Chapter 8
Method Using Command Abstraction Library for Iterative Testing Security of Web Applications

Seiji Munetoh
The Graduate University for Advanced Studies (SOKENDAI), Japan & IBM Research, Japan

Nobukazu Yoshioka
National Institute of Informatics (NII), Japan & The Graduate University for Advanced Studies (SOKENDAI), Japan

ABSTRACT

A framework based on a scripting language is commonly used in Web application development, and high development efficiency is often achieved by applying several Agile development techniques. However, the adaptation of security assurance techniques to support Agile development is still underway, particularly from the developer's perspective. The authors have addressed this problem by developing an iterative security testing method that splits the security test target application into two parts on the basis of the code lifecycle, application logic ("active development code") and framework ("used code"). For the former, detailed security testing is conducted using static analysis since it contains code that is changed during the iterative development process. For the latter, an abstraction library at the command granularity level is created and maintained. The library identifies the behavior of an application from the security assurance standpoint. This separation reduces the amount of code to be statically inspected and provides a mechanism for sharing security issues among application developers using the same Web application framework. Evaluation demonstrated that this method can detect various types of Web application vulnerabilities.

DOI: 10.4018/978-1-5225-3422-8.ch008

INTRODUCTION

Ensuring the security of Web applications is essential since they are connected to the Internet and exposed to attack. Various approaches have been taken to improving their security. For example, software vulnerabilities, weaknesses, and attack patterns have been enumerated and are being maintained by the MITRE Corporation[1] to support the sharing of security issues by software developers. In addition, application security has been improved through application of such practices as Security Development Lifecycle, Secure by Design, and Secure by Default (Microsoft, 2011). Unfortunately, the problem of Web application security remains for various reasons, including the evolution of software development methodologies, human errors in programming and configuration, and intentional avoidance of the default security features as a workaround.

An idealistic approach is to rely on the programming language and application framework to completely hide security-related issues from application programmers. While such concealment is gradually becoming a reality, security assurance is still a complex and tedious task in application development since security is related to non-functional requirements. There are new security issues associated with new features, and there are security functions closely related to application functionalities, e.g., access control. Hence, new methods and automation tools that facilitate awareness of the latest security issues and topics in application development are still required.

On the one hand, conventional security assurance methods such as security requirements documentation, secure design and implementation, and security testing align with and have been proven in the waterfall-style development process. These methods have been used for large-scale system development by leading software companies. They require professional human resources such as security experts, ample budget, and substantial development time. On the other hand, Agile development methods have recently become widely used, particularly by small teams with a limited budget. There is therefore a mismatch between Agile development methods and conventional security assurance methods.

At present, it is necessary to use a combination of various tools and methods to achieve security assurance. A unified tool would simplify security assurance and be better suited to Agile development. In addition, the tool itself should be developed using the Agile development approach to respond to changes in the environment and in technology.

In this paper, we propose a comprehensive method that uses an automation tool to facilitate the security assurance of Agile Web application development using a scripting language and an application framework. The target application is divided into two parts, the application logic ("active development code") and the framework ("used code"). From a static code analysis point of view, the boundary between the application and the framework is unclear since both are written in the same scripting language. We clearly distinguish the two parts on the basis of the differences in lifecycle and maturity and deal with them using completely different approaches.

For the former, detailed security testing is conducted using static analysis, including data flow analysis and control flow analysis. For the latter, an abstraction library at the command granularity level is created that identifies both security features and potential weaknesses provided by the framework. This separation reduces the amount of code to be statically inspected during iterative software development. The effect of the library is not only to reduce the inspection time but also to facilitate the sharing of security knowledge among developers using the same application framework and components. The library provides security knowledge with more detailed granularity than conventional methods and identifies vulnerabilities and security countermeasures at the command level.