Threat and Security Modeling for Secure Software Requirements and Architecture

Michael Shin, Don Pathirage, Dongsoo Jang
Department of Computer Science, Texas Tech University
Lubbock, Texas, USA
michael.shin@ttu.edu, don.pathirage@ttu.edu, dongsoo.jang@ttu.edu

Abstract – Most of the threat modeling approaches do not stipulate when and what types of threats should be identified and modeled in each software development phase. This paper addresses a threat and security modeling approach in software requirements and architecture. The threats to software systems are classified and modeled as input and output, class and message threats in software requirements, and message communication threats in software architecture so that the security countermeasures are modeled and designed against the threats. The modeling of threats and security countermeasures is described by means of the underlying meta-models of software requirements and software architecture models. An online shopping system is used to demonstrate the approach.

Keywords – Threat; Security Modeling; Meta-Model; Software Requirements; Software Architecture.

I. INTRODUCTION

The identifying and modeling of threats to software systems are the starting point of developing secure systems that protect sensitive assets from the threats. A secure system is specified and designed together with security countermeasures to neutralize the threats. A secure system enables to defeat only the attacks caused by the identified and modeled threats. A security-sensitive system can be vulnerable to attacks if any critical threats are not identified and modeled in software development.

Several threat modeling approaches [Schneier99, Abi-Antoun06, Torr05, Sindre05, McDermott99, Srivatanakul05, Lund11, Shevchenko18] have been suggested to model the threats to software systems. However, most of the threat modeling approaches do not stipulate when and what types of threats should be identified and modeled in software development. The approaches might overlook the threats unnoticed because they may require all threats to be identified in only one development phase (e.g., requirements specification or design).

This paper aims at developing a threat and security modeling approach that describes when and what types of threats are identified and modeled in software development as well as the modeling of security countermeasures against the threats. The approach to modeling threats and security is described by means of the meta-model, which is a model of software system models.

DOI reference number: 10.18293/SEKE2020-055

II. RELATED WORK

Threat Modeling. Threats in a system have been modeled by several approaches, which include attack trees [Schneier99], data flow diagrams [Abi-Antoun06, Torr05], and UML-based modeling [Sindre05, McDermott99, Srivatanakul05]. Attack trees [Schneier99] provide an approach to modeling and analyzing the threats of systems, and the threats are analyzed in terms of attacker's capabilities. The design models in the research [Abi-Antoun06, Torr05] are specified with data flow diagram, and the threats to the models are identified and analyzed using scenarios of each function in a system. Several threat modeling approaches, such as misuse cases [Sindre05], abuse cases [McDermott99], and HAZOP (Hazard and Operability Analysis) [Srivatanakul05], have been developed for object-oriented software systems. The approaches model threats using the use case model in UML and capture security requirements against the threats. The CORAS in [Lund11] is a model-driven risk analysis that assesses the risks of assets in the system. The authors in [Shevchenko18] describes threat modeling methods that target different parts of the development process.

Secure Software Development. The studies in [Lodderstedt02] proposed a new modeling language based on UML for the model-driven development of secure distributed systems. The research in [Turpe17] describes the interplay of three dimensions: threats, security goals, and system design. Security patterns in [Fernandez13, Schumacher06] address the broad range of security issues that should be considered in the stages of software development lifecycle.

In earlier work by a coauthor in [Gomaa04], an approach has been described to model complex applications by modeling security requirements and designs separately from application requirements and designs using the UML notation. In later work by the coauthor in [Shin07], an approach has been described for modeling the evolution of a non-secure application to a secure application in terms of a requirements model and a software architecture. The recent work of the coauthors proposes the design of reusable secure connectors [Shin17a] and describes security failure-tolerant requirements specification and analysis [Shin17b, Shin18].

III. THREATS IN SOFTWARE REQUIREMENTS AND ARCHITECTURE MODELS

The threats to systems are determined by considering the sensitive assets that are described in software requirements and

architecture. Fig. 1 depicts the threats to sensitive assets in software requirements and software architecture. The threats to input and output (I/O) are induced by an actor's interactions with a system, which are described in software requirements specification that is modeled in this paper by means of the use case model in the unified model language (UML) [Booch05, Rumbaugh04, Gomaa11]. An actor's sensitive input to and output from a system can be compromised, whereas an actor's untrusted input can compromise the system.

