OPC-MFuzzer: A Novel Multi-Layers Vulnerability Detection Tool for OPC Protocol Based on Fuzzing Technology

Xiong Qi, Peng Yong, Zhonghua Dai, Shengwei Yi, and Ting Wang

***Abstract*—With the rapid development of information and Industrial Technology, as the common data accessing interface for data provider, OPC technology is more and more widely deployed in the acquiring and sharing of production data. Yet, traditional OPC technology usually runs in the closed environment, always ignoring security defense, will cause serious consequence under malicious attack. For the complexity structure of OPC, with the feature of underlying layers like DCOM and RPC, which provide basic network service for upper layer, act as the critical causes for the faults of OPC protocol, unfortunately cannot be tested for vulnerability directly with traditional Fuzzer. In this paper, a vulnerability detecting tool for OPC protocol based on Fuzzing technology named OPC-MFuzzer is proposed and implemented; three different test case generating mechanisms for the testing of OPC, DCOM and RPC are developed separately. Finally three commercial OPC servers are selected for the experiment of vulnerability testing. The result shows that some vulnerability can be tested with the tool proposed, which prove the effective of such tool.**

***Index* Terms—Vulnerability detecting, security testing, fuzzing technology, OPC protocol.**

I. Introduction

OLE for Process Control [1] (OPC) is a popular communication protocol for sharing industrial production data among numerous data sources in process control system. With OPC, industrial devices from different manufactures can exchange data without difficulty. For the great convenience, OPC technology is widely used in industrial field including Electric power, chemical industry, water treatment, building intelligence and defense. However, with the deep integration of information technology and industrialization, more and more traditional isolated industrial control system is interconnected with Ethernet or directly connected to internet, which greatly improve the efficiency of industrial producing process, yet inevitably introduced some network security threats like virus and Trojan [2] at the same time. Although Classical Industrial network protocols like OPC becomes more and more popular in industrial control networks, security is not considered properly, which makes it vulnerable to cyber attack.

Compared with traditional data transmission oriented IT protocols, industrial network protocol is more powerful in the remote control [3] of physical devices, yet more feasible under malicious threat, if attacked, will behave abnormal and is most likely causing serious consequence like casualties, environmental pollution even endangering the society and countries.

With the importance of OPC, the existence of security vulnerability will lead to serious consequences. Therefore, it is extremely urgent to develop vulnerability detecting method for OPC specifically. Fuzzing is a technology that attempts to discover security vulnerabilities by sending random input to an application/device. As such, it is widely used to test for security bugs in input validation as well as in the application logic. Due to the specific features of OPC protocol including structure complexity and variable communicating port, traditional Fuzzing technology cannot be directly applied, and some necessary improvement is seriously needed. To solve this problem, this paper presented a novel vulnerability detecting tool for OPC protocol based on Fuzzing technology named OPC-MFuzzer, and a multi-layer test case generating mechanism is developed, which can generate different test case according to the structure of different protocol layers to improve the efficiency of vulnerability detecting.

The rest of this paper is organized as follows. In Section two, the related works on the vulnerability analysis for OPC protocol is discussed. Structure analysis for the OPC protocol is introduced in Section three. The multi-layer vulnerability detecting tool based on Fuzzing technology is presented in Section four. In Section five, the structure and the workflow of the tool proposed in part four is described in detail and some experimental results like the vulnerabilities detected with the tool are given to show the validity of the work mentioned. Finally, the paper is concluded in Section six where some future works are also discussed.

II. Related Works

The research area on the vulnerability analysis of OPC protocol attracted researcher's attention since the appearance of Stuxnet virus within Iran in the last few years. Lots of researchers from both academic and industrial circles have made some achievements in this field. Some of them with representatives can be classified into following two categories.

Manuscript received January 5, 2014; revised April 3, 2014. The research is supported in part by the National Science and Technology Major Project of China under Grant No. 2012ZX03002002, National Natural Science Foundation of China under Grant No. 90818021 and the Research Project of China Information Technology Security Evaluation Center under Grant No. CNITSEC-ZY-2013-016.

The authors are with the China Information Technology Security Evaluation Center, Beijing 100085, China (e-mail: xiongq@itsec.gov.cn, pengy@itsec.gov.cn, daizh@itsec.gov.cn, yisw@itsec.gov.cn, wangt@itsec.gov.cn).

