Journal of Information Processing Vol.22 No.3 536–546 (July 2014)

[DOI: 10.2197/ipsjjip.22.536]

Regular Paper

MASG: Advanced Misuse Case Analysis Model with Assets and Security Goals

Takao Okubo[1,a)] Kenji Taguchi[2,b)] Haruhiko Kaiya[3,c)] Nobukazu Yoshioka[4,d)]

Received: July 11, 2013, Accepted: February 14, 2014

Abstract: Misuse case model and its development process are useful and practical for security requirements analysis, but they require expertise especially about security assets and goals. To enable inexperienced requirements analysts to elicit and to analyse security requirements, we present an extension of misuse case model and its development process by incorporating new model elements, assets and security goals. We show its effectiveness from the quantitative and qualitative results of a case study. According to the results, we conclude the extension and its process enable inexperienced analysts to elicit security requirements as well as experienced analysts do.

Keywords: security, requirements analysis, misuse case and goal oriented requirements analysis

1. Introduction

One of the mandatory practices when developing a software system is to elicit and analyse the security requirements in an early stage of the system development process. Several requirements engineering methodologies have been proposed to address various security issues. Notable examples are KAOS [27] and i* [13]. It is, however, hard to use these methodologies efficiently in industry. We believe that there are two possible reasons why companies are failing to use them. One is a lack of substantial connection to the existing system development process and the other 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 Unified Modeling Language (UML). A use case diagram is used to capture the system's functionalities used by the actors and the use cases including several relationships between them.

Security is a non-functional requirement; others 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 system's vulnerabilities. It is hard to model security features without any description of the attackers and their threats to a system. Misuse case diagrams [22], [23], which are extensions of the use case diagram, have been proposed to model the requirements related to security issues. We can explicitly model the potential attackers and how they could harm the system as well as how their threats can be mitigated. The diagrams are friendly to industrial practitioners thanks to their simple graphical syntax and their underlying semantics in comparison with other complex methodologies, e.g., i* and KAOS.

The main concern in analysing the system's vulnerabilities is how to protect *assets* in the system. Assets are resources with potentially great value to the system's stakeholders. We can understand the intentions and reasons why we need to protect them from threats from the viewpoint of *security goals*. However, it is hard to specify what should be protected and why we intend to do so in misuse case diagrams because of a lack of supporting model elements available for them. This is why many engineers without comprehensive knowledge of security fail to identify critical threats or appropriate countermeasures. On the basis of this observation, we present an extension of the misuse case diagram called *the misuse case with assets and security goals* (MASG), which incorporates *assets* and *security goals* into the modelling elements. We also present an elicitation process for the models.

In an MASG diagram, an asset is associated with a misuse case, which means that the misuse case might cause harm to the asset. In addition, a security goal is associated with an asset, meaning that the security goal is intended to protect the asset from threats. Eventually, security goals are operationalized into countermeasures.

Our contribution is twofold. First, we incorporate an asset-based viewpoint into the misuse case diagram, which is supported by security goals. In other words, we specify new diagram notation including assets and security goals in addition to a misuse case diagram. Second, we propose a process model, which supports this new extension to the misuse case diagram. This paper is a revised version of a paper [20] in which the initial concept has been proposed. In this paper, we present a detailed comparison with related works, give a precise definition of the diagram, and illustrate the effectiveness of the approach through a case study.

[1] Institute of Information Security, Yokohama, Kanagawa 221–0835, Japan
[2] National Institute of Advanced Industrial Science and Technology, Amagasaki, Hyogo 661–0974, Japan
[3] Kanagawa University, Hiratsuka, Kanagawa 259–1293, Japan
[4] National Institute of Informatics, Chiyoda, Tokyo 101–8430, Japan
[a)] okubo@iisec.ac.jp
[b)] kenji.taguchi@aist.go.jp
[c)] kaiya@kanagawa-u.ac.jp
[d)] nobukazu@nii.ac.jp

The rest of the paper is organized as follows. In the next section, we first review researches on security analysis related to assets and goals. We also review and compare existing methods for eliciting security requirements using use cases such as those given in Ref. [23]. In Section 3, we introduce our security requirements diagram and the security requirements elicitation process called *MASG*. We also compare MASG with existing methods based on use cases to show that it is more effective than others. To validate our method, we conducted a comparative experiment, which is reported in Sections 4 and 5. Finally, we conclude with a summary of our contributions and mention future issues.

2. Related Work

Because assets and goals play an important role in MASG, we first review research related to them. We then review security requirements analysis methods based on the use case modelling because MASG is also based on use case modelling. We explain modelling using the method MUC (misuse case) in detail because MASG is an extension of MUC.

2.1 Assets

In the context of information system management, security is the protection of information [9] and the information is a kind of asset. One textbook [6] gives the following similar but more direct statement: "security is about the protection of assets." Computer security rests on confidentiality, integrity, and availability [1], and these aspects are frequently used as basic security requirements. Although an asset is the central concept for security, there has been little discussion about assets in the context of requirements engineering. We think that one of the main reasons is related to the characteristics of well-known notations such as use case and goal models. In these notations, explicit writing of data is neither preferred nor allowed.

We briefly review existing security requirements research related to assets. To summarize, such studies fail to combine asset analysis with functional requirements analysis. Assets are categorized into information, software, physical assets, services, people, and intangibles in a standard [9], but such categorization does not fully help us to elicit security requirements. The British standard BS7799 also has similar categories. Supaporn et al. [24] proposed a method to identify assets by using formal grammar and its patterns. In this research, the relationships between assets and the system's functionalities are not clear. Jaatun et al. [10] proposed a method to prioritize each asset with respect to confidentiality, integrity and availability. This method also does not focus on why the assets cause vulnerabilities. A study by Marino et al. [15], [16] is similar to Jaatun's research [10]. Long et al. propose the AVT Vector (AVT: asset, vulnerability, trustworthiness) for security requirements quantification [14]. On the basis of this research, it is not easy to trace the causes of threats to each asset because BS7799 asset categorization is used and the categorization is very rough. In the CORAS method [2], an asset is modelled only with respect to the value to its holder. Such value is useful for prioritising assets, but not for finding why assets cause vulnerabilities. Microsoft's threat modelling technique [25] directly uses data flow diagrams although its focus is not requirements engineering.

