DICOM-Fuzzer: Research on DICOM Vulnerability Mining Based on Fuzzing Technology

Zhiqiang Wang[1,2,3], Quanqi Li[1], Qian Liu[1], Biao Liu[1(✉)], Jianyi Zhang[1(✉)], Tao Yang[3], and Qixu Liu[4]

[1] Beijing Electronic Science and Technology Institute, Beijing, China
{wangzq,liubiao}@besti.edu.cn, liquanqi_China@163.com, qianniu_l@163.com, nese@163.com
[2] State Information Center, Beijing, China
[3] Key Lab of Information Network Security, Ministry of Public Security, Shanghai, China
[4] Key Laboratory of Network Assessment Technology, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
liuqixu@iie.ac.cn

Abstract. In recent years, the medical equipment and related information systems show the characteristics of mobility, networking, intelligence. At the same time, security incidents caused by medical equipment emerge in an endless stream, which brings a huge threat to the information security of users and causes serious harm. Most medical devices use open source protocol library, which brings great security risks to the digitalization and informatization of medical devices. Therefore, in the face of growing security threats and challenges, it is urgent to study the security of medical equipment. In this paper, the vulnerability mining of DICOM was studied, the most commonly used communication standard for high-performance medical devices, and a vulnerability mining model based on Fuzzing technology was proposed. This model constructed a vulnerability mining environment by simulating PACS system, and implemented a prototype system DICOM-Fuzzer. The system includes initialization, test case generation and other modules, which can complete large-scale automatic testing and exception monitoring. Then, three different versions of the open source library were selected to test the 1000 test cases generated respectively. It was found that when the received file data was greater than 7080 lines, the overflow would occur, resulting in the denial of service of the system. Finally, the security suggestions and repair measures were put forward, and the future research was described.

Keywords: DICOM · Fuzzing · PACS · DCMTK

1 Introduction

With the development of cloud computing, Internet of things, and mobile Internet, medical equipment and related information systems show intelligent, mobile,

H. Gao et al. (Eds.): ChinaCom 2019, LNICST 312, pp. 509–524, 2020.
https://doi.org/10.1007/978-3-030-41114-5_38

networked and other characteristics. Due to the lack of safety awareness of medical equipment manufacturers in the research and development process, the digitization and informatization of medical equipment have huge safety risks.

DICOM (Digital Imaging and Communications in Medicine) is an international standard for medical imaging and related information (ISO 12052). It is the basis of all medical imaging technologies, and it defines medical image formats that can be used for data exchange to meet clinical needs in quality. Conceptually, DICOM mainly includes two aspects: communication and service for medical image. That is to say, DICOM can be understood as a format standard for medical digital image communication. DICOM is widely used in radiological medicine, including cardiovascular imaging and radiological diagnostic equipment (X-ray, CT, MRI, ultrasound, etc.). The DICOM protocol solves the problem of how to associate the information of patients with medical images. However, since DICOM did not involve communication security at the beginning of its design, a large number of security threats and hidden dangers have been introduced into the digitization and informatization process of medical equipment. Since 2008, the medical equipment has exposed a lot of security vulnerabilities and attack events, such as remote attacks on heart pacemakers, insulin pumps, bluetooth defibrillator, X-ray machine, and so on, which have caused serious injury and huge losses to patients. Therefore, facing the growing security threats and challenges, there is an urgent need to conduct research on the security of medical devices.

This paper mainly focuses on the research of vulnerability mining for DICOM, the most commonly used communication standard for high-performance medical equipment. This paper proposes a DICOM vulnerability mining model based on Fuzzing technology, which constructs a DICOM vulnerability mining environment by simulating PACS (Picture Archiving and Communication Systems) system, and constructs test cases by combining strategies based on generation, variation and artificial construction. On this basis, a prototype system DICOM-Fuzzer is implemented, which can complete large-scale automatic test and abnormal monitoring. Finally, a kind of DoS (Denial of Service) vulnerability is found through the test of DICOM parsing library DCMTK, which proves the effectiveness of the model and tool.

The innovations and contributions of this paper are as follows:

(1) A DICOM vulnerability mining model based on Fuzzing technology is proposed for the first time, which combines three strategies to realize test case construction and can improve the efficiency of test case construction.
(2) The prototype system DICOM-Fuzzer implemented in this paper can realize the automatic test of PACS system and greatly improve the test efficiency.
(3) We found the DoS vulnerability of DICOM parsing library DCMTK, which will affect all medical devices using the open source library in a relatively large scope.

The other chapters of this paper are organized as follows: Sect. 2 introduces relevant work; Sect. 3 introduces the architecture of the model. Section 4

introduces the implementation of DICOM-Fuzzer tool. Section 5 introduces the experimental results and analysis. Section 6 introduces the conclusion and future research direction.