In the field of vulnerability testing of OPC server, Maria B. Line performed penetration testing on OPC [4], which is a central component in process control systems on oil installations, shown how a malicious user with different privileges can fairly easily compromise the integrity, availability and confidentiality of the system. In additional, a commercial Fuzzing test suite Achilles [5] is presented by Wuldtech Corporation, which integrated the function of OPC monitor as a novel anomaly detecting mechanism along with the monitoring mechanism based on ICMP and TCP abnormal flags, and providing the Fuzzing test function for OPC specifically through the updating of plug-in.

On the security defending of OPC, Tofino Corporation presented Tofino OPC Enforcer LSM [6], which inspects, tracks and secures every connection that is created by an OPC application and dynamically opens only the TCP ports that are required for each connection, permitting only between the specific OPC client and server that created the connection. In addition, Huang Renjie discussed the OPC UA security issues from the two views [7] of network environment security and communication security in OPC UA applications. The network security deployment solution based on distributed firewall was proposed to ensure the host of OPC UA server and client against the different attacks. Sequentially the improved OPC UA security model based on the existing model was presented.

These achievements described partially solve the problem of OPC vulnerability analysis. Yet most of the works proposed mainly focus on analyzing the vulnerability from aspect of taking advantage of legal function to launch attack like ARP spoofing, Man-in-the-Middle and Packet Replaying, vulnerability detecting and exploiting is neglected, or only considered the security testing of OPC protocol itself without correlating it with underlying protocols like DCOM and RPC, which limited the further application of these achievements. The works on the security defending only cover the attack exploiting known vulnerabilities; the 0-day vulnerability related attack cannot be defended well. In a word, the defects mentioned above hindered the further application of these works and a vulnerability detecting mechanism which can satisfy the multi-layer testing requirements of OPC is seriously needed.

III. Analysis of OPC Protocol

A. Summary of OPC Deployment

Standard OPC architecture can be deployed like Fig. 1 in Client/Server mode [8]. The OPC server acquires data from PLCs periodically through Ethernet interface over industrial network protocols like MODBUS TCP, DNP3, etc and provides data access to OPC client with OPC/DCOM.

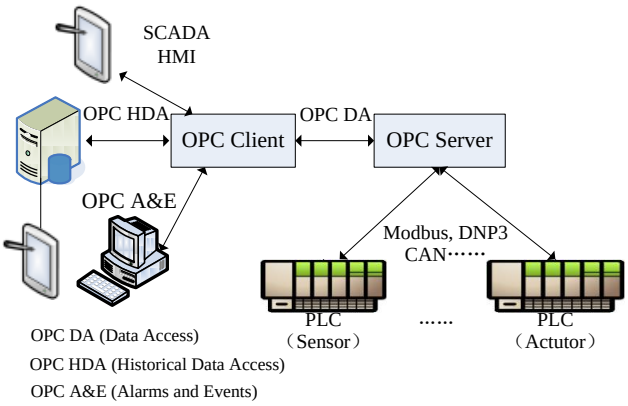

Fig. 1. Classical OPC architecture.

OPC can work in three modes, including OPC Data Access (OPC DA), OPC History Data Access (OPC HDA) and OPC Alarm and Event (OPC A&E). The OPC Data Access is mainly used to transfer real-time data acquired from PLC devices to control interface such as HMI. The OPC HDA mainly focused on history data access. The OPC A&E defines an interface for alarm monitoring and acknowledgment. Unlike DA, A&E does not provide a continuous stream of data between client and server, but instead supplies a value only when a specific event occurs.

B. Hieratical Structure of OPC Diagram

The structure of OPC Packet Diagram from top to bottom can be divided into six layers, like Fig. 2, including OPC, DCOM, RPC, Transport, Internet and Network, the upper layer works using the service provided by the lower layer. The former three layers can be considered to work together as the payload of TCP diagram, and OPC packet diagram is encapsulated in RPC packet for transfer.

OPC service works in "challenge-response" mode, waiting for TCP connection request on port 135 and UDP connection request on port 137. The process of OPC connection establishing can be divided into two steps. Firstly, OPC client query the TCP port number randomly generated by OPC server that can be used for data communication in the following steps, then the OPC client make connection with the port number obtained at the first step. Once the connection is established, the OPC client can get data transferring service from OPC Server.

