Vulnerability Discovery Technology and Its Applications

Zhunyang Pan
National Digital Switching System Engineering & Technological Research Center, Zhengzhou, China
Email: per_kong@126.com

Caixia Liu, Shuxin Liu and Shuming Guo
National Digital Switching System Engineering & Technological Research Center, Zhengzhou, China
Email: lcxtxr@163.com, liushuxin11@126.com, gsm@mail.ndsc.com.cn

***Abstract*—Vulnerability discovery technology becomes more and more important in software development and network security. This paper presents the classification of vulnerability discovery technologies and discusses the advantages, disadvantages and the extent of application of each class. Then we emphasize the procedure and the improvement methods of the Fuzzing test combined with protocol analysis. Furthermore, according to protocol vulnerability discovery methods, we analyze the issues of network vulnerability discovery and propose the directions of future research.**

***Index Terms*—vulnerability discovery, fuzzing test, protocol analysis, network vulnerability**

I. INTRODUCTION

People should attach great importance to the security of any communication product. There are vulnerabilities in almost all kinds of software and system, and the attackers tend to find them within a short time. Then the attackers may use the vulnerabilities to steal data and information or to make the target work abnormally or even crash. Therefore, it is very important to discover the vulnerabilities and remove them before the attackers make use of them.

Vulnerabilities refer to the defects of the system, which are generated in the design and development including pre and post configuration application. They affect the safety performance of the system seriously. Generally, vulnerabilities are caused by nonstandard writing of the procedure or imperfect system architecture. Reference [1] elaborates the definitions of information security vulnerabilities in different periods.

So far, people have studied vulnerabilities in many IT fields.

Reference [2] discussed and studied the vulnerabilities of the computer system. It introduced the present condition of network security, researched computer vulnerabilities and exposures (CVE) in depth from the manifold point, and established the Chinese version of the CVE standard. G. Lorenz and his colleagues studied the vulnerabilities of SS7 telecommunications network and presented an attack taxonomy. They also described the architecture of a system for detecting and responding to SS7 network attacks [3]. Reference [4] analyzed test model for security vulnerability in Web controls and put forward an improved test model. Reference [5] made a model-based vulnerability analysis of IMS network. It established a comprehensive vulnerability analysis model of IMS network based on TVRA method and made a systematic analysis of it. Ronald W. Ritchey and the others pointed out the vulnerabilities which come from the multi-host architecture in the network. They also analyzed it using model checking [6]. Frank Piessens proposed a structured taxonomy of software vulnerabilities, analyzing the common vulnerabilities of the software and classifying the most frequently occurring causes of vulnerabilities to avoid common pitfalls [7]. Reference [8] discussed the methods for the prevention, detection and removal of software security vulnerabilities. It gave a brief description of the source code security checkers available to partially automate security analysis of the software and made a discussion of functional programming techniques. Simon Byers and his colleagues performed a brief analysis of the movie production and distribution process and identified potential security vulnerabilities that may lead to unauthorized copies becoming available to those who may wish to redistribute them. They also offered recommendations for reducing security vulnerabilities in the movie production and distribution process [9].

In addition, people refine the object to analyze the vulnerabilities of a particular software or system to study other similar products. For example, Matt Bishop studied the vulnerabilities of the UNIX system and network and made an analysis of how to use Protection Analysis to improve the security of existing systems, and how to write programs with minimal exploitable security flaws [10].

The research of vulnerabilities is greatly helpful to improve the security performance of the system. It also plays an important role in improving network security. The research of vulnerabilities mainly includes vulnerability discovery, vulnerability analysis and vulnerability exploitation.

Vulnerability discovery

Vulnerability discovery refers to exploring and finding out the potential vulnerabilities of the system using different kinds of detection technologies and exploration tools.

doi:10.4304/jsw.8.8.2000-2007

Vulnerability analysis

Vulnerability analysis refers to analyzing the vulnerabilities which have been discovered, evaluating the threat levels and the utilization value, determining the effect they may result in and providing a basis for the following patching.

Vulnerability exploitation

Vulnerability exploitation means the concrete exploitation of the vulnerabilities that have been discovered, including both the attack using the vulnerabilities and the defense against the vulnerabilities.

Logically, vulnerability discovery is the basis of vulnerability analysis and vulnerability exploitation. At the same time, it is the prerequisite of concrete analysis and assessment.

