Test Model for Security Vulnerability in Web Controls based on Fuzzing

Guoxiang Yao
Information Technology Science College, Jinan University, Guangzhou, China
yao@jnu.edu.cn

Quanlong Guan and Kaibin Ni
Network and Education Technology Center, Jinan University, Guangzhou, China
sanson_guan@hotmail.com

***Abstract*—The number of Web controls' security vulnerability surged with ever-changing varieties of attacks. Therefore this paper analyzes test model for Web controls' vulnerability, and put forward a improved test model for Web controls' vulnerability. Be aimed to test vulnerability of Web ActiveX controls combining static analysis and dynamic analysis, as well as put forward a proposal of optimizing the generation engine for test data using "heuristic rule". Experiment results show that test model for Web controls' vulnerability based on fuzzing is effective and feasible, and it is able to manipulate interaction problems.**

***Index Terms*—Fuzzing test, Web controls, vulnerability test, vulnerability analysis**

I. INTRODUCTION

The number of Web controls' vulnerability has been growing rapidly these years. According to figures of Web controls vulnerability from celebrated Software Security Company called Symantec, it is claimed in the report during 2007 and 2009 that the number of Web vulnerability increases in geometry order of magnitude. There are two aspects of reasons for the large number of Web controls' vulnerability: one is the high market occupancy of the Web controls. Web ActiveX controls are independent of development platform. And the ActiveX controls based on one kind of programming language is needless to modify when they are used in another kind of programming language. As a result many software companies are intent on developing ActiveX controls, what's more, ActiveX controls are popular at many developers. Another is the high utility value of the ActiveX controls' vulnerability. ActiveX controls' vulnerability are the remote vulnerability, the same as the IE vulnerability of the Windows system. The attackers may make use of this king of vulnerability to execute the code willfully.

Web applications enable much of today's online business including banking, shopping, university admissions, and various governmental activities. Consequently, vulnerabilities that allow an attacker to compromise a web application's control of its data pose a significant threat[1]. Vulnerabilities that may lead to the compromise of sensitive information are being reported continuously, and the costs of the resulting damages are increasing. The main reasons for this phenomenon are time and financial constraints. Limited programming skills, or lack of security awareness on part of the developers[2]. Validating dynamic Web application is hard. Even professionally-developed applications often contain multiple faults. To prevent faults, programmers must make sure that the application creates a valid HTML page on every possible exection path[3]. Web browsers are of high interest when it comes to client-side vulnerabilities. For example, the image rendering library used by a web browser may crash when processing a crafted JPEG image[9].

With the release of Internet Explorer 3.0 in 1996, Microsoft introduced support for ActiveX, which originated as the Component Object Model, or COM[5]. COM allaws developers to make reusable objects that can be used by other application. COM objects can be written in a variety of programming languages. The minimum requirement is that the object implements the IUnknown interface[6]. A COM object that has been designed for ues n the Internet Explorer web browser is commonly referred to as an ActiveX control[7]. With its support for ActiveX controls, Internet Explorer allowed for the creation of web pages that had never-before seen levels of functionality. ActiveX controls are not limited by a sandbox like Java applets[8], and any Windows developer could easily make their code available for use in the Internet Explorer web browser. Internet Explorer's support for both ActiveX and scripting languages gives the browser a large attack surface[4,10,11,12] and a high level of control, which makes it a primary target for attacks.

A high number of security vulnerabilities that are today published on various security mailing lists are detected by using a fuzzing method. The fuzzing method is based on the fault injection technique that, by sending various input data to target application, tries to detect a security vulnerability.

doi:10.4304/jsw.7.4.773-778

II. TRADITIONAL WEB CONTROLS' VULNERABILITY TEST MODEL

A. Fuzzing vulnerability test model

The choice of the fuzzing vulnerability test methods depend on different factors, such as target programs, the format for the test data, and the investigators' skill etc. But the procedures are relatively consistent. As a result, we can abstract the model. Figure 1 is the diagram of the fuzzing vulnerability test model.