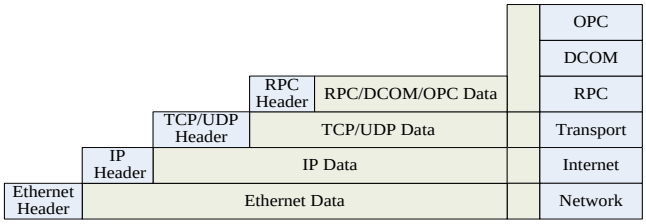

Fig. 2. Hieratical structure of OPC diagram.

C. Data Structure of OPC

The data access object of OPC can be divided into three layers, like Fig. 3, including Server, Group and Item, which

are organized like tree from top to bottom. OPC server maintains the information of server and acts as the container of the OPC group object. OPC group maintains the information related to itself and accommodates item objects, managing items. OPC items lay on the leaf of the tree, is a logical component, representing the single data including the I/O point of PLC or the instant value of the meter, like temperature, pressure, etc.

Standard OPC server provides seven server object calling interfaces, while standard OPC group provides eight group object calling interfaces. With these calling interfaces, OPC data accessing process can be divided into four steps. Firstly, OPC client creates an instance of server object, then use IOPCServer Interface to add a group into server, and IOPCItemMgt interface is used to add an item to the group, finally, use IOPCSyncIO interface to read an item.

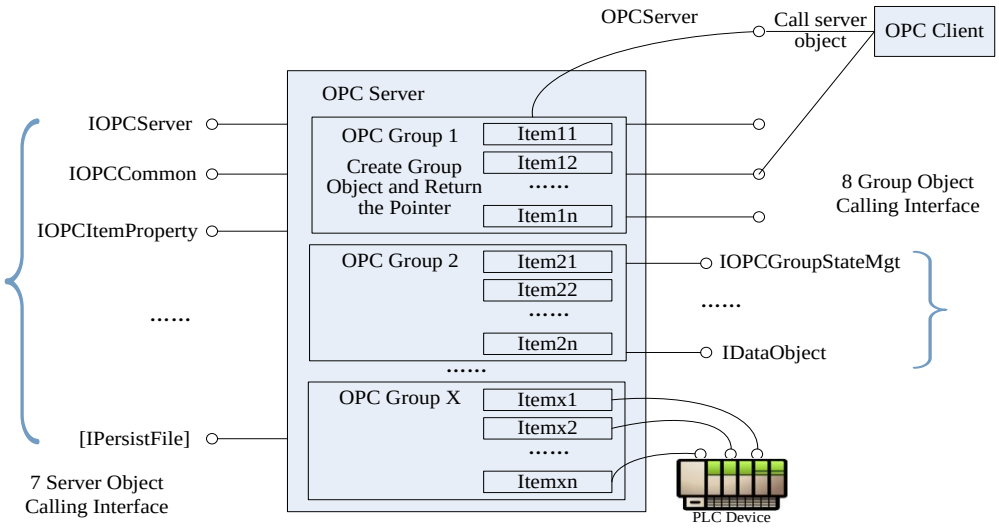

Fig. 3. Data structure of OPC and related object calling interface.

IV. Vulnerability Detecting for OPC Protocol with Fuzzing Technology

A. The Formation Nature of the Vulnerability in OPC

The vulnerabilities in OPC can be classified into two categories according to the formation nature. One class is caused by calling the API in the unsecure manner. This kind of vulnerability is caused by the defects on the design, assigning high privilege to some of the interface function without necessary security defense. If exploited, the vulnerability of this kind will bring high privilege like remote code executing to the attacker. The other is for the defects of input validating mechanism, which cause the error input of invalid parameters, change the workflow of the system process and result in the malicious code injection.

B. Summary of Fuzzing-Test

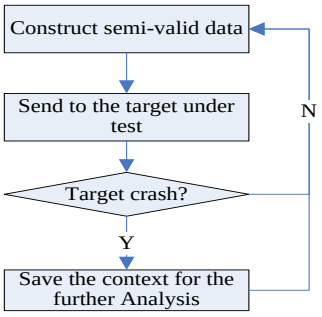

Fig. 4. General testing process of fuzzing.