This paper discusses the classification, advantages and disadvantages of vulnerability discovery technologies and also introduces some applications of the current ones. Nowadays we rely on communication network increasingly. And the number of terminal equipment which is connected to the network is very large. If there are vulnerabilities in the network, it will be a great threat to the security of personal information. Vulnerabilities of the network commonly occur on the communication protocols or the network architecture. So here we discuss the popular technologies of protocol vulnerability discovery and network vulnerability discovery.

II. STUDIES AND CLASSIFICATION OF VULNERABILITY DISCOVERY TECHNOLOGIES

A. Studies of Vulnerability Discovery Technologies

In recent years, vulnerability discovery technology becomes more and more important in the fields of software and system. It has been an indispensable aspect. The security personnel can find out the vulnerabilities of the system using vulnerability discovery technology and remove them in time in order to ensure the safety of the system. The ideal goal of vulnerability discovery technology is to detect the vulnerabilities totally automatically and to adjust adaptively. However, there are several problems in the current detection technologies, such as unique target, low efficiency and high rate of false alarm and missing report.

For the reasons above, people research vulnerability discovery technology constantly to overcome such problems. Ziyad S. AI-Salloum and his colleagues proposed a link-layer-based vulnerability discovery method to probe vulnerabilities within an enterprise network [11]. Reference [12] proposed a new Fuzzing method using multi data samples combination. Sung-Whan Woo and the others analyzed the vulnerability discovery process in web browsers and presented a quantitative characterization of browser vulnerabilities [13]. Andy Ozment pointed out that many software vulnerability discovery processes are unsound. He proposed a standard set of definitions relevant to measuring the characteristics of vulnerabilities and discovery processes and described the theoretical requirement of the vulnerability discovery models [14]. Omar H. Alhazmi and the others described and evaluated some new vulnerability discovery models for major operating systems. They also discussed the applicability of the proposed models and the significance of the parameters involved [15]. Reference [16] proposed a new Weibull distribution based on vulnerability discovery model and compared it with the existing AML Model.

B. Classification of Vulnerability Discovery Technologies

Table 1 shows the classification results under different rules.

TABLE I.
CLASSIFICATION RESULTS UNDER DIFFERENT RULES

Rules	Operation of the target	Mastery of the target
Classification results	Static analysis	White-box testing
	Dynamic analysis	Black-box testing
	Combined analysis	Gray-box testing

Vulnerability discovery technologies can be classified into static analysis and dynamic analysis according to the operation of the target [17]. Furthermore, we can also combine them. Reference [18], for example, proposed a static program analysis assisted dynamic software vulnerability discovery method.

Static Analysis

Static analysis refers to the analysis in which we do not need the actual operation of the target, but only do with the mastered data of the target. We just detect the logical problems, grammatical mistakes and implementation issues by analyzing the source code or the structure systems.

Static analysis does not need the target to be in motion. It has such advantages as simple operation and high detection speed. And if we discover the vulnerabilities, we could ascertain the cause quickly and remove them timely.

However, static analysis has the following disadvantages. First, it has a high degree of dependence on the data of the target. We need to have the source code of the target or the assembly code which are gained by disassembling. Second, it needs a testing rule base with a high degree of coverage to reduce the rate of missing report. It also needs to update the testing rule base constantly. Third, it always has a high rate of false alarm, so it will take a lot of manpower to screen the result and the analyst has to have a good work experience. In addition, static analysis lacks of the ability to discover the vulnerabilities which may emerge during the execution process. However, the source code is not always open. So this analysis method has certain limitations. Generally, it is used to test and improve the safety performance of the software by the manufacturer. Although we can get the assembly code or the scripting language of the procedures by taking advantage of reverse engineering, it will cost too much time, manpower and resource and need experienced researchers.

Static analysis technology [1] mainly includes lexical analysis technology, data flow analysis technology, symbolic

execution technology, model checking technology and so on [19].

Dynamic Analysis

Dynamic analysis refers to recording and analyzing the status and the output data of the target during the run session to find out the vulnerabilities.

It monitors the status of the target during the run session and discovers the vulnerabilities by detecting and analyzing the abnormal situation. The rate of false alarm, in this case, could be relatively low. Besides, we can input the testing data to the target during its running and analyze its output or response to detect the vulnerabilities. In this case, we need to construct suitable testing data. Otherwise, there will be too many false alarms and missing reports, leading to inefficiency.

Inputting and tracking testing method, stack comparison method and fault injection analysis are common dynamic analysis methods [20].