2 Related Work

Through the investigation of the research status of DICOM medical protocol, the specific analysis is as follows:

Farhadi et al. investigated the protection of Iran's domestic PACS [11,15], and evaluated the security control of PACS. After analysis by SPSS software, they found that PACS did not record the transmission path, nor use digital signatures or watermarks to protect medical images. This paper evaluates the protection of Iran's medical imaging information system as a whole, and analyzes the information security requirements of the medical imaging information system [5].

Gutierrez-martinez et al. proposed a business model for ISO/IEC 27002:2013 standard and security and privacy standards to improve the information security management level of large-scale PACS, aiming at the lack of effective management mechanism in medical institutions. The method associated with this pattern can be used to monitor the data flow in PACS, thus facilitating the detection of unauthorized access to images and other abnormal activities [6].

Tim Elrod et al. conducted vulnerability mining research on two PACS's protocols DICOM, Health Level 7(HL7) [1,2,4,7,14], and electronic medical information recording system. They used Fuzzing technology and penetration technology to find several security vulnerabilities, including the use of Web browser to enable medical staff to automatically distribute drugs using the dispensing cabinet made by Omnicell, and the use of "forced browsing" attacks giving unauthorized users unauthorized access to the control of a hospital medicine dispenser operated by Integris Heath [10].

Anirudh Duggal conducted attack and defense technology research for HL7 2.X protocol, he mainly used the Fuzzing technology to test HL7 protocol stack, discovering the man-in-the-middle attack, message source and size not validated, denial of service attacks and other security threats, and proposed some security suggestions and threat elimination measures, such as message size verification, enforcing two-way TLS connection, content input filtering, adding the checksum [1].

Codenomicon company issued unknown vulnerability management guidelines for medical equipment, putting forward using the Fuzzing testing technology based on generating to structure test cases, and to detect unknown security vulnerabilities. This guide mainly introduces the testing process of "Analyzing Attack Surface", "Reducing Risk" and "Generating-based Fuzzing". Finally, it recommends safety recommendations for testing and validation for medical equipment manufacturers. Among them, Fuzzing technology [3,9,12,17–22] is an effective flaw injection vulnerability mining method, with a certain degree of randomness and blindness. This guide does not propose a solution to this problem [13].

3 System Architecture and Design

This chapter mainly introduces the system architecture and the design of three core modules: test case construction module, DICOM test module and exception monitoring module.

3.1 System Architecture

The system architecture consists of six modules, including system initialization module, test case construction module, DICOM protocol test module, exception monitoring module, exception verification module and log output module. The core architecture of the system is shown in Fig. 1.

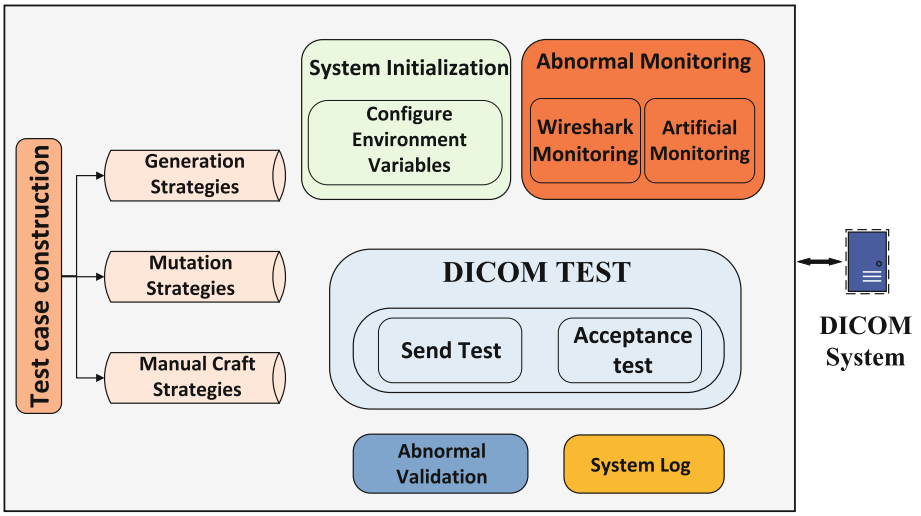

Fig. 1. DICOM vulnerability mining system architecture.

The system initialization module is responsible for configuring the environment required for operation, setting up the open source library DCMTK of DICOM protocol, uninstalling irrelevant object files, and installing object files to be tested. The test case construction module uses three types: generation-based construction, mutation-based construction and manual construction to construct the test case package. The DICOM test module sends the constructed DICOM packet file to the DICOM system to be tested for test. The anomaly monitoring module catches and monitors the test process and analyzes the feedback data concretely. Exception validation module is to re-verify the monitored exception to confirm the existence and availability of the exception. The log output module records the abnormal information that may be generated during the test.

Among them, test case construction module, DICOM test module and exception monitoring module are the core parts of the system, which are described in detail in the following chapters.

