International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:9, 2010

Research on Software Security Testing

Gu Tian-yang, Shi Yin-sheng, and Fang You-yuan

***Abstract*—**Software security testing is an important means to ensure software security and trustiness. This paper first mainly discusses the definition and classification of software security testing, and investigates methods and tools of software security testing widely. Then it analyzes and concludes the advantages and disadvantages of various methods and the scope of application, presents a taxonomy of security testing tools. Finally, the paper points out future focus and development directions of software security testing technology.

***Keywords*—**security testing, security functional testing, security vulnerability testing, testing method, testing tool

I. Introduction

WITH the wide use of computer, software becomes more complicated and large-scale, which also results in more software security problems increasingly. Software security is the ability of software to provide required function when it is attacked [1]. There is growing concern about security testing, because it is regarded as an important means to improve security of software.

Software security testing is the process to identify whether the security features of software implementation are consistent with the design. Software security testing can be divided into security functional testing and security vulnerability testing. Security functional testing ensures whether software security functions are implemented correctly and consistent with security requirements basing on security requirement specification. Software security requirements mainly include data confidentiality, integrity, availability, authentication, authorization, access control, audit , privacy protection, security management,etc. Security vulnerability testing is to discover security vulnerabilities as an attacker. Vulnerability refers to the flaws in system design, implementation, operation, management. Vulnerability may be used to attack, resulting in a state of insecurity, Security vulnerability testing is to identify software security vulnerabilities. In this paper , the current methods, techniques and tools of security testing are analyzed and summarized.

II. Major Methods of Security Testing

With growing concern about software security, research on security testing has made some progress.

A. Formal security testing

The basic idea of formal method is to build a mathematical model of the software, and provides software form specification supported by some formal specification language. Formal security testing methods can be classified into theorem proving and model checking. Theorem proving transforms program into logical formula, and then uses the axioms and rules to prove the program is a valid theorem. Model checking describes the behavior of the software by state transfer system S, using formula F, constructed by sequential logic, computation tree logic or the μ calculus,, to define requirements of the software implementation, and finds software vulnerabilities through automatic search for the state in S, which does not meet F.

A laboratory named Jet Propulsion Laboratory (JPL) of National Aeronautics and Space Administration (NASA) has carried out a formal security testing project [2]. Its main idea is to establish formal model of security requirements, such as state machine model. Security testing is accomplished by searching the state space for the specific state in violation of security restriction, which is also an unsafe state .As the model size and complexity increase, the state space grows exponentially, JPL developed a formal modeling framework (Flexible Modeling Framework, FMF) using SPIN to solve state explosion problem, and developed a testing tool based on property (Property Based Tester, PBT).

Method of formal security testing has some limitations. For theorem proving is difficult to achieved automatically, it requires high-quality staff to analyse, which is very time-consuming. So it is generally used to verify the design correctly rather than actual code. For the model checking method, exhaustive algorithm needs all practical implementation states, so it is hard to test infinite state system.and low efficient.

B. Model-based security testing

Model-based testing constructs a model by the behavior and structure of software, then derives test cases from test model. Finally drive software to run the test cases[3]. The behavior of Software system can be described by input and output sequence, activity diagram, sequence diagram, collaboration diagram, condition or data stream. Software behavior model and structure model is the exact description of the tested software, which can be used to generate test cases. Software testing models are commonly used , such as finite state machine, UML model, Markov chain.

Mark Blackburn made research on model-based security

GU Tian-yang Author is with Beijing Institute of System Engineering, Beijing, CO 100101 China (e-mail: somhan@ 163.com).
SHI Yin-sheng Author is with Beijing Institute of System Engineering, Beijing, CO 100101 China (e-mail: topsec@yeah.net).
FANG You-yuan Author is with Beijing Institute of System Engineering, Beijing, CO 100101 China (e-mail: fangyouyuan@gmail.com).
This work is supported by the National High Technology Research and Development Program of China under the grant No.2009AA01A404

functional testing [4]. The main project is Automated Security Functional Testing of NIST Computer Security Division (CSD). The main idea is to use SCRModeling tool for software to model the security functional requirements, which uses the form to describe security behavior model and then transforms it into test specification model. The T-VEC tool will generate test vectors (a set of input variables and desired output variables) basing on test specification model. It is also necessary to develop test driver schema and build mapping between target object and test environment. Finally test vectors will be driven to run over tested software (Figure 1). It is a general security functional testing method, which depends on the scope of security modeling capabilities, especially applies to the software, of which security function can be easily expressed by logical relations 'and' or 'or', such as authorization, access control.

