Towards Transformation Guidelines from Secure Tropos to Misuse Cases (Position Paper)

Naved Ahmed
Institute of Computer Science
University of Tartu
J. Liivi 2, 50409 Tartu, Estonia
naved@ut.ee

Raimundas Matulevičius
Institute of Computer Science
University of Tartu
J. Liivi 2, 50409 Tartu, Estonia
rma@ut.ee

ABSTRACT
Progressive increase in developing secure information systems (IS) requires that the security concerns should be properly articulated well ahead in early requirement engineering (RE) along with other functional and non-functional requirements. In this paper, based on the domain model for IS security risk management (SRM) we propose a set of transformation guidelines to translate Secure Tropos models to the misuse case diagrams. We believe that such a model translation would help developers to elicit real security needs by integrating the security analysis starting from early requirement stages to all the stages of development process. The translation aligns the IS security concerns with functional requirements and maintains traceability of the security decisions to their origin.

Categories and Subject Descriptors
D.2.1 [Requirements/Specification]: Languages, Methodologies

General Terms
Management, Design, Security, Languages.

Keywords
Information systems; Requirements engineering; Secure Tropos; Misuse cases; Model transformation.

1. INTRODUCTION
Nowadays, information systems (ISs) are of core importance in any business infrastructure, which is complicatedly integrated by involving many stakeholders. This broadens the IS scope and increase the access levels for resources, objectives and tasks. This raises the importance of IS security and turns it into necessity rather than a choice. Typically, security analysis should be performed throughout the whole system development cycle starting from the early stages (a.k.a., requirements engineering (RE) and system design) and leading to the late stages (e.g., implementation and testing). However this is not the case in practice [11] [22] where security is considered only when the system is about to be implemented or deployed. ISs perform different business needs and activities of an organisation and operate closely with its environment; therefore, assessing IS security separately from functional requirements creates a gap between requirement analysis and its actual implementation which usually does not reflects the real security needs. Thus, to eradicate this gap and to align the security and functional requirements analysis with the IS environment the model transformation is an important step. This transformation would benefit developers (*i*) to understand threats, their consequences and countermeasures, (*ii*) to save the iteration cost by discarding design alternatives, and (*iii*) to avoid IS to become complicated by implementing pre-defined unrealistic security features especially when some of its functionality is not really required.

Another problem is that the security modelling languages address security concerns specifically within the certain stages of IS development and due to lack of this linkage a formal methodology is needed to utilized a model generated in one language into another language. For example, Tropos [1] [2] and its extension Secure Tropos [17] [18] [19] could support analysis of the early and late IS concerns along with the system environment. Use cases [20] and their extension misuse cases [22] focus on elicitation of the software system functionality and here lies a gap between these two modelling languages. Furthermore, addressing different security concerns at various development stages, there exists only limited guidelines [13] [23] on how to transform a security model created in one language to a security model of another language, thus leading to the systematic IS development addressed through different viewpoints.

In this paper we propose a set of transformation rules to translate modelling notations from Secure Tropos to misuse cases diagram. Our proposal is based on the language alignment to the IS security risk management (ISSRM) domain model [4] [16]. Such a transformation could ensure that developers would benefit from both modelling languages; i.e., to apply Secure Tropos in order to answer *why* different security decisions are made, and to use misuse cases to specify *how* security concerns should be specified with respect to the system functionality.

The paper structure is as follows: in Section 2 we give the study background. In Section 3 we propose a set of rules to translate a Secure Tropos model to misuse case diagram. Section 4 discusses our proposal, and Section 5 concludes our study.

