University "Ss. Cyril and Methodius"
Faculty of Computer Science and Engineering

Proceedings of the Tenth International Conference on Informatics and Information Technology

CIIT 2013

April 18-21, 2013, Molika, Bitola, Macedonia

Editors:
Igor Mishkovski and Sasko Ristov

Skopje, 2014

Igor Mishkovski and Sasko Ristov

University "Ss. Cyril and Methodius"
Faculty of Computer Science and Engineering
Rugjer Boshkovikj 16, P.O. Box 393, 1000 Skopje, Macedonia
http://www.finki.ukim.mk/

Technical editors: Magdalena Kostoska and Goran Velkoski
Cover page design: Dejan Mladenovski
Publisher: Faculty of Computer Science and Engineering, Skopje, Macedonia
Printed in Skopje, Macedonia

Total print run: 150
Skopje, 2014

CIP - каталогизација на публикација
Народна и универзитетска библиотека „Св.Климент Охридски“, Скопје

004.7:621.39(062)
004(062)

PROCEEDINGS of the Tenth Conference on Informatics and Information Technology (10 ; 2013 ; Bitola)
Proceedings of the Tenth Conference on Informatics and Information Technology : CIIT 2013, April, 18-21, Molika, Bitola, Macedonia / editors Igor Mishkovski and Sasko Ristov. - Skopje : Faculty of Computer Science and Engineering, 2013. - 311 стр. : граф. прикази ; 30 см

Библиографија кон трудовите

ISBN 978-608-4699-01-9
1. Mishkovski, Igor [уредник] 2. Ristov, Sasko [уредник]
а) Информациско-комуникациски технологии - Собири б) Компјутерски науки - Собири

COBISS.MK-ID 92848138

Preface

Continuing the tradition, the decennial 10th Conference of Informatics and Information Technology – CIIT2014 was held for the tenth time, traditionally in Bitola, Macedonia during April 18-21, 2013. The first 8 years this conference was being organized by the Institute of Informatics at Faculty of Natural Sciences and Mathematics, while these last two years it was organized by the Faculty of Computer Science and Engineering (FCSE), Ss. Cyril and Methodius University in Skopje, which is the successor of the two largest institutions in the area of informatics and computer technologies in Macedonia, i.e., the Institute of Informatics at Faculty of Natural Science and Mathematics and the Institute of Computer Techniques and Informatics at Faculty of Electrical Engineering and Information Technologies. FCSE continues the tradition of giving the opportunity to researchers to present their latest results in the field of Informatics and Information Technologies.

During the three days of the conference, 66 accepted papers were presented in 12 regular sessions. Additionally, a special PhD student workshop was held with 10 presenters, as well as a student session with 9 accepted presentations of FCSE's student projects.

Professor Marjan Gusev, the founder/founding father of the CIIT conference, gave the invited talk "Ten Years of CIIT".

Apart from the variety of topics and areas that were covered by the presentations, the participants broadened their scientific knowledge and social networking during the several social events.

We believe that the growing trend of papers and presenters will continue to promote the CIIT conference as an influential international conference with great impact on ICT research and development.

The Editors,

Igor Mishkovski and Sasko Ristov

Editors

Igor Mishkovski and Sasko Ristov

Program committee

Smile Markovski, PhD
Dragan Mihajlov, PhD
Margita Kon-Popovska, PhD
Vancho Kusakatov, PhD
Lupcho Kocarev, PhD
Marjan Gushev, PhD
Katerna Zdravkova, PhD
Suzana Loshkovska, PhD
Zhaneta Popeska, PhD
Kosta Mitreski, PhD
Verica Bakeva, PhD
Vladimir Trajkovikj, PhD
Ana Madevska Bogdanova, PhD
Dejan Gjorgjevikj, PhD
Dimitar Trajanov, PhD
Sonja Gievska, PhD
Andrea Kulakov, PhD
Ljupcho Antovski, PhD
Slobodan Kalajdzhiski, PhD
Nevena Ackovska, PhD
Marija Mihova, PhD
Sonja Filiposka, PhD
Anastas Mishev, PhD
Ivan Chorbev, PhD
Goce Armenski, PhD
Dejan Spasov, PhD
Goran Velinov, PhD
Vesna Dimitrova, PhD
Boro Jakimovski, PhD
Lasko Basnarkov, PhD
Igor Trajkovski, PhD
Ivica Dimitrovski, PhD
Igor Mishkovski, PhD
Gjorgji Madzarov, PhD
Smilka Janeska-Sarkanjac, PhD
Sasko Ristov, PhD
Aleksandra Mileva, PhD
Pece Mitrevski, PhD
Sime Arsenovski, PhD
Adrijan Bozinovski, PhD

