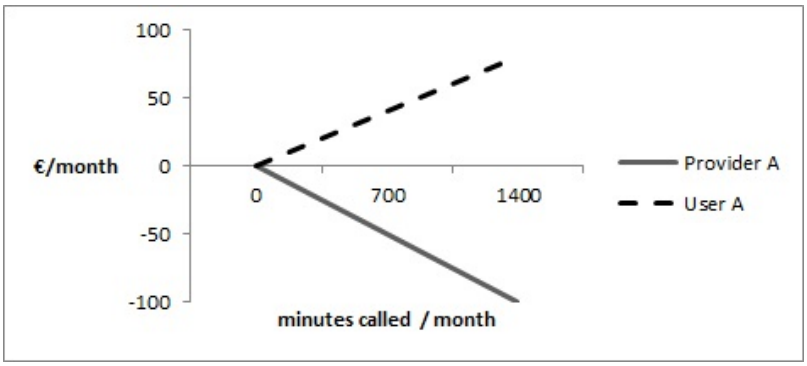
To make matters worse, in this sub-ideal scenario we assume User A does not intend to pay his monthly fee to Provider A. As it is a non-occurring transfer with respect to the ideal model of the ToA, the *Subscription Fee* value transfer is represented using a *dotted* line.

User A will now place as many calls as possible per month with provider B. As can be seen by following the dependency path, the *same* user A also terminates the call, but with his phone hosted by provider B. For each terminated call, user A receives a bonus. Considering that, in addition, he also intends to default on his payment of the Subscription fee, he is the position to make a generous profit.

The e3fraud Tool: The original e3value toolkit only allows static analysis of individual models. However, for the results to be useful in Risk Assessment, we need to view and compare two computations: the ideal case and one non-ideal case. Furthermore, in pay-per-usage environments, such as in telecommunication, the magnitude of the risk is dependent on the scale of usage. For example, in the case of toll fraud, both the damages



(a) Ideal case



(b) Sub-ideal case

Figure 3.5.: Example profitability graphs

incurred by the provider and the gain of a fraudster are dependent on the number of minutes called. As such, a tool was needed that could generate dual charts showing the dependency of the profitability with regard to usage, as well as the discrepancy between the ideal and sub-ideal models.

A Java tool was designed that takes as input e3fraud models, generates potential sub-ideal models based on the heuristics described in Section 3.2.3.1 and plots profitability graphs for both the ideal and non-ideal cases. The user may select which variable represents the horizontal axis and which range of values it can take. The tool also supports the ranking and grouping of sub-ideal models based on several criteria.

The two graphs in Figure 3.5 are an example of the type of analysis the software tools allow, based on 3.4. The financial outcome expected by the telecommunication provider, in normal usage conditions, is visible in Figure 3.5(a). Here, the user has a fixed cost, the monthly cost of the subscription. The costs of Provider A increase with each call, due to the termination bonus paid to Provider B for the interconnection. Operating costs are not represented here as they are unknown and assumed to be negligible for an individual user, but could be easily included in the model. The non-ideal case (Figure 3.5(b)) is significantly different. Besides the obvious loss for the provider, the fraudster's financial motivation is now clearly visible.

4. Conclusions

This deliverable documents the findings of the TRE$_S$PASS project on how to develop socio-technical security models with support for requirements intrinsic to the organisation that cannot be handled by the model yet. After a discussion of relevant aspects from our case studies, we have described several approaches to this challenge, partly through internal embeddings into the existing language, partly through extensions of the language, and partly through specialised tools.

Models can be extended in three ways:

- By translating new concepts into the existing models, as we have done for adding geolocation and routing in the ATM and the cloud case studies,
- By extending the language, as we have done when adding new actions to deal with, e.g., locking doors, or
- By developing new tools that interface to the TRE$_S$PASS process, as we have done with e3fraud for the telecommunications case study.

Clearly, the level of involvement increases dramatically from translating new concepts into the existing model over extending the language to developing new tools. While a translation is often fairly simple, a language extension requires changes to very likely all involved components of the risk assessment process, and developing a completely new tool even requires new interfaces to the existing tools and a meaningful interpretation of input to and output from the new tool. The drawback of a simple translation is that certain aspects of a system are difficult or impossible to model, or require an inordinate amount of work. Another drawback of simple translation is that other tools in the risk assessment process such as analysis and visualisation loose the high-level semantic information about what the low-level actions actually “mean”.

We suggest that an extension approach should follow these three steps:

1. Experiment with simple, straightforward translations to low-level instructions.
2. If the lack of support for concepts is manageable, then extend the language with constructs and add support to other tools in the risk assessment process.
3. If the lack of support for the new concepts is too big, then investigate the development of new tools.

This approach enables quick exploration of modelling possibilities and, once the potential of extensions has been identified, to improve the exploitation of the new elements in the risk assessment process.