In addition, we could also divide vulnerability discovery technology into white-box testing, black-box testing and gray-box testing according to the mastery of the target [21]. Detecting the vulnerabilities when the source code or design information is mastered is called white-box testing. If we do not have the original materials, we could only input testing data and observe the results. The detecting in this situation is called black-box testing [22]. And the situation of the gray-box testing ranges between the two cases above. In this case, we do not have the original materials, but we can get the assembly code or other information through reverse engineering.

III. PROTOCOL VULNERABILITY DISCOVERY

There are many kinds of vulnerabilities in network protocols [23]. People have made certain progress in the field of protocol vulnerability research, but generally for a specific protocol. Reference [24], for example, studied the vulnerabilities of EMAP (an efficient radio frequency identification mutual authentication protocol). Reference [25] detected the vulnerabilities of SIP in the IMS network and established vulnerability discovery models on the basis of Fuzzing.

We can use static analysis method, dynamic analysis method or their combination to detect the vulnerabilities of the protocols. Static analysis method for protocols is mainly dependent on the source code of the protocol. It analyzes the realization of the related field of the source code to find out the security problems of the code. Meanwhile, it combines the corresponding functions in the specific field to discover the vulnerabilities of the protocol. However, since the protocol is always being used to carry relevant information or to achieve particular control in the communication process, it relates to specific operation of the protocol in this process. Therefore, using dynamic test to discover protocol vulnerabilities is more intuitive and specific. Currently, most vulnerability discovery technologies for protocols are based on dynamic analysis or utilize dynamic analysis for auxiliary proving. And utilizing Fuzzing to detect vulnerabilities of the protocols is a common method of dynamic testing.

A. Fuzzing Test

Fuzzing was first proposed by Miller B.P. and his colleagues [26]. And it was first used to find out reliability problems. Then it was widely used in the fields of vulnerability discovery. Microsoft always detects the vulnerabilities of its products before formally pushing them to the market. And 20% to 25% of the security vulnerabilities are discovered using Fuzzing [27].

The idea of Fuzzing test comes from black-box testing. It is less demanding on the mastery of the source code or other original data of the test object. It constructs the test cases according to certain formation rules (changing the value of the data or increasing the length of the data randomly). Then input the cases into the external interface of the target, analyze the running condition of the target to judge whether the target works abnormally or not, and then find out the vulnerabilities. Generally, the cases constructed are malformed and aggressive. They are semi-effective to be distinguishable and processible. And the cases, at the same time, would make the target fail in the course of processing the malformed part and work abnormally or even crash. The flow of Fuzzing is shown in Fig. 1.

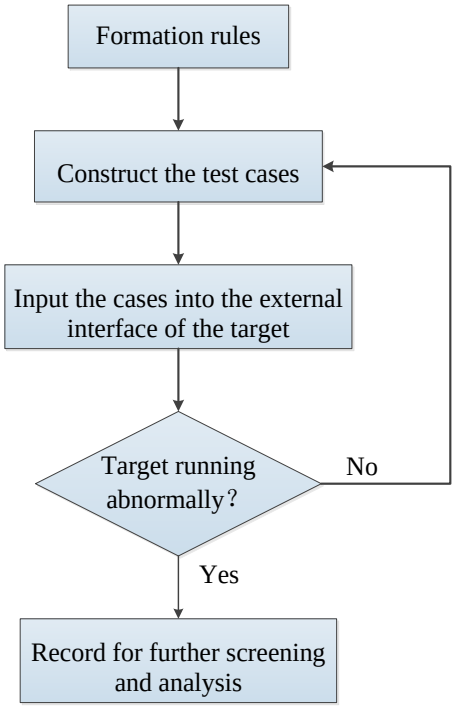

Figure 1. The basic flow of Fuzzing test

We simulate the network communication flow when we detect the vulnerabilities of the protocol. We construct the malformed protocols as the testing cases and simulate the interaction between the entities and monitor the running state of the target. If the result is abnormal, it will be recorded for further screening and analysis. If the system does not process the cases, the cases will be discarded and new testing cases will be constructed. Vulnerability discovery using Fuzzing test has several advantages. On one hand, we do not have to know the source code or the working details of the system. On the other hand, this method has a high degree of automation.

However, protocol vulnerability discovery using Fuzzing is inefficient. Fuzzing test constructs the malformed protocols mainly through changing the length of the packet and the characters in certain positions. Cases constructed in this way may be more likely to be discarded. Then it will take a long time to do the testing.

In order to solve such problems, researchers combine protocol analysis [28] or specific algorithms with Fuzzing to increase the processed probability of the semi-effective cases.

