

Abstract—Security is now the most critical feature of any computing systems. Eliciting and analyzing security requirements in the early stages of the system development process is highly recommended to reduce security vulnerabilities which might be found in the later stages of the system development process. In order to address this issue, we will propose a new extension of the misuse case diagram for analyzing and eliciting security requirements with special focus on assets and security goals. We will also present the process model in which business requirements and system requirements related to security features are separately analyzed and elicited in different phases. This process model helps us to analyze the requirements related to business goals in an earlier phase and to the system goals in a later phase so that any concerns related to them are dealt with separately. We will illustrate our approach with a case study taken from an accounting software package.

I. INTRODUCTION

One of the mandatory practices when developing a system is to elicit and analyze the security requirements in the early stage of the system development process. Several requirements engineering methodologies have been proposed to address various security issues. Notable examples are KAOS [11] and i* [4]. Unfortunately we can barely find any evidence of industry-wide adoption of these methodologies. We believe that there are two possible reasons why companies are failing to do so. One is a lack of substantial connection to the existing system development process and another is their complex semantics and learning cost. In order to avoid these problems, it is best to adopt widely used methodologies such as a use case diagram in the UML (Unified Modeling Language). A use case diagram is used to capture the functionalities of a system by the actors and the use cases including several relationships between them.

Security is a non-functional requirement. Other non-functional requirements include reliability and performance among others. What makes security completely different from other non-functional requirements is that it tries to assume the types of attackers that will try to harm the system. Attackers have malicious intent and often have some means of exploiting the vulnerabilities of a system. It is hard to model security features without any description of the attackers and their threats to a system. Misuse case diagrams, which are extensions of the use case diagram, are proposed to model the requirements related to security issues. We can explicitly model the potential attackers and how they can harm the system as well as how their threats can be mitigated. Misuse case diagrams [9], [10] are friendly to industrial practitioners thanks to their simple graphical syntax and their underlying semantics compared with other complex methodologies, e.g., i* and KAOS.

The main concern in analyzing a system’s vulnerabilities would be how to protect assets of the system. Assets are resources with potentially great value to the system. Security goals represent the intentions/reasons why we need to protect them from threats. However, it is hard to specify what should be protected and why we intend to do so in misuse case diagrams due to the lack of supporting model elements available for them. Based on this observation, we will present a new extension of the misuse case diagram, which incorporates assets and security goals into the modelling elements. We will also present a two-phase elicitation process model in which the preliminary goals are business-oriented and those for the system-to-be are elicited in a later phase. This goal distinction helps us to analyze the requirements related to business (security) goals in an earlier phase and to the system (security) goals in a later phase.

Our intended contribution is twofold. First is to incorporate an asset-based viewpoint into the misuse case diagram, which is supported by security goals. Second is the proposal of a process model, which supports this new extension to the misuse case diagram. We will illustrate our approach with a part of a larger case study, which is taken from an accounting software package.

This paper is organized as follows. The next section explains our proposed notation. Section 3 presents the two-phase process model for our extended diagrams. Section 4 illustrates the case study and Section 5 compares related work. We will finally concludes the paper in Section 6.

II. EXTENDED MISUSE CASE DIAGRAMS

The misuse case diagram we talk about here was proposed by Sindre and Opdahl in [9], [10]. The diagram is a user-friendly graphical notation thanks to their simple syntax and its underlying semantics. One can explicitly model misusers/attackers and how they intentionally or inadvertently threaten a system’s functionalities, and how they can be mitigated by their countermeasures. The purpose of the diagram is to provide the functional aspect of the security properties of a system. There are some variations, e.g., abuse cases by McDermott and Fox [5] and security use cases by Firesmith.
[2] and some recent works by the first author of this paper [7], [8].

We will reuse the basic model elements, actors, attackers, use cases, and misuse cases in the misuse case diagram. We will also explain our extension of the misuse case diagram in this section.

A. Asset-based extension

In our diagram an asset is a first-class citizen that can be represented by its name and the stereotype <<asset>>. An asset can be associated with a use case, if a use case uses the asset as a data object or resource. A threat is denoted by a use case with the stereotype <<threaten>>. If an asset is associated with a misuse case with a dashed line to the stereotype <<threaten>>, it means that the misuse case might cause harm to the asset. A threat may be mitigated by a countermeasure, which is represented as a use case with the particular stereotype <<countermeasure>>. This way, we can specify what should be protected (by asset), what harms the asset (by misuse), and finally what mitigates the threat (by countermeasure).