List of Participants

Aleksandar Kotevski
Adrijan Bozhinovski
Aleksandar Bahtovski
Aleksandar Donevski
Aleksandar Stojmenski
Aleksandra Bogojeska
Aleksandra Kanevce
Aleksandra Mileva
Aleksandra Popovska-Mitrovikj
Ana Madevska Bogdanova
Anastas Mishev
Andrea Kulakov
Andrea Naumovski
Andrej Gajduk
Biljana Risteska Stojkoska
Biljana Stojchevska
Bisera Dugalikj
Bojan Ilioski
Bojan Kostadinov
Bojan Najdenov
Bojana Koteska
Boro Jakimovski
Christian Huemer
Dalibor Serafimovski
Daniel Kareski
Daniela Kjurchievska
Danijel Novakovic
Darko Martinovikj
Dejan Gjorgjevikj
Dejan Konevski
Dejan Kovachev
Dejan Spasov
Dimitar Trajanov
Dobre Blazhevski
Dragan Jovanovski
Dragan Mihajlov
Dragi Zlatkovski
Eftim Zdravevski
Elena Janevska
Elena Stojanova
Elena Vlahu-Gjorgievska
Elvis Imeroski
Emil Stankov
Eva Blazhevska
George Tanev
Georgina Mirceva
Gjorgi Kakasevski
Gjorgji Madjarov
Goce Gjorgjijoski
Goran Stanoev
Goran Velinov
Goran Velkoski
Hristina Mihajloska
Igor Kulev
Igor Mishkovski
Ilija Jolevski
Ilinka Ivanoska
Ivan Chorbev
Ivan Kitanovski
Ivana Gjorgievska
Ivica Dimitrovski
Jasna Veljanovska
Josip Kolic
Jovan Janevski
Jugoslav Achkoski
Katarina Trojacanec
Kire Jakimoski
Kliment Mahoski
Kosta Mitreski
Kristina Spirovska
Leonid Djinevski
Ljupco Antovski
Magdalena Kostoska
Maja Siljanoska
Marija Mihova
Marija Petkovska
Marina Vasileva
Marjan Gusev
Martina Toceva
Mile Jovanov
Milos Jovanovik
Monika Simjanoska
Natasa Suteva
Nevena Ackovska
Nevena Serafimova
Nevenka Pop-Angelova
Nikola Koteli
Pece Mitrevski
Petre Lameski
Predrag Tasevski
Radmila Kotevska
Radoslav Bozhinovski
Riste Stojanov
Roman Brandt
Sanja Angelovska
Sashka Ivanovska
Sasho Gramatikov
Sasko Ristov
Saso Koceski
Sime Arsenovski
Slobodan Kalajdziski
Smile Markovski
Sonja Filiposka
Sonja Gievska
Stefan Spasovski
Suzana Loshkovska
Tatjana Vasileva–Stojanovska
Tome Dimovski
Toni Janevski
Toni Malinovski
Trajche Kocev
Vangel Ajanovski
Vasil Vangelovski
Vasja Pavlov
Vasko Popovski
Veno Pachovski
Verica Bakeva
Vesna Dimitrova
Vesna Kirandziska
Vladimir Popovski
Vladimir Trajkovik
Vladimir Zdraveski
Zaneta Popeska
Zlatka Trajcheska
Zoran Milevski
Zoran Zdravev

