A Model Transformation from Misuse Cases to Secure Tropos

Naved Ahmed[1], Raimundas Matulevičius[1], and Haralambos Mouratidis[2]

[1] Institute of Computer Science, University of Tartu, Estonia
{naved,rma}@ut.ee
[2] School of Computing and Technology, University of East London, UK
h.mouratidis@uel.ac.uk

Abstract. In current practices security concerns are typically addressed at the design or implementation stages, leaving aside the rationale for security analysis. The reason is that a systematic approach to address security from late development stages to early analysis stages does not exist. This paper presents transformation rules to perform model translation from misuse case diagram to Secure Tropos model. The translation justifies the system security concerns, and keep the traceability of the security decisions. Our proposal is based on the systematic domain model for information systems security risk management (ISSRM); thus, it preserves the semantics of both security languages' constructs and synchronise the mechanisms across language boundaries to elicit, correct and complete security requirements. An example from banking sector demonstrates the applicability of our proposal.

Keywords: Information System (IS), Requirements engineering, Secure Tropos, Misuse cases, Model transformation.

1 Introduction

It is recognised that blemishes in requirements, on one hand, cost 10 to 200 times more once handled [3], and glitches in early requirements analysis stages outcomes a high percentage of system failures [12]. On another hand current practice starts develop security only after the system design or implementation is done [6]. However, this might lead to a gap between requirement analysis and the actual implementation. Although security modelling languages are used at different stages of the system development, they still lack dedicated constructs to identify the security concerns [7, 8], such as vulnerabilities, risks and their countermeasures. There exists little effort to integrate different security modelling languages into the coherent modelling approach so that developers could benefit from various modelling viewpoints along different system development stages. Such integration could also contribute to security traceability across the development cycle, thus, also keeping the rationale for the security decisions.

In this paper we introduces a set of transformation rules to translate misuse case diagrams [11] to Secure Tropos models [10]. This is a continuation of our

previous effort [1], where we reported on the opposite transformation from Secure Tropos to misuse case diagrams. Both these model translations are based on the language semantic alignment [7, 8] to the domain model [9] of the Information Systems Security Risk Management (ISSRM). Since the major question of the goal modelling languages, like Secure Tropos, are to understand why certain system is build, in this paper we focus on capturing the security decision rationale from the misuse case models and representing it using Secure Tropos.

The structure of the paper is organised as follows: in Section 2 we give the background knowledge of security languages and introduce their alignment to the ISSRM domain model. In Section 3 we introduce the transformation rules to translate misuse cases to Secure Tropos. We illustrate our proposal through an online banking example [6]. In Section 4, we discuss benefits, completeness and limitations. Finally, we conclude our study in Section 5.

2 Background

2.1 ISSRM Domain Model

The ISSRM domain model [9] is used to align the security languages. It provides a systematic guidance for security risk analysis and supports modelling, assessing and treating risks on the basis of the likelihood and severity of failures as Tropos Goal-Risk framework [2]. The ISSRM domain model [9] (see Fig. 1) is inspired by, and compliant with the existing security standards (see details in [9]). Additionally as compared to Tropos Goal-Risk framework, ISSRM supports the definition of security for the key IS constituents and addresses the IS security risk management process at three different conceptual levels, i.e., asset-related, risk-related, and risk treatment-related concepts (described later). This gives details about the IS which is abstractly defined in a 3-layer architecture of Tropos Goal-Risk framework and helps to quantitatively measure the risk its likelihood, impact and cost of implementing security controls with respect to asset's value.

***i*) Assets-related concepts** describe the organisation's assets classified as *business* and *IS assets* along with the *security criteria* for business assets expressed in terms of confidentiality, integrity and availability.

***ii*) Risk-related concepts** define *risk*, composed of a threat with one or more vulnerabilities. An *impact* is the consequences of an event that negates the security criterion. An *event* is an aggregation of threat and one or more vulnerabilities. A *vulnerability* is the characteristics of IS assets that expose weakness or flaw. A *threat* is an incident initiated by a threat agent to target one or more IS assets. A *threat agent* is an agent who has means to harm IS assets intentionally. An *attack method* is a standard means to execute threat.

***iii*) Risk-treatment related concepts** describe a *decision* (e.g., avoidance, reduction, retention, or transfer) to treat the risk and *security requirement* is its refinement. A *control* is the implementation of requirements.