The threats to classes occur when the objects of classes in the static model for requirements analysis are engaged in processing or storing sensitive data. The static model is used in UML to depict the static structural aspects of a system by defining the classes in the system, their attributes, and relationships between classes.

A sensitive message passed between objects in the communication model in requirements analysis can be threatened. The dynamic aspect of a system is captured in the communication model of UML through objects and the messages passed between objects. The sensitive messages passed between objects can be threatened.

The messages communicated between components in software architectures can be compromised when the message communication between components are insecure. A message in software architectures is sent by a component to another via a connector that encapsulates the detail of message communication. Insecure connectors can breach the sensitive messages communicated between components.

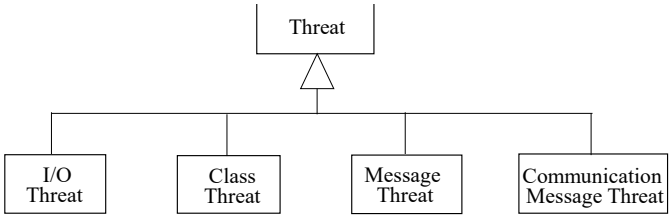

Fig. 1 Threats in software requirements and architecture

IV. Threat and Security Modeling in Requirements Specification

The threats and security in requirements specification are modeled by extending the underlying meta-model [Gomaa08] of the use case model, which is depicted in Fig. 2 in which the threat and security each are modeled as meta-classes. A meta-model is described with the meta-classes and relationships between the meta-classes. The meta-classes associated with threat and security are highlighted in gray in Fig. 2.

The threats of requirements specification concentrate on the threats to I/O that can be caused by the interaction between actors and the system. An actor's input to the system can contain sensitive data, such as a user's credit card number, which needs to be protected. On the other hand, an attacker can exploit a system with a suspicious input to the system, so the input must be verified. Some actor's access to a system must be authenticated and/or authorized in order not to disclose sensitive data in the system.

The threats to I/O are identified by means of analyzing the use case description that describes an actor's input to the system and the system's responses to the actor. The threat is described [Shin17b, Shin18b] in terms of threat name, threat type, threat point, security asset, description, and security need.

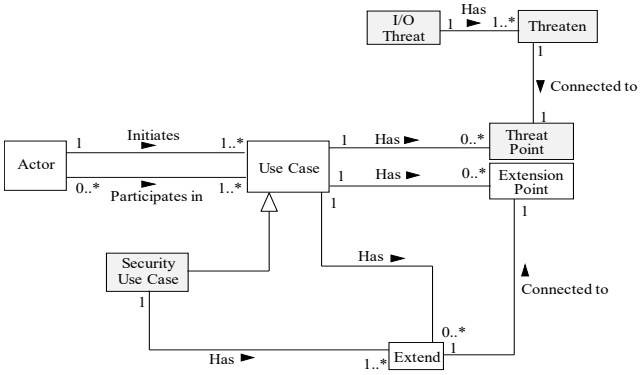

Fig. 2 Meta-Model of Use Case Model with threat and security

A threat to I/O is modeled together with a use case (Fig. 2) in which the threat is represented using the use case notation. Fig. 3 depicts an *unauthenticated ID* input threat, which threatens the *make order request* use case at the order request threat point. The *make order request* use case is an application use case in an online shopping system [Gomaa11], verifying a customer credit card payment and processes a customer's order request.

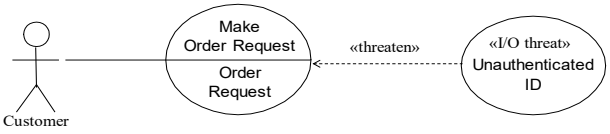

Fig. 3 Threat to Make Order Request use case

A security countermeasure against an I/O threat is specified using a security use case [Gomaa04], separately from application use cases (Fig. 2). When a system requires a security countermeasure, the security use case extends the application use case at an extension point. The *check account password* security use case (Fig. 4) is taken for the *make order request* use case against the *unauthenticated ID* input threat (Fig. 3).