SESS'11, May 22, 2011, Waikiki, Honolulu, HI, USA

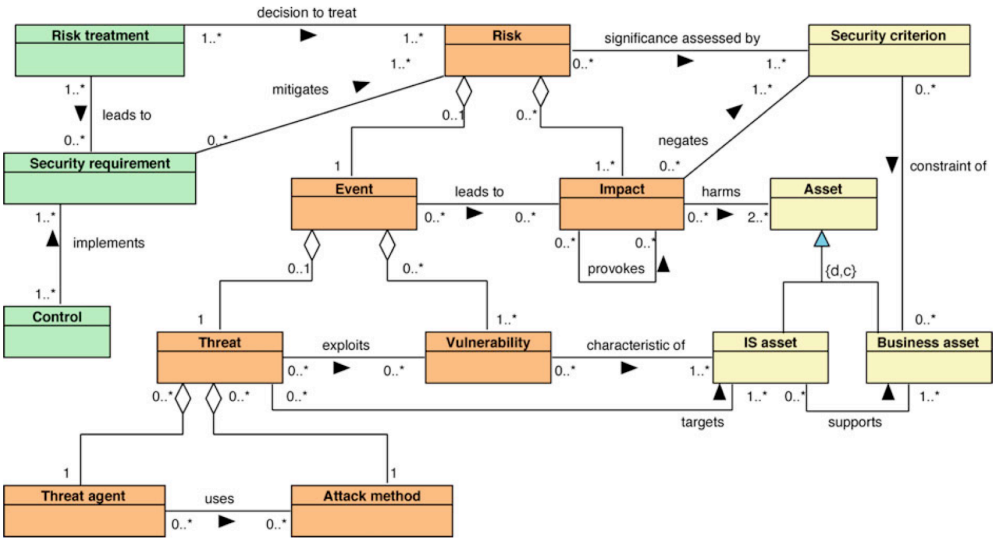

Figure 1. ISSRM Domain Model (adapted from [4] [16])

2. BACKGROUND

2.1 ISSRM Domain Model

The ISSRM domain model [4] [16] (see Figure 1) is inspired by, and compliant with the existing security standards, e.g., [3] [6] [10]. It supports 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*.

Assets-related concepts describe organisation's assets and their security criteria. Here, *asset* is anything that is valuable and that plays a vital role to accomplish organisation's objectives. A *business asset* describes the information, processes, capabilities and skills essential to the business and its core mission. An *IS asset* is the IS component, valuable to the organisation since it supports business assets. A *security criterion* is the property or constraint on business assets describing their security needs, which are, typically, expressed through *confidentiality*, *integrity* and *availability* of business assets.

Risk-related concepts introduce a *risk* definition. A *risk* is composed of a *threat* with one or more *vulnerabilities* that leads to a negative *impact* on one or more assets by harming them. An *impact* is the consequences of an *event* that negates the security criterion defined for business assets in order to harm assets. An *event* is an aggregation of *threat* and one or more *vulnerabilities.* A *vulnerability* is the characteristics of the IS assets which expose weakness or flaw. A *threat* is an incident initiated by a *threat agent* using *attack method* to target one or more IS assets by exploiting their *vulnerabilities*. A *threat agent* is an agent who has means to harm intentionally IS assets. An *attack method* is a standard means by which a threat agent executes threat.

Risk-treatment related concepts describe the concepts to treat risk. A *risk treatment* is a decision (e.g., avoidance, reduction, retention, or transfer) to treat the identified risk. A *security requirement* is the refinement of a risk treatment decision to mitigate the risks. A *control* designates a means to improve the security by implementing the security requirements.

The ISSRM application follows the general risk management process that is also based on the existing security standards, e.g. [3] [6] [10]. It is an iterative process consisting of six steps. Firstly a developer needs to define the *organisational context* and *assets* that needs to be secured. Then, one *determines security objectives* (e.g., confidentiality, integrity, and availability) based on the level of protection required for the identified assets. Next, *risk analysis and assessment* help identify potential risks and their impacts. Once risk assessment is performed *risk treatment decision* should be taken. This would result in *security requirements definition*. Finally, security requirements are implemented into the *security controls*. The risk management process is *iterative*, because new security controls might also open the possibility for new (not yet determined) security risks. In Section 3 we will implicitly apply this security management process to develop the rules for the security language translation.

2.2 Secure Tropos

Tropos is an agent-oriented software development methodology [1] [2] that helps developing the IS by considering its major social actors, defining their dependencies, understanding actor major goals, capabilities and intentions, and satisfying them through the newly defined software system. Using Tropos the IS is systematically developed from the early and late requirements to the architectural and detailed design. In our work we will specifically focus on the two requirements stages, which help to understand *why* the IS is developed.

Secure Tropos [17] [18] [19] enriches a set of Tropos constructs (*actor*, *goal*, *softgoal*, and *resource*) by introducing security related constructs such *as security constraint* and *threat*. An *actor* is an entity that has strategic goals and interests within the system or within the organisational setting. A (hard) *goal* represents an actor's strategic interest. A *softgoal*, unlike goal, does not have a clear criterion to measure its satisfaction; therefore it is subjected to interpretation (i.e. satisficed). A *plan* represents a way of doing something to satisfy actors' goals. A *resource* is a physical or informational entity required by actors. These constructs are common both in Tropos and Secure Tropos. In addition in Secure Tropos concepts of *security constraint* and *threat* are defined. A *security constraint* is a security restriction that the system must possess. A *threat* represents circumstances, which lead to an event

that endangers the security features of system. In addition to these concepts in our work we will use the notion of *vulnerability point* (any system's weakness) introduced by Elahi and Yu in [5]. The illustration of major concepts is provided in Figure 2, 4, and 6.

