An Automatic Meta-revised Mechanism for Anti-malicious Injection

Jin-Cherng Lin[1], Jan-Min Chen[1,2], and Hsing-Kuo Wong[3]

[1] The Dept. of Computer Sci & Eng, Tatung University, Taipei 10451, Taiwan
[2] The Dept. of Information Management, Yu Da College of Business Miaoli 36143, Taiwan
[3] Chung-shan Institute of Science *and* Technology, Taiwan
jclin@ttu.edu.tw, ydjames@ydu.edu.tw, davidwong1536@gmail.com

Abstract. "Invalidated Input" is Top One Critical Web Application Security Vulnerabilities according to have been released by Open Web Applications Security Project (OWASP) on July 14, 2004. Many web application security vulnerabilities result from generic input validation problems. Some sites attempt to protect themselves by filtering malicious input, but it may not be viable to modify the source of such components. We have tried to develop an automatic defense mechanism that can produce a proper input validation function on security gateway to filter malicious injection. To verify the efficiency of the tool, we picked the websites made up of some Web applications often contain third-party vulnerable components which was shipped in binary form. Among our experiments, the defense mechanism can automatically organize validation functions to avoid malicious injection attack. *abstract* environment.

Keywords: Black box testing, Malicious injection, Input validation, Security gateway.

1 Introduction

Many web application security vulnerabilities result from generic input validation problems. Some sites attempt to protect themselves by filtering malicious input, but a surprising number of web applications have no validation mechanisms. Many tools have been developed to detect Web application vulnerabilities but hackers are still successfully exploiting Web applications. A possible reason is that most tools just scan Web application vulnerabilities, but few tools can automatically revise these vulnerabilities. An advanced tool producing a proper input validation function depending on the database server and the application framework has been developed and verified its efficiency [2], but source code needed for inserting input validation function to revise injection vulnerabilities. If we can't modify the source of such components (either because the code was shipped in binary form or because the license agreement is prohibitive), above method can't be used.

In this paper, we present an advanced proposal adopting concept of application-level security gateway and more effectively resolving the problem than similar

T. Enokido, L. Barolli, and M. Takizawa (Eds.): NBiS 2007, LNCS 4658, pp. 98–107, 2007.

gateways or proxies. Our system consists of black box testing, validation functions and redirection mechanism. Black box testing can find all entry pointers and produce vulnerability lists. Validation functions can be dynamically organized and filter HTTP requests / responses to avoid malicious injection attacks. Redirection mechanism can avoid attack requests propagated to the web-server and return an error page to the user.

The remainder of the paper is structured as follows: Section 2 surveys a number of web application vulnerabilities and discusses related works have be proposed. In Section 3 we describe the technical details of our system for anti-malicious injection. Our system implementation is discussed in Section 4. The efficiency of our implementation is evaluated in Section 5, finally, Section 6 concludes.

2 Web Application Vulnerabilities and Related Work

Of all vulnerabilities identified in Web applications, problems caused by unchecked input are recognized as being the most common [1]. Static analysis can be used to analyze Web application code, for instance, ASP or PHP scripts. However, this technique fails to adequately consider the runtime behavior of Web applications and we must get source code. Recently, Y.W. Huang, S.K. Huang, T.P. Lin, and C.H. Tsai have developed a tool called WebSSARI (Web application Security Analysis and Runtime Inspection) [4,5]. The tool can be successfully used for automated Web application security assessment.

Another method called a black-box approach is adopted to analyze Web applications externally without the aid of source code. A black-box security analysis tool can perform an assessment very quickly and produce a useful report identifying vulnerable sites. Y.W. Huang, S.K. Huang, T.P. Lin, and C.H. Tsai have developed a remote, black-box security testing tool for Web applications is also called the Web Application Vulnerability and Error Scanner (WAVES) [6]. It can be used to analyze the design of Web application security assessment mechanisms in order to identify poor coding practices that render Web applications vulnerable to attacks such as SQL injection and cross-site scripting. The Open Web Application Security Project (OWASP) [7] has launched a WebScarab project. Two other available commercial scanners include SPI Dynamics' WebInspect [8] and Kavado's ScanDo [9].

Above works just only focus on detection, they seldom propose efficient method to automatically fix program's vulnerabilities. An advanced tool producing a proper input validation function depending on the database server and the application framework has been developed and verified its efficiency by us [2], but program or component's source code needed for inserting input validation function to revise injection vulnerabilities. Scott and Sharp [3] take the programmatic approach of specifying a security policy explicitly to provide a web application input validation mechanism-a rule-based security gateway- to protect against common application-level attacks. However, to enforce a security policy across a large web-application is difficult and adapt this mechanism