B. Fuzzing Test Combined with the Protocol Analysis

Before we start protocol vulnerability detection, we can analyze the protocols first. Then we may construct suitable testing cases. Protocol analysis refers to the research and analysis of the structural features of the protocol, the environment and the way of use, the specific function and realization method. It includes resolving the meaning and function of every field in the protocol. There is a standard form of a protocol. Through analyzing the protocols, we can distinguish the functions of each part of the protocol and ascertain the changeable part and the unchangeable part. In the construction process of the test cases, changing certain data and keeping the fixed part according to the results of protocol analysis can reduce the blindness of construction and increase processed probability. Fuzzing test flow combined with protocol analysis is shown in Fig. 2.

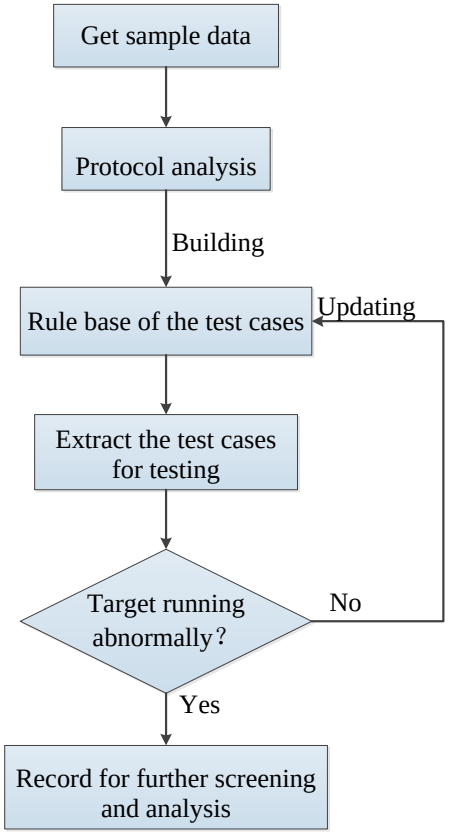

Figure 2. Fuzzing test flow combined with protocol analysis

Thus, the rule base of the test cases is built by analyzing the content of the packet in the workflow before the malformed protocols are constructed. Then extract the test cases for testing. It will record and wait for further analysis if the target works abnormally. And if the target works normally or does not respond, the results will be fed back to the database for updating and there will be a new round of testing.

For example, in view of SIP (Session Initiation Protocol, an application layer signaling control protocol based on text) vulnerability discovery, we can first resolve it for a certain environment (Fig. 3 is a SIP parsing example). Then we change the value of each field according to its meaning and send the changed protocol. And the last step is to record and analyze the relevant response. For example, to detect the network resource vulnerabilities of SIP, we may change the value of Max-Forward or CSeq field, send the changed protocols as test messages, record and analyze the changes of system processing delays.

```
Request-Line: REGISTER sip:192.168.1.2 SIP/2.0
Message Header
Via: SIP/2.0/UDP 192.168.1.3:6036;branch=z9hG4bK-d87543-f561ab6c7a0dc50f
-1--d87543-;rport
Max-Forwards: 70
Contact: <sip:10085@192.168.1.3:6036;rinstance=f95bf222099cd7be>
To: "10085"<sip:10085@192.168.1.2>
From: "10085"<sip:10085@192.168.1.2>;tag=82211425
Call-ID: c103a0532b36e029@Z3NobC1QQw..
CSeq: 1 REGISTER
Expires: 3600
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE,
SUBSCRIBE, INFO
Supported: eventlist
User-Agent: eyeBeam AudioOnly release 3015c stamp 27106
Content-Length: 0
```

Figure 3. SIP parsing example

Analyzing the protocols before constructing the cases can improve testing efficiency significantly and reduce the blindness of the testing. At the same time, we can build the rule base of a certain protocol to provide basis for vulnerability discovery of similar protocols in future work.

The problems in the Fuzzing test based on protocol analysis are as follows.

The realization of the protocol analysis process

Analyzing and finding out the changeable part of the protocols automatically and completely are the key to construct the cases successfully. It will directly determine the efficiency and the rate of missing report. Protocol analysis always relies on the analysts, because the results of the automated analysis often have a high rate of missing report.

The establishment of the testing rule base

Each protocol has its own format and transmission mode so that we need to select an appropriate rule base for a certain protocol. So in order to make this method generally

applicable, it will consume a large resource to establish the rule bases.

The discovery of vulnerabilities in the fixed part of the protocol

This kind of method can distinguish the so-called fixed part through protocol analysis to improve the efficiency of vulnerability discovery. However, in some networks, the changes of the fixed part may also cause abnormal work of the target network. In this case, there will be a missing report.