Constructs are connected using relationships: *dependency* (and its subtype of *secure dependency*), *decomposition*, *means-ends*, *contribution*, *restricts* and *attacks*. Two actors are connected by *dependency link*, which shows that one actor (*depender*) depends on another actor (*dependee*) to attain some *dependum* (e.g., goal, softgoal, plan or resource). A *secure dependency* is a dependency restricted by the security constraint that must be respected by both actors for the dependency to be satisfied (see Figure 2).

Once the dependencies between actors are identified a goal model helps a deeper understanding of how the actors reason about their goals, tasks and resources. A *means-end* link is a relationship between a *means* (plan/goal/ resource) and *end* (goal) indicates how the goal (ends) is satisfied. A *decomposition* relationship represents a breakdown of plan into several plans or goals. A *contribution* relationship shows the impact of one construct on another (typically a softgoal); this impact will either be positive or negative. *Restricts* and *attacks* relationships are introduced in Secure Tropos where former shows how a goal achievement is restricted by security constraint and prior indicates the target of attacker's plan. These relationships are illustrated in Figures 2, 4, and 6.

2.3 Misuse Cases

Misuse cases [22] are a security-oriented extension of the UML *use cases* [20], which support elicitation of software system functional requirements. Like use cases, misuse cases have two representations a graphical diagram and a textual template. In this work we will primarily focus on the *misuse case diagram*.

In addition to the standard use case concepts (*actor*, *use case*, and *includes* relationship), misuse case diagrams are extended with *misuser*, *misuse case*, and *security use cases* constructs and *threatens* and *mitigates* relationships (see Figure 7). A *misuser* is an actor who has an intention to harm the software system. A *misuse case* is a goal of the *misuser* and has communication association to use case actor *misuser.* Misuse case is accomplish by a sequence of actions executed either by combine efforts of several other misuse cases, or independently by *misuser*. *Threatens* and *mitigates* are relationships between use cases and misuse cases. *Threatens relationship* means a misuse case is potentially a threat to the use cases and can possibly harm it. *Mitigates relationship* indicates that a use case is countermeasure against any misuse case and, thus, mitigates it. *Security use case* is a special use case to perform countermeasure against identified threat.

As illustrated in Figures 3, 5, and 7, *misuse cases* are integrated in use case diagrams and show the system unwanted behaviour initiated by a *misuser*, who *threatens* the legitimate behaviour of the software system. This depiction results in a generation of a special type of use cases called *security use cases* in order to mitigate the threats caused by *misuser*.

2.4 Alignment of ISSRM and Security Modelling Languages

As discussed in [4] [14] [15] [16] the ISSRM domain model could be used to guide application of the security modelling languages with respect to the security risks analysis.

Alignment of ISSRM and Secure Tropos. The detailed explanation of the alignment between the ISSRM domain model and Secure Tropos is provided in [15]. In this section we summarise the major points shown in Table 1 (column 1). At the level of asset-related concepts, *actor, goal, resource* and *plan* are considered as the ISSRM *business* and *IT assets*. The ISSRM *security criterion* is expressed in through the Secure Tropos *security constraint* and *softgoal* constructs. At the level of risk-related concepts, Secure Tropos does not represent *risk* but it can express risk *impact* (as directed *contribution* between threat and softgoal) and *event* (as a *threat* construct). At the lower level of risk related concepts we can define the ISSRM *threat* through the Secure Tropos *goals/plans*, the ISSRM *threat agent* as the Secure Tropos *actor*, and the ISSMR *attack method* as the Secure Tropos *plan* (with *attack relationship*). We are not able to identify the ISSRM vulnerability, but it is possible to indicate the IS assets that have a vulnerable points. At the level of risk treatment-related concepts, the ISSRM *security requirements* are defined through the Secure Tropos *actors, goals, resources, plans* and *security constraints*.

Alignment of ISSRM and misuse cases. The detailed explanation of misuse cases analysed with respect to the ISSRM domain model in given in [14] and summarised in Table 1 (column 2). At the level of asset-related concepts, the standard use case constructs are applied. Thus, language constructs such as *actor*, *use case* and *system* are used to express the ISSRM *business* and *IS assets*. At the level of risk-related concepts, the misuse case *misuser* is used to present ISSRM *threat agent*, and the *misuse case* corresponds to the ISSRM *attack method.* Following the definition, the ISSRM *threat* is a defined as a combination the *misuser* and *misuse case* in the misuse case diagram. Finally at the level of risk treatment-related concepts it is possible to present the ISSRM *security requirements* as the *security use cases* in the use case diagram.