Table of Contents

eWorld - eWork, eCommerce, eBusiness, eLearning

Intelligent Systems, Robotics, Bioinformatics

Computer Networks, Multimedia and Digitalization

Theoretical Foundations of Informatics and Signal Processing

Security and Cryptography

Computer Architecture and Parallel Processing

Distributed Systems, GRID and Cloud Computing

Wireless and Mobile Computing

EVALUATION AND TESTING OF SEVERAL FREE/OPEN SOURCE WEB VULNERABILITY SCANNERS

Nataša Šuteva
Faculty of Computer Science, UGD
Štip, Macedonia

Dragi Zlatkovski, Aleksandra Mileva
Faculty of Computer Science, UGD
Štip, Macedonia

ABSTRACT

Web Vulnerability Scanners (WVSs) are software tools for identifying vulnerabilities in web applications. There are commercial WVSs, free/open source WVSs, and some companies offer them as a Software-as-a-Service. In this paper, we test and evaluate six free/open source WVSs using the web application WackoPicko with many known vulnerabilities, primary for false negative rates.

I. INTRODUCTION

Our everyday live heavily depends on using different web applications, as web e-mail clients, web instant messaging clients, Voice over IP services, e-learning portals, social networks, electronic banking, e-commerce platforms, etc. Because of this, the web applications became the most interest target for attackers to gain an unauthorized account access, steal sensitive data and identity, etc.

The OWASP (Open Web Application Security Project) Top Ten 2013 [16] offers a list of the most critical Web application vulnerabilities, including different types of injection, broken authentication and session management, cross-site scripting, cross-site request forgery, etc. This list is often used also as a minimum standard for website vulnerability assessment and PCI compliance according to Payment Card Industry Data Security Standard (PCI DSS). Classification of web application vulnerabilities can be found also in Common Vulnerabilities and Exposures database [3] and Web Application Security Consortium (WASC) Threat Classification v2.0 [20].

Web Application Security Scanners (WASSs) or Web Vulnerability Scanners (WVSs) are software tools for identifying potential vulnerabilities in the web applications, independently of the particular technology used for their implementation. They access the web applications in the same manner as user do, through the web front-end. Usually they are black-box testers, because they do not have access to the source code, so they detect vulnerabilities by actually performing attacks or by looking for known vulnerabilities and report potential exposures.

The beauty of WVSs hides in automatically and cost-effective conduction of security checks and production of the final report. Almost every report includes a remedy for found vulnerability, which is necessary for PCI compliance. Today there are more than 130 scanning vendors approved for PCI compliance [11]. Vulnerability scanning is essential part of maintaining security in a given organization and should be used continuously, especially when new version of web application or new equipment or technology is planning to use. But WVSs are not all-in-one oracles, they are not capable of detecting all of the possible vulnerabilities and attack vectors that exist. There are several reports showing that today WVSs fail to detect a significant number of vulnerabilities in test applications [1, 4, 12, 14, 15, 22]. Bau et al [1], testing eight WVSs, showed that WVSs need to be improved in detection of the “stored” and second-order forms of XSS and SQLI, and in understanding of active content and scripting languages. Khoury [7, 8] analyzed three state-of–art black box WVSs against stored SQLI, and their results showed that stored (persistent) SQLI are not detected even when these automated scanners are taught to exploit the vulnerability. They propose also a set of recommendations for increasing a detection rate in WVSs for this type of vulnerability. Doupé et al [4] tested eleven WVSs, and found that eight out of sixteen vulnerabilities were not detected by any of the used scanners. They discuss also a critical limitations of current WVSs, lack of better support for well known, pervasive technologies as JavaScript and Flash, and the need for more sophisticated algorithms to perform “deep” crawling and track the state of the application under test.

Kals et al [6] implement an automated black box scanner SecuBat which targets XSS and SQLI vulnerabilities. McAllister et al [10] also implement an automated black box scanner which targets reflected and stored XSS utilizing user interactions. Maggi et al [9] discuss techniques applicable to black box testing, for reducing the number of false positives. Fonseca et al [5] evaluated the XSS and SQLI detection performance of three WVSs via automated software fault-injection methods.

