World Scientific Research Journal Volume 5 Issue 8, 2019
ISSN: 2472-3703 DOI: 10.6911/WSRJ.201908_5(8).0005

Software Security Vulnerability Detection

Guping Zheng[1], Jinhua Li[1, a, *] and Jingang Cao[1]

[1]Department of Control and Computer Engineering, North China Electric Power University, Baoding, China

[a]1844623784@qq.com

Abstract

Vulnerabilities are some functional or security logical defects in the system, including all factors that cause threats and damage the security of the computer system. They are the existence of computer systems in hardware, software, protocol implementation or system security policies. Defects and deficiencies. For various reasons, the existence of vulnerabilities is inevitable. Once some of the more serious vulnerabilities are discovered by attackers, they may be exploited to access or destroy computer systems without authorization. Before the attacker finds and fixes the vulnerability in time, it can effectively reduce the threat from the network. Therefore, it is of great significance to actively explore and analyze system security vulnerabilities. This paper mainly introduces three technologies in software security vulnerability detection, namely static detection technology, dynamic detection technology and hybrid detection technology. After analysis, three technologies are compared. Different software can be used to detect software. Security vulnerabilities to protect software security.

Keywords

Manuscripts; software security; static detection technology; dynamic detection technology; hybrid detection technology.

1. INTRODUCTION

Network security has become an important issue that people are paying more and more attention to. According to the statistics of the CNCERT/CC 2007 network security work report, the number of vulnerabilities has shown a clear upward trend in recent years. Not only that, the time from the publication to the use of new vulnerabilities is getting shorter and shorter, and hackers analyze and research the vulnerability information released, often these vulnerabilities can be successfully exploited in a very short time. In addition to exploiting known vulnerabilities, hackers are also good at exploiting and exploiting unpublished vulnerabilities, launching virus attacks, or selling vulnerabilities to meet economic goals. Compared to hackers, security researchers appear to be relatively passive and lagging in vulnerability research. Therefore, research on vulnerability mining should be intensified in order to take a more proactive and reasonable approach to various types of vulnerabilities.

Many security organizations and individuals at home and abroad are engaged in research on vulnerabilities. Two of the more authoritative vulnerability vendors are CVE (Common Vulnerabilities and Exposures) and CERT (Computer Emergency Response Team). In addition, foreign eEye, LSD and other organizations also timely track and analyze the latest vulnerabilities, and give corresponding vulnerability solutions. NSFOCUS, Venus Star and other units are representatives of domestic security research organizations.

Vulnerability refers to some functional or security logical defects in the system, including all the factors that cause threats and damage the security of the computer system. It is the specific

ISSN: 2472-3703 DOI: 10.6911/WSRJ.201908_5(8).0005

implementation of the computer system in hardware, software, protocols or system security policy. Defects and deficiencies. For various reasons, the existence of vulnerabilities is inevitable. Once some of the more serious vulnerabilities are discovered by attackers, they may be exploited to access or destroy computer systems without authorization. Before the attacker finds and fixes the vulnerability in time, it can effectively reduce the threat from the network. Therefore, it is of great significance to actively explore and analyze system security vulnerabilities.

2. STATIC DETECTION TECHNOLOGY

Static analysis is a tactical, grammatical, and semantic analysis to detect potential security problems in a program. When it finds a security vulnerability, its basic method is also a static scan analysis of the program source program, so it is also classified as static detection analysis. Static analysis focuses on checking function calls and return status, especially for function calls that do not perform boundary check or boundary check (such as strcpy, strcat, etc., which may cause buffer overflow), functions provided by the user, and buffers in the user. A program for performing pointer arithmetic, and the like. The current popular software vulnerability static analysis technology [1] mainly includes source code scanning and disassembly scanning, which are all analysis techniques that can analyze the possible vulnerabilities in the program without running a software program.

Source code scanning is mainly for open source programs. By detecting file structures, naming rules, functions, stack pointers, etc. that do not comply with security rules, the security flaws that may be implied in the program are discovered. This vulnerability analysis technique requires proficiency in the programming language and pre-defines the rules for reviewing unsafe code, checking the source code by means of expression matching. Since the program is dynamically changed when running, if you do not consider the parameters and calling environment of the function call, there is no way to accurately grasp the semantics of the program without lexical analysis and parsing of the source code. Therefore, this method cannot find the dynamic operation of the program. Security hole.