References

Brantingham, P. L. (2010). Crime pattern theory. In S. P. L. Bonnie S. Fisher (Ed.), *Encyclopedia of victimology and crime prevention* (1st ed., p. 193-199). SAGE Publications, Inc. doi: http://dx.doi.org/10.4135/9781412979993

Clarke, R. V. G. (1997). *Situational crime prevention*. Criminal Justice Press Monsey, NY.

Cloud Security Alliance. (2010). *Top threats to cloud computing v1.0.* https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf.

Cohen, L. E., & Felson, M. (1979). Social change and crime rate trends: A routine activity approach. *American sociological review*, 588–608.

Egelman, S., Brush, A. B., & Inkpen, K. M. (2008). Family accounts: a new paradigm for user accounts within the home environment. In *Proceedings of the 2008 acm conference on computer supported cooperative work* (pp. 669–678). New York, NY, USA: ACM. doi: 10.1145/1460563.1460666

ENISA. (2009). *Cloud Computing Risk Assessment* (Tech. Rep.). Author.

Hay, B., Nance, K., & Bishop, M. (2011). Storm Clouds Rising: Security Challenges for IaaS Cloud Computing. In *Proceedings of the 2011 44th hawaii international conference on system sciences* (pp. 1–7). Washington, DC, USA: IEEE Computer Society.

Ionita, D., Wieringa, R., Wolos, L., Gordijn, J., & Pieters, W. (2015). Using value models for business risk analysis in e-service networks. In *The practice of enterprise modeling - 8th IFIP WG 8.1 working conference, poem 2015, valencia, spain, november 10-12, 2015. proceedings* (Vol. 235). Springer.

Josey, A., et al. (2014). *The open FAIR body of knowledge, a pocket guide: A taxonomy and method for risk analysis*. Van Haren Publishing.

Lenzini, G., Mauw, S., & Ouchani, S. (2015). Security analysis of socio-technical physical systems. *Computers and Electrical Engineering*. Retrieved from http://www.sciencedirect.com/science/article/pii/S0045790615000671 (Available online 6 April 2015)

Mell, P., & Grance, T. (2009a, October). *Effectively and Securely Using the Cloud Computing Paradigm.*

Mell, P., & Grance, T. (2009b, October). *The NIST Definition of Cloud Computing.*

Nidd, M., Ivanova, M. G., Probst, C. W., & Tanner, A. (2015). Tool-based risk assessment of cloud infrastructures as socio-technical systems. In R. Ko & K.-K. R. Choo (Eds.), *The cloud security ecosystem.* Syngress.

O'Leary, M. (2011). Modeling criminal distance decay. *Cityscape*, 161–198.

Pieters, W. (2011). Security and privacy in the clouds: a bird's eye view [Technical Report]. In S. Gutwirth, Y. Poullet, P. De Hert, & R. Leenes (Eds.), *Computers, privacy*

and data protection: an element of choice (pp. 445–457). Dordrecht: Springer. http://eprints.eemcs.utwente.nl/19837/.

Probst, C. W., & Hansen, R. R. (2008). An extensible analysable system model. *Information security technical report*, *13*(4), 235–246.

The Open Group. (2013). *ArchiMate® 2.1 specification.* Van Haren Publishing.

The TRE$_S$PASS Project, D1.2.2. (2015). *Final policy-specification language.* (Deliverable D1.2.2)

The TRE$_S$PASS Project, D1.3.1. (2013). *Initial prototype of the socio-technical security model.* (Deliverable D1.3.1)

The TRE$_S$PASS Project, D1.3.3. (2015). *Dynamic features of socio-technical security models.* (Deliverable D1.3.3)

The TRE$_S$PASS Project, D7.3.1. (2014). *Results from case study b.* (Deliverable D7.3.1)

A. Project Summary

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

Information security threats to organisations have changed completely over the last decade, due to the complexity and dynamic nature of infrastructures and attacks. Attacks like StuxNet involve technical and human factors, and they damage physical infrastructure. The recent attack on a German steel mill [1] was a combination of both targeted phishing emails and social engineering attacks. The phishing helped the hackers extract information they used to gain access to the plant's office network and then its production systems. As a result, the technical infrastructure of the mill suffered severe damage.

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

The main innovation of the 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.

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

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.

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

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.

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.

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

A.1. Case Studies

The TRE$_S$PASS 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 TRE$_S$PASS we identify social-engineering and trust-based attacks on such systems.

A.2. Overview of TRESPASS Integration

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

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

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

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

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

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

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

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

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

- **Components customisation (optional)** takes place before or during the TRESPASS model creation to create specialised custom model components.
- **Attacker profile creation** creates the attacker profile that the TRESPASS model analysis should consider, based on ready-made attacker profiles.
- **Defender/target profile creation** creates similar profiles for the other actors in the model based on the social data gathered in the social data collection activity.

e3value model creation This interactive activity involves using the *e3value toolkit* [2] to create business value models. These models structure the commercial information gathered in the data collection stage in a formal way.