Table 1. Alignment table of ISSRM Model concepts with Secure Tropos and Misuse Case constructs

ISSRM Model Concepts			Secure Tropos Construct	Misuse Case Constructs
0			1	2
Asset	a	**Asset**	Actor, goal, softgoal, plan, resource	Actor and use case
	b	**Business asset**		
	c	**IS Asset**		System
	d	**Security criteria**	Security constraint, softgoal	-
Risk	e	**Risk**	-	-
	f	**Impact**	Contribution between threat and softgoal	-
	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	l	**Risk treatment**	-	-
	m	**Security requirement**	Actor, goal, softgoal, plan, resource, security constraint	Security use case
	n	**Control**	-	-

3. TRANSFORMATION RULES

The main research objective of our study is to define a set of transformation rules from the Secure Tropos model to the misuse case diagram. The transformation rules, presented in this section are based on the alignment [13] [14] between the ISSRM domain model and security modelling languages as illustrated in Table 1. They are presented following the *Meeting Scheduler* [7] [12] [22], a well-established exemplar in RE.

3.1 Translating Asset-related Concepts

In Figure 2 we present a content of the Meeting Scheduler modelled using Secure Tropos. Here, on the one hand, an Initiator is using a Scheduler to schedule a meeting (e.g., see the goal Meeting be scheduled). On the other hand, Participants are dependent on the Scheduler to get the information about the meeting Agreement (e.g., its place and time). We identify, that the Agreement is an important business asset, thus, security criteria are defined for its *confidentiality* (e.g., Only used by the participants of the meeting), *integrity* (e.g., Agreed data cannot be changed once agreement achieved), and *availability* (e.g., Agreement availability must be ensured).

To transform such the Secure Tropos model to the (mis)use case diagram we apply the following transformation rules:

TR1: An *actor* that presents the (software) system in Secure Tropos is translated to a misuse case *software system boundary*;

The TR1 is based on the alignment between the Secure Tropos *actor*, misuse case *system boundary* and the ISSRM *IS asset*, as indicated in Table 1, line ***c***. We translate the Secure Tropos actors Scheduler (see Figure 2) to misuse case software system boundary that carries name Scheduler, as illustrated in Figure 3.

TR2: A Secure Tropos actor that presents the organisational actor (or the actor from the system environment) is translated to a misuse case *actor*;

Similarly to TR1, the TR2 is based on lines ***a*** and ***b*** as illustrated in Table 1. In addition to this alignment we observe that the (external) actor concept is defined similarly both in Secure Tropos (actor carries out actions to achieve goals [1] [2]) and in use cases (where an actor is using a system to accomplish a goal [20]). In Figure 3 we present actors Initiator and Participant that are translated from the corresponding actors shown in Figure 2. Next, three transformation rules – TR3, TR4, and TR5 – are defined according to Table 1 lines ***a*** and ***b***.

TR3: Secure Tropos *goals* and *plans* are translated to *use cases* in the misuse case diagram;

Goals and plans in Secure Tropos and use cases in (mis)use case diagrams are used to present (i) processes that carry value for the actors; and/or (ii) functions of (software) system that support business processes. For example we transform Secure Tropos goal Meeting be scheduled (Figure 2) to Meeting be scheduled use case in Figure 3 (similarly it is done for other Secure Tropos goals and plans). Finally, we define two rules – TR4 and TR5 – that describe how Secure Tropos relationships are translated:

TR4: *Secure dependency* links from Secure Tropos are translated to *communication* association in the misuse case diagram;

TR5: Secure Tropos *means-ends* and *decomposition* links are translated to misuse case *includes* relationship;