C. Fuzzing Test Combined with Specific Algorithms

We can also add constraints to the formation of the testing cases by combining them with some algorithms. The general idea of this method is to find a suitable algorithm according to the protocol type and its characteristics first, and then to determine the variable value in the algorithm. We can use statistic methods to determine the value. The first step is controlling the generation of the test protocols by changing the value for many times. Then we observe and analyze the statistic results of the generated test messages being processed and determine the optimal value of the variable. The process of determining the best variable value is shown in Fig. 4. After determining the algorithm and its variables, the generation of test message can be limited specifically and quantitatively to improve testing efficiency.

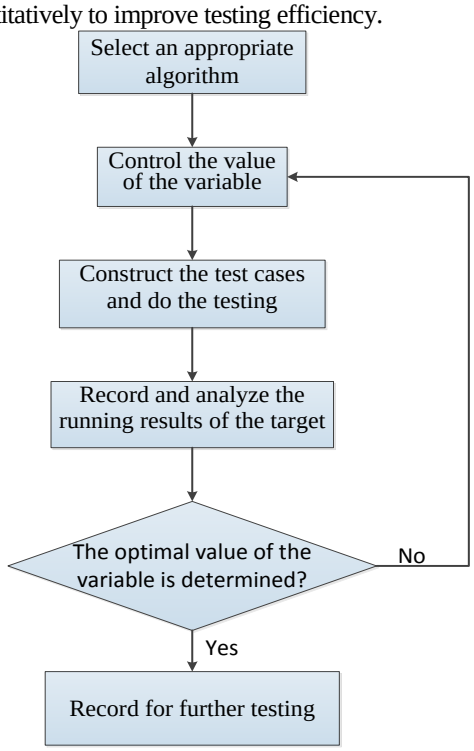

Figure 4. Process of determining the best variable values

For example, a limited distance formation method for the testing cases is proposed to detect the vulnerabilities of the Diameter protocol in the IMS. The method is based on equivalence partitioning. It limits the differences between the AVPs to make the testing cases more likely to be processed by the target. This method can also be applied to other network protocols. The crux is to find out the characteristics of the effective malformed protocols and then utilize the suitable algorithm.

The efficiency of vulnerability discovery can be improved significantly by limiting the conditions of the formation of the testing cases. This method applies to the situation that the testing period is required to be stringent. However, it needs to have an insight into the protocols to utilize this method. And there may be a high rate of missing report because of the limitations of the variety of the testing cases. Therefore, the target of this method is relatively simple

IV. NETWORK VULNERABILITY DISCOVERY

Network vulnerability mainly refers to the security problems in the architecture, communication protocols and processes stipulated in the network standards. Currently, the researches on network vulnerability mainly focus on the terminal part of the computer networks and the communication networks. The studied protocols are mainly low-layer protocols, such as TCP and UDP. For example, the security issues of TCP were discussed in References [29] [30] [31].

However, researches on the core network are relatively fewer. Vulnerabilities in the core network will make the whole network face security threats. So the study on the vulnerabilities of the core network is going deeper. Likewise, we can use static analysis and dynamic analysis to detect network vulnerabilities.

A. Static Analysis for Network Vulnerability

Static analysis for the network is different from the one for the software. Its main objects are not the source code, but the macroeconomic data, such as the architecture of the network, the interface specification and so on. The Evolved Packet System (EPS), for instance, will surely replace the traditional 2G/3G network for its good performance. It can provide a high level of security and confidentiality to the users and the operators. However, after simply analyzing its communication architecture, we could discover some problems. In the early stage of the network construction, considering the investment protection of the CS, the operators may use the old voice solutions in the CS domain to provide voice services. The process is called CS Fallback [32]. In this case, communication services will get back to be provided by the 2G/3G network, and the EPS network has to face vulnerabilities in the traditional network. The architecture of CS Fallback is shown in Fig. 5.