For evaluating and testing WVSs, vulnerable test applications are needed. These applications need to have exactly listed known vulnerabilities, so one can obtain the false positive and false negative rates also. Unfortunately, no standard test suite is currently available. There are several well-known, publicly-available, vulnerable web applications like DVWA (Dam Vulnerable Web Application) [13] and WebGoat [17], but their design is focused more on teaching web application security rather than testing WVSs. The exception is the realistic and fully functional web application WackoPicko [21] with 16 known vulnerabilities, created by A. Doupé, and used in [4] for their testing. We use this web application for our experiments. Additionally, WASC [19] has published evaluation criteria for web application scanners.

Because most of the research papers are concentrated on commercial WVSs, we decided to test and evaluate only free/open source WVSs. After Introduction Section, Section II is devoted to basic architecture of the black box WVSs. In Section III we give brief explanation of used testbed application, used six WVSs with their general characteristics and input vector support, followed by used methodology and obtained results on the false negative rates at the first place. At the end, we give short concluding remarks.

II. BLACK BOX WEB VULNERABILITY SCANNERS

Conceptually, almost all WVSs consist of three main components: a crawling component, an attacker component, and an analysis component.

At the beginning of the scanning process, the user enters at least one URL, with or without user credentials for the given web application. Using these data, the crawling component identifies all the reachable pages in the application, and all the input points to the application, such as the parameters of GET requests, the input fields of HTML forms, etc. After user sets the scanning profile, scanners can proceed automatically or with user interaction. We used only automated mode for our experiments.

The attacker component analyzes discovered data and for each web form, for each input and for each vulnerability type for which the WVS has test vectors, the attacker module generates values that are likely to trigger a vulnerability. Then, the form content is send to the web server using either a GET or POST request, and appropriate response is obtained from the server via HTTP.

Next, the analysis module has to parse and interpret the server response. Decision if a given attack was successful is made by calculation of confidence value, by using attack-specific response criteria and keywords.

III. EXPERIMENTS AND RESULTS

A. Vulnerable web application

Vulnerable WackoPicko application is a photo sharing and photo-purchasing site. Users of WackoPicko can upload photos, browse other user's photos, comment on photos, and purchase the rights to a high-quality version of a photo. It has 10 vulnerabilities accessible without authentication (reflected and stored XSS, reflected XSS behind JavaScript, predictable Session ID for admin, weak admin password, reflected SQLI, command line injection, file inclusion, unauthorized file exposure, and parameter manipulation), and 6 vulnerabilities accessible after logging into the web site (multi-step stored XSS, stored SQLI, directory traversal, forceful browsing, logic flaw and reflected XSS behind Flash).

The web server hosting WackoPicko and used in our experiments was run in the OWASP Broken Web Applications Project virtual machine [18], which has numerous intentionally vulnerable applications (we ignore other applications). The following technologies are used: Apache 2.2.14 (Ubuntu), PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch, and MySQL 5.0.67.

B. Tested Web Vulnerabilities Scanners

The scanners were run on a machine with a Pentium (R) Dual Core 2 x 2.00GHz CPU, 4 GB of RAM, and Windows 7 Home Premium.

Table 1 lists the six free/open source WVSs used in our study and their general characteristics. All have graphical user interface and support for proxy mode (manual crawling). Only NetSparker Community Edition and N-Stalker Free 2012 run only on Windows, and other four can be installed on Linux and OS X also. W3Af additionally is available on FreeBSD and Open BCD. Their input vector support are given on Table 2. Many different characteristic comparisons on older versions of these WVSs can be found on Chen's web site SecToolMarket [2].