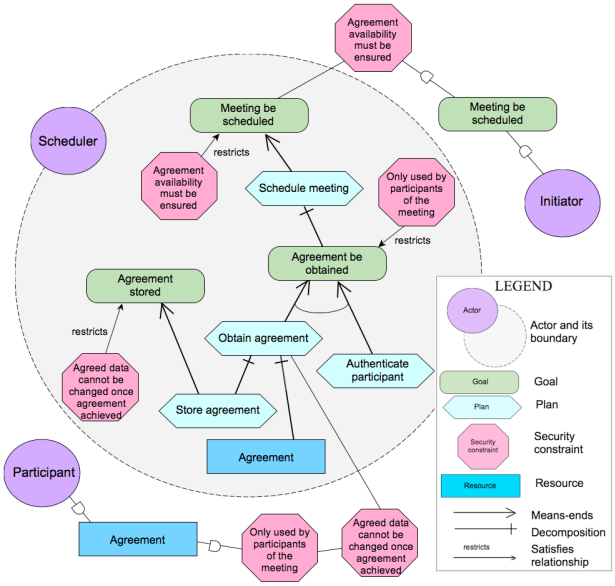

Figure 2. Asset Modelling using Secure Tropos

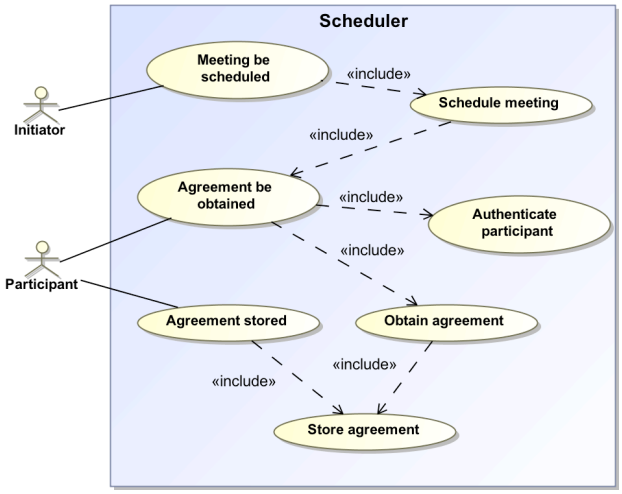

Figure 3. Asset Modelling using (Mis)use Case Diagram

Regarding TR4, Secure Tropos *actors* are dependent on the (software) system in order to achieve their goals. Similarly, in (mis)use cases diagram, a *use case* defines an interaction between actor and the (software) system to accomplish a goal. In addition, since a Secure Tropos *security constraint* always restricts some goal, we define that in misuse case diagram the *communication* link from an *actor* should be linked to the *use case* that in Secure Tropos is a goal restricted by the *security constraint*. To illustrate the TR4 we define communication links between Initiator and Meeting be scheduled, between Participant and Agreement be obtained, and between Participant and Agreement stored in Figure 3. In the misuse case diagram we are not able to present the ISSRM *security criteria* (see Table 1, line ***d***). This means that the security criteria in misuse cases should be recorded additionally using other means (e.g., misuse case textual template [22]).

Finally, the TR5 defines how the hierarchical structure of Secure Tropos constructs is translated to the misuse case diagram (see Figures 2 and Figure 3 for the illustration).

3.2 Translating Risk-related Concepts

Next step is to investigate potential risks to the Scheduler system. More specifically, in Figure 4 we concentrate on the *confidentiality* (e.g., Only used by participants of the meeting) security criteria defined for the business asset Agreement. There might exist an actor, called Attacker, who attacks the Agreement having the purpose to disclose it to unintended audience. To reach his goal (e.g., Disclose agreement) the Attacker needs to Steal agreement and to Reveal stolen data. This becomes possible because the Scheduler software system has vulnerability in its participant authentication plan (e.g., Authenticate participant): it is possible to repeat the authentication procedure multiple times[1].

As shown in Table 1 (line ***e***) we are able to express the ISSRM *risk* concept neither using Secure Tropos, nor misuse case diagrams; however using these security languages we can define the lower level ISSRM concepts, such as *threat agent*, *attack method*, and *threat*. Thus, to translate the Secure Tropos diagram (e.g., Figure 4) to the misuse case diagram (e.g., to Figure 5), we define the following rules:

TR6: A Secure Tropos *actor* who *exploits* and/or *targets* elements of other actors, is translated to the *misuser* in the misuse case diagram;

This rule is based on the correspondence between the Secure Tropos *actor* (e.g., Attacker in Figure 4) and misuse case *misuser* (e.g., Attacker in Figure 5) to the ISSRM *threat agent*, as shown in Table 1, line ***j***.

TR7: Secure Tropos *goals* and *plans* that belong to the actor who *exploits* and/or *targets* constructs of other actors, are translated to the *misuse cases* in the misuse case diagram;

The TR7 is based on the lines ***h*** and ***k*** (see Table 1). Following this rule, in Figure 5 we introduce four misuse cases, like Disclose agreement, Steal agreement, Try scheduler authentication repeatedly, and Reveal stolen data. Further, to keep the constructs hierarchy consistent in both models, we apply the transformation rule TR5, which transforms the Secure Tropos *decomposition* and *means-ends* links to the *includes* relationships in the misuse case diagram.

In the next step we define the relationships between the ISSRM *threat* and the attacked *IS asset*:

TR8: A Secure Tropos *exploits* relationship is translated to a *threatens* relationship in the misuse cases diagram;

For example in Figure 5 we define *threatens* relationship from the misuse case Try authentication repeatedly to the use case Authenticate participant. This relationship corresponds to the ISSRM *exploits* relationship (see Figure 1) As discussed in [13], on the one hand we are not able to model *vulnerabilities* using Secure Tropos; but we can express the *IS asset* which has the vulnerable point (e.g., notion of *vulnerability point* is defined in [5]). On the other hand, we are not able to express the ISSRM *vulnerability* in the misuse case diagrams neither. In addition, in the misuse case diagram we do not model the static secure resources (e.g., Agreement, which is an ISSRM *business asset* as presented in Figure 4), because (mis)use case diagrams are used to specify (malicious) acts [20] [22]. This means that we do not address the Secure Tropos *attacks* link (aligned with the ISSRM *targets* relationship) in the misuse case diagram.