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

1. In the **attacker profile selection**, the user selects the attacker profile to use in the analysis.

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

2. 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.
3. The **scenario selection** selects a scenario, consisting of a single pair of attacker and attacker goal, to run the TRESPASS analysis on.
4. 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.
5. **Attack generation** transforms the TRESPASS model to an attack tree.
6. **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.
7. The **attack tree analyses** compute quantitative properties of attacks, *e.g.*, utility for the attacker or success probability of the attack.

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

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

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

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

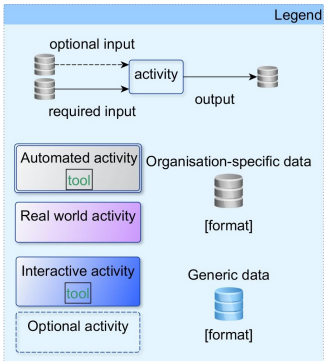


Figure A.1.: Legend for the Integration diagram in Figure A.2.

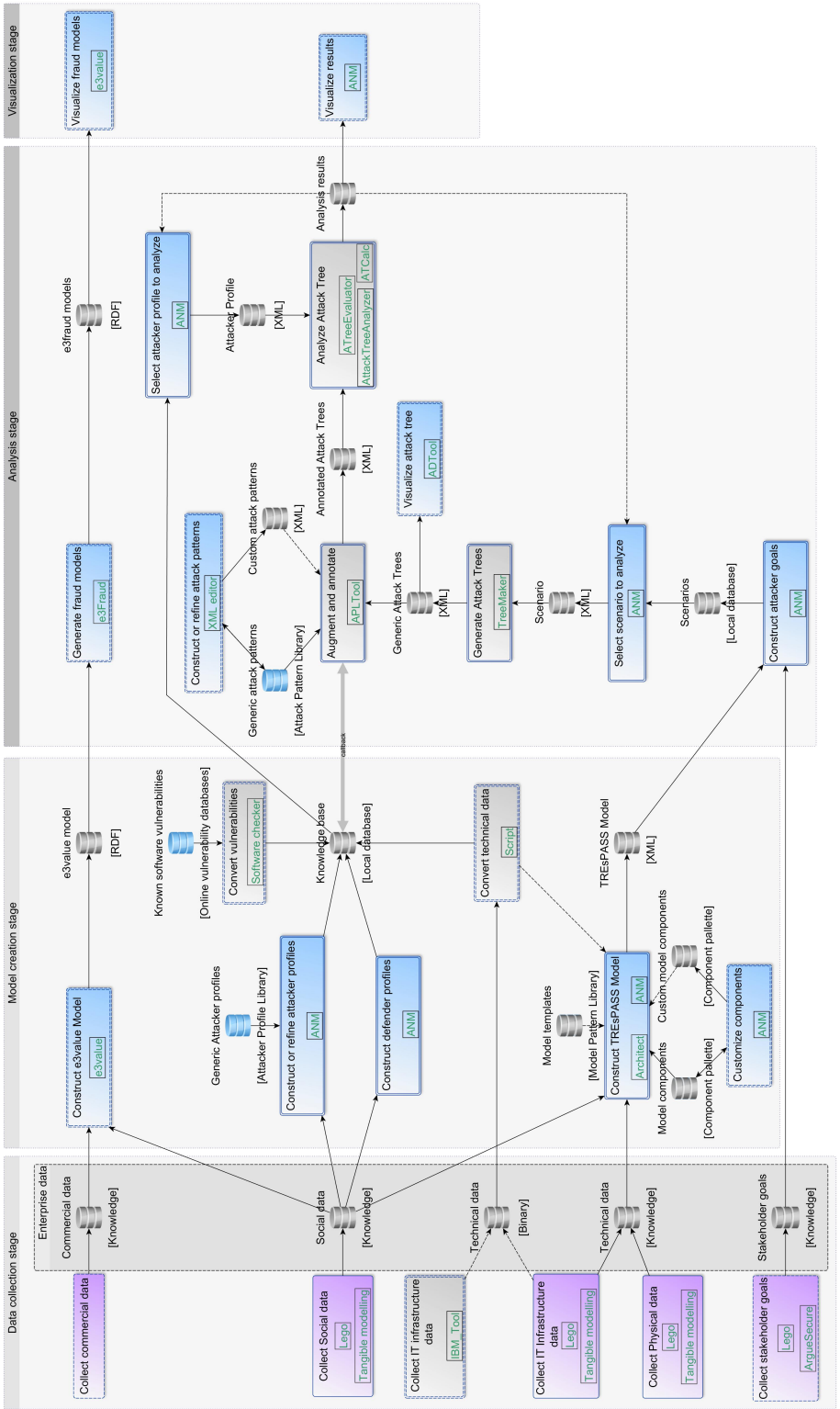


Figure A.2.: Integration diagram for the TRESPASS project.