The ISSRM application follows the general risk management process based on the security standards (see details in [9]). Firstly, define *organisational context and identify assets*. Then, determine *security objectives for assets*.

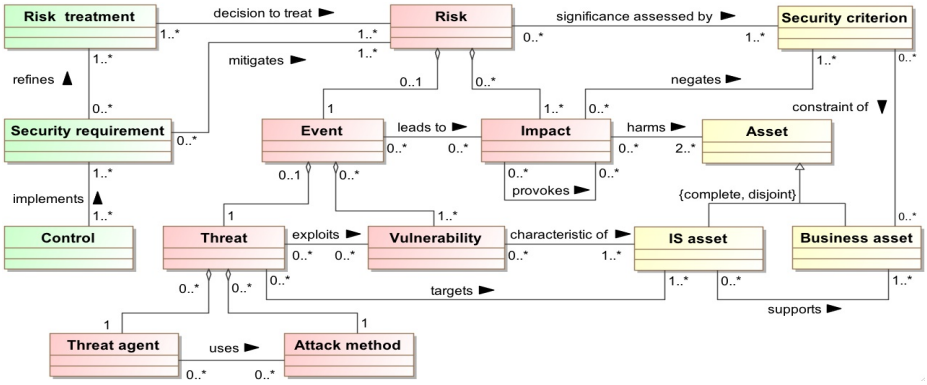

Fig. 1. ISSRM Domain Model, adapted from [9]

Next, *risk analysis and assessment* to identify potential risks and their impacts. Then, *risk treatment decisions* are taken resulting in *security requirements*. Finally, security requirements are implemented into *security controls*. This process is iterative, because new security controls might originate new security risks.

2.2 Misuse Cases

Misuse cases [11] are a security-oriented extension of the Use cases. Misuse case diagrams are extended with misuser, misuse case, and security use cases constructs including threatens and mitigates relationships (see Fig. 2). A *misuser* intends to harm the software system. A *misuse case* is a goal of misuser, the association is represented by a communication association. Misuser executes misuse case either by combine efforts of several misuse cases, or independently. *Threatens* relationship means a misuse case is potentially a threat to the use case. *Mitigates* relationship indicates that a use case is countermeasure against misuse case. *Security use case* performs countermeasure against the identified threat.

As illustrated in Fig. 2 misuse cases are integrated in use case diagrams to express the system unwanted behaviour (e.g., misuse cases `Money stolen`, `Enter pin code result repeatedly`, and `Transfer money to own account`) initiated by a misuser (e.g., `Attacker`). This depiction results in security use cases e.g., `Perform cryptographic procedures`.

2.3 Secure Tropos

Secure Tropos [10] is an extension of Tropos [4]. It enriches Tropos by introducing security related constructs (see Fig. 3). In Tropos, an *actor* (e.g., `Customer`, `Bank officer` and `Banking IS`) is an entity that has strategic goals and interests within the system. A *goal* (e.g., `Transaction be performed`, `Account privacy guaranteed`) is an actor's strategic interest. A *plan* (e.g., `Perform transaction`, `Keep data up to date`) represents means to satisfy actors' goals.

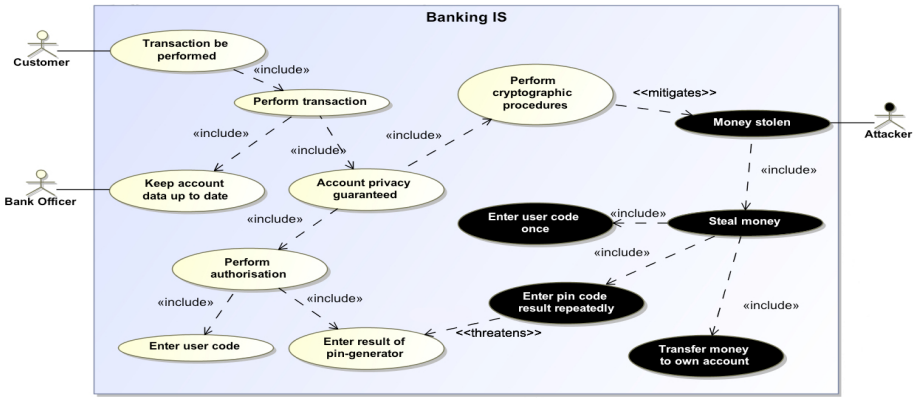

Fig. 2. Misuse Case Diagram

A *resource* (e.g., `Account`) is an entity required by actors. In Secure Tropos, *security constraint* (e.g., `Only by bank customer` and `Only by bank officer`) is a constraint that the system must possess. A *threat* (e.g., `Money stolen`) represents an event that endangers the security features of system. Additionally, vulnerability point is represented by a black circle in Fig.3 (adapted from [5]).