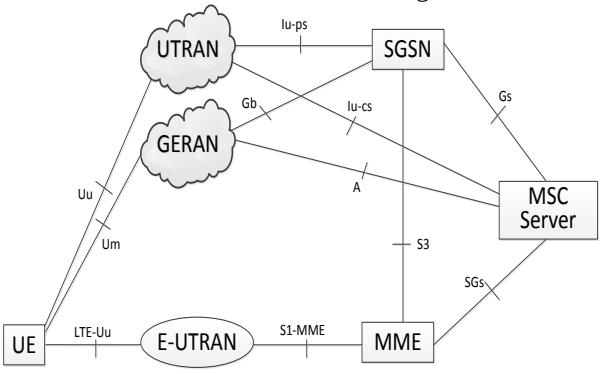

Figure 5. The architecture of CS Fallback

B. Dynamic Analysis for Network Vulnerability

The communication of the network is based on the protocols. So protocol vulnerability is also the key factor affecting the performance of network security. But network vulnerability discovery is different from simple protocol vulnerability discovery. Protocol vulnerability discovery technologies described in the last section are mainly for stateless protocols. But the protocols involved in network communication flow are always state. It means that every protocol message is related to several other ones. So, at this time, we should detect network vulnerabilities according to the state mechanisms of the protocols [27], combined with the communication flow.

A simple diagram of network communication process is shown in Fig. 6. A and B are both the entities of the network. A sends Request 1 to B, and then it sends Request 2 to B if it receives response from B. B will give the response for Request 2 only after the above flow is over. The previously described protocol vulnerability discovery can only be used to test Request 1. If we want to detect the vulnerabilities in the processing procedure of Request 2, we need to make A send Request 1 correctly first, and then send the malformed testing Request 2 after receiving Response 1 from B. If we make A send the malformed testing Request 2 directly, B may discard the testing message immediately for the lack of previous necessary procedure. And then there will be a missing report.

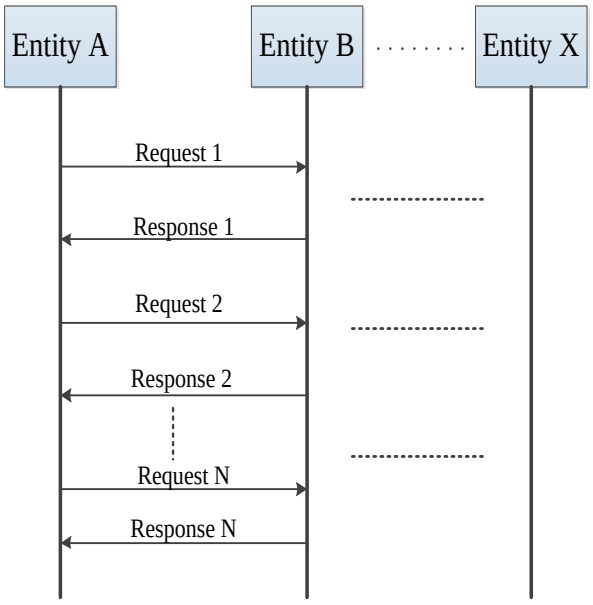

Figure 6. Network communication process

Taking into account the above situation, we need to determine the process mode of the network processing the protocol at the beginning of the testing process. The flowchart shown in Fig. 7 shows the process of detecting the vulnerabilities of the network based on Fuzzing.

After the target is selected (Here, we choose Request 2 in Fig. 6 as the example), it sends the normal Request 2 to the target directly and monitors the response of B. If the response is normal, it will continue the following steps in accordance with protocol vulnerability discovery. And if A does not receive normal response from B, it means that Request 2 is state. So it needs to send Request 1 first, and then it sends testing message 2 to do the detecting after receiving Response 1.

In this way, we can test every signaling message in the communication process. So the rate of missing report can be reduced significantly. But the testing process will be very tedious when the interaction of network signaling messages is complex and there are too many state protocols. It needs to simulate the associated processes for every testing case. So the testing time will be relatively long. Therefore, to reduce the blindness and to improve the relevance of the test during the discovery process of network vulnerabilities, we should first analyze the architecture and the communication flow artificially.