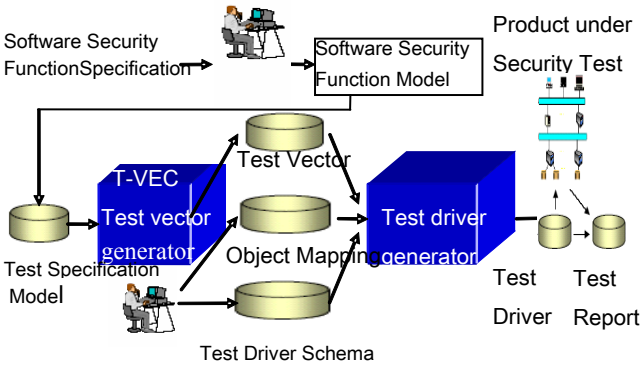

Fig. 1 Process flow diagram of automatic security functional testing

C. Fault injection-based security testing

Wenliang Du used fault injection technique for software security testing, which established fault mode of Environment-Application Interaction, EAI. [5] [6]. Fault injection focuses on the interaction points of application and environment, including user input, file system, network interface, environment variable. Related project is PROTOS Security Testing of Protocol Implementations of OUSPG (University of Oulu, Secure Programming Group), the target of the project is to test the protocol's security.

The main idea is to test whether software can response correctly using various types of protocol packets. In order to discover software security vulnerabilities, some fault data should be injected into various protocol packets, such as juggling the value of certain protocol's field. Protocols supported are HTTP, SIP, WAP, SNMP, etc. The main fault injection tools are CECIUM, DOCTOR, ORCHESTRA, NFTAPE, LOKI, Mendosus, OGSA, FAIL-FCI. Fault injection can effectively simulate a variety of abnormal behavior of software. The fault injection function can make the software mandatory reach a certain state, which can not reach easily by the other testing technology.

D. Fuzzy testing

Fuzzy Testing is effective to discover security vulnerability, which gets more and more attention. Fuzzy testing would inject random data into program to test whether it can running normally under the clutter input. Fuzzy testing is illogical, just creates clutter data. Fuzzy testing would find flaws of tested software, which are difficult for the other logical testing method.

E. Vulnerability scanning testing

Vulnerability scanning, as an important method to find software security risks, includes testing space scanning and known defects scanning. Testing space scanning deals with network port, string, procedure data, network data and other elements scanning, for example, through network port scanning, it can be found whether the port of software is opened which should not open. Known defects scanning finds known flaws usually basing on the defect library.

F. Property-based testing

Paper [7] describes a method of property -based testing. The method transforms security property of software into specification described by TASPEC language. It would extract the code in relation to specific property by program slicing technology, and discover violation of the code against security property specification. Property-based testing focuses on some specific security properties, which can meet requirement of classification and priority.

G. White box-based security testing

Static analysis, as one of common white-box based testing methods, is good at finding security bug, such as buffer overflow. The main technologies of static analysis are deducing, data flow analysis and constraint analysis [8]. Ben Breech and Lori Pollock suggested a dynamic compiler based security testing framework [9], which could insert attack code into the running program and test security mechanism of software. It is mainly used to test program-based attacks, such as stack buffer overflow.

H. Risk-based security testing

Brad Arkin, Gary McGraw, etc made research on risk-based security testing [11], which combined the risk analysis, security testing with software development lifecycle, as early as possible to find high-risk security vulnerabilities. This approach emphasized SDL (Security Development Lifecycle). Misuse model, abnormal scenario, risk analysis and penetration testing would be used in various stage of software development.

III. Taxonomy and Function of Security Testing Tools

Security testing tools are used as automatic or semi-automatic way to validate whether the system security function is correct, the security mechanism is effective and to find potential vulnerabilities. It can not only improve efficiency of testing but also reduce software security risk. There have emerged a large number of powerful security testing tools recently. Taxonomy of security testing tools would

be various. A good taxonomy should satisfy the requirements of clarity, orthogonality, objectivity, universality and usability[12]. Clarity indicates that every tool should be classified uniquely. The orthogonality requires that each category should be significantly different. Objectivity requires that each tool should be easily classified by attribute comparison, which reduces subjective factors. Universality suggests that the taxonomy should cover testing tool as widely as possible. Usability means the taxonomy is simple and easy to use. According to the tested object, security testing tools can be divided into host security testing tools, network security testing tools, and application security testing tools. NIST SAMATE (Software Assurance Metrics and Tool Evaluation) project suggests four standards for taxonomy of testing tools, that is, the life cycle stage testing tool applies to, method or technique testing tool adopts, automatic level of testing tool, and angle of view for testing tool. According to the different stage testing tool applies, we can divide tools by stage of requirement, design, coding, testing, maintenance. According to different method or technology testing tool adopts, we can divide tools by exclusion, detection, response, evaluation, and others. According to automatic level, we can divide tools by assisted analysis, semi-automation and automation. According to angle of view, we can divide tools by external and internal tools. For example, static source code analyser is a coding stage, semi-automation, detection and internal tool.