3.2 Test Case Construction

Construction Strategy. The construction strategy adopts the test case construction method based on Fuzzing technology. Fuzzing technology is an automated software testing technology. It is one of the important means of vulnerability mining by inputting a large amount of abnormal data to the tested target and monitoring its anomalies to find vulnerabilities. Test case construction strategies are divided into automatic random construction based on generation, analysis construction based on variation, and manual construction, which are described in detail as follows.

Automated Random Construction Based on Generation. This construction strategy adopts the idea of Fuzzing test to construct random DICOM data packet as input, and uses random function to arbitrarily change any value of this data packet, or modifies any bit of this binary DICOM data packet by cyclic modification. This generation strategy can allow the computer to construct a large number of random data packets, which has the advantages of low cost, high degree of automation, and can explore many unexpected vulnerabilities. However, most deformed data packets constructed by this method are invalid and will be abandoned by the equipment to be tested, leading to low efficiency of the final test process.

Analysis Construction Based on Mutation. This strategy based on mutation analysis is more purposeful and targeted than random construction. According to existing vulnerability analysis, most vulnerabilities are prone to appear on the boundary value. For example, normally required input type is int, while the maximum value defined by int can be input when testing. Applying this idea to the construction of packets, it is speculated that setting a byte in the DICOM packet as the boundary value may lead to triggering vulnerability.

Artificial Structure. Artificial construction is to construct test cases purposefully and pertinently through the constructor's understanding of protocol and other contents. Based on experience, the author guesses the place where the vulnerability may be triggered, and sets the numerical value manually. For example, there is a string "DICM" with a length of four bytes in the header file Header of DICOM packet. Changing the string to a value that is not the string "DICM" to observe whether the modified packet can trigger the vulnerability.

After the test case construction is completed, it is verified by sending it to the test program. This is done by constructing the fuzzer. Figure 2 shows the general testing process of the fuzzer.

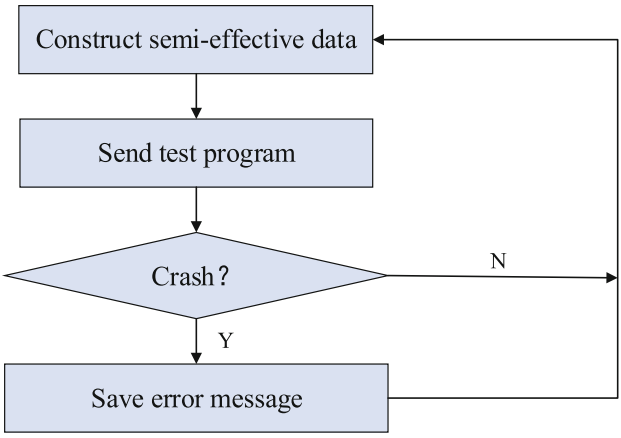

Fig. 2. General testing process for Fuzzer.

Vulnerability Analysis. A DICOM file generally consists of a DICOM file header and a DICOM data set. Introduction to the file, consisting of 128 bytes. DICOM prefix, which can be used to determine whether the file is a DICOM file based on whether the 4-byte string equals "DICM".

The main component of a DICOM file is a data set. Including:

(1) Tag: an ordered pair of 16-bit unsigned integers, with the first 8 bits representing the group number and the last 8 bits representing the element number.
(2) Value representation: indicating the data type in the data element.
(3) Value length: an unsigned 16-bit or 32-bit integer representing the length of the data field.
(4) Data domain: the data type of the value that exists in this field is determined by the value representation of this data element, and its storage length is even bytes.

Analyzing the file structure in Fig. 3, and paying attention to the parts of "value length" and "value range". It can be seen that the ultra-long value length and the transmission of odd bytes of data may lead to buffer overflow and other anomalies:

Extra Long Value Length. When implementing the DICOM protocol, programmers may assume that an unsigned integer with a value length of 16 or 32 bits in the DICOM file structure is sufficient to meet the usage requirements. This approach is not rigorous and scientific, and may cause some unexpected consequences. Therefore, the future vulnerability mining can start from this aspect.

Odd Data Field. As mentioned above, the data type of the value stored in the data domain is determined by the value of the data representation, and its

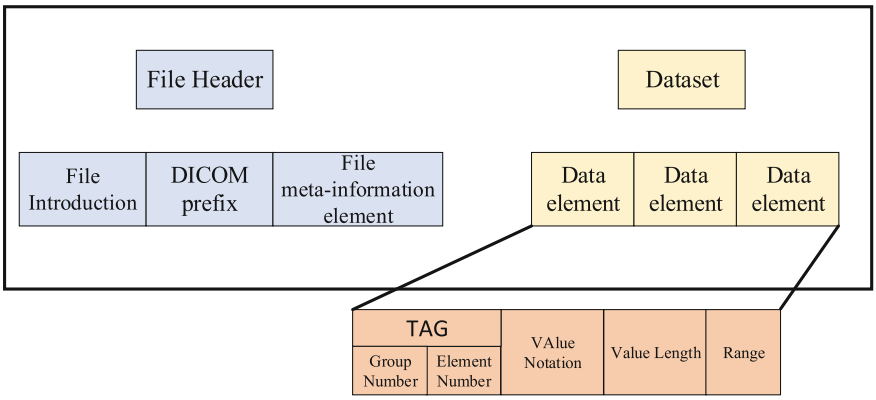

Fig. 3. DICOM file structure diagram.

storage length is an even number of bytes. If the data stored in data domain is odd in the process of data transmission, whether the DICOM protocol still can correctly handle the situation is one of the areas that the tester concern and needs to test.