Disassembly scans are often the most effective way to discover security vulnerabilities for programs that do not expose source code. There is a wealth of experience in analyzing disassembled code, and auxiliary tools can be used to help simplify the process. But there can be no completely automated tool to complete the process. For example, an excellent disassembler IDA can be used to get the assembly script language of the target program, and then use the scan method for the compiled script language to further identify some suspicious assembly code sequences. The benefit of looking for system vulnerabilities through disassembly is that, in theory, no matter how complex the problem can always be solved by disassembly. Its shortcomings are also obvious. This method is time-consuming and labor-intensive, requires a high level of technical skill, and cannot detect security vulnerabilities generated during the dynamic operation of the program.

The static analysis method is efficient and fast, and the source code can be checked quickly, and the inspector does not need to understand the implementation of the program, so it is very suitable for automatic program source buffer overflow check. In addition, it provides a more comprehensive coverage of system code and reduces false negatives. However, this method also has great limitations. The ever-expanding feature library or dictionary results in a large set of detection results and high false positive rate. The static analysis method focuses on analyzing the "features" of the code without concern for the program. The function does not have an analysis check for the function and program structure.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.201908_5(8).0005

3. DYNAMIC DETECTION TECHNOLOGY

In the actual detection process, in addition to open source software, it is difficult to obtain the source code information of the system under test, thus limiting the application of static detection technology. The dynamic detection technology is to debug the running software system by constructing non-standard input data, and check the abnormality of the running result according to the system function or data flow direction to determine whether the tested software system has a loophole. Dynamic detection technology usually uses an input interface or an operating environment as a starting point to detect vulnerabilities in the system. Although the dynamic detection technology has the advantage of high accuracy, its efficiency is very low. Because the functions and processes of various software systems are different, they cannot be uniformly scanned like static detection technology, but should be targeted at software systems. The function is dynamically detected, which will inevitably lead to a decrease in efficiency [4]. And dynamic detection can only determine the scope of the vulnerability, further tracking and analysis, and the type and utilization of security vulnerabilities can be determined through experience, so it is difficult to apply to large software.

Dynamic analysis technology is a dynamic detection technology. The target program is run in the debugger. By observing the running state of the program during execution, the memory usage and the value of the register, etc., to find potential problems and find loopholes. It starts from the two aspects of code flow and data flow: dynamically track the target program code flow by setting breakpoints to detect defective function calls and their parameters; perform two-way analysis on the data stream, trigger potential errors by constructing special data and the result Analyze. Dynamic analysis requires the help of debugger tools. SoftIce, OllyDbg, WinDbg, etc. are powerful dynamic trace debuggers.

Common dynamic analysis methods include input tracking test method, stack comparison method, fault injection analysis method, etc. [2]. Dynamic analysis is performed when the program is running, and the discovered vulnerability is a program error, so it has a high accuracy; it can specifically check the target system, so as to accurately determine the corresponding function or module of the target system. System performance. In addition, the dynamic analysis technique is very similar to the black box test. Without the source code, it can be analyzed by observing the input and output of the program, and various checks are performed to verify whether the target program has errors. This method can be used as a buffer overflow vulnerability caused by input.

4. HYBRID DETECTION TECHNOLOGY

Because the static detection technology requires the target program source code, and has the defects of large detection scale and high false positive rate, the dynamic detection technology has the defects of low coverage and low efficiency. Therefore, in recent years, hybrid detection technology has developed rapidly. It effectively combines the advantages of static detection and dynamic detection, avoids the shortcomings of static detection and dynamic detection, and effectively improves detection efficiency and accuracy. Hybrid detection technology behaves in a form similar to dynamic detection technology. However, according to the prior knowledge of the program, the tester designs the test case in a targeted manner during the test. This test can be tested directly against the boundary conditions of interest in the data stream, making it more efficient than dynamic detection.

At present, hybrid detection technology is mainly realized by an automated vulnerability digger, Fuzzing detection technology. The vulnerability exploit first analyzes the operating environment, functions, and interfaces of the target software, constructs malformed data, and generates test cases. Then, pass the test case through the interface and run the program. Finally, the monitor program is used to run the program. If the program runs abnormally, the program

ISSN: 2472-3703 DOI: 10.6911/WSRJ.201908_5(8).0005

running environment and input data are further recorded to analyze the abnormal information. Different vulnerability drills have different structures and mining methods due to different mining objects. At present, there are mainly file type vulnerability miner, FTP vulnerability miner, web vulnerability miner, operating system vulnerability miner, and so on. According to the different way of constructing test cases for excavators, Fuzzing technology can be divided into two categories [4], Dumb Fuzzing and Intelligent Fuzzing. Dumb Fuzzing detection technology is similar to black box testing in that it is based on random input to find problems.