Fuzz-Testing [9] is a kind of security testing technology in which the bad inputs are carefully constructed in attempt to make the target crash. As such, it is widely used to test for security vulnerabilities in the input validation mechanism or the application logic of network protocol, ActiveX or software. Compared with other vulnerability detecting technology, Fuzzing technology directly tests executable programs instead of source code, has become one of the most effective technologies in the field of security vulnerability detecting for its high degree of automation and wide adaptability. The general testing process can be described like Fig. 4.

There are two key points on employing Fuzzing-test to detect vulnerabilities in OPC protocol. First, the multi-layer feature of OPC should be considered, for OPC is based on DCOM and RPC from the aspect of hierarchy, and the vulnerabilities of OPC may be truly caused by the protocols below the OPC. To satisfy this requirement, the expected test case should be provided accordingly, which need to consider the interface of OPC and the lower protocols, and three different test case generating mechanisms should be developed separately. Secondly, the generating of semi-validate test case is critical, which seems valid from cursory look according to the principle of input, yet partially invalid in the detail. The test case should appear valid for the program with vulnerability may employ input validating mechanism with defects. The test data with obvious error cannot pass the validating process and will not be able to execute the kernel code. However, some key parts of the test case should be invalid in order to trigger the deep level vulnerabilities. The ability of balancing the generating of valid and invalid test case is a critical metric for assessing the performance of Fuzzing-test tool.

C. Design and Implementation of OPC-MFuzzer

1) High level design of OPC-MFuzzer

The design of OPC-MFuzzer is based on Fuzzing technology, can be used to detect vulnerabilities in the OPC server. The structure of OPC-Mfuzzer is made up of four components including OPC structure analyzing module,

Test case generating module, Testing module and anomaly detecting module, which is described like Fig. 5.

OPC structure analyzing module is responsible for acquiring the basic information from OPC, DCOM and RPC, including the list of interfaces and the functions contained in the interface. Test case generating module is used to generate semi-valid test case for three protocol layers separately according to the principles. Testing module is responsible for sending test case and supervising for the anomaly information of process and network connection captured by anomaly monitoring module. If anomaly is detected during the testing process, the related context information will be recorded and the following test case will be executed according to the sequence.

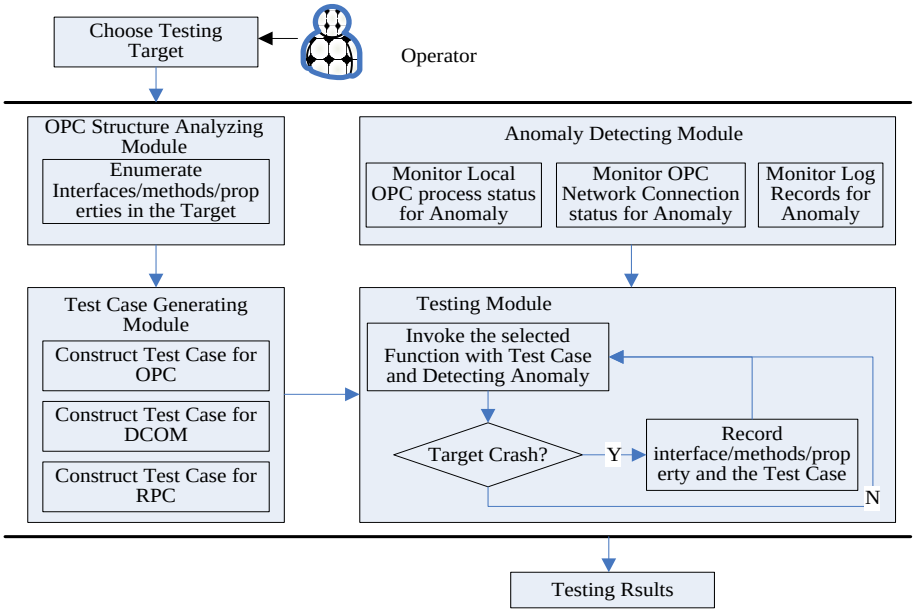

Fig. 5. Structure of OPC-Mfuzzer.

2) OPC structure analyzing module

OPC use IDL (Interface Description Language) to describe the interface provided. The type library compiled from interfaces definitions will be published together with OPC server, which contains the complete description of the interfaces. At the beginning, OPC structure analyzing module usually call the EnumClassesOfCategories and GetClassDetails function of IOPCServerList interface to enumerate the OPC services on the target.