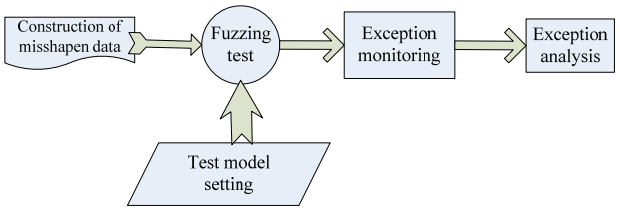


Figure 1. fuzzing vulnerability test schematic diagram

The first step is to construct the misshapen data that triggers vulnerability on every procedure of fuzzing test, whatever the object of test is. This step is the decisive factor for if the fuzzing test can detect the vulnerability. There are two ways of generating the test data. They are generating test data with the preestablished data as seed and with mutating the existing test data. It depends on the test target and the data format to decide the way of generating the data. Whatever, the procedure of generating the test data must deal with the automation problem.

Fuzzing test is a significant step for the fuzzing vulnerability test model. Generally it can set up test mode. It can appoint that the test model is violence test, equivalence class test, boundary value test, or the combination of field test etc. Generally fuzzing test can adjust the test model and data construction mode to make the test procedure more efficient.

Exception monitoring is an important step in fuzzing test. Because we can detect the data set or document which triggers the system abnormity on the procedure of sending test data set or document to the fuzzing test system. What's more, the exception monitoring takes notes of some important information under abnormal conditions, such as the CPU status, register status, and the stack status etc, which can be consulted for the follow-up abnormality analysis.

The assignment of anomaly analysis is to make sure if this anomaly is a flaw or if it is utilized. Because not only the software vulnerability can trigger the system anomaly, but also the software bugs can cause the system breakdown. Anomaly analysis is almost the weakness for every fuzzing system. Because it involves too much manual analysis, it is hard to make use of automatic mechanism.

B. Traditional Web controls' vulnerability test model

The traditional model is composed of three stages including preparation before test, test phases, and report after test. The preparation before test includes analysis of Web controls' attribute, the function with the parameter and the generation of the test data. The test phase mainly consists of simulating the customers to open the test instance, dynamic Exception monitoring, and the anomaly report.

1). Preparatory phase before test

Firstly, we make sure the objective Web Active controls. The main work of it is to find the CLSID of the Web controls. Secondly, we confirm if the IObjectSafety secure interface is actualized, if the script is safe, and if it is setup with the killbit bit. We can acquire these information from the corresponding the information in the registry entries. As long as the assurance of the IObjectSafety secure interface, script safety, and the circumstance of the KillBit bit, we can carry on the follow-up steps. Because it is lack of any of the three factors, the Web controls can't be invoked, which make the following analysis meaningless.

Lastly, we will obtain the attribute list, function list, and the function parameter list of the ActiveX controls. Only when we are clear about these lists can we pertinently construct the fuzzing test data, which is one of the specialties for the Web ActiveX controls' vulnerability test based on fuzzing.

2). Test phases

We generate some test instances according to the attributes, functions and he function parameters on the preparation phase. The work on the test phase is sending every test instances into the target program. The document formats of the test instances are htm, html, wsf, and PDF on the procedure of the Web controls' vulnerability test. We need to open all the test documents with the program simulating the manual manipulation before the test.

After opening the test document, IE may point out if it is going to load the corresponding Web controls, which also can be operated by the program simulating the users. The procedures of the Web controls load, Web controls initialization and the test instances invoking the controls can all be possible of appearing exception. We make use of the interface and the structuring exception handling SHE from windows to have the fuzzing tools attached to the IE browser as the debugger. On this way, we can receive, handle as well as record kinds of debug and abnormal events. When the exception emerges, the windows systems will always pop-up error prompt windows.

A perfect fuzzing test tool must stimulate users to close the shut window. This function can be realized by HOOK technology. We will analyze and deal with the problem in detail in the subsequent research.

3). Test report phase

The test report includes exception of the fuzzing test and the test document of the corresponding abnormal events.

