Detecting and Preventing SQL Injection and XSS Attack using Web Security Mechanisms

Ms. Aboli Vairagade, Prof. D.M. Sable, Prof. V. R. Wadhankar
Agnihotri Collge of Engineering Nagthana Wardha

Abstract:-In this paper we proposed a system prototype tool to evaluate web application security mechanisms. The methodology is based on the idea that injecting realistic vulnerabilities in a web application and attacking them automatically can be used to support the assessment of existing security mechanisms and tools in custom setup scenarios. To provide true to life results, the proposed vulnerability and attack injection methodology relies on the study of a large number of vulnerabilities in real web applications. To remove the vulnerabilities by implementing a concrete Vulnerability & Attack Injector Tool (VAIT) for securing web applications.

To prevent various attacks like follows:
1. SQL Injection (SQLi)
2. Cross Site Scripting (XSS)
3. Brute Force Attack
4. Shoulder surfing Attack
5. Social Attack.
6. Dictionary Attack

*****

1. INTRODUCTION

Nowadays there is an increasing dependency on web applications, ranging from individuals to large organizations. Almost everything is stored, available or traded on the web. Web applications can be personal websites, blogs, news, social networks, web mails, bank agencies, forums, e-commerce applications, etc. The omnipresence of web applications in our way of life and in our economy is so important that it makes them a natural target for malicious minds that want to exploit this new streak.

Conceptually, the attack injection consists of the introduction of realistic vulnerabilities that are after wards automatically exploited (attacked). Vulnerabilities are considered realisticbecause they are derived from the extensive field study on real web application vulnerabilities presented in [16], and are injected according to a set of representative restrictions and rules defined in [17]. The attack injection ethodology is based on the dynamic analysis of information obtained from the runtime monitoring of the web application behavior and of the interaction with external resources, such as the backend database. This information, complemented with the static analysis of the source code of the application, allows the effective injection of vulnerabilities that are similar to those found in the real world. Although this methodology can be applied to various types of vulnerabilities, we focus on of the most widely exploited and serious web application vulnerabilities that are SQL Injection (SQLi) and Cross Site Scripting (XSS) [3], [6]. Attacks to these vulnerabilities basically take advantage of improper coded applications due to nchecked input fields at user interface. This allows the attacker to change the SQL commands that are sent to the database (SQLi) or through the input of HTML and scripting languages (XSS).

A Brute-Force Attack, or exhaustive key search, is a cryptanalytic attack that can, in theory, be used against any encrypted data[1] (except for data encrypted in an information-theoretically secure manner). Such an attack might be used when it is not possible to take advantage of other weaknesses in an encryption system (if any exist) that would make the task easier. It consists of systematically checking all.

Shoulder surfing can also be done at a distance using binoculars or other vision-enhancing devices. Inexpensive, miniature closed-circuit television cameras can be concealed in ceilings, walls or fixtures to observe data entry. To prevent shoulder surfing, it is advised to shield paperwork or the keypad from view by using one's body or cupping one's hand.

A Dictionary Attack is based on trying all the strings in a prearranged listing, typically derived from a list of words such as in a dictionary (hence the phrase dictionary attack). [1] In contrast to a brute force attack, where a large proportion of the key space is searched systematically, a dictionary attack tries
only those possibilities which are deemed most likely to succeed. Dictionary attacks often succeed because many people have a tendency to choose short passwords that are ordinary words or common passwords, or simple variants obtained.

Social Attack, in the context of information security, refers to psychological manipulation of people into performing actions or divulging confidential information. A type of confidence trick for the purpose of information gathering, fraud, or system access, it differs from a traditional "con" in that it is often oneof many steps in a more complex fraud scheme. The term "social engineering" as an act of psychological manipulation is also associated with the social sciences, but its usage has caught on among computer and information security professionals.

2. RELATED WORK:

[1] Jose Fonseca, Marco Vieira, and Henrique Madeira produce the evaluation of web security mechanisms using vulnerability & attack injection.