3.3 DICOM Test

PACS is a system applied in the hospital imaging department, the main task is to digitally store everyday of all kinds of medical images (including the MRI, CT, ultrasound, X-ray machines, infrared instrument, the microscopic instrument equipment) through various interfaces (analog, DICOM, network), when needed, it can be used quickly under certain authorization back to use, adding a few auxiliary diagnosis and management functions. Because of the numerous interface categories of medical imaging devices and the large amount of data generated every day, how to transfer data between various imaging devices and how to organize the storage of data are crucial to the system.

The benefits brought to hospitals by PACS are obvious, including the following aspects: (1) reduction of material cost. (2) reduction of management costs. (3) improve work efficiency. (4) improve the medical level of hospitals. (5) provide resources accumulation for hospitals. (6) make full use of our hospital resources and other hospital resources.

The relationship between the test case and the PACS system is shown in Fig. 4. The purpose of fuzziness testing is to test whether various network protocols or related applications have security vulnerabilities. Fuzzing testing is a well-known black box technology for application security testing [16]. The principle is to send test data in a specific format to the target through the Socket API and monitor the abnormities in the target [8,16]. The testing process can be roughly divided into five steps [16]. First, identifying the target to be tested and getting more details about the target. Second, determining the input and

potential variables, such as file headers, file names, environment variables, and so on. Then configuring the target to be ready for testing. After generating fuzzy semi-valid data, using the fuzzy semi-valid data as input to execute the program and monitor for exceptions. Try to find application vulnerabilities and quickly find vulnerabilities such as large buffer overflows and formatting strings.

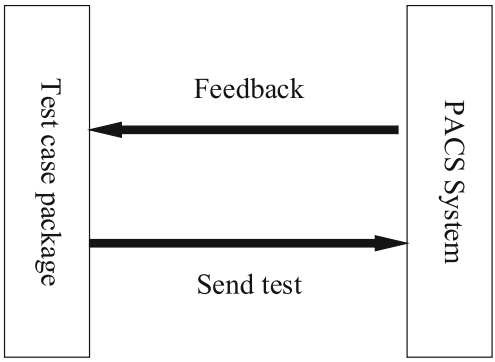

Fig. 4. Test structure diagram.

Fuzzy half valid data is for some applications, most of the data is valid, when testing, the application will first consider it to be valid data, but because of the rest of the invalid data, so there is a high probability of compiling or executing exceptions when a program processes data. Such exceptions include access cross-border, data overflow and so on, which eventually lead to application crash, delay, and so on. Figure 5 shows the general testing process for Fuzzing.

3.4 Abnormal Monitoring

In this paper, the abnormal monitoring mainly adopts two means, including Wireshark monitoring and manual monitoring.

Wireshark Monitoring. Wireshark is the world's foremost and widely-used network protocol analyzer. It lets you see what's happening on your network at a microscopic level and is the de facto (and often de jure) standard across many commercial and non-profit enterprises, government agencies, and educational institutions.

Wireshark has a rich feature set which includes the following:

- Deep inspection of hundreds of protocols, with more being added all the time.
- Live capture and offline analysis.
- Standard three-pane packet browser, etc.

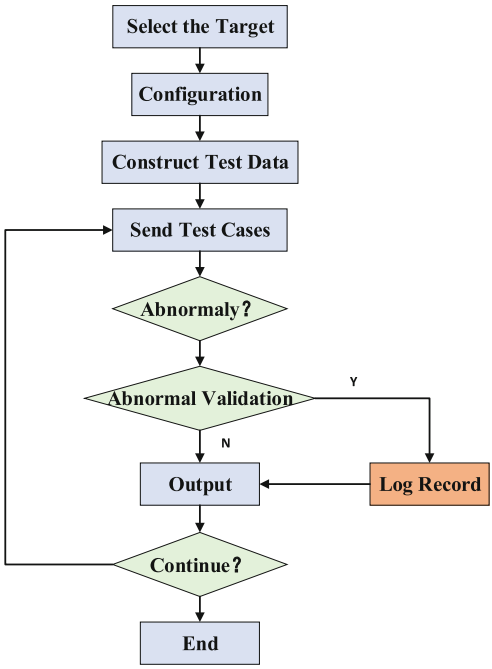

Fig. 5. Flow chart of test program.