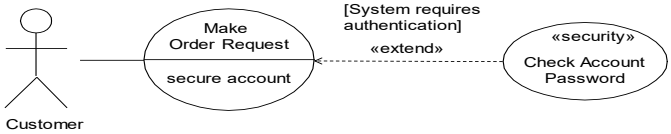

Fig. 4 Security use case for Make Order Request use case

V. Threat and Security Modeling in Requirements Analysis

A. Class Model

The threats and security in the static model for software requirements analysis are described in the underlying meta-model [Gomaa08] of the class model (Fig. 5b). The threats to classes are described as the class threat meta-class that threatens the class meta-class through the threaten meta-class. The security countermeasures are described as the security class meta-class that is specialized from the class meta-class. The

class meta-class supports the use case meta-class in the use case meta-model (Fig. 5a), which needs classes to implement the functionality of the use case.

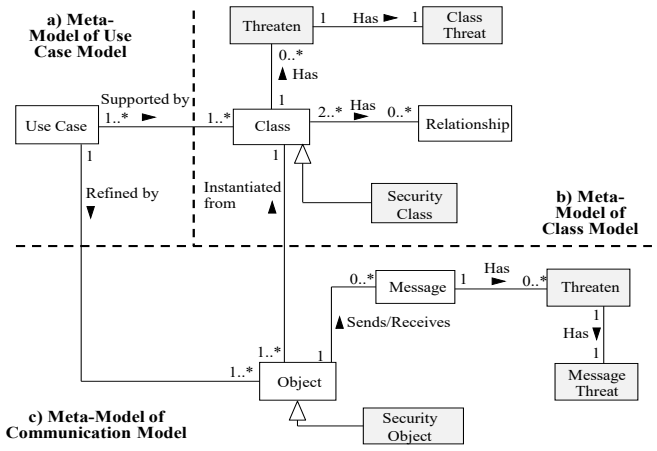

Fig. 5 Meta-Model of Requirements Analysis

The class threats are identified by considering the role of each class, such as interface, entity, control, or application logic [Gomaa11]. An interface class that interfaces to and interacts with an actor must be secure to receive sensitive input from or send sensitive output to the actor. An entity class that stores data might have sensitive data (e.g., patient medical record) that requires security. A control class that contains the system's sensitive state information or coordination logic might be compromised. An application logic class that contains the details specific to applications might be tampered with.

Part of the static model for *make order request* use case (Fig. 4) is depicted in Fig. 6a using the class diagram, which includes the *Customer Account* class that stores customer's account data in the *make order request* use case (Fig. 4), and *Account Password Checker* security class that verifies a customer's account password in the *check account password* security use case (Fig. 4). A credit card disclosure class threat (Fig. 6b) is identified and modeled for the *make order request* use case. Also, the security countermeasure against the credit card class disclosure threat is modeled with *Encryption* and *Decryption* security classes (Fig. 6a). Fig. 6b depicts the class threat description for credit card disclosure class threat.

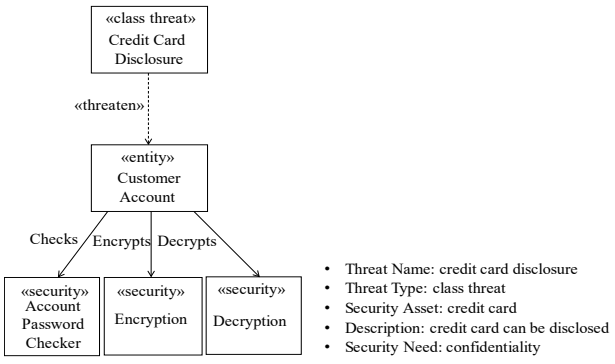

Fig. 6 Credit Card Disclosure class threat and its description

B. Communication Model

The meta-model [Gomaa08] of the communication diagram (Fig. 5c) describes the threats to messages passed between objects and the security against the threats. The security object meta-class is specialized from the object meta-class (Fig. 5c), which is utilized to refine the use case meta-class in the use case model (Fig. 5a).

The message threats in the communication model can be the threats to confidentiality, integrity, or non-repudiation security of messages from the application perspective. When a sensitive message is sent by an object to another, the message must be confidential or must not be tampered with. Also, non-repudiation may be required to prove the message presence between objects later. The message threats are determined by examining the messages passed between objects on the communication diagrams. The security objects against the message threats are determined and incorporated into the communication diagrams.