Haley et al. presented a security requirements engineering framework [7]. 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.

2.2 Goals

As mentioned in the introduction, it is not easy to find security requirements without knowing the stakeholders' goals. Knowing such goals enables a requirements analyst to elicit security requirements that are consistent with them. KAOS [28] and i* [29] are representative goal oriented requirements analysis methods and notations. In i*, goals are identified through the dependencies among actors, and each goal is decomposed into tasks of each actor who has the responsibility to achieve the goal. In KAOS, goals are identified on the basis of the domain model, and each goal is decomposed into requirements, assumptions, and domain properties. A system to be developed has to achieve the requirements. The actors related to the system have to meet the assumptions. The real world rules the domain properties. Each requirement is refined into an operation, and the operation closely corresponds to a use case. There already exist various kinds of extensions of KAOS and i* for security requirements [13], [17], [19], [27], but most of them are too complex to analyse assets and other issues. In MASG, goals and assets are naturally combined with other elements such as use cases and actors, and goals are used as a trigger to find and validate security requirements together with assets.

For finding and refining security requirements from goals, predefined catalogues of goals and their refinement are useful. The NFR framework [3] is a representative catalogue, and it offers general security goals such as confidentiality, integrity and availability. Saeki et al. reused the contents of product documents certified by common criteria (CC) to derive security requirements in a goal model [21]. Although MASG does not provide any specific catalogues or contents, we intend to use such existing techniques. CC contains explicit relationships among assets, threats, security objectives, and security functions. Such relationships are useful information for security requirements modelling, and such modelling is also useful information for the relationships. Taguchi et al proposed a security-modelling framework for combining CC with use case modelling for aligning security requirements and security assurance [26]. This framework is similar to ours because a part of CC is similar to a goal model.

2.3 Use Cases

The advantages of the original use cases [4], [11] are as follows. First, the use case model is simple enough to define the boundary between a system and its related actors. Most stakeholders including businesspersons can easily understand it. Second, the use case model does not restrict variations in system architecture and/or design as much as possible. One of the reasons why data should not be specified in a use case comes from this advantage. Here, we compare existing techniques that extend use cases to security.

Abuse cases [18] is the oldest one. In this technique, a misuse

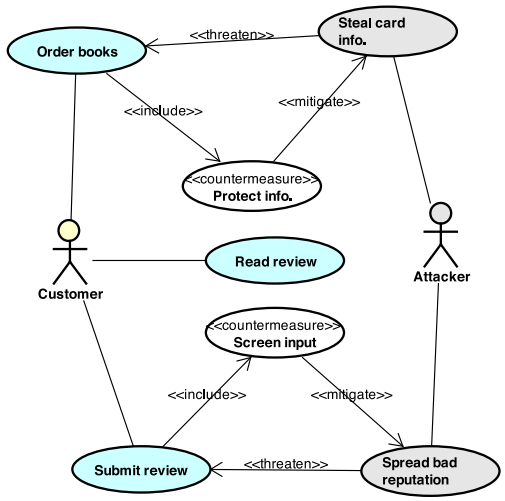

Fig. 1 Example of an MUC model. This model is a simplified and modified version of the model in Fig. 1 of Ref. [23].

Table 1 Comparison of security requirements elicitation methods using use cases in terms of language and notation.

	Abuse [18]	MUC [23]	Sec. use cases [5]
Use case	√	√	√
Misuse case	√	√	√
Actor	√	√	√
Malicious actor	√	√	√
Assets			
Goals			
Relationships among use cases and misuse cases		√	√

Table 2 Comparison of security requirements elicitation methods using use cases in terms of method and process.

	Abuse [18]	MUC [23]	Sec. use cases [5]
Assets		√	
Goals		√	

case model is written separately from a use case model. It is thus hard to analyse the relationships among requirements, threats, and security requirements. The process of writing abuse cases is also shown in Ref. [18]. In this process, actors in a use case model are candidates of attackers.

Misuse cases [22], [23] is the most well-known security extension for use cases. As mentioned in the introduction, MASG is an extension of MUC. In an MUC model, attackers (misusers) and attacks (misuses) are explicitly represented as well as actors and use cases. Each misuse case has also one or more explicit associations to one or more use cases. Each such association specifies a misuse that could be a threat for the use case. To mitigate and/or avoid such misuses, additional use cases are added. Such an additional use case has also one or more explicit associations to one or more misuse cases. Each such association specifies an additional use case that could be a countermeasure for a misuse case. The process for developing the misuse case model is also proposed, and assets and goals are focused on in some steps.

A simple example of MUC for an online shopping system like amazon.com is shown in **Fig. 1**. This MUC contains three normal use cases: "Order books," "Read review" and "Submit review." An attacker threatens the use case "Order books" by stealing the customer's credit card information in the system. To avoid or mitigate this threat, an additional use case "Protect info." is added in this MUC. The attacker also threatens the use case "Submit review" by damaging reputation for books. To mitigate this threat, an additional use case "Screen input" is also added, as shown in the figure. Although the model in the figure is worthwhile for representing security requirements, it is not so easy for a requirements analyst to identify the threats and their countermeasures (security requirements). Experienced analysts will identify such threats because they know what kinds of assets are related to each threatened use case. For example, an analyst will focus on a review article itself while identifying a countermeasure "Screen input" because he or she notices that the article has been read by many people, and they believe the article.

