SUMMARY OF Ph.D. DISSERTATION

School Science for Open and Environmental Systems	Student Identification Number	SURNAME, First name SHIMAMURA, Makoto

Title

A study on automatic behavioral analysis of shellcode

Abstract

The Internet is one of the most important infrastructures. Various services, such as online banking and online shopping, are provided in the Internet. Remote attacks that target servers providing services show no sign of decline. Remote attacks prevent the target servers from functioning properly and incur significant damage. Thus, protecting servers from remote attacks is an important challenge to network security.

Many remote attacks aim to force victim servers to execute malicious machine instructions, often called *shellcode*, in a malicious message. To protect servers against shellcode execution, server administrators use defense systems, such as intrusion detection systems. Defense systems protect servers from shellcode by using characteristics of shellcode, called *signatures*. For example, a host-based intrusion detection system detects shellcode if there is a file whose name matches one of the signatures of filenames, and a network-based intrusion detection system blocks a message if it contains a match to one of the signatures of byte patterns. A defense system requires a signature for each shellcode. As a result, it cannot provide enough protection against shellcode if it does not have a signature for the shellcode. When new shellcode appears, a vendor of defense systems must create a new signature. To create a signature for new shellcode, it is necessary to analyze the shellcode.

In shellcode analysis, an analyst uses disassemblers and debuggers to extract system calls and API calls that shellcode issues to access computation resources, and then the analyst creates signatures for the shellcode with the extracted information. However, manually analyzing shellcode is time-consuming and error-prone. Moreover, shellcode analysts must analyze a lot of shellcode on a daily basis because a vendor usually collects ten thousands of shellcode in a day to quickly find new shellcode. To reduce the burden on shellcode analysts, some researchers propose automatic shellcode analyzers.

However, skillful attackers recently try to evade shellcode analyzers. For example, an attacker encrypts shellcode which is decrypted at runtime. This encryption prevents analyzers from disassembling the shellcode body. Shellcode also detects the symptom that it runs on an analyzer or a debugger by inspecting the result of system calls. If the shellcode succeeds to detect such an environment, it terminates immediately to prevent the analysis. Unfortunately, existing shellcode analyzers are evaded by these techniques. To analyze shellcode accurately, a shellcode analyzer must be robust to such evasion techniques.

This dissertation proposes a novel shellcode analyzer, *Yataglass*, which is more difficult to evade than existing shellcode analyzers. Yataglass analyzes shellcode by emulating the execution of shellcode. By doing so, encrypted shellcode cannot evade Yataglass. To prevent shellcode from detecting Yataglass by inspecting results of system calls, Yataglass finds a conditional branch that inspects the results of system calls with dynamic taint analysis, and then it analyzes both execution paths of the branch. To make evasion of Yataglass more difficult, this dissertation also proposes a countermeasure for the memory-scanning attack. Memory-scanning attack disrupts existing shellcode analyzers by accessing data in victim server's memory. To analyze shellcode that incorporate memory-scanning attacks, Yataglass infers data that shellcode makes use of with symbolic execution.

The dissertation shows the experimental results using a prototype implementation of Yataglass. According to the results, Yataglass successfully analyzed real shellcode and ones generated by MetaSploit. The experimental results also show that Yataglass can analyze shellcode that evade existing analyzers by inspecting results of system calls and memory-scanning attack.