[2] D. Avresky, J. Arlat, J.C. Laprie, and Y. Crouzet, produce the fault injection for formal testing of fault tolerance.
[3] J. Arlat, A. Costes, Y. Crouzet, J.-C. Laprie, and D. Powell, produce the fault injection and dependability evaluation of fault-tolerant systems.
[4] N. Neves, J. Antunes, M. Correia, P. Ver_ıssimo, and R. Neves, produce the using attack injection to discover new vulnerabilities.
[5] N. Jovanovic, C. Kruegel, and E. Kirda, produce the, precise alias analysis for static detection of web application vulnerabilities.
[6] IBM Global Technology Services produce the, IBM internet security systems x-force 2012 trend & risk report.
[7] M. Fossi, produce the semantic report on the underground economy, semantic security response.
[8] D. Powell and R. Stroud, produce the, Conceptual Model and Architecture of MAFTIA.
[9] V. Krsul, produce the, software vulnerability analysis.
[10] J. Fonseca and M. Vieira, produce the mapping software faults with we security vulnerabilities.

3. PROPOSED SYSTEM AND WORK

The methodology proposed was implemented in a concrete Vulnerability & Attack Injector Tool (VAIT) for web applications. The tool was tested on top of widely used applications in two scenarios. The first to evaluate the effectiveness of the VAIT in generating a large number of realistic vulnerabilities for the offline assessment of security tools, in particular web application vulnerability scanners. The second to show how it can exploit injected vulnerabilities to launch attacks, allowing the online evaluation of the effectiveness of the counter measure mechanisms installed in the target system, in particular an intrusion detection system.

In practice, the use of both static and dynamic analysis is a key feature of the methodology that allows increasing the overall performance and effectiveness, as it provides the means to inject more vulnerability that can be successfully attacked and discarded those that cannot.

The proposed methodology provides a practical environment that can be used to test countermeasure mechanisms (such as intrusion detection systems (IDSs), web application vulnerability scanners, web application fire-walls, static code analyzers, etc.), train and evaluate security teams, help estimate security measures (like the number of vulnerabilities present in the code), among others. This assessment of security tools can be done online by executing the attack injector while the security tool is also running; or offline by injecting a representative set of vulnerabilities that can be used as a test bed for evaluating a security tool.

3.1 SYSTEM ARCHITECTURE:

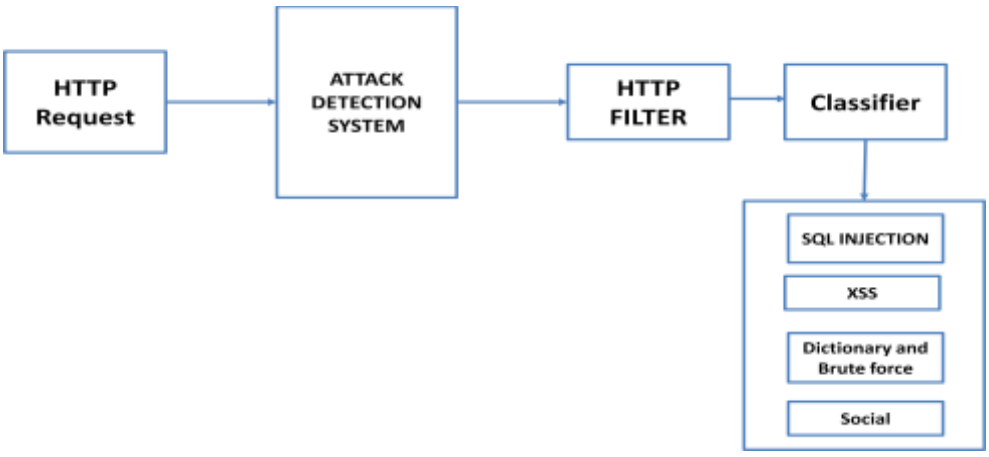


Fig1. System Architecture

3.2 MODULES:

3.2.1 DETECTING AND PREVENTING SQL INJECTION ATTACK

SQL injection is a code injection technique, used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker).[1] SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literalescape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

3.2.2 DETECTING AND PREVENTING XSS ATTACK

Cross-Site Scripting (XSS) vulnerabilities are a type of computer securityvulnerability typically found in Web applications. XSS vulnerabilities enable attackers to injectclient-side script into Web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy. Cross-site scripting carried out on websites accounted for roughly 84% of all security vulnerabilities documented by Symantec as of 2007.[1] Their effect may range from a petty nuisance to a significant security risk, depending on the sensitivity of the data handled by the vulnerable site and the nature of any security mitigation implemented by the site's owner.