Security use cases [5] is almost the same as misuse cases. The differences are as follows. First, additional use cases for avoiding attacks are explicitly called "security use cases." Second, various kinds of patterns for security use cases are provided.

A summary of our comparison of use case extensions is given in **Tables 1** and **2**. A tick (√) in a cell indicates that an extension has the corresponding feature. A blank cell means that we could not find any evidence in the literatures that the extension has the feature. For example, an abuse case has use cases, misuse cases, actors, and malicious actors. It does not have other features such as relationships among use and misuse cases.

2.4 Problems to Be Solved

As shown in Tables 1 and 2, MUC seems to be a better method than the others because it takes goals and assets into account in its analysis process. In addition, it has the following advantages. First, its model is easy for any stakeholders to understand. Second, it lets us analyse how attackers intentionally or inadvertently threaten the functionalities of a system to be developed (i.e., misuse cases or attacks) because attackers are explicitly modelled in a misuse case diagram. Third, countermeasures for mitigating the misuse cases can be naturally introduced because there are explicit relationships between misuse cases and countermeasures.

However, we think that goals and assets should be explicitly modelled for the following reasons. First, it is hard for novice requirements analysts to predict attacks and their countermeasures without knowing the asset to be attacked and the goals for its protection. Second, novice analysts tend to overlook the analysis of assets and goals even if the process recommends that they analyse them. Third, novice analysts cannot easily find typical ways to review the relationships among assets, goals, use cases and misuse cases even if they remember the need for asset and goal analysis. Fourth, modelling assets and goals explicitly lets us easily design case tools for guiding security requirements elicitation.

We cannot define proper security requirements without clarifying assets. On the other hand, asset clarification usually imposes some limitations and restrictions on the design and architecture. One research challenge is to balance these two issues during asset clarification at the requirements elicitation stage. MASG lets us clarify assets to a suitable extent for balancing them.

3. MASG: Misuse Case with Assets and Security Goals

On the basis of the problems described in the previous section, we define a MASG model and the process for developing it. **Figure 2** shows the meta-model of MASG. It is an extension of the original MUC model [23]. In Fig. 2, new elements are shown in grey, and the changed link is shown in red. We have added assets and security goals to the original model, and changed the relation between misuse cases and use cases. The meta-model is explained in detail below.

3.1 Asset-based Extension

In our diagram an asset is a first-class citizen that can be represented by its name and the stereotype <<asset>> (see the example shown in **Fig. 3**). There are two types of asset: data assets, and use case assets. A data asset can be associated with a use case, if a use case uses the asset as a data object or resource. A use case asset can be described by adding <<asset>> to use cases. Use cases themselves have some value that may be reduced by attackers, or their functionality may be used as a tool for attacking other assets. For example, a use case that provides a messaging function can be used by attackers to send spam. Such use cases should be identified as "use case assets."

A threat is denoted by a use case with the stereotype <<misuse>>. 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>>. In this

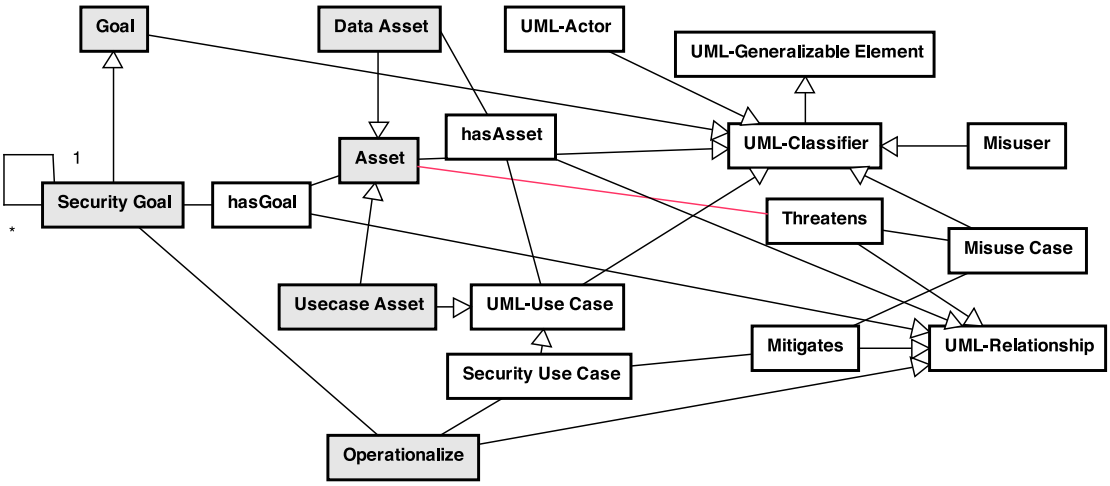

Fig. 2 MASG metamodel (extension of MUC metamodel).