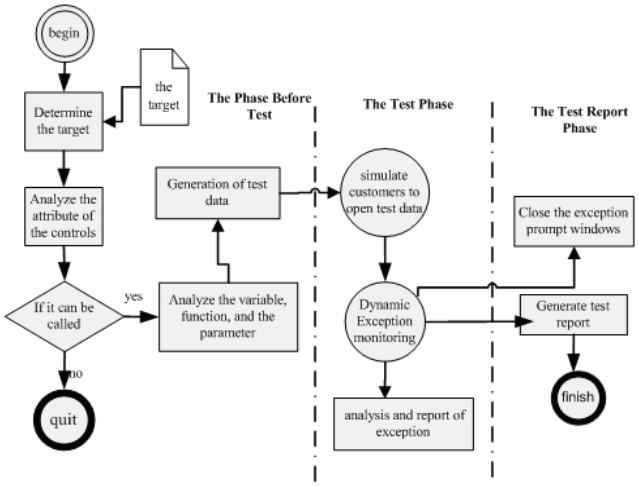


Figure 2. traditional ActiveX controls' vulnerability test model

C. The deficiency of the traditional web controls' vulnerability

There are deficiencies of the traditional model. Firstly, it is significant to make a comprehensive test for the callable methods of Web ActiveX controls. If the list of the callable methods for ActiveX controls is quite long, the working capacity for the fuzzing vulnerability test procedure will be greatly increased. It is showed in two aspects including the number of the generating test instance and the time of the test procedure. This paper advances the program scanning analysis technology to solve this problem. Secondly, it is the problem of the randomness of the test data generation. We adopt the method of the heuristic test data generation. Lastly, the test is quite single. The dynamic analysis can examine the code on the implementation. It is good at discovering the errors in the running time. However the static analysis examines the errors of the program with the algorithmic examination. The traditional model accents on static analysis more, which is difficult to discover the defect generated in interactive procedure. This article adopts the combination of the dynamic analysis and the static analysis.

III. IMPROVED MODEL FOR WEB CONTROLS' VULNERABILITY TEST

This paper puts forward a proposal of the improved model combining the dynamic analysis and the static analysis. The improved model is added with the code scanning analysis module, heuristic method of generating the test data model, and the OllyDbg analysis test model etc.

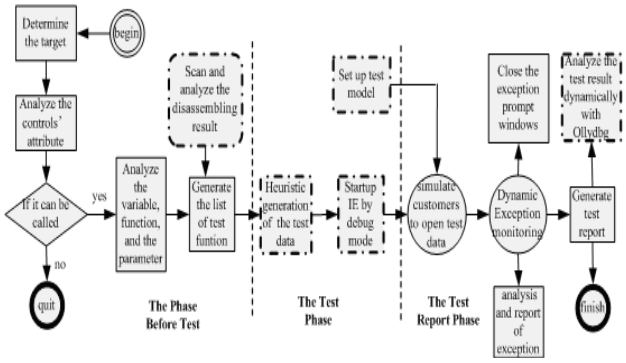


Figure 3. improved model for Web controls' vulnerability test

The code scanning analysis model: Firstly we carry on the disassembling to the binary file of target program with the disassembling tool such as IDA. Secondly we utilize the insecurity method such as the controls' function called strcpy to scan and analyze the disassembling results.

We make the function the under test function. It makes the list of the under test function decrease greatly. The model also searches for the "tagged word" in the function address space. It offers support to the structure of the test data.

Heuristic generating test data module: In the original model, after a fully preparation the process of test data generation have a certain purpose. In order to improve the efficiency of vulnerability testing, we use the heuristic method to generate test data. So that the process of testing is more targeted, which is favorable to detect potential vulnerabilities better.

OllyDbg analysis of test results module: OllyDbg is a popular open-source software used to debug binary codes dynamically. OllyDbg not only has strong anti-assembly function, but also has excellent dynamic debugging engine. We can easily load the test data that caused the exception by OllyDbg interface. Further more, it is more convenient for the further analysis of vulnerabilities.

