ATLANTIS PRESS

Advances in Computer Science Research, volume 82

7th International Conference on Social Network, Communication and Education (SNCE 2017)

Survey of Software Vulnerability Discovery Technology

Wei Wang

Information Engineering Department, Capital Normal University, Beijing, China

wjwblabla@163.com

Keywords: Vulnerability discovery; Fuzzing; Software security

Abstract. The 21st century is the information age. The rapid development of computer technology supports the rapid development of the information age. With the rapid spread of computers and networks, more and more software products play an important role in people's daily life. In computer security[1], a vulnerability[2] is a weakness which allows an attacker to reduce a system's information assurance. Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw. Due to software developer's negligence in the development process of software or programming language limitations, software products often have security and functional flaws which damage software, known as software vulnerabilities. Software vulnerability discovery aims at discovering vulnerabilities that already exist in software, and then developers patch vulnerabilities to eliminate damages brought to software products. Now, vulnerability discovery ,in the field of information security, is becoming increasingly important. This paper mainly introduces the main methods of vulnerability discovery.

Introduction

Vulnerabilities are security flaws in computer system or software, including functional or logical defects, that pose a potential threat to the security of computer system and software.

Vulnerabilities have the following characteristics: necessity, permanence, and harmfulness. Because the computer program is not computable, we cannot use the mathematical method to completely eliminate all the logical defects and vulnerabilities in the software, so the vulnerability has necessity[3]. And in general, vulnerabilities are usually exposed with the use of software user, the computer system and software may have been running for a long time, so the vulnerability has permanence. At the same time, the existence of vulnerabilities is apt to damaging computer system and software, so the vulnerability has harmfulness.

The research of vulnerability is divided into vulnerability discovery and vulnerability analysis. Vulnerability discovery technology refers to the exploration of unknown vulnerabilities, comprehensively using various technologies and tools to discover potential vulnerabilities in software as much as possible. Vulnerability analysis technology is to discover the details of the vulnerabilities by analyzing deeply.

According to the vulnerability discovery objects, vulnerability discovery can be divided into two categories: source-based vulnerability discovery and binary-based vulnerability discovery. Source-based vulnerability discovery technology can obtain source code of software, then use automatic tools or manual testing methods analyzing the source code to discover software vulnerabilities. Binary-based vulnerability discovery technology includes white box analysis, black box analysis and gray box analysis. White box analysis is an analysis method of vulnerability with the binary code of target software, or source code changing from binary code by reverse engineering[4]. Black box analysis is an analysis method of vulnerability without binary code, which can control program input, then observe program output getting information to discover vulnerability[5]. Gray box analysis is an analytical method of vulnerability that combines above two analysis methods to improve the efficiency and quality of vulnerability discovery[6].

This paper mainly investigates the methods and techniques of vulnerability discovery. The next

section introduces the main existing methods of vulnerability discovery, including manual analysis, fuzzing, static analysis technology and dynamic analysis technology.

Technologies of Vulnerability Discovery

As Table 1 shows, main vulnerability discovery methods include manual testing technology, Fuzzing, static analysis technology, dynamic analysis technology, etc.

Table 1 Advantages and disadvantages of vulnerability discovery technologies

Vulnerability discovery technology	Manual testing	Fuzzing	Static analysis	Dynamic analysis
Advantage	It is mainly used in program with interface.	There is less false positives, high efficiency and able to detect a variety of vulnerabilities.	It is mainly used in program without source code.	It is mainly used in program without source code or dealt with reverse engineering.
Defeat	It highly depends on the analyst's experience and skills.	There is false negative and not common.	The result set to analysis is large and false positive rate is high.	It is usually not automatic.

Next, this paper will introduce the above four kinds of vulnerability discovery technologies.

Manual Testing. Manual testing is a vulnerability discovery technique for the target program, according to manually constructing a variety of input data, then finding the problem by observing the state and output of the target program.

The manual constructed input data includes valid data and invalid data. And output data also includes normal output data and abnormal output data. Abnormal output is likely to be caused by the vulnerability, and abnormal status of target program is also a foretaste of vulnerability.

As Table 1 shows, manual testing is highly dependent on the experience and skills of the analyst, and is usually used for target programs that has human-machine interaction interfaces, such as the manual analysis methods in web vulnerability discovery.

Fuzzing. Fuzzing was first thought by Barton Miller Lars Fredriksen and Bryan So casually[3]. Fuzzing is an automatic software testing technology based on defective injection. It is an effective automatic vulnerability discovery technology[7]. The principle is sending some random combination of data and code through the software external interface, to make the target software throw exception, and then discovering vulnerability[8]. Because these data and code do not need to understand and satisfy the contextual relationship of the target software, fuzzing technology can be used to automatic vulnerability discovery and analysis.