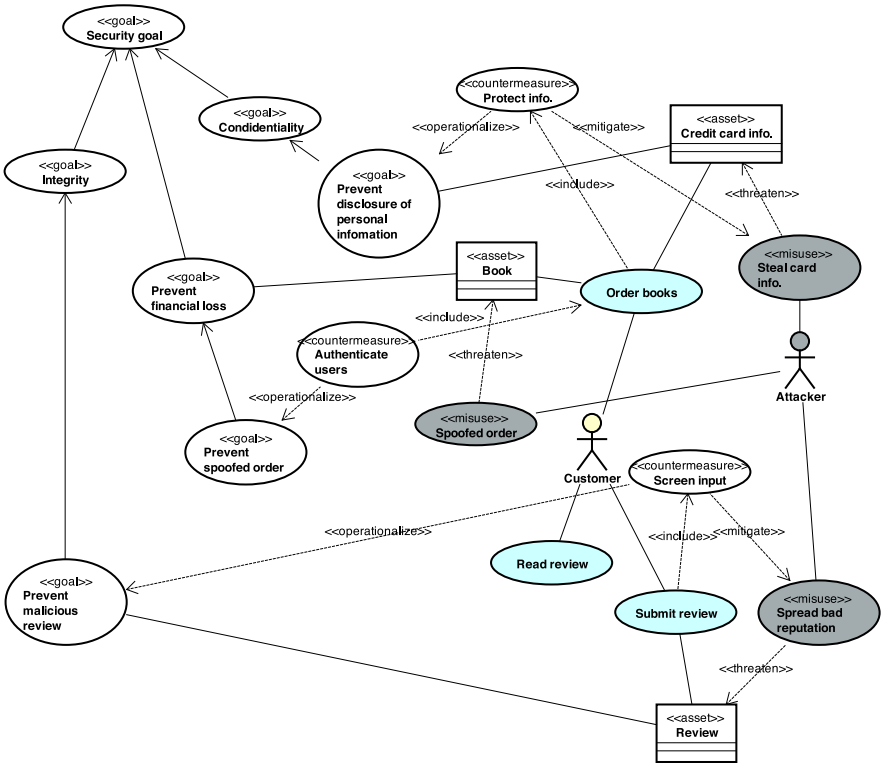

Fig. 3 Example of MASG corresponding to the MUC model in Fig. 1.

way, we can specify what should be protected (*asset*), what harms the asset (*misuse*), and finally what mitigates the threat (*countermeasure*).

3.2 Security-goal-based Extension

Our extension 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 developer has in order to secure a 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 usecase that is stereotyped by <<goal>>. Security goals may be associated with one or more other security goals. 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>>.

We explain how we elicit and analyze security requirements in the next two sections.

3.3 MASG Example

Figure 3 shows an example of the analysis result with MASG that corresponds to the MUC model in Fig. 1. As shown in Fig. 1, threats are found without the help of any syntactic elements in MUC. On the other hand, assets are explicitly modelled and related to some use cases in MASG as shown in Fig. 3. For example, an asset "Credit card info." is related to a use case "Order books," and another asset "Review" is related to use cases "Submit review" and "Read review." We can predict how such assets might be threatened more easily than how use cases might be. For example, we can easily predict that a bad reputation can be spread because the asset "Review" can be submitted and read. The left hand side of Fig. 3 shows a goal model, that helps us to protect the assets. Even if we do not know how each asset should be protected, the goal model suggests ways to protect each one. In this example, sub-goals of integrity and confidentiality are provided, and the sub-goals help us to identify countermeasures because the countermeasures are normally operationalized goals of the sub-goals.

3.4 Security Requirements Elicitation Process

In this section, we present a security requirements elicitation process. Our proposed requirements elicitation process is described in **Fig. 4** with a UML activity diagram and the process is explained step by step below.

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

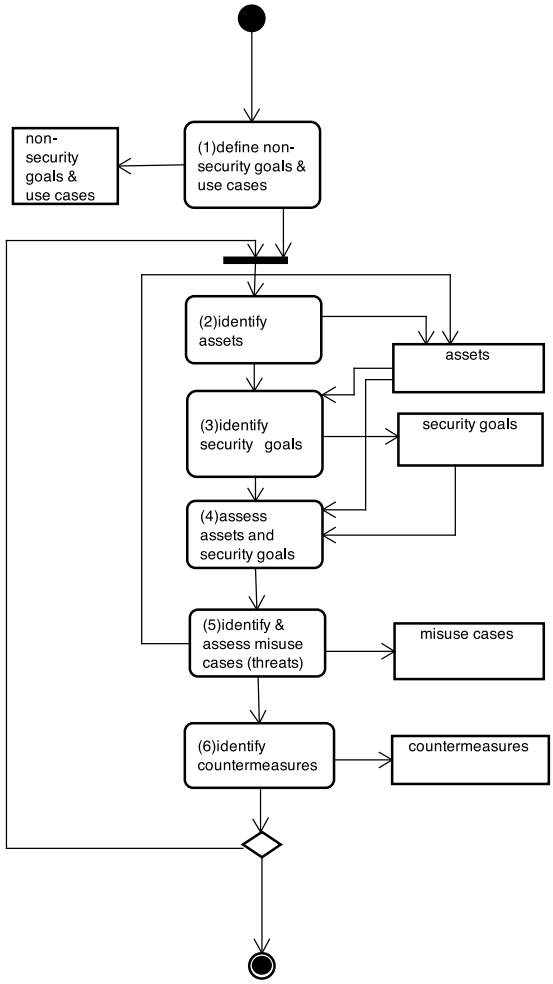

Fig. 4 Security requirement elicitation process.

(2) Identify assets (candidates).
We identify the data assets and use case assets of the target system. As emphasized in the previous sections, asset identification is a crucial step for our security requirements analysis method. The use cases identified in Step 1. and their associated data objects are identified as data assets. Some of the use cases can be identified as use case assets.
Assets related with use cases can be identified by extracting noun phrases from use case diagrams and use case description. Analysts can also use domain knowledge about the use cases. However, analysts without sufficient security knowledge might not determine whether the extracted data or use cases actually need to be protected before considering security goals and threats. Therefore at this step identified assets might be only the 'asset candidates.'

(3) Identify security goals.
This step identifies the security goals. Analysts decompose the top security goal to sub goals like the goal oriented approach. Well-known security properties – confidentiality, integrity, and availability – are automatically given as subgoals of the top level of the *security goal*.

(4) Assess assets and security goals.
Security sub-goals are identified by assessing the need for security for assets (candidates) identified in the assets iden-

tification steps. We must analyze the misuse cases that can be obstacles to the sub-goals. If a misuse case is harmful for some assets related to the assets, it must remain; if not, it should be removed. Then, we add objectives to mitigate the misuse cases against the security goals as refinements of the goals.

(5) Identify and assess misuse cases (threats).
We add the misuse cases (security threats) identified in the previous step as *misuse cases*.