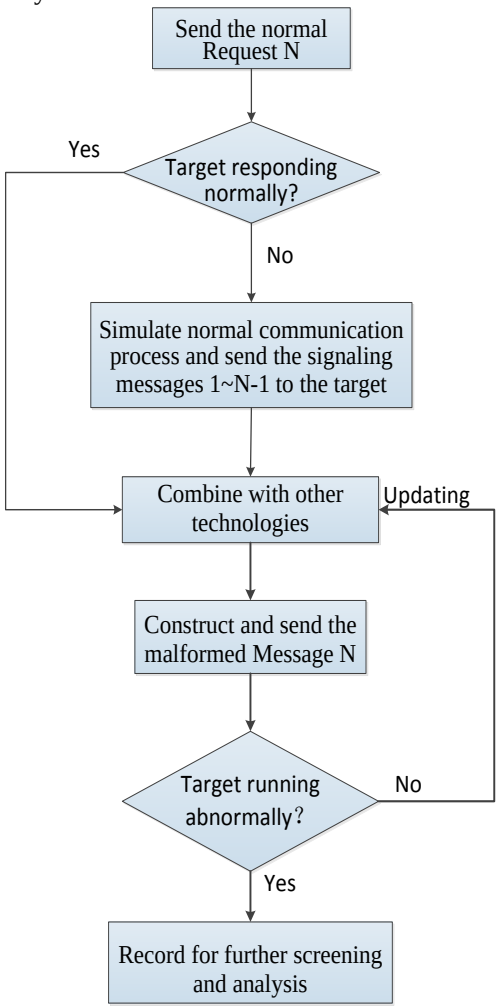

Figure 7. Fuzzing test flow for the network

V. CONCLUSION AND PROSPECT

This paper introduces the development of vulnerability researches and the importance of such researches for the field of information. It makes an overview of vulnerability researches in different areas. Then we present the classification of vulnerability discovery technologies as well as the extent of application, advantages and disadvantages of each class. We discuss the methods of protocol vulnerability

discovery and point out the strengths and weaknesses of each method in detecting protocol vulnerabilities. Finally, according to protocol vulnerability discovery technologies, we analyze the issues in the process of detecting network vulnerabilities and propose the idea of network vulnerability discovery combined with the communication flow.

Vulnerability discovery technology has played an important role in solving the security problems in the field of information technology. The research is being more and more systemic. Current vulnerability discovery technologies have been improved. But the direction of improvement is single. Researchers keep seeking a vulnerability discovery technology which is highly automated, high-efficient with a low rate of false alarm and missing report. But the difficulty can be imagined. Therefore, a good research direction is to seek an appropriate balance among the above aspects. In addition, we should know the advantages and disadvantages of the vulnerability discovery technologies and choose the right one according to the actual situation.

ACKNOWLEDGMENT

This work was supported in part by a grant from a project numbered 2011ZX03006-003.

REFERENCES

[1] Shizhong Wu, "Review and Outlook of Information Security Vulnerability Analysis," *Journal of Tsinghua University (Natural Science)*, vol. 49(2), pp.2065-2072, 2009.

[2] Chunying Wang, Daxin Liu and Danyu Zhang, "Analysis and Research of Computer System Common Vulnerabilities & Exposures (CVE)," *The Seventh International Conferece on Electronic Measrement and Instruments.* ICEMI 2005.

[3] G. Lorenz, T. Moore, G. Manes, J. Hale, S. Shenoi , "Securing SS7 Telecommunications Networks," *Workshop on Information Assurance and Security*, vol. 2, pp. 273-278, June 2001.

[4] Guoxiang Yao, Quanlong Guan, and Kaibin Ni, "Test Model for Security Vulnerability in Web Controls Based on Fuzzing," *Journal of Software*, vol. 7, No. 4, pp. 773-778, April 2012.

[5] Dong Wang, and Chen Liu, "Model-based Vulnerability Analysis of IMS Network." *Journal of Networks*, vol. 4, No.4, pp. 254-262, June 2009.

[6] Ronald W. Ritchey, and Paul Ammann, "Using Model Checking to Analyze Network Vulnerabilities," *Security and Privacy, IEEE Symposium on*, 2000.

[7] Frank Piessens, "A Taxonomy of causes of Software Vulnerabilities in Internet Software," *Supplementary Proceedings of the 13th International Symposium on Software Reliability Engineering. IEEE Computer Society Press, Los Alamitos*, CA, 2002.

[8] Jay-Evan J. Tevis, and John A. Hamilton, "Methods for the Prevention, Detection and Removal of Software Security Vulnerabilities," *Proceedings of the 42nd annual Southeast regional conference*. ACM, 2004.

[9] S. Byers, L. Cranor, D. Kormann, P. McDauiel, E. Cronin, "An Analysis of Security Vulnerabilities in the Movie Production and Distribution Process," *Telecommunications Policy*, vol. 28, pp. 619-644, 2004.

[10] M. Bishop, "A Taxonomy of UNIX System and Network Vulnerabilities," *Technical Report CSE-95-10, Department of Computer Science*, University of California at Davis, 1995.

[11] Ziyad S. Al-Salloum, and Stephen D. Wolthusen, "A Link-Layer-Based Self-Replicating Vulnerability Discovery Agent," *Computers and Communications (ISCC), IEEE Symposium on*, 2010.

[12] Xueyong Zhu, Zhiyong Wu, and J. William Atwood, "A New Fuzzing Method Using Multi Data Samples Combination," *Journal of Computers*, vol. 6, No. 5, pp. 881-888, May 2005.