The above taxonomy is so coarse grain that it's hard to choose for test staff , In this paper, it is proposed to divide security testing tools into 11 categories, source code analyser, bytecode scanner, binary code scanner, database frangibility scanner, network vulnerability scanner, Web application vulnerability scanner, Web service scanner, dynamic analyser, configure analyse tool, requirement verification tool, design validation tool.

Static source code analyser scans the source code to detect buffer overflow, race condition and other security vulnerabilities by matching specific security flaws code mode. Some supervisor analyser can analyse data flow and control flow to reduce misreport, and provide report according to the type or priority of security vulnerability. Source code analyser is popular for its faster scanning, high-performance, and developers can find security vulnerabilities earlier during coding stage. Higher misreport is a disadvantage to this tool, and it is applicable to fewer languages, such as C, C++. For the languages, such as. NET, Java, the tool is not mature enough to discover security vulnerabilities efficiently. Bytecode scanner is similar with source code analyser, except that its scan object is Java bytecode.

Binary code scanner uses disassembly and pattern recognition technology to discover security vulnerabilities by scanning the executable binary code or DLL file. Advantage of the binary code scanner is its independence of source code. Disadvantage of the binary code scanner its dependence on reverse engineering and disassembly technology and higher misreport.

Database frangibility Scanner is a professional tool which is used to find database application security problems. It runs as SQL client to execute various SQL queries, and find out weakness of the database configuration, such as weak password, authorization, access control, etc. There are two typical operating modes for database frangibility scanner, infiltration attack mode and audit mode. Audit model requires the administrator account. Advantage of this tool is easy to use and effective to find security vulnerabilities of user management, privilege management, authentication and other configuration management of database. Lack of semantic comprehension of sensitive data in the database is its disadvantage.

Network vulnerability scanner scans remotely the open port, running service, operating system types, etc. It is good at finding the security vulnerabilities of operating system, service, protocol of network.

Web application vulnerability scanner simulates web client to scan privileged URL, weak CGI. It would record http interaction and inject malicious code in later interaction, then examine the response. It would find cross-site scripting, SQL injection, directory traversal, Cookie virus efficiently.

Web services scanner is a relatively new class of security testing tool, which focuses on testing web service security function and identifying vulnerabilities. Its typical functions include WSDL file scanning, listing and calling methods provided by web services, testing XML message encryption, XML message signature, signature verification and other security functions, conformance test for WS-Security.

Dynamic analyser constructs exceptional scenarios at run-time to test the existence of vulnerabilities. Typical functions include intercepting system calls, recording the implementation of procedure, detecting border of clusters, fault injection of file systems, network interfaces and system resources, identifying memory leaks, type mismatch, and other security issues.

Program configuration file, host configuration file, application server configuration file would be analysed by configure analyse tool to find security problems. Requirement verification tool would be used to verify validity, maturity, conformance and accuracy of software security requirement specifications.

Design validation tool focuses on detecting vulnerabilities in software design model. Various types of security testing tools are listed , as shown in Table 1. Due to requirement verification tool is immature, it is not listed.