Table 1: General characteristics of the evaluated scanners

	NetSparker Community Edition	N-Stalker Free 2012	OWASP ZAP	W3Af	Iron WASP	Vega
Company/ Creator	Mavituna Security	N-Stalker	OWASP	W3Af Devel.	L. Kuppan	Subgraph
Version	2.5	7.1.1.126	2.0.0	1.2-r6654	0.9.5.0	1.0 (beta)
Licence/ Technology	Freeware .Net 3.5	Freeware Unknown (Win32)	ASF2 Java 1.6.x	GPL2 Python 2.6.x	GPL3 .Net 2.0 SP2	EPL1 Java 1.6.x
Operating System	Windows	Windows	Windows Linux OS X	Windows Linux OS X FreeBSD OpenBSD	Windows Linux OS X	Windows Linux OS X
Authent.		Yes	Yes	Yes		
Report		Yes	Yes	Yes		
Scan Log	Yes		Yes	Yes	Yes	Yes

NetSparker Community Edition have many features disabled, compared to its commercial version, but still you can scan and exploit SQL injection vulnerabilities without any false-positives.

N-Stalker Free 2012 provides a restricted set of features, compared to its commercial version, and will inspect up to 500 pages within target application.

OWASP Zed Attack Proxy (ZAP) is an easy to use integrated scanning and penetration testing tool, and it is designed to be used by people with a wide range of security experience.

Table 2: Supporting input vectors by the evaluated scanners

	NetSparker Community Edition	N-Stalker Free 2012	OWASP ZAP	W3Af	Iron WASP	Vega
HTTP Query String Parameters	Yes	Yes	Yes	Yes	Yes	Yes
HTTP Body Parameters	Yes	Yes	Yes	Yes	Yes	Yes
HTTP Cookie Parameters		Yes		Yes	Yes	
HTTP Headers		Yes		Yes	Yes	Yes
HTTP Parameter Names					Yes	
XML Element Content				Yes	Yes	
XML Attributes					Yes	
XML Tags					Yes	
JSON Parameters					Yes	
Flash Action Message Format	Yes					
Custom Input Vector					Yes	
SUMMARY	**3**	**4**	**2**	**5**	**10**	**3**

W3Af stands for Web Application Attack and Audit Framework, it is written in Python, and it was started by Andres Riancho in March 2007. In July 2010, W3Af announced its sponsorship and partnership with Rapid7. It uses more than 130 plug-ins. Users have available a command-line interface also.

IronWASP stands for Iron Web application Advanced Security testing Platform, created by Lavakumar Kuppan. It uses various external libraries, as IronPython, IronRuby, Json.NET, Jint, etc, making it more powerful. It has a scripting shell for both Python and Ruby giving full access to the IronWASP framework, and this can be used by the pen testers to write their own fuzzers, create custom crafted request, analysis of logs, etc

Vega includes an automated scanner for quick tests and an intercepting proxy for tactical inspection.

C Methodology

In our experiments, scanners that support authentication, were run without logging and with logging, and only the default values for configuration parameters were used. In the NO_LOG mode, the scanner was directed to the initial page of WackoPicko and told to scan for all vulnerabilities. In the LOG mode, the scanner was given first a valid username and password. We did not use proxy mode for scanners that have support for it. For N-Stalker Free 2012 we start automated mode with OWASP Policy. W3Af is run with activated plugins: audit, auth, bruteforce, grep and mangle.

D. Results

Figure 1 plots the time needed for each scanner to scan used web application. One can see, that running time ranges from 3 minutes to 9 hours and 52 minutes.

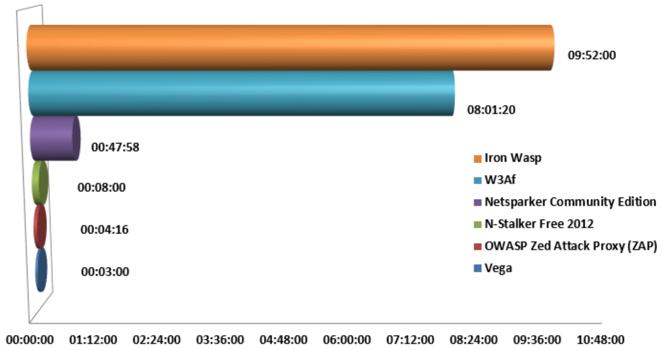

Figure 1: Running times of evaluated scanners

Table 3: Numbers of vulnerabilities of the evaluated scanners according to their severity without logging