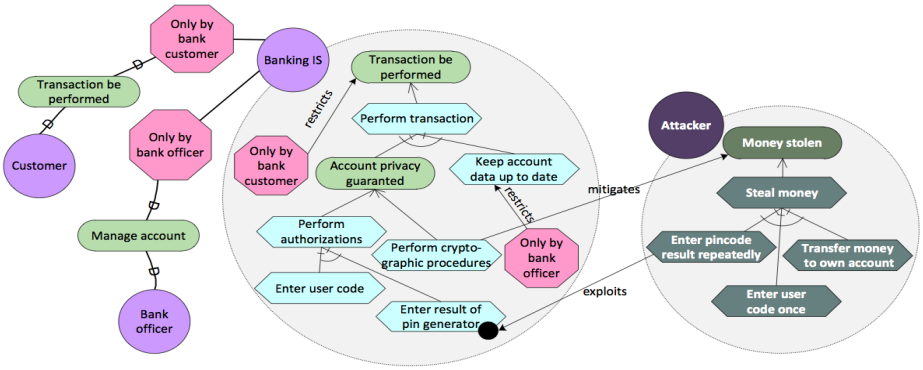

Fig. 3. Secure Tropos Diagram

Secure Tropos uses relationships to connect constructs. *Dependency link* shows that one actor (depender) depends on another actor (dependee) to attain some dependum (e.g., goal, plan or resource). A *secure dependency* is restricted by the security constraint that must be respected by both actors (e.g., relationship between `Customer` and `Banking IS`). A *means-end link* indicates how the goal (end) is satisfied. A *decomposition relationship* represents a breakdown of plan into several plans or goals. *Restricts* and *attacks relationships* are intro-

duced in Secure Tropos where former shows a security constraint restriction on a goal achievement and prior indicates the target of attacker's plan.

Tropos methodology covers the overall IS development, however we limit our scope to the goal and security attack scenario modelling (which correspond to the Tropos late requirements stage [4]).

2.4 Alignment of ISSRM and Security Modelling Languages

As discussed in [9] the ISSRM domain model guides the application of the security modelling languages with respect to the security risks analysis. The detailed alignment of ISSRM domain model with Secure Tropos and Misuse Cases is provided in [8] and [7], and summarised in Table 1 (column **1** & **2**).

Table 1. Alignment of ISSRM Concepts with Modelling Languages Constructs

ISSRM Model Concepts			**Secure Tropos Constructs**	**Misuse Case Constructs**
0			**1**	**2**
Asset related concepts	**a**	Asset	Actor, goal, plan, resource	Actor and use case
	b	Business asset		
	c	IS asset		System
	d	Security Criteria	Security constraint	–
Risk related concepts	**e**	Risk	–	–
	f	Impact	Contribution between threat and other construct	–
	g	Event	Threat	–
	h	Threat	Goal, plan	Misuser & Misuse case
	i	Vulnerability	Vulnerability point	–
	j	Threat agent	Actor	Misuser
	k	Attack method	Plan, attacks relationship	Misuse case
Risk-treatment related concepts	**l**	Risk treatment	–	–
	m	Security requirement	Actor, goal, plan, resource, security constraint	Security use case
	n	Control	–	–

3 Transformation Rules

3.1 Transformation from Misuse Cases to Secure Tropos

This section introduces a set of rules for translating Misuse cases to Secure Tropos model. They are based on ISSRM model and its application process.

Asset-related concepts are translated using following transformation rules:

TMS1. *A system boundary that presents software system in the misuse case diagram is translated to the Secure Tropos actor.*

This rule is based on alignment between the Secure Tropos *actor* and misuse case *system boundary* to the ISSRM *IS asset* as introduced in Table 1 (line ***c***). In Fig.3 we present a Secure Tropos actor `Banking IS` with its boundary.

TMS2. *A use case is translated either to Secure Tropos goal or plan belonging to the boundary of the system actor. Correspondingly, an includes link is translated either to means-ends relationship (where ends is the goal and means is the*

plan) or to decomposition relationship (where some plan is decomposed).
Note: we assume $OR \Rightarrow$*means-ends*, and $AND \Rightarrow$*decomposition* in Secure Tropos model.