TABLE1 TAXONOMY OF MAINSTREAM SECURITY TESTING TOOLS

Type of tools	Commercial	Open source or Free
Source code analyser	Source Code Analysis Suite of Fortify Software Co ; Prexis of Ounce Labs ; CodeAssure of Secure Software Co ; K7 of Klocwork Co ; Prevent of Coverity Co ; DevPartner SecurityChecker of Compuware Co ; C++Test, JTest, TEST, WebKing of Parasoft Co ; CodeCenter of CenterLine Systems Co ; CodeScan of CodeScan Labs ; CodeSonar of GrammaTech Co ; DevInspect of SPI Dynamics Co ; SoftCheck Inspector of SofCheck Co ; PolySpace of PolySpace Technologies Co.	Rough Auditing Tool for Security (RATS) ; lawFinder ; BOON ; ASTREE ; C Code nalyzer(CCA) ; Csur ; CQual ; Jlint ; ITS4 ; Splint ; UNO ; LAPSE ; PHP-Sat ; PMD
Bytecode scanner	BugScan of LogicLab Co.	FindBugs
Binary code Scanner	AspectCheck of Aspect Security Co ; BEAST of Security Innovation Co ; BugScan of LogicLab Co.	FxCop ; BugScam
Database frangibility scanner	AppDetective of Application Security Co ; Database Scanner of Internet Security Systems Co.	MetaCortex
Network vulnerability scanner	Internet Security Scanner of Internet Security Systems Co ; NTOSpider of NT Objectives Co ; GFI LANguard of GFI Co ; Retina of eEye Co ; Security Auditor's Research Assistant (SARA) of Advanced Research Co ; SAINT of Saintcorporation Co.	NMAP ; Nessus ; SuperScan ; Metasploit Framework
Web application vulnerability scanner	WebInspect of SPI Dynamics Co ; AppScan of Watchfire Co ; N-Stalker Web Application Security Scanner of N-Stalker Co ; Acunetix Web Vulnerability Scanner of Acunetix Co ; Burp suite of portswigger.net Co.	OWASP WebScarab ; OWASP Berretta ; Nikto ; Wikto ; Paros Proxy Spike Proxy ; EOR ; Pantera
Web service scanner	SOATest of Parasoft Co ; SOAPbox of Vordel Co ; WebServiceTester of Optimyz Co ; SOAPSonar of CrossCheck Co ; XRay Diagnosis of Forum Systems Co ; SOAPScope of MindReef Co ; e-TEST suite for Web Services Testing of Empirix Co ; QEngine of AdventNet Co ; LISA Complete SOA Test Suite of ITKO Co.	Foundstone WSDigger ; Pushtotest TestMaker
Dynamic analyser	Compuware BoundsChecker	Foundstone .NETMon ; CLR Profiler ; NProf
Configure analyse tool	Desaware CAS/Tester	Foundstone SSLDigger ; PermCalc
Design validation tool	SDMetrics of SDMetrics Co.	

IV. CONCLUSION

This paper analyses the definition, classification, major methods and tools of security testing, and suggests a classification which divides security testing into security functional testing and security vulnerability testing. Then proposes a taxonomy of security testing tools.

The directions of software security testing techniques include security functional model of authorization, access control; formal security testing; risk-based security testing and its application in software engineering; threat model and attack tree based security testing. In addition, along with the development of web service recently, how to deal with security testing for web service is a new challenge.

ACKNOWLEDGMENT

Gu Tianyang thanks all the participants for the valuable discussions and thanks his mother and wife for their support to his work.

REFERENCES

[1] Gary McGraw, Bruce Potter. “Software Security Testing”[J]. IEEE Security & Privacy, 2004, 2(5):81-85.
[2] David P. Gilliam, John D. Powell, Matt Bishop. “Application of Lightweight Formal Methods to Software Security”[C]. In proc. 14th IEEE International Workshops on Enabling Technologies (WETICE 2005), 13-15 June 2005, Linköping, Sweden.pp.160-165.
[3] Yan Jiong, etc. “Survey of Model-Based Software Testing” Computer Science, 2004.31(2)
[4] Ramaswamy Chandramouli, Mark Blackburn. “Automated Testing of Security Functions Using a Combined Model and Interface-Driven Approach”[C]. In proc. 37th Hawaii International Conference on System Sciences (HICSS-37 2004), 5-8 January 2004, Big Island, HI, USA.
[5] Du Wenliang , Mathur A P. “Vulnerability Testing of Software System Using Fault Injection”[R]. Coast TR 98-02, 1998.
[6] Du Wenliang, Aditya P. Mathur. “Testing for Software Vulnerability Using Environment Perturbation”[C]. In proc. DSN 2000.pp.603-612.

[7] George Fink, Matt Bishop. “Property Based Testing: A New Approach to Testing for Assurance”[J]. ACM SIGSOFT Software Engineering Notes, 1997, 22(4):74 ~ 80.

[8] Xia Yi-min, etc. “Security Vulnerability Detection Study Based on Static Analysis”. Computer Science, 2006.33(10).

[9] Ben Breech, Lori Pollock. “A Framework for Testing Security Mechanisms for Program-Based Attacks”[J]. ACM SIGSOFT Software Engineering Notes, 2005, 30(4).

[10] Lieven Desmet, Bart Jacobs, Frank Piessens, Wouter Joosen. “Threat modeling for web services based web applications”[C]. In proc. Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS 2004), September 2004, UK.pp.161-174.

[11] Brad Arkin, Scott Stender, Gary McGraw: “Software Penetration Testing”[J]. IEEE Security & Privacy ，2005，3(1): 84-87.

[12] Shi Yin-sheng, Deng Shi-wei, Gu Tian-yang, “Software security testing methods and tools”, Computer Engineering and Design, January 2008,Vol.29,pp.27-30

Gu Tianyang is an assistant researcher at Beijing Institute of System Engineering. His technical interests include data sharing, computer security, software testing, virtual reality. Contact him at Beijing Institute of System Engineering, Beijing 100101, China; somhan@163.com