In this article, you can use Wireshark to monitor the vulnerability mining system under test. In the running environment of vulnerability mining system, after opening Wireshark, you can easily select the information you want to capture, which can help analyze the cause of this exception by analyzing and reading the captured packet data. Wireshark can select the Npcap Loopback Adapter interface to capture various protocol packages, and then select DICOM protocol in the capture filter to capture the DICOM protocol packages, and conduct the required exception analysis through manual analysis.

Artificial Monitoring. Manual monitoring means that security personnel conduct penetration testing on the protocol according to their experience and monitor possible abnormal results. Due to the participation of human beings, the identification and reliability of manual monitoring are relatively high. However, its disadvantage is that it is purely manual monitoring, which cannot find loopholes thoroughly and requires the participation of engineers with rich experience.

4 Experiment

In this paper, different versions of DCMTK open source library are selected to conduct DICOM vulnerability mining test, and methods such as test case construction, fuzzy test and exception monitoring are introduced.

4.1 DCMTK Library

DCMTK is a collection of libraries and applications implementing large parts the DICOM standard. It includes software for examining, constructing and converting DICOM image files, handling offline media, sending and receiving images over a network connection, as well as demonstrative image storage and worklist servers. DCMTK is written in a mixture of ANSI C and C++. It comes in complete source code and is made available as "open source" software.

DCMTK is one of the three open source libraries of DICOM, which is an open source library with C/S architecture, namely Client/Server structure. SCP (Service Class Provider) is the Server, which is responsible for providing various services for image data and playing the role of Server. The SCU (Service Class User) is the Client, which is the party using these services. DCMTK is an open source library based on C++. Its basic structure is a number of packages, and this article mainly uses the content of DCMNET. The DCMNET package is a network library and available tools. This module contains all function sets to realize DICOM network communication, namely, upper finite state machine of DICOM, the element of association control service, and the element of DICOM message service. The purpose of this chapter is to successfully build a server that can receive DICOM files from the client and simulate a DICOM running environment. DCMTK is used to prepare for the smooth implementation of the design scheme of the following vulnerability mining system.

In order to verify the effectiveness of the Fuzzing technology-based DICOM protocol security analysis method proposed in this paper, it is tested and analyzed for DCMTK3.5.3, DCMTK3.5.4, DCMTK3.6.0–DICOM open source protocol library.

4.2 DCMTK Test

This system runs under a virtual machine configured with Intel(R) Core(TM) i7-8750h dual processor of 2.20 GHz and 2.21 GHz, memory of 2 GB, and the system version is WIN7.

In this paper, three different versions of DCMTK3.5.3, DCMTK3.5.4 and DCMTK3.6.0 of DICOM open source protocol library DCMTK were selected for comparison test, and 1000 test cases were respectively used to perform Fuzzing test on the above three different versions of open source libraries.

The test is based on the analysis of the DICOM protocol. Besides, it also includes basic programming language knowledge such as reading and writing files, loops, sockets, random number and so on. In the specific Fuzzing, the writing of the data packet sending and receiving test program mainly uses Socket to send data. Through Socket, a computer can accept the data of other computers and send data to other computers.

```
sockaddr_in sockAddr;
memset(&sockAddr, 0, sizeof(sockAddr)); // Each byte is padded with zero
sockAddr.sin_family = PF_INET; // IPv4 address
sockAddr.sin_addr.s_addr = inet_addr("127.0.0.1"); // IP address
sockAddr.sin_port = htons(1234); //Port
```

The Fuzzing test of the DICOM open source protocol library DCMTK3.5.3 is shown in the Figs. 6 and 7.

Fig. 6. DCMTK3.5.3 SCP receiving test.

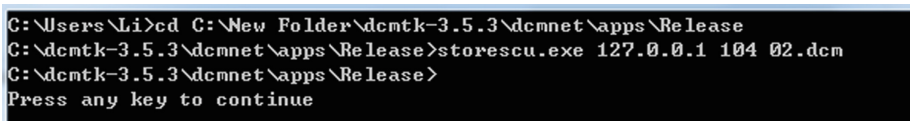

Fig. 7. DCMTK3.5.3 SCU send test.

As shown in Figs. 6 and 7, the test environment runs successfully. Figure 6 is the server-side SCP, and Fig. 7 is the client-side SCU. It can be seen that the SCU successfully sends DICOM files (02.dcm) to SCP, and the SCP receives them successfully. The port selected by SCU is 104, and the IP address selected is 127.0.0.1.

The Fuzzing test of the DICOM open source protocol library DCMTK3.5.4 is shown in the Figs. 8 and 9.

