Realizing the Potential of Attack Patterns for Secure Software Development

Xiaohong Yuan, Francis Akowuah, Huiming Yu
Department of Computer Science
North Carolina A&T State University
1601 East Market St., Greensboro, NC 27411, USA
xhyuan@ncat.edu

Abstract

A critical challenge in secure cyberspace is engineering more secure software. To address this challenge, it is important to integrate security-related activities and deliverables to each of the phases of software development life cycle (SDLC). To develop secure software, software developers need to think like an attacker. Attack patterns are a mechanism that captures the attacker’s perspective, and approaches used by attackers to generate exploits against software. Attack patterns have the potential to be used in each phase of SDLC. We propose to realize the potential of attack patterns for secure software development by developing methods and tools for using the attack patterns in the Common Attack Pattern Enumeration and Classification (CAPEC) library in the requirements, design and testing phases of SDLC.

1. Introduction

A critical challenge in secure cyberspace is engineering more secure software. In recent decades, it has been realized that the majority of the security defects are due to known software defects [1]. As a result, a number of secure software engineering initiatives have emerged. Examples of these initiatives include Microsoft’s Trustworthy Computing initiatives [2], the Build Security In initiative [3], the CERT Software Assurance initiative [4], etc. These secure software engineering initiatives propose a number of secure Software Development Life Cycles (SDLCS), which integrate security-related activities and deliverables to each of the phases of software development life cycle.

To develop secure software, software developers need to think like an attacker. Derived from the concept of design patterns, attack patterns are a mechanism that captures the attacker’s perspective, and approaches used by attackers to generate exploits against software. Attack patterns can be used to teach software developers how software is exploited in reality and how to avoid the attacks. An attack pattern typically includes the following information: Pattern Name and Classification, Attack Prerequisites, Description, Related Vulnerabilities or Weaknesses, Method of Attack, Attack Motivation-Consequences, Attacker Skill or Knowledge Required, Resources Required, Solutions and Mitigation, etc. [5]

According to [6], attack patterns have the potential to be used in each phase of the SDLC. During requirement gathering phase, attack patterns allow requirement gatherers to define how the software should react if the software is under the attacks described by the attack patterns. Attack patterns can be useful in the architecture and design phase because some attack patterns describe attacks that directly exploit architecture and design flaws in software. Attack patterns can help architects to determine which architectural and design features to avoid or to incorporate since they provide a useful context for the threats that the software is likely to face. During the implementation and coding phase, applicable attack patterns can be used to guide developers to avoid implementation bugs, and guide the development of subsets of search rules for static analysis tools which target security weaknesses. Attack patterns can be used to develop test cases during the testing phase. Attack patterns can also guide the design of secure operational configurations and procedures, as well as the generation of policies and standards that are used to develop secure software [6].

It currently includes 474 attack patterns contributed by the community.

Considering the potential value of attack patterns in all phases of SDLC, and the vast amount of resources on attack patterns, a number of research questions can be raised: How can the potential value of attack patterns for developing secure software be realized? In another word, how can attack pattern resources such as CAPEC be utilized in developing secure software? Though Barnum and Sethi [6] illustrates using examples how attack patterns can be used in each phase of SDLC, a systematic process of utilizing existing attack patterns in all phases of SDLC is lacking. Also, can automated tools be developed to support the use of attack patterns listed in CAPEC in developing secure software? How can attack pattern resources such as CAPEC be used as an educational tool for software developers as well as computer science or software engineering students in colleges and universities?

Pauli and Engebretson [9] proposed a software tool that is based on the CAPEC dictionary. The software tool takes as an input a system prerequisite and presents to the user related attack patterns and necessary mitigation strategies. Pauli and Engebretson [10] also proposed a hierarchy-based approach for teaching CAPEC attack patterns. A hierarchy framework are utilized to present attack pattern information logically, and help students gain deeper understanding of relevant attack pattern elements. Despite these limited efforts of utilizing CAPEC attack patterns in software development and software security education, much research needs to be conducted to answer the research questions raised above.

2. Proposed Research

In order to realize the potential of attack patterns for secure software development, we propose to develop methods and tools for using CAPEC attack patterns in the requirements, design and testing phases of SDLC. The following research objectives are proposed:

1. Develop a method of using CAPEC attack patterns to elicit and refine software security requirements
2. Develop a method of using CAPEC attack patterns to generate abuse cases
3. Develop a method of using CAPEC attack patterns for developing test cases
4. Develop a tool for recommending applicable attack patterns during the requirement, architecture and design phases. This will extend Pauli and Engebretson’s work [9] since the tool will be used in requirements, and architecture phase, not just design phase.

We will start with conducting case studies. For example, to develop a method of using CAPEC attack patterns to elicit and refine software security requirements, we can select an application such as an open source web application, and write requirements including security requirements for the application. We will then analyze the requirements and select applicable attack patterns from the CAPEC attack patterns. We will examine the process of selecting applicable attack patterns, and the indicators of applicability of attack patterns based on existing requirements. This will shed light on developing a tool for recommending applicable attack patterns during the requirement elicitation phase. Based on the selected applicable attack patterns, we will elicit and refine security requirements. After conducting several of such case studies, and we will then generalize the process into a method of using CAPEC attack patterns to elicit and refine software security requirements. This approach can also be used for the other research objectives listed above.

3. References