In addition, OPC structure analyzing module also executes actions including diagram border identification, dangerous values analysis and the checksums. Border identification is to use the model placed inside to identify a given packet, is responsible for acquiring the basic information from OPC, DCOM and RPC, including interface list, function list contained in the interface, the type of the parameters for the interface function. This information is stored in the form of XML, which will be called by the other modules.

3) Test case generating module

Test Case Generating is the step of great importance in the Fuzzing test process, which determines the success rate of vulnerability detecting. OPC-MFuzzer mainly focuses on detecting vulnerabilities in OPC, DCOM and RPC. So the test case generating strategy should be developed according to the formation nature of these vulnerabilities.

Construct test case for OPC. To generate test case for OPC, the OPC .net library provided by OPC foundation is employed with c# API to generate legal request, currently supporting OPC DA, OPC HDA and OPC A&E, which will be mutated with the mutating module from Peach [13] to produce test case and allocate a Mutator for each interface type. Although the OPC .net library will filter some illegal test data according to the inherent rule, which limited the quality of test case, this method is still popular in OPC testing for its ease of use and the snapshot of the testing process can be described like Fig. 6.

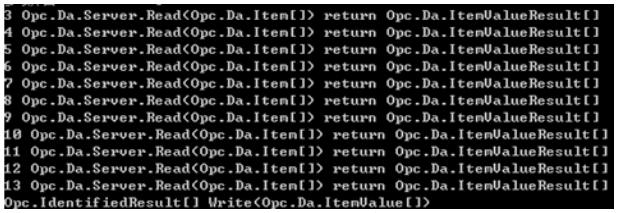

Fig. 6. Testing process for OPC.

Construct test case for DCOM. For testing DCOM, we use C++ API to access the official interfaces provided by Microsoft DCOM library. After the legal DCOM requests is produced, then the mutate mechanism is employed for the test case generating, and each parameter is allocated a Mutator separately. Although some semi-valid test case will be considered illegal and filtered by the lower DCOM library, this method is still widely used in the testing of DCOM and part of the test cases generated for DCOM can be described like Fig. 7.

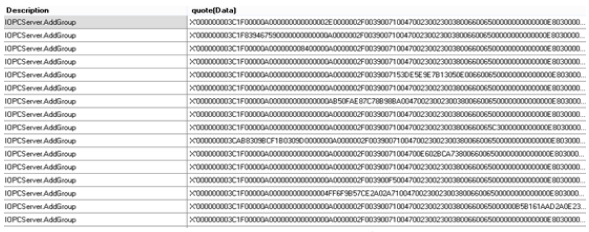

Fig. 7. Test cases generated for DCOM.

Construct test case for RPC. This method is different with the former two, which rebuilds the total function of DCOM with python, and construct RPC request without employing any third part library. This method holds the best performance, yet relatively complex in engineering and the quality of the test case will not be limited for libraries. The snapshot of the test process can be described like Fig. 8.

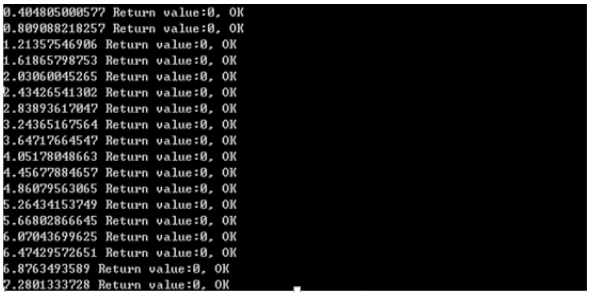

Fig. 8. Testing process for RPC.

4) Testing module

Testing Module is the center of OPC-MFuzzer, which forward the test case generated from test case generating module to target and schedule the exception monitoring and log recording task. Testing module is composed of four sub functions, including sends and receives data, processing request and identifying local network interfaces.

5) Anomaly detecting module

Anomaly Detecting Module is responsible for monitoring the exception caused by specific test case. There are three different ways for exception catching, including heartbeat, process state monitoring and log recording. Heartbeat takes advantage of the send and reply mode to judge whether the target process is living. Process state monitoring captures the exception with windows system driver and memory usage monitor. Log recording mechanism gets meta-data from network traffic and process monitor and record the exception to the log, which can be used for further research on the reason of the vulnerability.