Fuzzing uses a large number of automatically constructed semi-valid input data to discover vulnerabilities if the input data can generate output abnormal or causing the application to crash in the program. Fuzzing is a special black box analysis technology ,which is different from the functional test, whose purpose is “crash”, “break” and “history”. Fuzzing test cases are usually offensive malformed data to trigger various types of vulnerabilities. Common fuzzing tool includes SPIKE Proxy, Peach Fuzzer Framework and so on.

There are mainly two main fuzzing techniques: Blind Fuzzing and Smart Fuzzing. Blind Fuzzing is inserting random data at random location to generate malformed files. This method is simple to implement, easy to quickly trigger error, but its full randomness will lead to a large number of invalid input or format. However, modern software often uses very complex private data structure. Blind Fuzzing exposes some shortcomings for complex files with complex data structures , for example, the strategies to generate test case lack are of pertinence, generate a large number of invalid test cases and it is difficult to find deep logic vulnerabilities of complex parsers.

For the lack of Blind Fuzzing, Smart Fuzzing is more and more proposed and applied. Smart

Fuzzing studies the protocol and file format of target application, function configuration, and understands the causes of various vulnerabilities, to write fuzzing tool on purpose. Usually Smart Fuzzing includes three characteristics: logic-oriented (Logic Oriented Fuzzing), data type (Data Type Oriented Fuzzing) and sample-based (Sample Based Fuzzing).

Logic Oriented Fuzzing refers to that the test goal is the logic of parsing file, but not the file itself. After determining the logical goal of test, we can change the specific location of the sample file as far as possible without destroying other data dependencies, when generating the deformed data. Data Type Oriented Fuzzing refers to that it can identify different data types and can use different rules to generate deformed data according to the type of target data. The deformed data by this method is usually effective and can largely reduce invalid deformed files.

Sample Based Fuzzing refers to that at first create a legitimate sample file, also known as template file, whose all data structure and logic must be legitimate, and then treating this sample file as a template to generate deformed file every time only modifying a small part of data and logic. This method is also called variation and this method is less difficult than the above two methods at generating deformed files for complex files. The above three features are not independent of each other, but can be exist at the same time.

As Table 1 shows, compared with other technologies, fuzzing rarely creates false positives, is high efficiency, can quickly find a real vulnerability, and can be used to detect a variety of vulnerabilities. Fuzzing technology can be used to detect multiple security vulnerabilities, including buffer overflow vulnerabilities, integer overflow vulnerabilities, formatted strings and special character vulnerabilities, competing and deadlock vulnerabilities, SQL injection, cross-site scripting, RPC vulnerability attacks, files System attacks, information disclosure and so on. At present, what is commonly used in industry is fuzzing technology. But, fuzzing also has shortcomings, such as, it cannot guarantee that the system has no vulnerabilities and it is not common.

According to the characteristics of the analysis objectives, fuzzing can be divided into three categories: file format fuzzing, protocol fuzzing, and dynamic web pages Fuzzing.

The file format fuzzing is a fuzzing method for file format parsing. File format fuzzing uses malformed files as input data, then observe whether the software throws exception further to discover vulnerabilities when software parses malformed files. Protocol fuzzing is a fuzzing method for network protocols. For example, FTPFuzz (Infigo FTPStress Fuzzer), a tool for exploiting FTP protocol, is a tool specifically designed to test FTP security. Its basic principle is replacing the command parameters of FTP protocol with dirty data, to construct the abnormal FTP command and sent to the FTP service program being tested, to detect whether the FTP service program is abnormal, and then to fuzz vulnerabilities. Dynamic Web pages Fuzzing is a fuzzing method for web program written in ASP, PHP, Java, Perl and the B / S architecture applications using such technologies. The typical tool is HTTP Fuzz.