(6) Identify countermeasures.
We identify the countermeasures that mitigate the threats. These countermeasures are added as *countermeasure* use cases of misuse. Then the analysts can verify that the identified countermeasures satisfy the security goals.
New assets might arise as a consequence of this step. Therefore, iteration from steps 2 to 4 is required until no further assets have been identified.

4. Our Study

We conducted an experiment to evaluate the effectiveness of MASG in comparison with MUC. The effectiveness of the assets and security goal definition in the security requirement elicitation was compared. Section 4.1 states the goal, questions, and metrics for this study. Section 4.2 describes the study hypothesis. Section 4.3 describes the participants. Section 4.4 describes the study design. Section 4.5 describes the data collection procedures. Finally, Section 4.6 describes threats to the study.

4.1 Goal, Research Questions, and Metrics

The goal of MASG is to give developers a better understanding and more secure analysis than MUC.

Although quantitative metrics are ideal, we have to take qualitative metrics because of limitations on time and subjects. We used the qualitative evaluation approach and the subjective impressions of the participants.

The following questions were aimed at achieving the overall goal.

(1) Can MASG provide more convenience in drawing, analyzing and understanding than MUC?
(2) Can MASG provide greater abilities for identifying threats / countermeasures than MUC?
(3) The main feature of the extension of MASG is related to assets and security goal. Are assets and security goals useful and essential for security requirements analysis?

4.2 Hypothesis

At first we considered comparing the analysis results using MUC with results using MASG. If the results with MASG are better than the results with MUC, we can assume that MASG has better ability of security analysis than MUC.

We also considered that if analysis results produced by inexperienced people were in the best case at almost the same level as those of security experts, then we can assume that MASG has the same or better ability of security analysis than MUC because the experts group are the professional engineers whose analysis skills are apparently higher than the inexperienced group.

4.3 Participants

In the experiment, we gave the subjects sample use cases of a software and let them elicit security requirements using MUC and MASG. We prepared the following four participants group.

We had group (A) and (C) use MUC for analysis and let group (B) and (D) use MASG.

(A) Students: 8
Graduate students in the risk engineering department. They have a little programming experience gained through university lecture courses. They also have some knowledge about general information security issues.

(B) Students: 10
Although they are same kind of people as (A), they are different people from (A).

(C) System engineers (SEs): 7
Experienced people whose business is software analysis and design. They have experience in developing several software applications. They also have some knowledge about security.

(D) System engineers (SEs): 6
Although they are same kind of people as (C), they are different people from (C).

All groups have knowledge of UML.

4.4 Study Design

The software application sample for the experiment was a web shopping site such as ™Amazon [*1]. The procedure of the experiment was as follow:

(1) Lecture on security requirements analysis
To give knowledge about security requirements analysis method of the minimum requirement, we gave the same 90-minute lecture on security requirements analysis to both groups. We introduced MUC notation and analysis procedure described in Ref. [23].

(2) Lecture on MASG (for group (B) and (D) only)
We gave a lecture on the notation and analysis procedure for MASG to group (B) and (D) only.

(3) Analysis
We gave the subjects the use case model sample shown in **Fig. 5** and let them perform a security requirements analysis within 100 minutes. A UML modeling tool [*2] was given for drawing MUC and MASG diagrams. MUC diagrams and MUC descriptions are required as the output for group (A) and (C), and MASG diagrams and MASG descriptions for group (B) and (D). Subjects of both groups were also required to answer the questionnaire on MUC/MASG.

4.5 Data Collection

The use case sample used in the study is shown in Fig. 5.

Subjects of both groups were also required to answer the following questions.

(1) Are MUC (MASG for group (B), (D)) diagrams easy to learn?
(2) Are MUC (MASG for group (B), (D)) diagrams easy to

[*1] http://www.amazon.com/
[*2] ™astah* professional (http://astah.net/editions/professional)

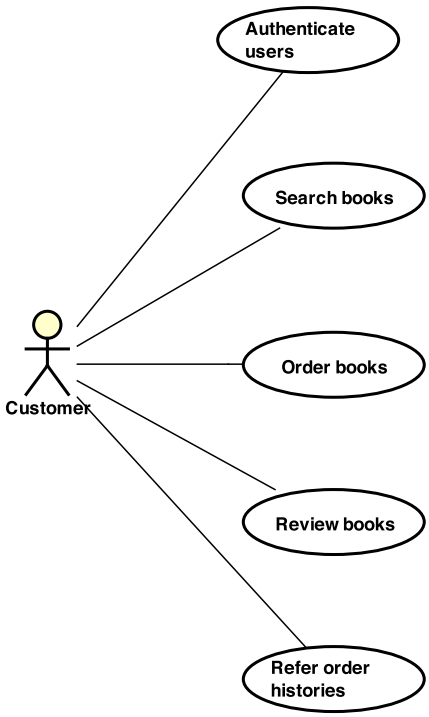

Fig. 5 Sample application.

draw?
(3) Are the analysis results with MUC (MASG for group (B), (D)) diagrams easy to be understood?
(4) Are MUC (MASG for group (B), (D)) diagrams easy to analyze?
(5) Is asset identification useful for analysis?
(6) Is security goal identification useful for analysis?
(7) Do you think that you could identify all the threats with MUC (MASG for group (B), (D))?
(8) Do you think that you could identify threats precisely with MUC (MASG for group (B), (D))?
(9) Do you think that you could identify all the countermeasures with MUC (MASG for group (B), (D))?
(10) Do you think that you could identify countermeasures precisely with MUC (MASG for group (B), (D))?

4.6 Threats to Validity

There are various types of threats to internal validity, external validity, construct validity and conclusion validity [12]. Here, we focus on only those considered relevant to our study.

4.6.1 Internal Validity

Internal validity deals with whether we can infer that a relationship between two variables is causal, and not due to any confounding factors [12]. Here, we discuss only risks that applied to our study and the procedures that we used to contain them.