This method is simple and easy to quickly trigger the wrong location of the vulnerability. Because it is not targeted, it is inefficient. Intelligent Fuzzing detection technology can effectively improve the efficiency of automated detection by studying the protocol, input, file format and other aspects of the target software, and can improve the efficiency of automated detection. Therefore, this method can more effectively exploit software security vulnerabilities.

5. CONTRAST

The current development trend of static detection technology is to combine various technologies of static detection to improve performance.

There are several ways to combine:

(1) Provide a framework to test the program using different detection techniques and analyze it after obtaining a large number of test results. From the false positive rate, the intersection of test results using multiple technologies to make vulnerability judgment decisions can reduce the false positive rate; from the false negative rate, the result of using multiple detection techniques for the same vulnerability can reduce the false negative rate.

(2) Directly combine detection technology, and obtain new methods through the combination of technology, such as adding type derivation technology in the process of symbol execution, and adding the derivation of its type characteristics in the process of variable simulation execution, which can obtain more High vulnerability detection rate.

In addition to the combination of multiple static detection techniques, a combination of dynamic and static detection has emerged. The SD (Static-Dynamic) method first uses a static method to analyze the program to obtain the internal features of the program, thereby screening the test data set to guide the next round of dynamic testing. The DSD method performs a dynamic test before the first round of static testing to eliminate some of the impossible input conditions and simplify the complexity of static detection.

Programs have many things in common during design and execution, such as control flow and data flow. The memory map of a process is divided into code segments, read-only data segments, data segments, resource segments, heaps, stacks, and so on. Of course, they will be different under different operating systems. For example, under UNIX, there will usually be uninitialized data segments (BSS) and so on. The dynamic detection method is a method for detecting the weakness of the program without changing the source code or even the binary code. This type of detection is mainly implemented by modifying the process running environment.

Because the static detection technology requires the target program source code, and has the defects of large detection scale and high false positive rate, the dynamic detection technology has the defects of low coverage and low efficiency. Therefore, in recent years, hybrid detection technology has developed rapidly. It effectively combines the advantages of static detection and dynamic detection, avoids the shortcomings of static detection and dynamic detection, and effectively improves detection efficiency and accuracy. Hybrid detection technology behaves in a form similar to dynamic detection technology. However, according to the prior knowledge of the program, the tester designs the test case in a targeted manner during the test. This test can

ISSN: 2472-3703 DOI: 10.6911/WSRJ.201908_5(8).0005

be tested directly against the boundary conditions of interest in the data stream, making it more efficient than dynamic detection.

6. SUMMARY

Software vulnerability mining is an important part of network attack and defense technology. This paper focuses on several common vulnerability mining techniques. Each method has its own advantages and disadvantages. In order to accurately mine software vulnerabilities, it can be used comprehensively in actual testing. For example, combining manual testing with Fuzzing technology, and using Debugger (IDA Pro, Ollydbg, etc.) for analysis and debugging is a good vulnerability mining idea [3]. Through manual testing, you can understand some basic information of the target, find out the clues that the program may be vulnerable, and use the Fuzzing technology to design and implement the specific fuzzer tool. While using fuzzer to test some of the vulnerable points of the target, use a Debugger tool to suspend the target, so that when an error occurs, it will be captured by the Debugger, so that the target program can be debugged with the help of the Debugger.

ACKNOWLEDGEMENTS

This research is supported by the Fundamental Research Funds for the Central Universities (2018MS072).

REFERENCES

[1] Zhou Liang. Application of Security Vulnerability Detection Technology in Software Engineering[J]. Network Security Technology and Application, 2017(02): 56-57.

[2] Gao Wei. Computer Software Security Vulnerability Detection Technology and Application [J]. Computer CD Software and Application, 2014(04): 172-173.

[3] Li Zhoujun, Zhang Junxian, Liao Xiangke, et al. Software Security Vulnerability Detection Technology[J]. Chinese Journal of Computers, 2015(04): 717-732.

[4] Sri Lanka .On the detection and analysis of software security vulnerabilities[]. IEEE Conference Publications, 2017.