3.2.3 DETECTING AND PREVENTING DICTIONARY ATTACK

It is possible to achieve a time-space tradeoff by pre-computing a list of hashes of dictionary words, and storing these in a database using the hash as the key. This requires a considerable amount of preparation time, but allows the actual attack to be executed faster. The storage requirements for the pre-computed tables were once a major cost, but are less of an issue today because of the low cost of disk storage. Pre-computed dictionary attacks are particularly effective when a large number of passwords are to be cracked. The pre-computed dictionary need only be generated once, and when it is completed, password hashes can be looked up almost instantly at any time to find the corresponding password.

3.2.4 DETECTING AND PREVENTING SOCIAL ATTACK

Some automated teller machines have a sophisticated display which discourages shoulder surfers from obtaining displayed information. It grows darker beyond a certain viewing angle, and the only way to tell what is displayed on the screen is to stand directly in front of it.

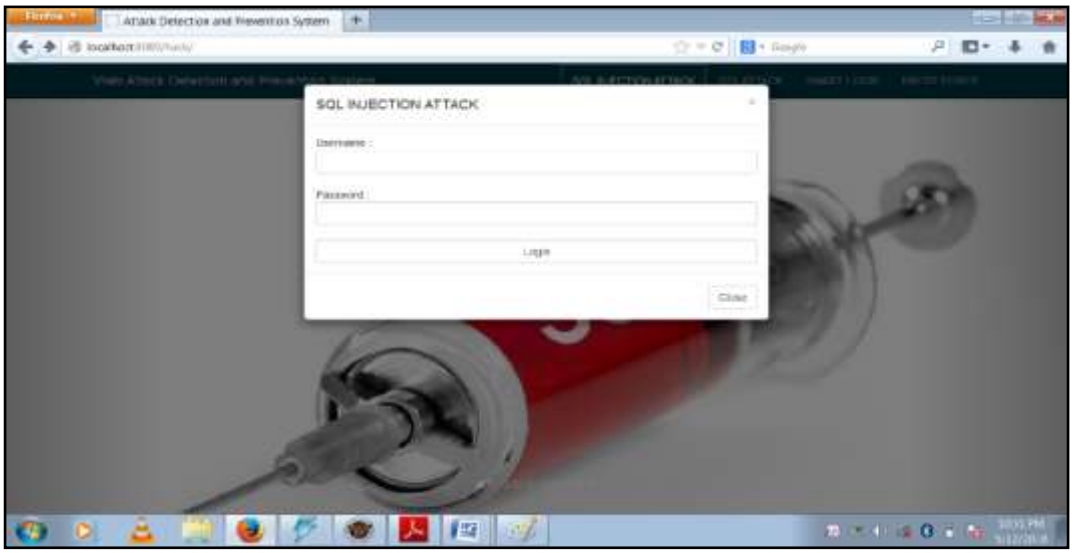
3.2.5 DETECTING AND PREVENTING BRUTE FORCE ATTACK

One of the measures of the strength of an encryption system is how long it would theoretically take an attacker to mount a successful brute-force attack against it. Brute-force attacks are an application ofbrute-force search, the general problem-solving technique of enumerating all candidates and checking each one