Maturation People change over time, such as during the course of an experiment or even between measurements, so we had to use each subject for only one testing.

Testing Testing refers to any change in the second administration of a test resulting from the subject of having taken the test [12] previously. We prepared two testing types: MUC and MASG. Since the two methods are similar, if we had had the same participants take both the MUC and MASG tests, the results of the earlier test might have led to improved analytical skill and domain knowledge about the target software. Although we chose the designs for the two groups, there was a risk of some participants having experience with MUC or MASG.

4.6.2 External Validity

External means the extent to which a study's results can be generalized to other situations and other people [12].

Population validity Our study presumed that if MASG is useful for inexperienced students, it is also useful for experienced people. However we could not collect data about MASG usage by experienced people to validate this premise because of the limitation on the participants.

Ecological validity We also tested MUC and MASG for only one case application. There is a risk if the results of our study are scalable with the number of use cases.

4.6.3 Construct Validity

Construct validity is the extent to which what was to be measured was actually measured [12]. In the study, we measured the numbers of identified threats, countermeasures and relations between them. Although our aim was to measure the quality of the analysis, the measured number may not indicate the quality, because it depends on the refinement level. For example, some people may identify more similar threats than others. To handle such a risk, we also examined not only the quantity but also the quality of the analysis results.

4.6.4 Conclusion Validity

Statistical conclusion validity refers to the ability to make an accurate assessment about whether the independent and dependent variables are related and about the strength of that relationship [12]. So the two key questions here are 1) Are the variables related? and 2) If so, how strong is the relationship? In order to answer the first question, we performed a null hypothesis testing procedure stating that any observed relationship is probably nothing more than normal sampling error or fluctuation. Then, we performed a t-test to examine whether there was any hypothetical significance (see Section 5.2.1).

5. Data Analysis, Results, and Interpretations

5.1 Data Analysis

The analysis results produced by 4 groups are shown in **Table 3**. The average value, maximum value and value determined by the experts for identified threats, identified countermeasures, and relations between threats and countermeasures are shown.

Table 4 indicates answers to the questionnaire mentioned above. Since some of the participants did not answer questions, the total number of each question is equal to or smaller than the number of participants.

5.2 Results and Interpretations

We now describe the results of the data analysis that was performed and our interpretation of them.

5.2.1 Statistical Hypothesis Testing

We performed a t-test on the number of threats, countermeasures and threat-countermeasure links for pairs of (A)-(B), (C)-(D), (A)-(C), (B)-(D), (B)-(C) and (A)-(D). The results are indi-

Table 3 Analysis results.

-	Identified threats	identified countermeasures	threat-countermeasure [*3]	total time
Average ((A)MUC/student)	5.63	1.13	1.63	70.0
Max ((A)MUC/student)	7	5	5	-
Average ((B)MUC/SE)	7.00	4.29	5.29	55.0
Max ((B)MUC/SE)	11	7	9	-
Expert (MUC)	6	3	6	-
Average ((C)MASG/student)	5.30	3.00	3.50	63.3
Max ((C)MASG/student)	11	5	5	-
Average ((D)MASG/SE)	5.83	3.83	5.17	52.0
Max ((D)MASG/SE)	8	8	8	-
Expert (MASG)	7	3	7	-

Table 4 Answers to the questionaire.

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Yes (MUC)	10	8	8	6	7	8	1	5	3	6
No (MUC)	1	3	3	5	4	2	10	6	8	5
Yes (MASG)	12	8	6	10	12	11	2	3	5	4
No (MASG)	3	6	8	5	3	4	13	12	10	11

Table 5 t-test results.

pair	test	threats	countermeasures	threats-countermeasures
(A)-(C)	f-test	0.092	0.855	0.593
	t-test	0.706	**0.031**	0.112
(B)-(D)	f-test	0.219	0.707	0.596
	t-test	0.359	0.739	0.933
(A)-(B)	f-test	0.054	0.325	0.470
	t-test	0.240	**0.014**	**0.012**
(C)-(D)	f-test	0.333	0.455	0.727
	t-test	0.623	0.393	0.202
(B)-(C)	f-test	0.687	0.216	0.803
	t-test	0.179	0.223	0.189
(A)-(D)	f-test	0.584	0.586	0.900
	t-test	0.774	**0.022**	**0.008**

cated in **Table 5**.

(c1) Comparing MUC with MASG ((A)-(C), (B)-(D)) (A)-(C) and (B)-(D) are the comparison between MUC and MASG analysed by people with the similar skill level and experiences. The significant difference appears at the number of identified countermeasures between student group ((A)-(C)).

(c2) Comparing students with SEs ((A)-(B), (C)-(D)) (A)-(B) and (C)-(D) are the comparison of the same methods performed by groups with different skill levels and experiences. According to the t-test results, the significant differences appear at the number of identified countermeasures and threat-countermeasure links for the group using MUC((A)-(B)). On the other hand, there are no significant differences for the group using MASG ((C)-(D)).

(c3) Comparing MASG/students with MUC/SEs ((B)-(C)) According to the t-test result, there is no significant difference. The results indicate that the analysis results obtained by inexperienced people are at almost the same level as those obtained by the security experts. This finding is concordant with our hypothesis.

(c4) Comparing MUC/students with MASG/SEs ((A)-(D)) It is the opposite comparison of (c3). The t-test result indicates that the significant differences appear at the number of identified countermeasures and the number of threat-countermeasure links.

5.2.2 Qualitative Evaluation

The application sample contains some threats that are difficult to identify for a person who has little knowledge or experience of requirements analysis such as those in group (C) since they require consideration of assets first. ‘Review with malicious intent’ and ‘Steal card info’ are such threats. Since ‘Card info’ does not appear in the use case ‘Order books’, it must be identified as an asset first. These threats were identified not only by group (B), (D) but also by several persons in group (C). The results indicate that MASG is effective for assisting people with less knowledge to elicit security requirements by identifying assets and security goals.