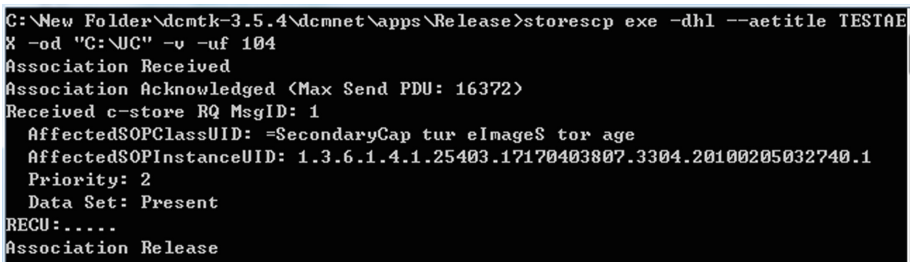

Fig. 8. DCMTK3.5.4 SCP receiving test.

The Fuzzing test of the DICOM open source protocol library DCMTK3.6.0 is shown in the Figs. 10 and 11.

We use the program to automatically generate a large number of DICOM file test cases, as shown in Fig. 12:

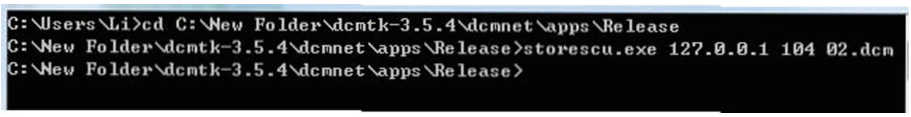

Fig. 9. DCMTK3.5.4 SCU send test.

Fig. 10. DCMTK3.6.0 SCP receiving test.

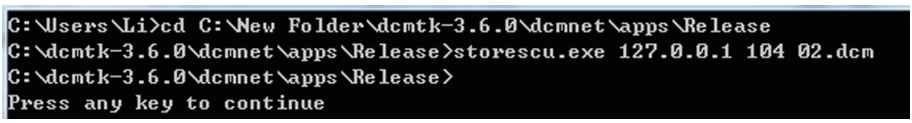

Fig. 11. DCMTK3.6.0 SCU send test.

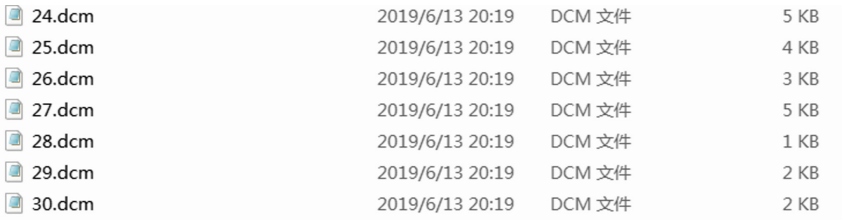

Fig. 12. Generate test cases of DICOM type.

5 Results and Analysis

The code for the Fuzzing test was successfully run, compiled, and executed on the server side. Automated testing after the server successfully receives the file.

After a large number of test cases are constructed and the system is repeatedly tested for a long time. A large number of data packets were generated, with valid input and many invalid input. One of the typical data packets was selected for testing.

An exception was found during the Fuzzing test of the DICOM open source protocol library DCMTK3.5.4.

As shown in Fig. 13, the system exception is caught in the process of fuzzy test of DCMTK3.5.4. The file of fuzzy test has been successfully sent to the server side of DCMTK3.5.4. After that, it can be found that the originally opened server side automatically stops working, the system crashes causing anomalies, and SCP stops working. This test resulted in a system crash caused by the bug. During the test run, Wireshark software was used to record and monitor packages.

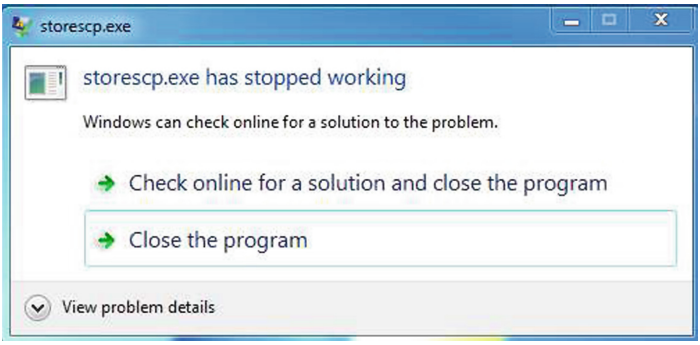

Fig. 13. Abnormal test results of DCMTK3.5.4.

Through packet capture analysis, it can be seen that the DICOM file has been transferred, that is to say, the DICOM file has been sent from the client to the server side, and then the server side crashes, so it is preliminarily determined that the problem is caused by the constructed test case.

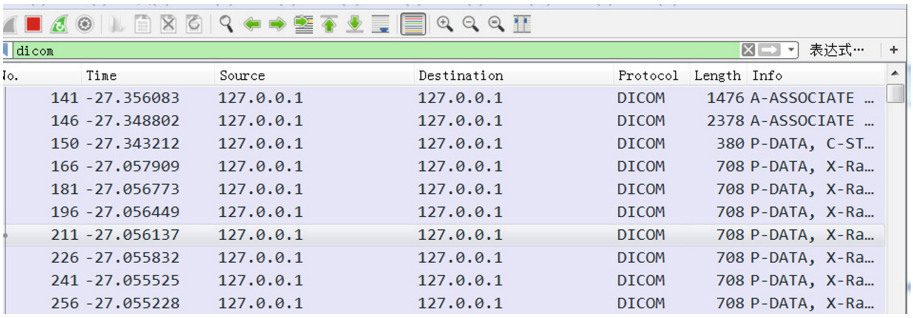

Fig. 14. Wireshark interface.