A. Web Control enumeration module

Two small modules are contained. They are all loadable Web Control module of enumeration target host and the module which enumerate the properties, functions, function parameters of a specific ActiveX control. The results enumerated will be formed a list and stored in database tables. Therefore, it is easy to query in the following analysis, and it eliminates the need of repeated enumeration as well.

B. Disassembling code analysis module

The binary files are disassembled into assembly codes by IDA and other disassembling engine tools. And then plug-ins provided by IDA are used to search strcpy and other unsafe method calls. Besides, the analysis module of disassembling module takes charge of finding out the "tagged word" in address range of derivative functions of ActiveX controls. This "tagged word" may be a character, a string, or a number. The "tagged word" is helpful for test data generation engine to construct better test data,

thus the Fuzzing system can detect vulnerabilities more efficiently.

C. Test data generation module

According to the comprehensive functions list of the two modules, test data or test documents are generated. The data may be in the form of "URL", and the documents may be in form of "htm", "swf" and so on. Actually, its essence is to use scripting languages to call the methods of ActiveX controls. The method of "boundary value" stress testing is usually used to generate test documents, but it is not efficient enough if this method is used alone. This paper introduces "heuristic rule" and " tagged word" to cope with stress testing, so as to enhance the intelligence of the test data generation engine.

D. The application interaction module

According to SEH (structured exception handling) mechanism of Windows, when an exception is not caught finally, the system will notify the user that "The program has crashed" with an error prompt window. While Loading the ActiveX, initializing ActiveX objects, or calling ActiveX methods or properties, an exception may occur. Application interaction module uses the technology called HOOK to deal with these system error prompt windows caused by the exceptions.

E. Dynamic exception monitoring module

Dynamic exception monitoring module is responsible for simulating the test documents generated by the clicks of users, and sending test data generated (such as URL data) to the target program; Besides, this module is also in charge of monitoring exceptions occurred by the target program (usually refers to the IE browser) during the testing, and recording these exceptions.

IV. TEST RESULTS AND ANALYSIS

In order to prove the feasibility and effectiveness of fuzzing test model for Web control vulnerability proposed by this paper, we test the "STORM" player (mainly with its mps.dll control), which has a larger amount of users. Test environment: Windows system. Internet Explore: V7.0. Test object: STORM player V2.9.

A. Enumerating the properties, methods and method parameters of ActiveX control

Install "STORM V2.9", and then enumerate the properties, methods, and method parameters of ActiveX control (mainly with mps.dll control of STORM player) by the improved Fuzzing test model. In Figure 4 is the enumeration list with the properties, methods and method parameters of mps.dll control of the Storm player.

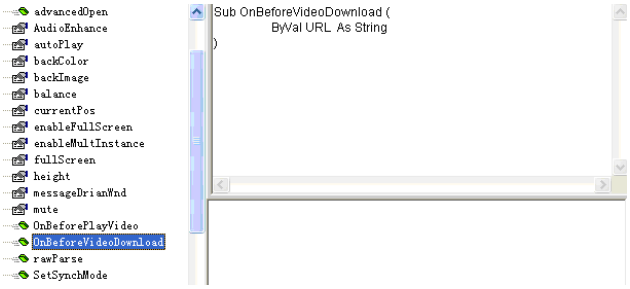


Figure 4. the list of ActiveX controls' properties, methods and method parameters

B. Tectonic test document

With the Fuzzing test model in this paper, right-clicking any property or method in the enumerated list can generate the corresponding test data. The model generates test documents in the form of VB Script, in order to be directly called easily by the wscript.exe. This following codes are in one of the Fuzzing test documents of the "OnBeforeVideoDownload ()" function derived from mps.dll control of Storm player.

```
<?XML version='1.0' standalone='yes' ?>
<package><job id='DoneInVBS' debug='false' error='true'>
<object classid='clsid:6BE52E1D-E586-474F-A6E2-
1A85A9B4D9FB' id='target' />
<script language='vbscript'>
'Wscript.echo typename(target)
targetFile = "C:\Program Files\StormPlayer \mps.dll"
prototype = "Sub OnBeforeVideoDownload ( ByVal URL As
String )"
memberName = "OnBeforeVideoDownload"
progid = "MPSLib.StormPlayer"
argCount = 1
arg1=String(8212, "A")
target.OnBeforeVideoDownload arg1
</script></job></package>
```