As Sindre et al. [23] state, the identification of threats and security goals is important in security requirements analysis. The answers to the questionnaire show that both groups believe that identifying threats and security goals is useful for analysis.

[*3] relation between threat and countermeasure

With respect to usability, MUC and MASG are rated high in drawability and understandability (see Table 4). However, according to the questionnaire answers, some subjects thought that both methods were difficult to understand and use for analysis because of their complexity. The number of use cases in the application sample is 5. Some subjects were afraid that if the number of use cases increases, the diagrams might become complicated which would affect visibility and drawability. MUC and MASG need simplified tool support such as filtering by use case or by asset.

With respect to comprehensiveness, few participants thought that the only use of MUC or MASG was to achieve the comprehensive analysis (see Table 4). Assuring the comprehensiveness requires group work using enough time and mutual verification since the quality of security analysis with MUC or MASG depends on human expertise about assets, threats, and countermeasures. Each experiment was done by a single person with limited time, so the answer indicates that the subjects understood the characteristics of both methods correctly. With respect to the correctness of analysis, more subjects considered that correctness was achieved with both methods. Some subjects of group (C) stated that the reason that correctness was not achieved is the lack of the experience of the subjects themself.

5.3 Summary of Results

The quantitative results (c1) and qualitative results indicate that MASG is thoroughly more effective than MUC for analysts with insufficient skills and experiences. Moreover, the quantitative results (c2)(c3) and qualitative results indicate that MASG is usable for inexperienced people in that they can elicit security requirements that are almost equivalent to those produced by people having expertise with MUC.

The questionnaire answers indicate the importance of identifying assets and security goals, which is a feature of MASG model.

The questionnaire results also indicate that there was a concern about scalability for both MUC and MASG. There is a need for tool support with functions filtered by use cases, assets, threats, and countermeasures.

5.4 Discussion

This section describes discussion on the result, comparison with other methods, and limitation / drawbacks of the proposed method.

5.4.1 Discussion

There were no significant difference found at the number of identified threats between MUC and MASG. We assume one of the reasons is that the sample application like “Amazon.com” is so well-known and familiar to students, so it may be easy to find threats. However, we can find differences at other elements (countermeasures and threat-countermeasure links). With the proposed analysis steps, analysts usually identify threats at first, and then identify countermeasures that mitigate threats. According to the result, we can estimate that MASG helps more effective countermeasure identification, or more effective analysis in total.

5.4.2 Comparison with Other Works

Haley et al. propose a security requirements engineering framework [7] with similar structure and activity model to our method which contains assets and security goals. However, it proposes the **framework**, or meta-model and activity of security requirements analysis. It is insufficient for analysts to elicit security requirements of actual software development, since it needs practical tools for threat analysis and a common language presenting assets, goals, threats and countermeasure for sharing requirements with stakeholders. Our MASG method provides a model (not meta-model) and steps as a threat analysis tool for analysts and UML-like language for sharing requirements with stakeholders.

5.4.3 Limitations and Drawbacks

The target application sample for the experiment contains five use cases. Some answer comments of the questionnaire pointed that it may become complicated when the number of use cases increases. Although the tendency is the same as MUC, one MASG diagram contains more elements than the MUC diagram presenting the same use cases. Scalability may be the drawback of MASG, because of the difficulty of comprehensiveness of mitigation. We consider tool support is essential for applying MASG to large scale systems. Use case diagrams can be decomposed to multiple layers. If MASG can be multi-layered, we can improve the readability of MASG.

MASG does not treat the following issues. They are out of scope of the paper.

- The way to identify assets from use cases
- Detail threat decomposition like threat modeling [8], [25]
- Threat assessment method

However, they are essential elements for security requirements elicitation. We propose an elicitation process combining MASG and threat analysis / assessment tool like threat modeling to complete the analysis work. At first, analysts identify the root threats with MASG. Then they decompose the threats to detail threats as attacks and conditions using threat analysis tools such as the threat tree. After that they assess each decomposed threats with tools such as DREAD.

We don’t think the proposed method is the perfect solution for inexperienced people. It is insufficient to identify potential threats, assess assets and goals and determine countermeasures. It is one of the limitations of our method. We consider introduction of security patterns with our model that will be the solution for the security knowledge issue.

6. Conclusion

In this paper, we proposed an extension for misuse case diagrams by incorporating some new model elements, *assets* and *security goals* for eliciting and analysing the security features of a computing system. We illustrated the extended model MASG with a process that helps us to analyse the requirements related to security goals, assets, threats, and security countermeasures. Moreover, we showed the effectiveness from the quantitative and qualitative results of a case study. In other words, MASG is usable for inexperienced people because they could elicit almost equivalent security requirements to people experienced with

MUC.

Future work includes further extensions of the diagrams with tool support. Security consists of features like confidentiality, availability, integrity, and accountability. These features may sometimes have interactions that 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 analyse any conflicts in the security features and how to reach a compromise among the security goals.