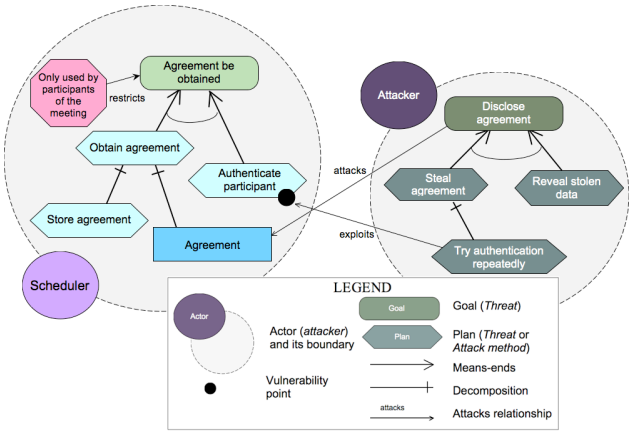

Figure 4. Threat Definition using Secure Tropos

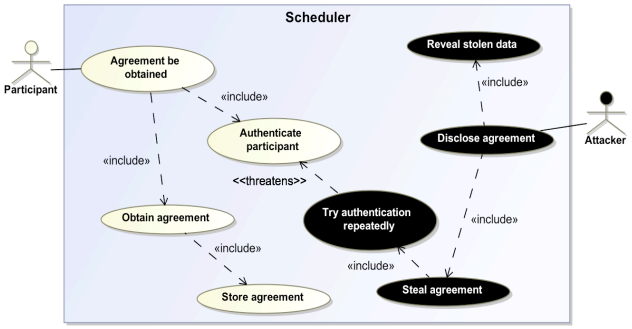

Figure 5. Threat Definition using Misuse Case Diagrams

3.3 Translating Risk Treatment-related Concepts

After the potential risks to the valuable system assets are understood, the next step is to specify risk countermeasures. For example, to mitigate the risk of agreement disclosure (e.g. Disclose agreement in Figure 6), we define a plan to Perform cryptographic procedures[2]. This means that even if the Attacker gets access to the Agreement, s/he is not be able to use it, because the Agreement is encrypted. The Secure Tropos *plan* Perform cryptographic procedures contributes to the satisfaction of the confidentiality security constraint (e.g., Only used by participants of the meeting).

TR9: A Secure Tropos *mitigates* link is translated to a *mitigates* relationship in the misuse cases diagrams.

[1] Let's suppose that the Attacker succeeds obtaining the Participant's login name of the. Then s/he is able to Try authentication repeatedly to the Scheduler in order to determine participant's password. The similar case is described in [12].

[2] As defined in Table 1, line ***g***, the Secure Tropos *threat* is aligned with the ISSRM *event*. According to ISSRM, the *event* is decomposed to a *threat* and *vulnerability*. Following Secure Tropos [17] [18] [19], we observe that Attacker and its boundary elements present the more detailed view of the Secure Tropos *threat* (e.g., Disclose agreement) as defined in Figure 6.

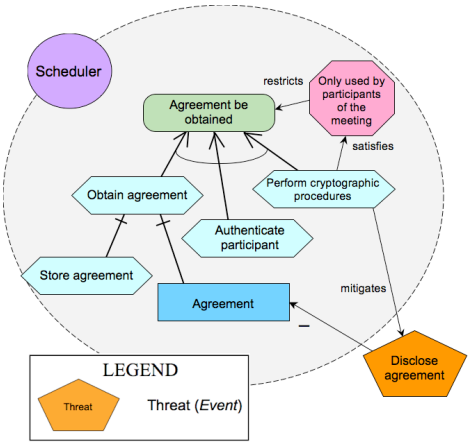

Figure 6. Risk Treatment using Secure Tropos

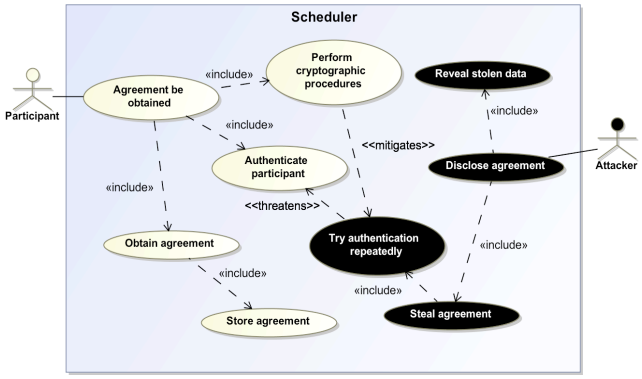

Figure 7. Security Requirements using Misuse Cases

To translate the risk countermeasures defined using Secure Tropos to the misuse case diagram, firstly, we can apply the TR3 and TR5. We follow the alignment between the Secure Tropos *plans* and *goals*, *security use cases* of the misuse case diagrams and the ISSRM *security requirements* as specified in Table 1, line ***m***. For example in Figure 7 we introduce a security use case Perform cryptographic procedures, which mitigates the identified misuse cases (e.g., Disclose agreement).