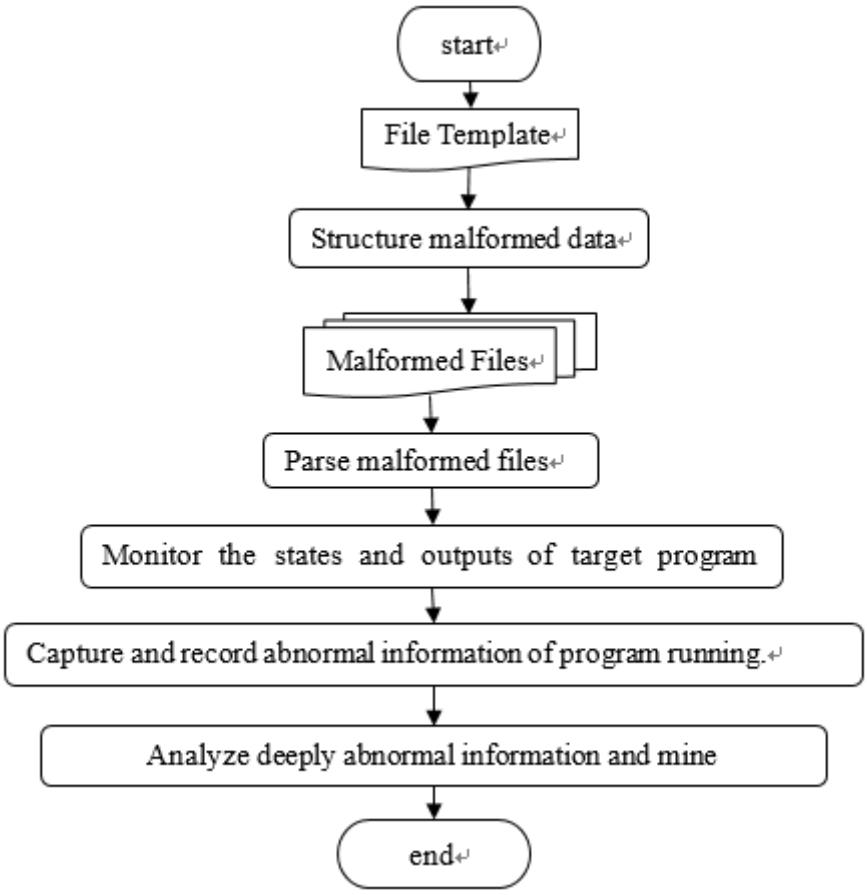

Figure 1. process of file format fuzzing

Figure 1 shows the process of file format fuzzing. At first, generate a number of deformed files according to certain rules using a normal file as test template file. Secondly, send the deformed files as input data to software to parse. Thirdly, inspect the status and output of software whether the software throws exception. Then, record errors the software produce, such as register status, stack status, etc. At last, in-depth analyze the errors and exceptions to find the useful information to discover vulnerabilities.

Static Analysis Technology. Static analysis technology is analyzing and detecting source program to discover security vulnerabilities or hidden dangers in the program[9]. It is a typical white box analysis technology whose methods mainly include static string searching and context searching. Static code Audit is a kind of static analysis method for source code target for discovering program errors, security vulnerabilities[10]. As Table 1 shows, code audit is checking source code of program to find shortcomings and errors, then analyzing these information to discover vulnerabilities.

Dynamic Analysis Technology. Dynamic analysis technology originated from software debugging technology, is to use the debugger as a dynamic analysis tool, but different from the software debugging technology is that it is often handled without the source code, or the code was reversed by reverse engineering[11]. As Table 1 shows, dynamic analysis requires running the target program in the debugger to discover vulnerabilities by observing the running status of the program, memory usage, and register values during execution. Common dynamic analysis tools include SoftIce, OllyDbg, WinDbg and so on. But the shortcoming of dynamic analysis technology is difficult to automate.

Summary

With the popularity of computer software in people's daily life, more and more countries and people concern about the security of software and vulnerabilities of software. The vulnerability discovery

technology is an important aspect in information security field. Vulnerability discovery technology, born out of the software testing theory and software development debugging technology, can greatly improve the security of the software. But the vulnerability discovery is a double-edged sword, which has become the mainstream technology for hacker to hack software.

All in all, the development prospect of vulnerability discovery technology is broad, with the information security is more and more attention, software development technology is more and more advanced ,and the new analysis means will also follow.

References

[1] Computer security on https://en.wikipedia.org/wiki/Computer_security
[2] Vulnerability on https://en.wikipedia.org/wiki/Vulnerability_(computing)
[3] B. Liu, L. Shi ,Z. Cai and M. Li: *Proc. MINES '12 Proceedings of the 2012 Fourth International Conference on Multimedia Information Networking and Security*(Washington, DC, USA, November 02 - 04, 2012). 2012, p.152-156
[4] White-box testing on https://en.wikipedia.org/wiki/White-box_testing
[5] Black-box testing on https://en.wikipedia.org/wiki/Black-box_testing
[6] Gray-box testing on https://en.wikipedia.org/wiki/Gray_box_testing
[7] Fuzzing on https://en.wikipedia.org/wiki/Fuzzing
[8] Fuzzing on http://baike.baidu.com/link?url=Vujua1EhJYHwFx4D2SdDmcK4aV-kb5LxIpNjISyMKimPuLpxFgc0DKH9iTxlwGBAxsNaLrUgpDZ2MiE6a3k9Y_
[9] Static program analysis on https://en.wikipedia.org/wiki/Static_program_analysis
[10]Static code analysis on https://www.owasp.org/index.php/Static_Code_Analysis
[11]Dynamic program analysis on https://en.wikipedia.org/wiki/Dynamic_program_analysis