C. Vulnerability testing and result analysis

It costs a long time to do the vulnerability testing after a series of test documents was constructed, especially when the quantity of properties or methods of the ActiveX control is large. In this case, the benefits of intelligence will be presented. It can deal with the interaction problem well. Therefore, this test can be done unattended, realizing automated Fuzzing test. Fuzzing test model proposed in this paper is able to handle the interaction problem properly.

1) exception list

Do the Fuzzing test on a property or method of Web ActiveX controls. And the results are presented in the form of a list table. From the table we can easily get which test document lead to the exception and the number of exceptions occurred. In Figure 5 is the exception table of function "OnBeforeVideoDownload () " in the Fuzzing test, which is derived from mps.dll control of Storm player.

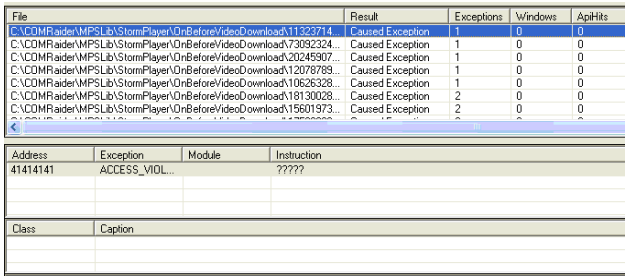


Figure 5. the exception list of the Web controls' vulnerability test

2) Detailed report on exception

We can check the detailed report on each item in exception table. The detailed report on test results is generated from the Fuzzing test on function "OnBeforeVideoDownload ()", which is derived from mps.dll control of Storm player. In Figure 6 is the detailed report on an item.

```
Exception Code:ACCESS VIOLATION
Disasm:41414141          ?????    ( )

Seh Chain :
------------------------------------------------------------------------
Called From                    Returns To
------------------------------------------------------------------------
Registers :
------------------------------------------------------------------------
EIP 41414141
EAX FFFFFF00
EBX 0180247C -> 00000000
ECX 41414141
EDX 00152908 -> 00000804
EDI 01802480 -> 00000000
ESI 01802468 -> 0169C4C4
EBP 01802478 -> 00000000
ESP 0013EB4C -> Asc : AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Block Disassembly :
------------------------------------------------------------------------
41414141        ?????          <--- CRASH

ArgDump :
------------------------------------------------------------------------
EBP+8   00000000
EBP+12  0000502A
EBP+16  00000001
EBP+20  FFFFAFD7
EBP+24  FFFFFFFF
EBP+28  00038AB4 -> 016974A4

Stack Dump:
------------------------------------------------------------------------
13EB4C 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  [................]
13EB5C 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  [................]
13EB6C 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  [................]
13EB7C 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  [................]
13EB8C 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  [................]
```

Figure 6. detailed report of exception

From the detailed report in Figure 6, we can easily get all the important information when the exception occurs, including the type of the exception, the state of current threads, the state of SEH list, the state of stack memories, the content of all important registers and the instructions near to the current EIP.

From the detailed report, we know that the value of the EIP register is "41414141", which is the hexadecimal code of classic long string "AAAA" used in test documents. It fully shows that if ingenious input parameters designed by the attacker are added, the EIP can be hijacked. And then the system can be boot to execute special codes designed by testers. Therefore, we can make sure there are vulnerabilities in function "OnBeforeVideoDownload ()" derived from mps.dll control. Experienced testers can analyze the specific location of vulnerability from the address of the collapse and the state of stacks and directions when collapse occurs in the detailed report on exceptions.