It is defined according to the lines ***a*** and ***b***. Here the developer decides whether a use case is translated to Secure Tropos *goal* or *plan*. In Fig. 3, we translate the use case `Transaction be performed` to goal meaning that the use case `Perform transaction` should be *plan*, because only a plan could be means to achieve the goal (ends) in Secure Tropos. On the other hand, the use case `Account privacy guaranteed` is translated to a goal. Here we also define two plans `Perform authorisation` and `Perform cryptographic procedures` that are the means to achieve this goal. We illustrate the *OR relationship* to specify two alternates to achieve the goal `Account privacy guaranteed`.

In Fig. 2, two actors (e.g., `Customer` and `Bank officer`) communicate to the `Banking IS`. Based on the Table 1 lines ***a*** and ***b*** we translate these actors to the Secure Tropos actors in Fig. 3 by introducing the following rule:

T_{MS}3. *An actor from the misuse case is translated to a Secure Tropos actor.*

An interaction of actor with system presents how actors collaborate to achieve their goals. In misuse cases it is defined by *communication links* while Secure Tropos uses *dependency links*. A communication link would be translated using either of the three following cases:

T_{MS}4. *(i) If the system is dependee, then the communication link is translated as depender and the use case to which the misuse case actor communicates is defined as dependum (according to $T_{MS}3$) in the Secure Tropos dependency;*
(ii) If the system is depender, then the communication link is translated as dependee and the developer specify the dependum manually, since it is not possible to capture it from the misuse case diagram;
(iii) A security constraint could be defined to restrict the goal/plan (as well as the dependum). The restricted goal/plan is translated from the use case, to which the actor communicates in the misuse case diagram.

Following $T_{MS}4$, the *communication links* (see Fig. 2) between actors `Bank officer` and `Customer` with `Banking IS` are translated to the dependency links (see Fig. 3). However, it is not possible to capture security constraints (Table 1, col ***2***, line ***d***). Although, we defined them manually (e.g., `Only by bank customer` and `Only by authorised bank officer`) by identifying the elements that needs to be restricted e.g. dependum goal `Manage account` in Fig. 3.

Translating ***risk-related concepts***, generate the Secure Tropos attack scenario (see Fig. 3) using the following transformation rules:

T_{MS}5. *A misuser is translated to Secure Tropos actor. In the discussion below we recall this actor as a threat agent.*

It is based on line ***j*** in Table 1, which identifies that the *misuser* and the Secure Tropos *actor* are aligned to the ISSRM *threat agent*. Thus in Fig. 3 we identify a threat agent as *Attacker*.

T_{MS}6. *A misuse case is translated to the plan of threat agent. Using $T_{MS}2$, an includes link is translated to the Secure Tropos decomposition relationship.*

In Table 1, this rule refers to lines ***h*** and ***k***, according to which ISSRM *threat* and *attack method* are presented as *misuse case* and *plan* (and *goal*). Therefore,

`Money stolen`, `Steal money`, `Enter pin code repeatedly`, `Enter user code once` and `Transfer money to own account` are translated to plan constructs in Secure Tropos model (see Fig. 3). To simplify the translation misuse cases are transformed to only Secure Tropos goals.

T_{MS}7. *A threatens relationship is translated to the Secure Tropos exploits link. The exploits link is pointed to the vulnerability point, which needs to be added to the appropriate Secure Tropos construct.*

In the example, *threatens relationship* is translated to Secure Tropos *exploits link* from the threat agent's plan `Enter pin code repeatedly` to the vulnerability point identified in the `Enter result of pin generator` (see Fig. 3). Secure Tropos *threat agent* and his plans; correspond to the combination of the ISSRM *threat agent*, *attack method* and *threat*. Following Table 1 define a Secure Tropos *threat* (aligned to the ISSRM *event*) as a generalisation of the Secure Tropos threat agent and its boundary. For example, `Money stolen`.

Translating ***risk treatment-related concepts***, a security use case `Perform cryptographic procedures` is already translated to the Secure Tropos plan (see Fig. 3) as discussed in rule $T_{MS}2$. Now we introduce that:

T_{MS}8. *A mitigates relationship from the misuse case diagram is translated to the mitigates link in Secure Tropos.*

In the ISSRM domain model (Fig. 1) the *mitigates relationship* indicates the mitigation of potential risk event by introducing appropriate security requirements. The security use case `Perform cryptographic procedures` mitigates the threat `Money Stolen`, thus it is translated to the Secure Tropos mitigates to reduce the risk event `Money stolen`.