This rule is based of alignment to the *mitigates* relationship defined in the ISSRM domain model (see Figure 1). We emphasise that in the misuse case diagram *mitigates* link should be defined from the *security use case* (e.g., Perform cryptographic procedures) to the top-level *misuse case* (e.g., Disclose agreement) as illustrated in Figure 7.

4. DISCUSSION

In this work we define a transformation from the Secure Tropos model to the misuse case diagram. Our rules a semi-automatic model translation. For example, when translating Secure Tropos model to the misuse case diagram, the modeller needs to identify the actor that presents the (software) system. This actor is then translated to the misuse case software boundary according to the transformation rule TR1. The remaining rules can be applied automatically.

The translated misuse case diagram should be considered as an initial step towards definition of the overall misuse case model. For instance, each use and misuse cases should be complemented with the textual templates[3], resulting in more detailed specification of the analysed system. However we also observe that some information captured in the Secure Tropos model is not possible to transfer to the misuse case diagrams. For instance secure dependencies are translated to the communication links (see TR4), thus resulting in the loss of the dependum and security constraint definition.

In [14] the textual misuse case template is analysed to support the IS security risks. The analysis provides the correspondences between the misuse case textual templates entries and the ISSRM domain model. A misuse case *textual description* can identify risks, security requirements and mitigation steps from the system environment. We acknowledge the importance of textual representation that how it complements and strengthen the transformation rules defined in Section 3; but due to the lack of space we do not discuss how the ISSRM concepts such as risk, impact, vulnerability, threat agent, attack method and security requirements could be captured from the Secure Tropos model. We only note that populating these concepts in misuse template require intensive manual interaction from the modeller.

5. RELATED WORK

Santander and Castro introduce a set of guidelines [21] to translate the *i** (predecessor of Tropos) models to the use case diagram. Firstly, they extract actors from the *i** strategic dependency model. Secondly, the actor dependencies are analysed to form use cases scenarios, which, finally is finalised through analysis of the *i** strategic rationale model. This work, like our proposal, defines the exogenous model translation. However, Santander and Castro base their guidelines on the syntactic language correspondences. In comparison, we propose our transformation rules based on the semantic alignment of the constructs of Secure Tropos and misuse case with ISSRM. In addition, we specifically target the security domain, thus capturing the smooth translation of the early security decisions to the later stages of the secured IS development.

In [19] Secure Tropos is integrated with the UMLsec approach [11]. As defined Secure Tropos helps define social security aspects of system's stakeholders to identify the security requirements. While UMLsec helps analyse security at the architectural and detailed design stages [11]. The work [19] integrates both approaches to define a smooth development of a secure IS, starting from early analysis to the detailed design. Our approach also emphasises the importance of the early security understanding and the transition to the later security development activities. In contrast, we propose a translation the Secure Tropos model to the misuse cases, as an intermediate technique between the early (using Secure Tropos) and detailed design (using UMLsec) stages. In addition we support the transformation (and not integration) approach to combine two modelling techniques.

6. CONCLUSION

In this paper we define a set of transformation rules to translate a Secure Tropos model to the misuse case diagram. We base our transformations on the Secure Tropos and misuse case alignment [14] [15] to the ISSRM domain model [4] [16]. Thus, it becomes

[3] Some entries of the textual template could be identified according to the ISSRM domain model [14]. However due to the space limits we do not discuss this in the paper.

possible to preserve the security-relate semantics expressed using these languages. The proposed transformation rules are also applicable to other *i** security extensions, such as Secure *i** [5] or other versions of Secure Tropos [8] by first aligning the model's constructs with ISSRM domain model.

Applying Secure Tropos to elicit security requirements helps understanding and reasoning on the benefits and security trade-offs [5] at the early IS stages. Thus the systematic translation of the Secure Tropos model to the misuse cases allows developers to relate together the security needs and software system functional features together. In addition such a representation using the (mis)use case diagrams directly addresses the important organisational needs and goals elicited using Secure Tropos.