References

[1] Bishop, M.: *Computer security*: *Art and science*, Addison-Wesley, Pearson Education, Inc. (2003).
[2] Brændeland, G. and Stølen, K.: Using model-based security analysis in component-oriented system development, *QoP*, pp.11–18 (2006).
[3] Chung, L., Nixon, B., Yu, E. and Mylopoulos, J.: *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers (1999).
[4] Cockburn, A.: *Writing Effective Use Cases*, Addison-Wesley (2000).
[5] Firesmith, D.: Security Use Cases, *Journal of Object Technology*, Vol.2, No.1, pp.53–64 (2003).
[6] Gollmann, D.: *Computer Security*, John Wiley & Sons (1999).
[7] Haley, C.B., Laney, R.C., Moffett, J.D. and Nuseibeh, B.: Security Requirements Engineering: A Framework for Representation and Analysis, *IEEE Trans. Software Eng.*, Vol.34, No.1, pp.133–153 (2008).
[8] Howard, M. and Lipner, S.: *The Security Development Lifecycle*, Microsoft (2006).
[9] International Standard: ISO/IEC 27002 Information technology – Security techniques – Code of practice for information security management (2005).
[10] Jaatun, M.G. and Tøndel, I.A.: Covering Your Assets in Software Engineering, *ARES*, pp.1172–1179 (2008).
[11] Jacobson, I., Christerson, M., Jonsson, P. and Overgaard, G.: *Object-Oriented Software Engineering – A Use Case Driven Approach*, Addison Wesley (1992).
[12] Johnson, R.B. and Christensan, L.B.: *Educational Research: Quantitative, Qualitative and Mixed Approaches*, Sage Publications, Inc (2010).
[13] Liu, L., Yu, E. and Mylopoulos, J.: Security and Privacy Requirements Analysis within a Social Setting, *International Conference on Requirements Engineering* (*RE 2003*), pp.151–161, IEEE (2003).
[14] Long, T., Liu, L., Yu, Y. and Jin, Z.: AVT Vector: A Quantitative Security Requirements Evaluation Approach Based on Assets, Vulnerabilities and Trustworthiness of Environment, *RE*, pp.377–378 (2009).
[15] Marino, B.D.R. and Haddad, H.M.: Asset Assessment in Web Applications, *ITNG*, pp.762–767 (2010).
[16] Marino, B.D.R., Haddad, H.M. and Molero, A.J.E.: A Methodological Tool for Asset Identification in Web Applications: Security Risk Assessment, *ICSEA*, pp.413–418 (2009).
[17] Massacci, F., Mylopoulos, J. and Zannone, N.: Computer-aided Support for Secure Tropos, *Automated Software Engineering*, Vol.14, No.3, pp.341–364 (2007).
[18] McDermott, J.P. and Fox, C.: Using Abuse Case Models for Security Requirements Analysis, *ACSAC*, pp.55–64 (1999).
[19] Mouratidis, H. and Giorgini, P.: Secure Tropos: A Security-oriented Extension of the Tropos Methodology, *International Journal of Software Engineering and Knowledge Engineering*, Vol.17, No.2, pp.285–309 (2007).
[20] Okubo, T., Taguchi, K. and Yoshioka, N.: Misuse cases + Assets + Security Goals, *Workshop on Software Security Process* (*SSP09*), IEEE, pp.424–429 (2009).
[21] Saeki, M. and Kaiya, H.: Security Requirements Elicitation Using Method Weaving and Common Criteria, *MoDELS Workshops*, pp.185–196 (2008).
[22] Sindre, G. and Opdahl, A.L.: Eliciting Security Requirements by Misuse Cases, *TOOLS* (*37*), pp.120–131 (2000).
[23] Sindre, G. and Opdahl, A.L.: Eliciting security requirements with misuse cases, *Requir. Eng.*, Vol.10, No.1, pp.34–44 (2005).
[24] Supaporn, K., Prompoon, N. and Rojkangsadan, T.: Enterprise Assets Security Requirements Construction from ESRMG Grammar based on Security Patterns, *APSEC*, pp.112–119 (2007).
[25] Swiderski, F. and Snyder, W.: *Threat Modeling*, Microsoft Press (2004).
[26] Taguchi, K., Yoshioka, N., Tobita, T. and Kaneko, H.: Aligning Security Requirements and Security Assurance Using the Common Criteria, *SSIRI*, pp.69–77 (2010).
[27] van Lamsweerde, A.: Elaborating Security Requirements by Construction of Intentional Anti-Models, *International Conference on Software Engineering* (*ICSE 2004*), pp.148–157, IEEE (2004).
[28] van Lamsweerde, A.: *Requirements Engineering: From System Goals to UML Models to Software Specifications*, Wiley (2009).
[29] Yu, E., Giorgini, P., Maiden, N. and Mylopoulos, J.: *Social Modeling for Requirements Engineering*, The MIT Press (2011).

Takao Okubo was born in 1966. He received his M.E. degree from Tokyo Institute of Technology in 1991 and had been engaged in the Fujitsu Laboratories limited since 1991. He received his Ph.D. degree from Institute of Information Security in 2009. He has been an associate professor of Institute of Information Security since 2013. His research interest is security software engineering. He is a member of IEEE and IPSJ.

Kenji Taguchi is an invited researcher of National Institute of Advanced Industrial Science and Technology (AIST) whose expertise lies in system assurance for safety critical/high integrity and secure systems. He has been mainly working on formal methods and system assurance, and has served several software engineering/formal methods/system assurance conferences/workshops as program committee member or program chair. He is a co-founder of the Integrated Formal Methods (IFM) international conference series and was a program co-chair of International Conference on Formal Engineering Methods (ICFEM) 2012.

Haruhiko Kaiya is a professor in Department of Information Sciences, Kanagawa University, Japan.

Nobukazu Yoshioka is a researcher at the National Institute of Informatics, Japan. He received his B.E. degree in Electronic and Information Engineering from Toyama University in 1993. He received his M.E. and Ph.D. degrees in School of Information Science from Japan Advanced Institute of Science and Technology in 1995 and 1998, respectively. From 1998 to 2002, he was with Toshiba Corporation, Japan. From 2002 to 2004 he was a researcher, and since August 2004, he has been an associate professor, in National Institute of Informatics, Japan. His research interests include Security and Privacy Software Engineering, Cloud computing, Agent Technology, object-oriented methodology, software engineering, and software evolution. He is a member of the Information Processing Society of Japan (IPSJ), the Institute of Electronics, information and Communication Engineers (IEICE) and Japan Society for Software Science and Technology (JSSST). He has been a board member of JSSST since 2012.