Starting the server and client of DCMTK3.5.4, opening Wireshark, selecting the required monitoring interface, and running the command "storescu.exe 127.0.0.1 104 test1" on the client. As shown in Fig. 14, the monitoring data of the system is obtained. In addition, regular expressions can be modified to further filter specific error messages, as shown in Fig. 15.

After the Fuzzing test were performed on DCMTK3.5.3 and DCMTK3.6.0, no PACS system exception were found after the test data were sent (no vulnerability were found). The summary of the experimental results is shown in the Table 1.

The experiment is the first to design vulnerability mining framework and the test of Fuzzing for DICOM protocol. This experiment verified the vulnerability mining framework proposed in Sect. 3.1. A total of three different versions of DCMTK were selected for Fuzzing testing, and comparisons were also made during the testing process. Among them, abnormalities were found during the

Table 1. Test results of three different versions of DCMTK.

DCMTK versions	Number of test cases sent	Fuzzing test results
DCMTK3.5.3	1000	No abnormalities found
DCMTK3.5.4	1000	Abnormalities found
DCMTK3.6.0	1000	No abnormalities found

DCMTK3.5.4 version testing process. No exceptions were found for the other two DCMTK versions (DCMTK3.5.3 and DCMTK3.6.0). During the test of DCMTK3.5.4, and it was found that when the data in the constructed data package was larger than 8070 rows, the system would crash, and the reasons for the system crash were preliminarily analyzed as the large DICOM data package and data overflow. It is recommended that developers check the length of data capacity when developing software. Avoid the harm such as denial of service, attack, process crash and outage of PACS system caused by vulnerability.

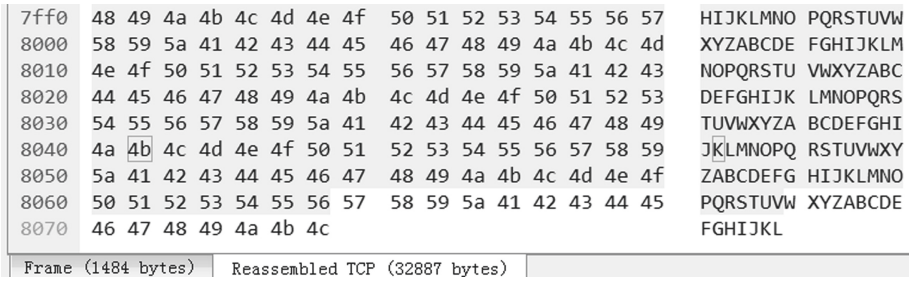

Fig. 15. Test case that caused storescp to crash.

6 Conclusion

DICOM is an international standard for medical images and related information (ISO 12052). DICOM solves the problem of how to associate patient information with medical images. Therefore, DICOM is widely used in radiation medicine, including cardiovascular imaging and diagnostic equipment. However, since DICOM did not involve communication security at the beginning of design, a large number of security threats and hidden dangers were introduced in the process of digitalization and informatization of medical equipment. In this paper, we studied the vulnerability mining of DICOM open source database, and proposed the framework of vulnerability mining for DICOM for the first time. We used the generated and manually constructed strategies to construct test cases, and built a security test system of DCMTK open source database, simulated the transmission process using DICOM protocol, and Fuzzing test different versions of the DCMTK open source database. The purpose of Fuzzing testing is

to test whether various network protocols or related applications have security vulnerabilities. Fuzzing testing is a well-known black box technology for security testing. Fuzzing technology is a kind of automatic software testing technology. It is one of the important means of vulnerability mining by inputting a large number of abnormal data to the tested object and monitoring its abnormalities. Secondly, through manual monitoring and Wireshark monitoring, the whole test process was monitored for exceptions. A total of three different versions of the DCMTK open source library were selected for Fuzzing test, and a comparison was made during the test. Among them, during the testing of the DCMTK3.5.4 version, the vulnerability caused by the constructed packet overflow was found. No exceptions were found for the other two DCMTK versions (DCMTK3.5.3 and DCMTK3.6.0). The overall test effect is satisfactory. The designed vulnerability mining system based on Fuzzing technology can dig out the vulnerabilities of DICOM protocol or the logic or design vulnerabilities existing in the system when using DICOM protocol to some extent. At last, some suggestions on development and repair are proposed to software developers, and it is hoped that the experimental results can provide references for the safe development and utilization of medical imaging applications.