4 Discussion

Semi-automated Transformation: The transformation rules could support a semi-automatic model translation. When translating the models, the developer needs to indicate if the *(mis)use cases* need to be translated to the *goal* or *plan* (see rule $T_{MS}2$). It influences the translation of *includes relationship* either to *means-ends* or *decomposition*. Also in $T_{MS}4$, the developer indicates whether the Secure Tropos actor (translated from the misuse case *software boundary*) plays the role of *dependee* or *depender* in the translated *dependency link(s)*. Additionally, the developer defines the labels for *dependum* and *security constraint(s)* (as illustrated in Fig. 3). The remaining rules could be applied automatically.

Transformation Completeness: The transformation does not contribute with complete model in the target language but helps developers to concentrate on the details, which give the added value for the target model. The transformation highlights the major overlapping semantic areas of two security languages. The translated Secure Tropos model give reason for the system security.

Transformations and Misuse Case Textual Template: Matulevičius *et al.* [7] have aligned misuse case textual template and ISSRM domain model. Although we do not have enough space to discuss the template translation. We acknowledge that the template would complement and strengthen the transformation.

5 Conclusion

In this paper we tackled to eradicate the gap between the functional (software) system requirements and their relation to early security requirement analysis. We define a set of transformation rules from misuse case diagrams to Secure Tropos models. The transformation highlights and preserves the security-related semantics. The resulted model helps understanding the environment and gives reasoning on the benefits and trade-offs of the security decisions taken. Therefore, it benefits the overall model maintainability management between the two different presentations of security problem. In the example we have illustrated the applicability of our proposal, we acknowledge the importance of the industrial case study to validate the rules. The translation can be applied to existing or legacy systems to find the missing rationale for implemented security primitives and can provide alternate security solutions to solve the problem.

We agree to the importance of validating the current work and as a future work we encourage to empirically validate the translation through perception, performance and correctness tests. Furthermore, we plan to expand the scope by introducing a semi-automated transformation rules for other security languages. Such approach would result in a systematic model-driven security engineering, which would facilitate systematic security definition from the early requirements to system design and implementation.

References

1. Ahmed, N., Matulevičius, R.: Towards Transformation Guidelines from Secure Tropos to Misuse Cases. pp. 36–42. SESS'11, ACM (2011)
2. Asnar, Y., Giorgini, P., Massacci, F., Zannone, N.: From Trust to Dependability through Risk Analysis. In: Proceedings of ARES. pp. 19–26. IEEE (2007)
3. Boehm, B.W., Papaccio, P.N.: Understanding and Controlling Software Costs. IEEE Trans. Software Eng. 14(10), 1462–1477 (1988)
4. Castro, J., Kolp, M., Mylopoulos, J.: Towards Requirements-driven Information Systems Engineering: The Tropos Project. Inf. Syst. 27(6), 365–389 (2002)
5. Elahi, G., Yu, E.S.K.: A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs. In: Conceptual Modeling - ER. pp. 375–390. Springer (2007)
6. van Lamsweerde, A.: Elaborating Security Requirements by Construction of Intentional Anti-Models. In: ICSE 2004, UK. pp. 148–157. IEEE (2004)
7. Matulevičius, R., Mayer, N., Heymans, P.: Alignment of Misuse Cases with Security Risk Management. In: Proceedings of ARES. pp. 1397–1404. IEEE (2008)
8. Matulevičius, R., Mayer, N., Mouratidis, H., Dubois, E., Heymans, P., Genon, N.: Adapting Secure Tropos for Security Risk Management in the Early Phases of Information Systems Development. In: CAiSE, Proc. pp. 541–555. Springer (2008)
9. Mayer, N.: Model-based Management of Information System Security Risk. Ph.D. thesis (2009)
10. Mouratidis, H., Giorgini, P.: Secure Tropos: A Security-oriented Extension of the Tropos Methodology. International Journal of SEKE 17(2), 285–309 (2007)
11. Sindre, G., Opdahl, A.L.: Eliciting Security Requirements with Misuse Cases. Requir. Eng. 10(1), 34–44 (2005)
12. Sommerville, I.: Software Engineering. Addison Wesley, 6th edn. (August 2000)