[13] Sung-Whan Woo, Omar H. Alhazmi, and Yashwant K. Malaiya, "An Analysis of the Vulnerability Discovery Process in Web Browsers," *Proceedings of the 10th IASTED International Conference on Software Engineering and Applications, Dallas, TX, USA*, 2006.

[14] A. Ozment, "Improving Vulnerability Discovery Models," *Proceedings of the 2007 ACM workshop on Quality of protection*, ACM, 2007.

[15] O. H. Alhazmi, and Y. K. Malaiya, "Application of Vulnerability Discovery Models to Major Operating Systems," *IEEE Transactions on Reliability*, vol. 57, No.1, pp. 14-22, March 2008.

[16] HyunChul Joh, Jinyoo Kim, and Yashwant K. Malaiya, "Vulnerability Discovery Modeling Using Weibull Distribution," *Software Reliability Engineering, IEEE International Symposium on*, pp. 299-300, 2008.

[17] Jun Gao, Zhida Xu, and Jian Li, "Survey of Automatic Discovery of Software Vulnerability," *Computer and Digital Engineering*, vol. 37, No. 1, pp. 100-104, 2009.

[18] Ruoyu Zhang, *Static Program Analysis Dynamic Software Vulnerability Discovery*, MS thesis, Shanghai Jiaotong University, 2010.

[19] E. M. Clarke, and E. A. Emerson, "Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic," *25 Years of Model Checking*, Springer Berlin Heidelberg, pp. 196-215, 2008.

[20] Qiang Chi, Hong Luo, and Xiangdong Qiao, "Vulnerability Discovery Analysis Techniques," *Computer and Information Technology*, pp. 90-92, 2009.

[21] Lin Shao, Xiaosong Zhang, and Enbiao Sun, "New Method of Software Vulnerability Detection Based on Fuzzing," *The Computer Applied Research*, vol. 26, No. 3 pp. 1086-1088, MARCH 2009.

[22] L. H. Tahat, B. Vaysburg, B. Korel, A. J. Bader, "Requirement-Based Automated Black-Box Test Generation," *Computer Software and Applications Conference, IEEE Annual International*, pp. 489-495, 2001.

[23] S. Whalen, M. Bishop, and S. Engle, "Protocol Vulnerability Analysis," *Department of Computer Science, University of California, Davis, USA, Technical Report CSE-2005-04*, 2005.

[24] Tieyan Li, and Robert Deng, "Vulnerability Analysis of EMAP-An Efficient RFID Mutual Authentication Protocol," *Availability, Reliability and Security, The Second International Conference on*, IEEE, pp. 238-245, 2007.

[25] Shuxin Liu, Jianhua Peng, Caixia Liu, Xiaolong Xie., "SIP Vulverability Discovery Model in IMS Based on Fuzz Testing," *Application Research of Computers*, vol. 29, No. 9, pp. 3456-3459, 2012.

[26] B. P. Miller, L. Fredriksen, and B. So, "An Empirical Study of the Reliability of UNIX Utilities," *Communications of the ACM*, vol. 33, No. 12, pp. 32-44, 1990.

[27] Baofeng Zhang, Chongbin Zhang and Yuan Xu, "Network Protocol Vulnerability Discovery Based on Fuzzy Testing," *Journal of Tsinghua University (Natural Science)*, vol. 49, No. S2, pp. 2113-2118, 2009.

[28] Chi Liu, Kangfeng Zheng, and Hui Li, "Research of Vulnerability Discovering Based on Protocol Analysis," *The 2009 Graduate Academic Exchange of Communication and Information Technology Proceedings*, pp. 302-308, 2009.

[29] B. Guha, and B. Mukherjee, "Network Security Via Reverse Engineering of TCP Code: Vulnerability Analysis and Proposed Solutions," *Network, IEEE*, vol. 11, No.4 pp. 40-48, 1997.

[30] R. Ritchey, O' Berry Brian, and S. Noel, "Representing TCP/IP Connectivity for Topological Analysis of Network Security," *Computer Security Applications Conference, Proceedings, 18th Annual*, IEEE, pp. 25-31, 2002.

[31] Aldar C-F. Chan, "Efficient Defence Against Misbehaving TCP Receiver DoS Attacks," *Computer Networks*, vol. 55, No. 17, pp. 3904-3914, 2011.

[32] Yihua Jiang, 3*GPP System Architecture Evolution (SAE) Principle and Design*, People's Posts and Telecommunications Press, China, 2010.