B. Security goal based extension

Our extension for incorporating assets into misuse case diagrams introduces a new viewpoint into the diagrams. Unfortunately just adding assets is not enough to elicit and analyze the requirements related to the security features of a system. Security requirements represented by countermeasures can model how to protect assets from any harm. However, they are insufficient for specifying what intention the system developer has in order to secure the system and how it is operationalized into countermeasures. Our second extension is to incorporate security goals into our misuse case diagram notation.

Security goals are represented with the same oval icon as a use case that is stereotyped by <<goal>>. Security goals may be associated with another security goal(s). An upper abstract goal can be refined into a lower concrete goal. If a security goal is associated with an asset, it means that the security goal is intended to protect the asset from threats. Security goals are operationalized into countermeasures, which protect the system. Their relationship is depicted by a dashed line stereotyped by <<operationalize>>.

Figure 1 is a basic diagram that illustrates all the model elements explained in this section. We will explain how we will elicit and analyze security requirements in the next two sections.

III. PROCESS

A. Overview

In this section we present a security requirements elicitation process that consists of two phases, business requirements elicitation and system requirements elicitation. The main reason why we have divided the process in two different phases is that it is difficult to identify the security goals or security requirements associated with a system in the very beginning of the requirements elicitation/analysis stage. Generally, system modelling, which is related to a system’s architectural/system environment, would be carried out comparatively later. Therefore, we propose the system modelling be carried out once the business security requirements are elicited. This distinction is implicit in a sense that it is not reflected in our extended misuse case diagrams. Figure 2, depicted as an activity diagram describes this proposed requirements elicitation process.

We will now explain the proposed process step by step.

(1-0) Define non-security requirements and goals.

Non-security requirements and goals are elicited in this step. This step is carried out using a traditional goal-oriented approach. The non-security goals and requirements are the outputs of this step and are also the inputs of the security analysis steps.

(1-1) Identify (business-oriented) assets.

Next, identify the assets of the target system. As we have emphasized in the previous sections, asset identification is a crucial step for our security requirements analysis method. The use cases identified at (1-1) and their associated data objects are identified as assets.

(1-2) Identify security (business) goals.

This step is to identify the security goals related to the business concerns. Well-known security properties, confidentiality, integrity, and availability are automatically given as the sub-goals of the top level of the security goal.

(1-3) Assess assets and security goals.

Assessing the need of the security sub-goals are identified at the previous step. We must analyze the threats that can be obstacles to the sub-goals. If a threat is harmful, the sub-goal related to the threat must remain. If not, it should be removed. Then, add objectives to mitigate the threats against the sub-goals as refinements of the sub-goals.

(1-4) Identify and assess misuse cases (threats).

Add the threats identified in the previous step as misuse cases.

(1-5) Identify countermeasures.

Identify the countermeasures that mitigate the threats. These countermeasures are added as countermeasure use cases of misuse. And then the analysts
can verify whether the identified countermeasures satisfy the security goals.
New assets might arise as a consequence of this step. Therefore, iteration from (1-1) to (1-5) and (1-1) to (1-4) are required until no further assets are identified.

(2-0) define system model.

The business requirements including the security have been elicited before this step processes. Then, the analysts should model the system architecture and some implementation specifications.

(2-1) Identify system assets.

New assets related to the system arise as a consequence of the system modelling. So, identification of these system assets is required next.

The steps that follow are similar to those for (1-2) to (1-5), so we will omit the explanation.

(2-2) Identify security (system) goals.

(2-3) Assess assets and security goals related with a system.

(2-4) Identify and assess misuse cases (threats).

(2-5) Identify countermeasures.

Iterations from (2-1) to (2-5) and (2-1) to (2-4) are also required after the first iteration is completed.

B. Discussion

The advantages of our process model over the existing methods, such as [3], are:

- The security goals and assets are classified in two types, business goals (assets) and system goals (assets). The steps for deriving the system goals and assets are located after the steps for the business goals and assets. Isolation of the business security goals is a better approach because they are system-independent. Even if the developers change the system architecture of the target software, it does not affect the business security goals. Moreover, the system architecture is defined after the use cases including the security functions are defined.

- Our model specifies both data and use cases(functions) as assets. This specification enables identifying not only information security threats such as disclosure, tampering, etc. but also system security threats such as DoS attacks, abuse, etc.

