Security Software Formal Modeling and Verification
Method Based on UML and Z

Kunyu Cao*, Xiaohong Li, and Jinliang Xing

School of Computer Science and Technology, Tianjin University, Tianjin, China
{kunyucao,xiaohongli,jinliangxing}@tju.edu.cn

Abstract. With the continuous expansion of software application range and the frequent occurrence of software security accidents, software security has become the focus of attention. This paper proposes a security software formal modeling and verification method, which uses UML to organize the basic structure of software, and uses Z specification to supply security semantics for UML model in order to support formal model verification. This method can guarantee the security of software design, while reducing the complexity of formal modeling. On this basis, this paper also introduces a website register function as case study and discovers the security flaws in the system design; the effectiveness and practicality of this method is then demonstrated.

Keywords: software security, formal method, weakness detection, theorem proving.

1 Introduction

In software design process, the security flaws introduced by lack of security considerations are important factors causing rapid growth of software security issues. Microsoft’s research report [1] pointed out that more than 50% of software security flaws stem from unsafe design, and will lead to higher repair costs in the later stages of software development. Therefore, the timely detection of defects in the earlier stages of software development is very necessary.

Modeling and verification is an important method to study the security of software design. However, due to the high level of difficulty and complexity of software modeling, there is not a standardized process to develop security software [2].

Software modeling can be divided into two major categories, formal and informal. Informal methods often cannot support model automatic verification effectively. Bau and Mitchell [3] proposed a conceptual framework that modeling system security from three aspects including system, threats and security attributes. Ali, El Kassas and Mahmoud [4] present a modeling method that combines SAM and threat modeling technique, but the formal refinement before model verification increases the complexity of the process.

* Corresponding author.

V. Khachidze et al. (Eds.): iCETS 2012, CCIS 332, pp. 390–401, 2012.
© Springer-Verlag Berlin Heidelberg 2012
Formal methods can effectively support model verification methods such as belief logic [5], model checking [6] and theorem proving [7], but most of them face the restrictions of limited expression ability and higher modeling complexity. Yoo and Jee [8] introduced a formal integrated development and verification environment for safety-critical software, mainly used for requirements analysis and design of the reactor protection system. Khan and Zafar [9] modeled and verified security properties of a railway control system formally using Z [10] specification. Bartels and Glesner [11] presented a modeling and verification method for LLVM program based on CSP [12] [13], but CSP's limited expression ability cannot describe complex temporal behaviors.

To solve these problems, scholars have put forward a method of combining UML and formal language. Hei et al [14] proposed a method that transformed UML state diagrams into Petri net model, which verifies UML model indirectly through the verification of the Petri net model. Moebius, Stenzel and Reif [15] present a model-driven method that can generate a formal model for security verification from UML diagrams. These new ideas improved, to some extent, purely formal methods.

This Paper presents a security software formal modeling and verification method. Firstly, this method builds formal models containing semantics of software using UML and Z specification. After that, we formalize software weaknesses based on analysis and abstraction. Finally, this method verifies software design security by theorem proving, using Z-EVES tool.

Compared with other methods, this method has several advantages: 1) Combination of rigorous formal methods with UML fetch up the lack of security considerations of UML; 2) Using formal model as intermediate bridges the gap between high-level design model with specific code; 3) Using UML view as the modeling framework can reduce the system formal modeling complexity efficiently. 4) Application of theorem proving verification can effectively find weaknesses in software design in order to reduce hem in a timely manner. To the best of our knowledge, there exists no other approach that combines all of these benefits.

The rest of the paper is organized into four sections, where section 2 describes the construction of software formal modeling, section 3 describes software security verification, and section 4 demonstrates our method using a case study of website register system. Finally, section 5 provides concluding remarks.

2 Software Formal Modeling Method

2.1 Existing Problems

UML is used widely as an important software design modeling language in software development process. However, due to its lack of security considerations, although the software meets the functional requirements there are often some serious security flaws. In addition, UML has no rigorous mathematical support, resulting in uncertainty and ambiguity factors that hinder the efficiency of software development.