The *Order Request Repudiation* message threat is modeled in Fig. 7a in which it threatens the *order request* message in the *make order request* use case when the *Customer Interface* object sends the *order request* message to the *Delivery Order* entity object. Fig. 7b depicts the message threat description for the *Order Request Repudiation* message threat. As the *Order Request Repudiation* message threat is identified, a digital signature is taken as a security countermeasure, which is realized by means of the *Digital Signature Generator* security class (Fig. 7a), and the *Digital Signature Verifier* security class (Fig. 7a).

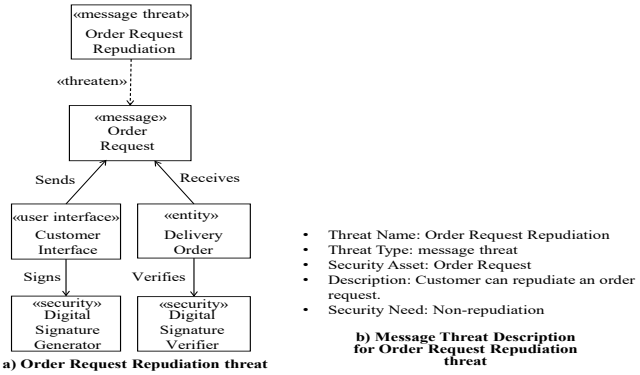

Fig. 7 Order Request Repudiation message threat and its Description

VI. Threat and Security Modeling in Software Architecture

The threats and security in software architecture are described in the underlying meta-model of software architecture model (Fig. 8b) in which a component sends a message to or receives it from another through a connector. The message communication threat meta-class threatens the message meta-class between components. The secure connector meta-class is specialized from the connector meta-class. The component meta-class consists of object meta-class in the meta-model of communication model (Figs. 8a and 5c).

The message communication threats are identified in terms of integrity, confidentiality, non-repudiation, authentication and authorization from the software architecture perspective.

The sensitive messages communicated between components can be tampered with or can be disclosed. Fig. 9a depicts a message communication threat to the *order request* message sent by a Customer component to a Delivery Order component in which the order request can maliciously be changed. Fig. 9b depicts the message communication threat description for the *order request* message.

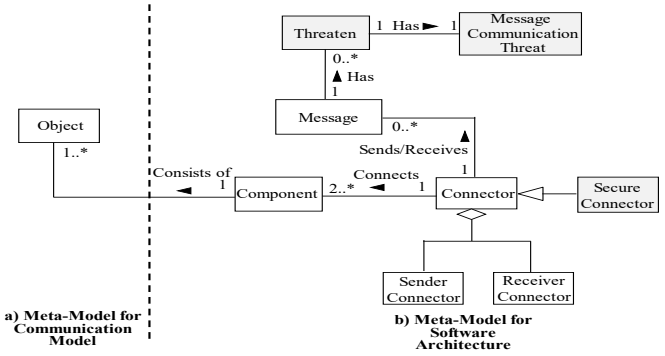

Fig. 8 Meta-Model of Software Architecture

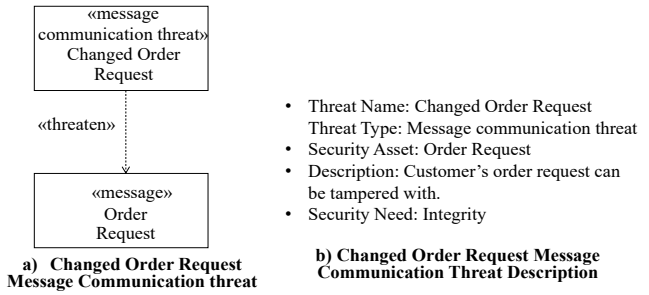

Fig. 9 Changed Order Request communication message threat

For countering the message communication threats, a secure connector [Shin17a, Shin18a] is designed by separately considering the message communication pattern and the security patterns required by application components. A secure connector (Fig. 8) is a distributed connector, which consists of a secure sender connector for a sender application component, and a secure receiver connector for a receiver application component.

VII. Conclusions and Future Work

This paper describes a threat and security modeling approach for software requirements and architecture, which is described by means of the meta-model of the use case model, static model, communication model, and software architectural model. The threats are classified and modeled as I/O threat, class threat, message threat and message communication threat. The security countermeasures are modeled as a security use case for an application use case, a security class for an application class, a security object for an application object, and a secure connector for a simple connector.

This paragraph describes future research for the threat and security modeling of software requirements and architectures. A way of future research could be designing security fault-tolerant secure connectors, which tolerate the breaches of software architectures.