- It is difficult to verify if the security goals are satisfied in the analysis stage. Our process enables this to happen by looking at the dependency between the countermeasure use cases and security sub-goals. By checking the coverage of the operationalized sub-goals, the analysts can
verify the satisfaction of the security goals.

IV. CASE STUDY

We carried out a case study on an accounting software package. We will discuss a part of the case study in this section in order to illustrate our approach.

The package provides several services such as journalize and update account title among others, which are modelled as ordinary use cases in Fig. 3.

This is the result of step (1-0) without any business goals. We will only illustrate a part of the larger diagram, which shows only the security related features of our extended diagrams.

(1-1) Identify business assets.

We identified the journal data and account title as the candidate (business) assets.

(1-2) Identify security (business) goals.

confidentiality, integrity, and availability are added as the sub-goals of the security goal, which is the top goal of the security goals.

(1-3) Assess assets and security goals.

From the assessment, the confidentiality and availability sub-goals are removed, since they are not relevant to the security features at this time. On the other hand, a tamper threat is identified. The prevent tampering and trail sub-goals are added as
The result of the refinement of the goal integrity.

(1-4) Identify and assess misuse cases (threats).

Now, tamper is added as a misuse case.

(1-5) Identify countermeasures.

Identify and authenticate, validate data, and lock data are added as the countermeasures against tamper.

Since the added countermeasures and data may require security protection. They should be examined if they are assets. As a consequence of (1-5), the validate data and lock data use cases are identified as new assets. Please note that countermeasures may be identified as assets in our analysis.

Now, the second iteration is needed. The result of the second iteration in which availability is revived to add the new security goal prevent unintended use.

After the iteration is completed, system modelling (2-0) will be carried out. The assets and threats related to the system model are identified in steps (2-1) to (2-5). After password asset and password attack and wiretap threats are identified, prevent password attack and encrypt communication countermeasures are defined as a result of this analysis. The final result of our process is shown in Fig. 4.

V. RELATED WORK

There are several extensions of use case diagrams with respect to security issues. Our proposal largely relies on those previous works and Haley et. al [3]. We will now compare the extensions of misuse cases.

The abuse case diagrams by McDermott and Fox [5] use abuse cases that represent those which harm the system as a result of their interactions with the system. The notation does not support any difference between normal use cases and abuse cases. A simple process model is presented on how to construct an abuse case diagram. Our proposed extension is mainly based on Sindre and Opdahl [9], [10]. They proposed misuse case diagrams in which misuse cases and misusers are introduced. They used coloured icons to represent both elements and did not use the stereotype as we did in our proposal. The most substantial difference between their proposals and ours are that they lack of crucial model elements to capture security concerns, namely, the assets and security goals.

Alexander [1] proposed an extension of misuse case diagrams with goals. In his notation, use cases are replaced by goals, and several types of conflicts, e.g., direct conflict between goals, and indirect conflict between hostile goal and ordinary goal may be described and analyzed. His proposed notation is regarded as a new type of goal-oriented requirements method and diverged from a conventional use case diagrams, since there are no longer any use cases.

The first author of this paper uses assets to denote resources which can be used between actors and use cases [7], [8]. They may be annotated by stereotypes <<I>> for integrity, <<A>> for availability and <<C>> for confidentiality depending on their nature. These papers also introduced several roles for misusers. This is quite similar to our approach, but assets are not independent entities in their approach so that it is hard to identify the assets themselves in the elicitation process.

Haley et. al presented a security requirements engineering framework. Their work could be the first to emphasize the importance of assets as well as security goals. They present the core artifacts in security requirements and their structure and activities in the framework. Mead et. al presented SQUARE [6], a process model for security requirements. She recently incorporated the asset-based viewpoint into SQUARE. Our work is along the same line with theirs but their works do not present any modeling notation so it is not easy in practice to adopt their methodologies.

VI. CONCLUSION

In this paper we have proposed a new extension for misuse case diagrams by incorporating some new model elements, assets and security goals for eliciting and analyzing the security features of a computing system. We have illustrated our approach using a process model in which the business (security) goals and system (security) goals are separately elicited in different phases. This process model helps us to analyze the requirements related to the business goals in an early phase and the system goals in a later phase in the security requirements elicitation process.

Future work includes further extensions of our diagrams. Security consists of features like confidentiality, availability, integrity and accountability. Sometimes these features may have interactions which cause conflicts. These features can be represented as types or attributes of a goal. Our plan is to explore how these new types of goals could help us to analyze any conflicts in the security features, and how to reach a compromise among the security goals.
REFERENCES