	NetSparker Community Edition	N-Stalker Free 2012	OWASP ZAP	W3Af	Iron WASP	Vega
High Vulnerabilites	7	2	4	1	45	3
Medium Vulnerabilites	15	4	18	1	78	1
Low Vulnerabilites	8	16	414		8	25
Informational Vulnerabilites	12	21	177	9	2	17
SUMMARY	**42**	**44**	**613**	**11**	**133**	**46**

The number of found vulnerabilities classified according to their severity is given on Table 3. The total number ranges from 11 to 613 vulnerabilities. High values for founded vulnerabilities do not mean better scanners.

From the evaluated scanners, we find that the report from OWASP ZAP is very confusing, because it mixes vulnerabilities with different severity.

At the start, we know that three scanners NetSparker Community Edition, IronWasp and Vega, do not support authentication, so they could not find any of the vulnerabilities accessible after authentication.

Table 4 summarized obtained results. An empty cell indicates that the given scanner did not discover the vulnerability. No_LOG means that the given vulnerability was found without authentication. One can see from the obtained results, that for WVSs that support authentication with scanning, the scanners did not find additional vulnerabilities. Also, W3Af for example, did not find any of the known vulnerabilities.

Table 4: Characteristics of the evaluated scanners

	NetSparker Community Edition	N-Stalker Free 2012	OWASP ZAP	W3Af	Iron WASP	Vega
Reflected SQLI	No_LOG		No_LOG		No_LOG	No_LOG
Stored SQLI						
Reflected XSS	No_LOG	No_LOG	No_LOG		No_LOG	No_LOG
Stored XSS	No_LOG		No_LOG		No_LOG	
Reflected XSS behind JavaScript	No_LOG	No_LOG				
Reflected XSS behind Flash						
Predictible Session ID						
Command line injections					No_LOG	
File inclusion					No_LOG	No_LOG
File Exposure						
Parameter Manipulation						
Directory Traversal						
Logic Flow						
Forceful browsing						
Weak passwords						

Table 5 summarized the numbers of found known vulnerabilities and the number of false negatives. All examined free/open source WVSs have very high rates of false negatives, running from 68,8% for IronWasp to 100% for W3Af. NetSparker can scan only SQLI and XSS vulnerabilities without authentication, so it performed very well, with finding all possible vulnerabilities of these kinds. N-Stalker Free 2012 offer only reduced analysis of XSS vulnerabilities, with or without authentication, so it found only two out of five XSS vulnerabilities. Other modules are disabled in this version.

Table 5: Number of false negative

	Number of found vulnerabilities	Number of false negative
NetSparker Community Edition	**4**	**12**
N-Stalker Free 2012	**2**	**14**
OWASP ZAP	**3**	**13**
W3Af	**0**	**16**
Iron WASP	**5**	**11**
Vega	**3**	**13**

IV. CONCLUSIONS

Because the web application WackoPicko is almost three years old, and has only 16 known vulnerabilities, and because it is only one of its type, there is a need of a new application with more recent vulnerabilities, with versions other than Apache/PHP/MySQL also. Also, OWASP Broken Web Applications Project, need to be updated with the latest versions of used technologies, because Apache 2.2.14 (Ubuntu) and PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch have known vulnerabilities and exploits, which have been detected by WVSs, and have made our tasks harder. Because of this, we did not gave the false positive rates.

REFERENCES

[1] J. Bau, E. Bursztein, D. Gupta and J. Mitchell, "State of the Art: Automated Black-Box Web Application Vulnerability Testing", In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2010.

[2] S. Chen, SecToolMarket, [Online]. Available: http://sectoolmarket.com/

[3] Common Vulnerabilities and Exposures. [Online]. Available: http://cve.mitre.org.