SUMMARY

As the Web ActiveX control has a particular feature and changes rapidly, the security testing of Web application has become quite difficult. Therefore, the Web security is becoming more and more important. This paper has a discussion on the model and the algorithm of Web security vulnerabilities detection. And the research has a breakthrough to deal with the Web ActiveX control vulnerabilities detection. The future work is as follows: enhancing the relevance of test data, in order to improve the ratio of the trigger vulnerabilities of test data; searching the "tagged word" in the address space of all the functions derived from ActiveX; searching functions derived from ActiveX and other specific implement methods which call unsafe methods.

ACKNOWLEDGMENT

This work was supported by the Industry-University-Research Institute Collaboration project of Guangdong province, China under Grant No. 2010B090400149、No. 2008B090500201、No. 2010B090400046, the scientific and technological plan project of Guangdong province, China under Grant No. 2008A010100001, Distinguished Young Talents in Higher Education of Guangdong, China under Grant NO.LYM10031 and the demostration (Innovation Cultivation) Base for The industry-University-Research Institute Collaboration on Network information security Detection, China Grant No. cgzhzd0807.

REFERENCES

[1] G. Wassermann and Z. Su. Sound and precise analysis of Web[M]. Applications for injection vulnerabilities. In PLDI, 2007.

[2] N. Jovanovic, C. Kruegel, and E. Kirda. Pixy: A static analysis tool for detecting web application vulnerabilities (short paper). In 2006 IEEE Symposium on Security and Privacy, Oakland, CA, May 2006.

[3] S. Artzi, A.Kie ˙zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst. Finding bugs in dynamic Web applications.Technical Report[J] .Science and Artificial Intelligence Laboratory,Feb. 2008.

[4] Guan Quan-long, Ye Saizhi，Yao Guo-xiang. Research and design of internet public opinion analysis system，SSME 2009, p 173-177.

[5] The Component Object Model: A Technical Overview, http://msdn2.microsoft.com/en-us/library/ms809980.aspx

[6] INFO: Difference Between OLE Controls and ActiveX Controls, http://support.microsoft.com/kb/159621

[7] Security Vulnerability Research & Defense : The Kill-Bit FAQ: part 2 of 3. http://blogs.technet.com/swi/archive/2008/02/07/The-kill_2DOO_Bit-FAQ_3AOO_-post-2-of-3.aspx

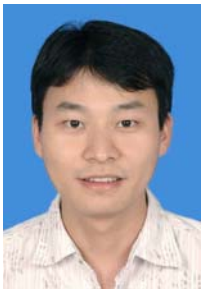
[8] Java Security Architecture: The Original Sandbox Model, http://java.sun.com/j2se/1.4.2/docs/guide/security/spec/security-spec.doc1.html#18313

[9] JPEG COM Marker Processing Vulnerbility in Netscape Browsers(2000), ttp://www.openwall.com/advisories/OW-002-netscape-jpeg/

[10] Howard, M., Pincus, J., and Wing, J.: Measuring Relative Attack Surfaces(2003), http://www.cs.cmn.edu/%7Ewing/publications/Howard-Wing03.pdf

[11] Yao, Guo-Xiang, Guan, Quan-Long,et al. Research and implementation of next generation network intrusion detection system based on protocol analysis, CCCM 2008, p 353-357.
[12] Guan, Quan Long; Yao, Guo Xiang; Ni, Kai Bin; Zhou, Mei Xiu. Research on fuzzing test data engine for web vulnerabilit,Advanced Materials Research, v 211-212, p 500-504, 2011
[13] Yao,Guo Xiang; Song, Ga Zi; Guan, Quan Long. Security model research and design based on separate storage of keys. 2010 2nd International Conference on Industrial Mechatronics and Automation, v 2, p 672-676, 2010

Guoxiang Yao was born in 1959. He is a male. He earned M.S. degree in computer science from University of Science and Technology of China. His major field of study is network information security, computer network and data security. He is a professor in Information Technology Science College, Jinan University.



Quanlong Guan was born in Shaoguan, Guangdong Province, China on 1981. He received his Bachelor's degree, Master's degree in computer science from Jinan University, Guangdong in 2003 and 2006 respectively.

He has been teaching at Jinan University, Guangdong, China since 2006. His current research interests include computer networks and information security.