References

[Abi-Antoun06] M. Abi-Antoun, D. Wang and P. Torr, "Checking Threat Modeling Data Flow Diagrams for Implementation Conformance and Security", International Conference on Automated Software Engineering, pp. 392-396, Atlanta, Georgia, USA, November 6-9, 2007.

[Booch05] G. Booch, J. Rumbaugh, and I. Jacobson, "The Unified Modeling Language User Guide", 2nd Edition, Addison Wesley, Reading MA, 2005.

[Fernandez13] Fernandez, E. B., 2013. Security Patterns in Practice, Wiley.

[Gomaa11] H. Gomaa, "Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures", Cambridge University Press, 2011.

[Gomaa04] H. Gomaa and M. E. Shin, "Modeling Complex Systems by Separating Application and Security Concerns" 9th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2004), Italy, April 2004.

[Gomaa08] H. Gomaa and M. E. Shin, "Multiple-View Modeling and Meta-Modeling of Software Product Lines," IET Software, Volume 2, Issue 2, April 2008, pp. 94 – 122.

[Gomaa11] H. Gomaa, "Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures", Cambridge University Press, 2011.

[Lodderstedt02] T. Lodderstedt, D. Basin, J. Doser, "SecureUML: A UML-Based Modeling Language for Model-Driven Security", Fifth International Conference on the Unified Modeling Language, pp. 426-441, Dresden, Germany, September 30 – October 4, 2002.

[Lund11] M. S. Lund, B. Solhaug, and K. Stølen, "Model-Driven Risk Analysis", Springer Berlin Heidelberg, 2011.

[McDermott99] J. McDermott and C. Fox, "Using Abuse Case Models for Security Requirements Analysis," In Proceedings of 15th Annual Computer Security Applications Conference (ACSAC`99), pp. 55-64, Phoenix, Arizona, December 1999.

[Rumbaugh04] J. Rumbaugh, G. Booch, and I. Jacobson, "The Unified Modeling Language Reference Manual", 2nd Edition Addison Wesley, Reading MA, 2004.

[Schneier99] B. Schneier, "Attack trees: Modeling security threats," Dr. Dobbs Journal, pages 21–29, December 1999.

[Schumacher06] Schumacher, M., Fernandez, E. B., Hybertson, D., Buschmann, F., Sommerlad, P., 2006. Security Patterns, Wiley.

[Shevchenko18] N. Shevchenko, T. A. Chick, P. O'Riordan, T. P. Scanlon, and C. Woody, "Threat Modeling: A Summary of Available Methods", SEI/Carnegie Mellon University, July 2018.

[Shin07] M. E. Shin and H. Gomaa, "Software Modeling of Evolution to a Secure Application: From Requirements Model to Software Architecture," Science of Computer Programming, Volume 66, Issue 1, pp. 60-70, April 2007.

[Shin17a] M. E. Shin, H. Gomaa, and D. Pathirage, "Model-based Design of Reusable Secure Connectors," 4th International Workshop on Interplay of Model-Driven and Component-Based Software Engineering (ModComp2017), September 17, Austin/Texas, USA, 2017.

[Shin17b] M. E. Shin and D. Pathirage, "Security Requirements for Tolerating Security Failures," 29th International Conference on Software Engineering and Knowledge Engineering, Pittsburgh, USA, July 5-7, 2017.

[Shin18] M. Shin, D. Pathirage, and D. Jang, "Analysis of Security Failure-Tolerant Requirements", The 30th International Conference on Software Engineering and Knowledge Engineering (SEKE2018), San Francisco Bay, California, USA, July 1-3, 2018.

[Sindre05] G. Sindre and L. Opdahl, "Eliciting Security Requirements with Misuse Cases," Requirements Engineering, vol. 10, Issue 1, pp. 34-44, January 2005.

[Srivatanakul05] T. Srivatanakul, "Security Analysis with Deviational Techniques," PhD thesis, Department of Computer Science, University of York, UK, 2005.

[Torr05] P. Torr, "Demystifying the Threat-Modeling Process," IEEE Security and Privacy, vol. 03, no. 5, pp. 66-70, September/October 2005.

[Turpe17] S. Turpe, "The trouble with security requirements", 25th International Requirements Engineering Conference (RE), pp. 122-133, Lisbon, Portugal, September 4-8, 2017.