V. Experiment Testing for the Tool

A. Achievements on the vulnerability detecting with OPC-MFuzzer

The ability of detecting vulnerabilities is the most critical metric for measuring the validity of Fuzzing-test tool. After applying the tool for vulnerability detecting on OPC server, we successfully verified the existence of some published vulnerabilities and detected some 0-day ones, which prove the correctness of the fuzzing strategy employed and embody the application value of the tool developed. Some of the most representative vulnerabilities verified with OPC-MFuzzer can be listed as follows.

1) OPC Server Takebishi Electric DeviceXPlorer Remote Code Executing Vulnerability (CVE-2007-1319). This vulnerability is caused by the defects of server handles validating mechanism, which allows an attacker with access to the Takebishi Electric DeviceXPlorer OPC Server be able to arbitrarily access server process memory, potentially allowing that attacker to execute arbitrary code or cause a denial-of-service.
2) OPC Systems.NET RPC Packet Remote Denial of Service Vulnerability (CVE-2011-4871). This vulnerability allows an attacker to exploit this issue to crash the affected application via a malformed .NET RPC packet on TCP port 58723, denying service to legitimate users.
3) Matrikon-OPC DNP3 OPC Server Denial of Service Vulnerability (CVE-2013-2791). This vulnerability is caused due to an unspecified error within the DNP3 Master station when handling certain communication packets, which can be exploited to crash the application by sending a specially crafted DNP3 packet.

In addition, some 0-day vulnerabilities are detected in the OPC server of some Mainstream brand Control Devices like Rockwell, Schneider etc, which have been submitted to CNNVD vulnerability database and related venders have been notified for patch developing. However, for the security consideration, the detail information can not be given here.

B. Comparison with Other OPC Security Tool

The OpcSecurityAnalyzer [9] application is a OPC tester proposed by Advosol corporation, which is designed to help with security permission settings related to OPC server access in order to find why the access to an OPC DA server is denied and check if the the server access is denied for all but the valid users. Yet the function of OpcSecurityAnalyzer is limited to check the correctness of access control mechanism and make simple log audit or configuration check. The vulnerability detecting mechanism is not included in this tool. In addition, the Achilles [8] fuzzing test device provides similar performance with OPC-Mfuzzer on testing OPC classic, yet the other OPC like A&E etc and the lower protocols like DCOM etc cannot be supported.

We use OPC-Mfuzzer to test Matrikon.OPC.Modbus [9], the OPC server and the OPC-Mfuzzer are deplyed on DELL R710 Server with Xen5450 3.0GHzx4, 16GB DDR3 memory, 320GB SATA2.0 HDD, and 100Mb LAN is employed for network connection. The topology of experiment can be described like Fig. 9.

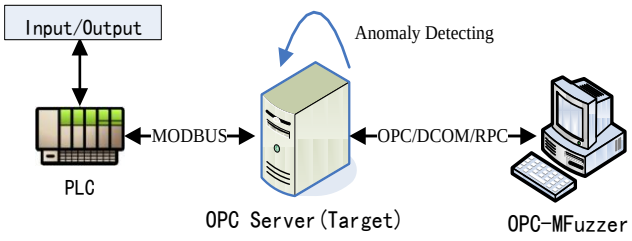

Fig. 9. Topology of experiment.

To record the anomaly detected, the log item is described as a ten Tuples, like (CaseId, time, SubType, uid, Response Time, Description, Exception_info, Hresult, Result, Data) to record the id of test case, test time etc. During the experiment process, 57694 test cases and 57694 log items are produced. 40 interfaces and 178 interface functions are tested by the OPC fuzzer, partially described as follows and there different OPC including DA, HAD, AE and the lower protocols like RPC are supported totally. The result of the experiment proves that the OPC-Mfuzzer tool mentioned can satisfy the requirement of the OPC server testing.

VI. Conclusion and Future Works

With the rapid development of industrial automation, the industrial network protocols employed in the SCADA for production data and control instruction transmission are facing more and more security threats. As one of the most critical protocols for data sharing, OPC is based on DCOM, which uses the service provided by RPC. In order to map vulnerabilities related to OPC, we have to consider vulnerabilities related to DCOM and RPC for the structure complexity of OPC. However, traditional Fuzzing technology cannot be directly applied, some improvement is seriously needed to satisfy the requirement of vulnerability detecting for OPC, DCOM and RPC together.