In general, the overall framework of this paper and the method of exploiting vulnerabilities by using Fuzzing technology are beneficial attempts to exploit vulnerabilities of DCIOM protocol. Especially in the current situation of the increasingly severe security situation in the form of network security, especially in the medical field, the results of vulnerability mining of DICOM protocol can provide references for medical device developers, and to some extent reduce the possible security risks of DICOM protocol. Currently, Artificial Intelligence (AI) technology is developing rapidly, and work efficiency is greatly improved in the process of generating and using large data, especially the application effect of generated adversarial network in large data is gratifying. Next, we will continue to perfect and improve the experimental methods, furthermore, the test case construction, screening and testing are optimized and reformed in combination with the generated adversarial network. By combining the static vulnerability mining and artificial intelligence to improve the efficiency of vulnerability mining as the future research direction, and the security vulnerabilities in DICOM protocol will be further explored in the future.

Acknowledgments. This research was financially supported by the National Key Research and Development Plan (2018YFB1004101), Key Lab of Information Network Security, Ministry of Public Security (C19614), Special fund on education and teaching reform of Besti (jy201805), the Fundamental Research Funds for the Central Universities (328201804, 328201910), key laboratory of network assessment technology of Institute of Information Engineering, Chinese Academy of Sciences.

References

1. Duggal, A.: Hl7 2.x security. In: The 8th Annual HITB Security Conference (2017)

2. Blazona, B., Koncar, M.: Hl7 and DICOM based integration of radiology departments with healthcare enterprise information systems. Int. J. Med. Inform. **76**, S425–S432 (2007)
3. Chen, Y., Wang, Z.: Progress in fuzzy testing. Comput. Appl. Softw. **28**(7), 291–293 (2011)
4. Dolin, R.H., et al.: Hl7 clinical document architecture, release 2. J. Am. Med. Inform. Assoc. **13**(1), 30–39 (2006)
5. Farhadi, A., Ahmadi, M.: The information security needs in radiological information systems–an insight on state hospitals of Iran, 2012. J. Digit. Imaging **26**(6), 1040–1044 (2013)
6. Gutiérrez-Martínez, J., Núñez-Gaona, M.A., Aguirre-Meneses, H.: Business model for the security of a large-scale PACS, compliance with ISO/27002: 2013 standard. J. Digit. Imaging **28**(4), 481–491 (2015)
7. Hasman, A., et al.: Hl7 RIM: an incoherent standard. In: Ubiquity: Technologies for Better Health in Aging Societies, Proceedings of Mie 2006, vol. 124, p. 133 (2006)
8. Liu, Q., Zhang, Y.: TFTP vulnerability mining technology based on fuzzing. Comput. Eng. **33**(20), 142–144 (2007)
9. Luo, Y.: Design and implementation of network security vulnerability scanning system. Ph.D. thesis, National University of Defense Science and Technology, Chang-Sha (2007)
10. Elrod, T., Morris, S.: I'm not a doctor but i play one on your network (2011)
11. Nagy, P., Bowers, G., Reiner, B.I., Siegel, E.L.: Defining the pacs profession: an initial survey of skills, training, and capabilities for PACS administrators. J. Digit. Imaging **18**(4), 252–259 (2005)
12. Pianykh, O.S.: Digital Imaging and Communications in Medicine (DICOM): A Practical Introduction and Survival Guide. Springer, Heidelberg (2009)
13. US Food and Drug Administration: Content of premarket submissions for management of cybersecurity in medical devices: draft guidance for industry and food and drug administration staff (2013). Accessed 1 May 2014
14. Vossberg, M., Tolxdorff, T., Krefting, D.: DICOM image communication in globus-based medical grids. IEEE Trans. Inf. Technol. Biomed. **12**(2), 145–153 (2008)
15. Wiese, M., Beck, K., Tschöpel, E., Reindl, P., Carl, P.: PACS-picture archiving and communication system. Der Urologe B **39**(3), 237–244 (1999)
16. Xu, Y.: Research and implementation of fuzzing test technology for streaming media protocol. Ph.D. thesis, Beijing University of Posts and Telecommunications (2009)
17. Zhang, B., Zhang, Y., Xu, Y.: Exploring network protocol vulnerabilities based on fuzzy testing. J. Tsinghua Univ.: Nat. Sci. Ed. **S2**, 2113–2118 (2009)
18. Zhang, G., Shi, X., Li, R., Ren, J.: Fuzzy test optimization scheme for NFC protocol. Hebei Ind. Sci. Technol. **34**(3), 155–161 (2017)
19. Zhang, X., He, Y.: Overview of software testing methods. Sci-tech horizon (4), 35–37 (2012)
20. Zhang, Y., Wang, Z., Liu, Q., Lou, J., Yao, D.: Research progress and development trend of near-field communication technology security. J. Comput. Sci. **39**(6), 1190–1207 (2016)
21. Zhuang, T.: The Application of Computer in Biomedicine. Science Press, Beijing (2000)
22. Zou, Q., et al.: From automation to intelligence: advances in software vulnerability mining technology (2018)