Vulnerability Mining Based on Hook Mechanism

Longjuan Wang¹,²*, Chunjie Cao¹,², Wenjie Zhong¹,², Fangjian Tao¹,², Chaosheng Tang¹,²

¹School of Computer Science and Cyberspace Security, Hainan Province, Hainan University, No.58, Renmin Road, Haikou 570228, China
²Key Laboratory of Internet Information Retrieval of Hainan Province, Hainan University, No.58, Renmin Road, Haikou 570228, China

*juanywong@126.com

Abstract: IOT devices are easy to be attacked by hackers because of their intelligence and openness. The vulnerability mining of IOT devices has become a key research field in the security field. This paper studies a method of vulnerability mining based on function hook mechanism, which can make AFL QEMU mode is suitable for the binary fuzzy test of firmware CGI program of IOT devices, it can effectively test the firmware CGI program. Through the experiment, we successfully found the common vulnerability of IOT router equipment and through the reverse analysis of firmware, we trace back to the assembly code of the corresponding vulnerability point, and then carry out the corresponding analysis.

Keywords: Internet of things, Vulnerability mining, hook, router.

1. Introduction

According to the report data of IOT Analytics [1], by the end of 2019, there are about 9.5 billion IOT devices connected to the network, which is much higher than the previous prediction of 8.3 billion devices. The main driving factors were the explosive growth of consumer smart home and the large demand for IOT intelligent devices in the Chinese market, especially new intelligent devices such as smart door locks and smart speakers. By 2025, it is estimated that the total number of connected IOT devices will reach 28 billion, driven by new low-power WAN connections and 5G.

On the one hand, more and more hackers begin to pay attention to the security of IOT devices, carry out security research and vulnerability mining, leading to frequent 0day vulnerability events of IOT devices, such as Dlink manufacturer router vulnerability [2], Haikang, Dahua camera back door vulnerability [3], and other kinds of intelligent door lock vulnerability events.
The mentioned by Zheng Yaowen et al[4] that due to the open characteristics of IOT devices, the vulnerability mining of IOT devices has become a new opportunity and challenge in security research.

2. Background

In the research process of vulnerability mining in the IOT, most domestic and foreign IOT security researchers pay more attention to the vulnerability research of device communication protocol security and device firmware security. Through the use of auxiliary static analysis tools, such as IDA [5], ghidra [6], binwalk [7] and other reverse analysis tools, the program reverse and vulnerability mining. Aiming at the low efficiency and time-consuming problem of manual static analysis vulnerability mining, the use of symbolic execution [8], fuzzystesting and other methods can effectively help security researchers improve the efficiency of vulnerability mining. Aiming at the problems of low path coverage and high randomness in fuzzing technology, some scholars proposed AFL fuzzy [9] technology based on path coverage, which can store the path information collected in the fuzzing process in shared memory, and continuously collect and feed back through genetic algorithm. Iotfuzzer [10] can also use fuzzing of some communication protocols in IOT devices controlled by app without source code to automatically discover some memory corruption vulnerabilities in IOT devices. Nathan Voss et al [11] have developed a method combining Unicorn instruction simulation execution and AFL fuzzy testing based on path coverage, so that the code simulated and executed based on unicorn engine, such as MIPS and arm instruction set of IOT devices, can be exploited through fuzzing technology.

This paper proposes a vulnerability mining method based on function hook mechanism. AFL QEMU pattern can be applied to binary fuzzing of firmware CGI program of IOT device. This method can effectively test the firmware CGI program. The general vulnerability of IOT router is found through experiments. Through the reverse analysis of firmware, the vulnerability can be traced back to the assembly code of the corresponding vulnerability point, and then the corresponding analysis is carried out.

3. Vulnerability mining method based on hook mechanism

3.1. Using AFL QEMU mode for fuzzing

To use AFL QEMU mode, the QEMU driver of corresponding architecture needs to be compiled separately. Because in the case of no source code, the fuzzing speed of QEMU mode is much lower than that of source code inserting mode. In view of the low speed and the inability to simulate fuzzing in the whole system, Jesse hertz and Tim newsham open the source of the triforceafl tool[12], which improves the fuzzing speed of non source closed source programs.

3.2. The AFL QEMU fuzzing based on hook mechanism

When we use web fuzzing to test CGI program, we interact with it by specifying HTTP protocol and fuzzing each data field of its protocol. In firmware fuzzing, we also choose external controllable interaction points as input points, construct abnormal data to input into CGI program, and monitor abnormal behavior of the program. Because AFL fuzzy is based on path coverage, AFL fuzzy is used as the fuzzing engine.
3.3. Interaction between user input and CGI

When using AFL for fuzzing, the corresponding test cases are generally input by standard input or command line, which leads to a problem. The interaction point between IOT device and user is generally received through a CGI handler, CGI function of httpd daemons or a program port, but cannot be directly communicated with it through binary level Each other. Taking intelligent router as an example, CGI program is used in binary level to process and parse HTTP packets submitted by users' HTTP protocol to the back-end. The program obtains input through some internal defined library functions, such as cgigetvalue, websgetvar, etc. For example, the cgigetvalue("name") code in the httpd daemon can obtain the value with name as the key name in the corresponding HTTP protocol packet. When the user submits the get packet or the post packet contains the key value of name, such as when the user requests index.cgi?name=Alice The function can capture the value Alice of the field. Therefore, the cgigetvalue function is needed to hook in this case.