Another benefit for relating these languages together is that it maintains and manages traceability links between two different representations of the problem. The security solutions expressed in the misuse case diagrams are directly derived from the Secure Tropos model; thus the constructs expressed in both modelling languages are related to each other and can easily be traced back and forth.

In this work we have illustrated the proposed transformation rules to the running example, but we certainly acknowledge the importance to validate our work using real business case. Further more we plan extension of the scope of our analysis for other security-oriented techniques, such as KAOS extensions to security [12] or framework introduced by Haley *et al*. [9]. Our larger target is to introduce a systematic model-driven security framework to facilitate security understanding from early requirements to system design and implementation.

7. REFERENCES

[1] Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. and Mylopoulos, J. 2004. Tropos: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*, Springer.

[2] Castro, J., Kolp, M. and Mylopoulos, J. 2002. Towards Requirements-Driven Information Systems Engineering: The TROPOS Project. *Information Systems* 27, 365-389.

[3] DCSSL 2004. EBIOS – Expression of Needs and Identification of Security Objectives.

[4] Dubois, E., Heymans, P., Mayer, N. and Matulevičius, R. 2010. A Systematic Approach to Define the Domain of Information System Security Risk Management. In *Intentional Perspectives on Information Systems Engineering*. Springer, 289-306

[5] Elahi, G. and Yu, E. 2007. A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs. In *Proceedings of ER 2007*. Springer, 375-390.

[6] ENISA, 2004. Inventory of Risk Assessment and Risk Management Methods.

[7] Feather, M. S., Fickas, S., Finkelstein, A. and van Lamsweerde A. 1997. Requirements and Specification Exemplars. *Automated Software Engineering*, 4, 419-438.

[8] Giorgini, P., Massacci, F., Mylopoulos J. and Zannone, N. 2005. Modeling Security Requirements Through Ownership, Permission and Delegation. In Proceedings of RE'05, IEEE Computer Society, 167-176.

[9] Haley, C. B., Laney, R. C., Moffett, J. D. and Nuseibeh, B. 2008 Security Requirements Engineering: A Framework for Representation and Analysis. IEEE Trans. Software Eng., 34 (1), 133-153

[10] ISO, 2005. Information technology – Security techniques – Information security management systems – Requirements, International Organisation for Standardisation.

[11] Jurjens, J. 2005. *Secure Systems Development with UML*. Springer-Verlag Berlin Heidelberg.

[12] van Lamsweerde, A. 2004. Elaborating Security Requirements by Construction of Intentional Anti-models. In *Proceedings of the ICSE'04*, 148-157.

[13] Matulevičius, R. and Dumas, M. 2011. Towards Model Transformation between SecureUML and UMLsec for Role-based Access Control. Databases and Information Systems VI, IOS Press, 339-352

[14] Matulevičius, R., Mayer, N. and Heymans, P. 2008. Alignment of Misuse cases with security Risk Management. In *Proceedings of ARES 2008*, IEEE Computer Society, 1397-1404.

[15] Matulevičius, R., Mayer, N., Mouratidis, H., Dubois, E., Heymans, P. and Genon, N. 2008. Adapting Secure Tropos for Security Risk Management during Early Phases of the Information Systems Development. In *Proceedings of CAiSE 2008*, Springer Heidelberg, 541-555.

[16] Mayer, N. 2009. *Model-based Management of Information System Security Risk*. Doctoral Thesis, University of Namur.

[17] Mouratidis, H., Giorgini, P. and Manson, G. 2003. An Ontology for Modelling Security: The Tropos Approach, Knowledge-based Intelligent Information and Engineering Systems. In *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 1387-1394.

[18] Mouratidis, H. and Giorgini, P. 2007. Secure Tropos: A security-oriented Extension of the Tropos Methodology. *International Journal of Software Engineering and Knowledge Engineering*, World Scientific Publishing.

[19] Mouratidis, H., Jurjens, J. and Fox, J. 2006. Towards a Comprehensive Framework for Secure Systems Development. In Proceeding of CAiSE 2006, Springer, Heidelberg, 48-62.

[20] OMG 2007. Unified Modeling Language (OMG UML), Superstructure, V2.1.2 http://www.omg.org/spec/UML/2.1.2/Superstructure/pdf.

[21] Santander, V. and Castro, J. 2002. Deriving Use Cases from Organizational Modeling. In *Proceeding of the RE'02*. IEEE Computer Society, 32

[22] Sindre, G. and Opdahl, A. L. 2005. Eliciting Security Requirements with Misuse Cases. *Requirements Eng*. 10 (1), Springer-Verlag.

[23] Zulkernine, M., Graves, M. and Khan, U. 2007. Integrating Software Specifications into Intrusion Detection. International Journal of Information Security (IJIS), Springer, 345-357.