[4] A. Doupe, M. Cova and G. Vigna, "Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners". In C. Kreibich, M. Jahne (Eds.) *Proceedings of the 7th International conference on Detection of Intrusions and Malware, and Vulnerability Assessment - DIMVA'10*, pp. 111-131, Springer Berlin Heidelberg 2010.

[5] J. Fonseca, M. Vieira, and H. Madeira, "Testing and comparing web vulnerability scanning tools for sql injection and xss attacks", In *Proceedings of the 13th IEEE Pacific Rim International Symposium. Dependable Computing (PRDC 2007)*, vol. 0, pp. 365–372, 2007.

[6] S. Kals, E. Kirda, C. Kruegel, and N. Jovanovic, "Secubat: a web vulnerability scanner". *In Procedings of the 15th International Conference World Wide Web (WWW '06)*, pp. 247–256, 2006.

[7] N. Khoury, P. Zavarsky, D. Lindskog and R. Ruhl, "Testing and assessing web vulnerability scanners for persistent SQL injection attacks", *First International Workshop on Security and Privacy Preserving in e-Societies (SeceS '11)*, New York, NY, USA, 2011.

[8] N. Khoury, P. Zavarsky, D. Lindskog and R. Ruhl, "An Analysis of Black-Box Web Application Security Scanners against Stored SQL Injection", In *Proceedings of the IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT 2011) and 2011 IEEE Third International Conference on Social Computing (SOCIALCOM 2011)*, Boston, USA, October 2011.

[9] F. Maggi, W. K. Robertson, C. Kr¨ugel, and G. Vigna, "Protecting a moving target: Addressing web application concept drift", In *Proceedings of the 12th Internationall Symposium Recent Advances in Intrusion Detection (RAID'09)*, pp. 21–40, 2009.

[10] S. Mcallister, E. Kirda, and C. Kruegel, "Leveraging user interactions for in-depth testing of web applications", In *Proceedings of the 11th Internationall Symposium Recent Advances in Intrusion Detection (RAID '08)*, pp. 191–210, 2008.

[11] Payment Card Industry Security Standards Council. Approved Scanning Vendors. [Online]. Available: https://www.pcisecuritystandards.org/approved_companies_providers/approved_scanning_vendors.php.

[12] Peine, H.: Security Test Tools for Web Applications. Technical Report 048.06, Fraunhofer IESE (January 2006)

[13] RandomStorm OpenSource project, DVWA (Dam Vulnerable Web Application), [Online]. Available: http://www.dvwa.co.uk/.

[14] L. Suto, "Analyzing the Effectiveness and Coverage of Web Application Security Scanners", [Online]. October 2007. Available: http://www.stratdat.com/webscan.pdf.

[15] L. Suto, "Analyzing the Accuracy and Time Costs of Web Application Security Scanners", [Online]. Feb 2010. Available: http://ha.ckers.org/files/Accuracy and Time Costs of Web App Scanners.pdf

[16] Open Web Application Security Project, "OWASP Top Ten Project" [Online]. Available: http://www.owasp.org/index.php/Category: OWASP Top Ten Project.

[17] Open Web Application Security Project (OWASP), OWASP WebGoat Project. [Online]. Available: http://www.owasp.org/index.php/Category:OWASP WebGoat Project

[18] Open Web Application Security Project (OWASP), OWASP Broken Web Applications Project. [Online]. Available: https://www.owasp.org/index.php/OWASP_Broken_Web_Applications_Project#tab=Main

[19] Web Application Security Consortium, "Web Application Security Scanner Evaluation Criteria", [Online]. Available: http://projects.webappsec.org/Web-Application-Security-Scanner-Evaluation-Criteria.

[20] Web Application Security Consortium, "Web Application Security Scanner Threat Classification", [Online]. Available: http://projects.webappsec.org/f/WASC-TC-v2_0.pdf.

[21] WackoPicko [Online]. Available: https://github.com/adamdoupe/WackoPicko

[22] Wiegenstein, A., Weidemann, F., Schumacher, M., Schinzel, S.: Web Application Vulnerability Scanners—a Benchmark. Technical Report, Virtual Forge GmbH (October 2006)