3.4. Redirecting input stream using hook mechanism

Since most of the functions that get input in CGI are library functions, hook technology is used to hook these CGI library functions, which makes hook functions receive data from standard input. After using AFL fuzzy to guide use case variation, these variation cases are transferred to CGI program through standard input, and finally achieve similar indirect fuzzing the CGI program, which is taking the CGI program of HTTP daemon httpd as an example, and the method flowchart is shown in Fig.1.

![Fig.1 The flowchart of method](image)

The hook mechanism will trigger and AFL fuzzy engine will start to work after the CGI program receives external input. When there is no external input, AFL fuzzy is equivalent to blocking the steps of inputting mutation cases into the program. Therefore, the external input here can repeatedly request CGI program through the requests module of Web fuzzing with HTTP request packets of various function points after parsing from burp suite, so that the fuzzy engine can work all the time without blocking.

The preparatory work for firmware fuzzing is to disassemble CGI service program, such as httpd, and find the library function to get input from HTTP protocol data package through IDA reverse analysis. Secondly, the shared library program is written. After the CGI program is hooked, the input
received by the library function is redirected to the standard input. When AFL fuzzy is used for fuzzing test, the external web fuzzing module constantly requests the HTTP service of CGI program, so it can successfully test a CGI program at the level of binary firmware.

4. **Experiment**

4.1. **Experimental environment**

After constructing the fuzzy test method of hook mechanism, this paper selects a home router on the market for actual fuzzing test. The experimental environment is as follows:

1. Operating system: Ubuntu 16.04
2. Software environment: AFL-fuzz, qemu
3. Physical equipment: ASUS home router.

4.2. **Experimental process**

This experiment selects a home router of ASUS manufacturer. The IP address of the background management interface of the router is 192.168.1.1.

Download the firmware of this model from ASUS official website, use binwalk tool to analyze the firmware, take out the httpd service program of arm architecture in the /usr/SBIN folder, load it into IDA, and find the CGI function corresponding to obtaining HTTP protocol parameters in the function window on the left.

Compile hook program, use arm cross compiler environment "arm linux gnuabi GCC -- shared -FPIC hook. C - O hook" command to compile the source code, and get hook shared library program. Execute the command "LD"_Preload =. / hook "AFL Fuzzy - Q - M none - I in - o out / --. / httpd" is tested and found that it can be fuzzed successfully.

At this time, the request module in Web fuzzing is used to continuously request its HTTP service. A uniq crashes is quickly found in AFL fuzzy, as shown in Figure 4. The AFL fuzzy process is interrupted. In the out / crashes directory of the current directory, a file is generated to save the crash content information.

5. **Conclusion**

Nowadays, the security research of IOT devices is a relatively hot topic and research field at home and abroad. More and more intelligent devices begin to enter thousands of households, and the number of connected IOT devices is also increasing. Therefore, vulnerability mining for IOT devices is a key point in the experimental work in the field of information security. In this paper, AFL QEMU mode is applied to the binary fuzzy test of firmware CGI program of IOT devices in firmware fuzzing. This method can effectively test the firmware CGI program, and successfully find the general vulnerability of IOT router equipment. Because this method is used in firmware, the efficiency needs to be improved. Therefore, the next step is to encapsulate the request module from external HTTP request to AFL fuzzy to reduce the difference of rate between external request and fuzzing.

**Acknowledgments**

This work was supported by Natural Science Foundation of Hainan Province under grant no.618MS025 and the National Natural Science Foundation of China under grant no. U19B2044.
References

[1] Knud Lasse Lueth 2019 IoT 2019 in Review: The 10 Most Relevant IoT Developments of the Year. https://iot-analytics.com/iot-2019-in-review/

[2] Xiao Y 2017 D. Research on security detection and optimal configuration of wireless router GuiZhou University

[3] Freebuf 2017 https://www.freebuf.com/news/128963.html

[4] Yaowen Z, Hui W and Kai C 2019 Overview of vulnerability mining technology in IOT. J. Journal of information security pp 61-75

[5] Publications S 2008 J. Journal of the Royal Society for the Promotion of Health

[6] NSA. Ghidra 2019 https://www.nsa.gov/resources/everyone/ghidra/

[7] DEVTTYS0.binwalk[EB/OL] 2019 https://github.com/devttys0/binwalk

[8] Jinbin L, Xiaofei Z and Hui L 2009 Research on symbol execution technology National Symposium on computer security

[9] Afl-fuzz 2019 american fuzzy lop (2.52b). https://lcamtuf.coredump.cx/afl/

[10] Jiongyi C, Wenrui D and Qingchuan 2018 IOTFUZZER: Discovering Memory Corruptions in IoT Through App-based Fuzzing. Network and Distributed Systems Security (NDSS) Symposium 2018 pp 18-21

[11] Nathan Voss 2017 afl-unicorn: Fuzzing Arbitrary Binary Code. https://medium.com/hackernoon/afl-unicorn-fuzzing-arbitrary-binary-code-563ca28936bf

[12] Nccgroup 2017 TriforceAFL https://github.com/nccgroup/TriforceAFL