In this paper, a vulnerability detecting tool for OPC protocol based on Fuzzing technology named OPC-MFuzzer is proposed and implemented; three different test case generating mechanisms for the testing of OPC, DCOM and RPC are developed separately. Finally three commercial OPC servers are selected for the experiment of vulnerability testing. The result shows that some vulnerability can be tested with the tool proposed, which prove the effective of such tool.

In the future, we plan to combine the Fuzzing test and OPC program analysis together, take advantage of the information output from OPC binary program analyzing process to improve the efficiency of black-box testing, acting as so called gray-box testing.

References

[1] C. Cadar, V. Ganesh, P. Pawlowski, D. Dill, and D. Engler, "EXE: Automatically generating inputs of death," *ACM Transactions on* Information *and System Security*, vol. 12, no. 2, pp. 1-38, Dec. 2008.

[2] R. Langner, "Stuxnet: Dissecting a cyber-warfare weapon," *IEEE Security and Privacy*, vol. 9, no. 3, pp. 49-51, May/June 2011.

[3] V. M. Igure and R. D. Williams, "Security and SCADA protocols," presented at International Conference of Nuclear Infrastructure Security: American Nuclear Society. Albuquerque, NM, United States, May 2006.

[4] F. Sandnes *et al.*, "Penetration testing of OPC as part of process control systems," presented at International Conference of Ubiquitous Intelligence and Computing, Springer Berlin Heidelberg, Sweden, April 2008.

[5] R. Huang, F. Liu, and D. Pan, "Research on OPC UA security," presented at The 5th International Conference on Industrial Electronics and Applications (ICIEA), IEEE, United States, March 2010.

[6] V. V. Tan, D. S. Yoo, and M. J. Yi, "Security in automation and control systems based on OPC techniques," presented at Strategic International Forum on Technology. IFOST, Korea, February 2007.

[7] G. Devarajan, "Unraveling SCADA protocols: Using sulley fuzzer," presented at the Defcon 15 Hacking Conference, United States, July 2007.

[8] A. Gervais, "Security analysis of industrial control systems," Ms. Thesis, KTH Stockholm and Aalto University, Sweden, 2012.

[9] X. Lu *et al.*, "An OPC technology based scada system design for wind power plants," *Dianli Xitong Zidonghua/Automation of Electric* Power *Systems*, vol. 32, no. 23, pp. 90-94, 2008.

Qi Xiong was born in Hubei Province, China, 1983. He received bachelor, master and Ph.D. degree from Computer School of Wuhan University at 2004, 2006 and 2010 separately. He is now an associate researcher at Technology Security Evaluation Lab of China Information Technology Evaluation Center (CNITSEC) since 2010. His research interests include the security of critical infrastructure, vulnerability detecting and fuzzing test.

Yong Peng was born in Shanghai, China, 1974. He received Ph.D degree from Beijing University of Posts and Telecommunication in 2013. He is now an associate researcher and acting as the director of Technology Security Evaluation Lab at China Information Technology Evaluation Center (CNITSEC). His research interests include information security, vulnerability detecting and fuzzing test.

Zhonghua Dai was born in Jiangsu Province, China, 1978. He received Ph.D degree from Wuhan University in 2013. He is now an assistant researcher and acting as the chief engineer of Technology Security Evaluation Lab at China Information Technology Evaluation Center (CNITSEC). His research interests include information security, security risk assessment, critical infrastructure protecting and security analysis of embedded system.

Shengwei Yi was born in Henan Province, China, 1977. He received Ph.D degree from Computer School of Beihang University at 2012. He is now an assistant researcher at the Technology Security Evaluation Library of China Information Technology Evaluation Center (CNITSEC) since 2012. His research interests include security of industry control system, data mining.

Ting Wang was born in Hubei Province, China, 1986. She received MS degree from Computer School at Hua-zhong University of Science and Technology (HUST) in 2012. She is now an assistant engineer in China Information Technology Evaluation Center (CNITSEC) since 2013. Her research interests include information security, vulnerability detecting and fuzzing test.