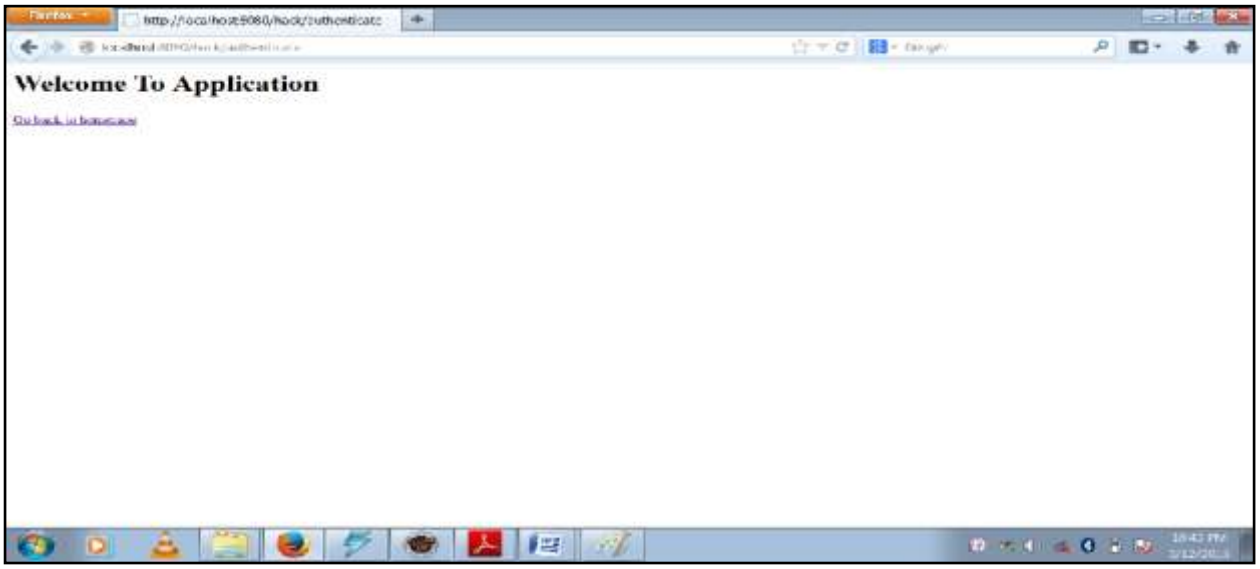
3.2.6 DETECTING AND PREVENTING SHOULDER SURFING ATTACK

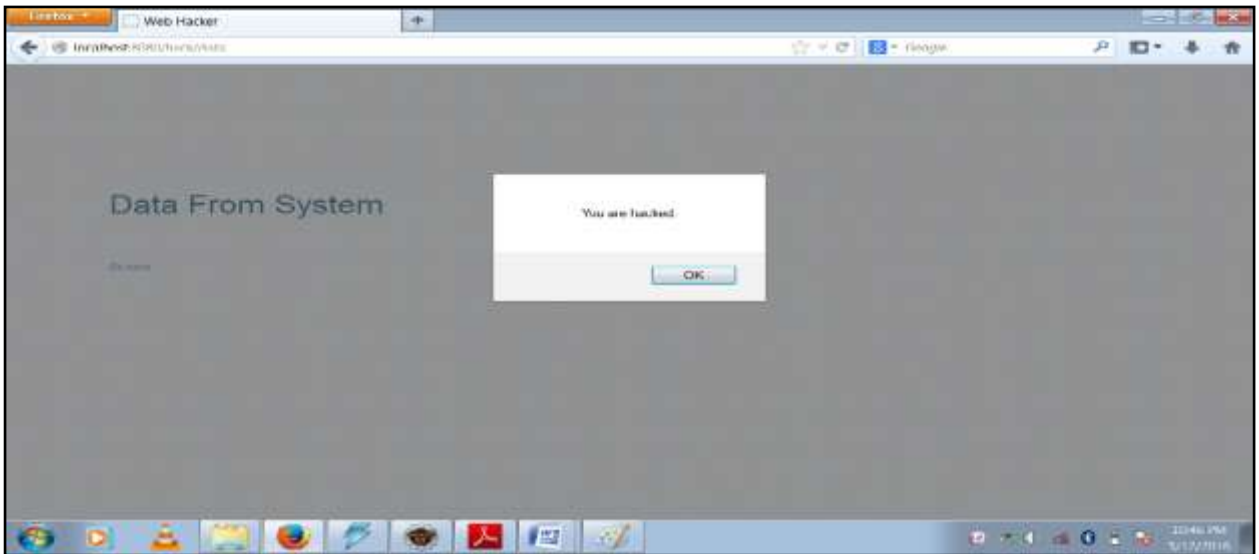
This technique can be used to fool a business into disclosing customer information as well as by private investigators to obtain telephone records, utility records, banking records and other information directly from company service representatives. The information can then be used to establish even greater legitimacy under tougher questioning with a manager, *e.g.*, to make account changes, get specific balances, etc.

4. SNAPSHOTS

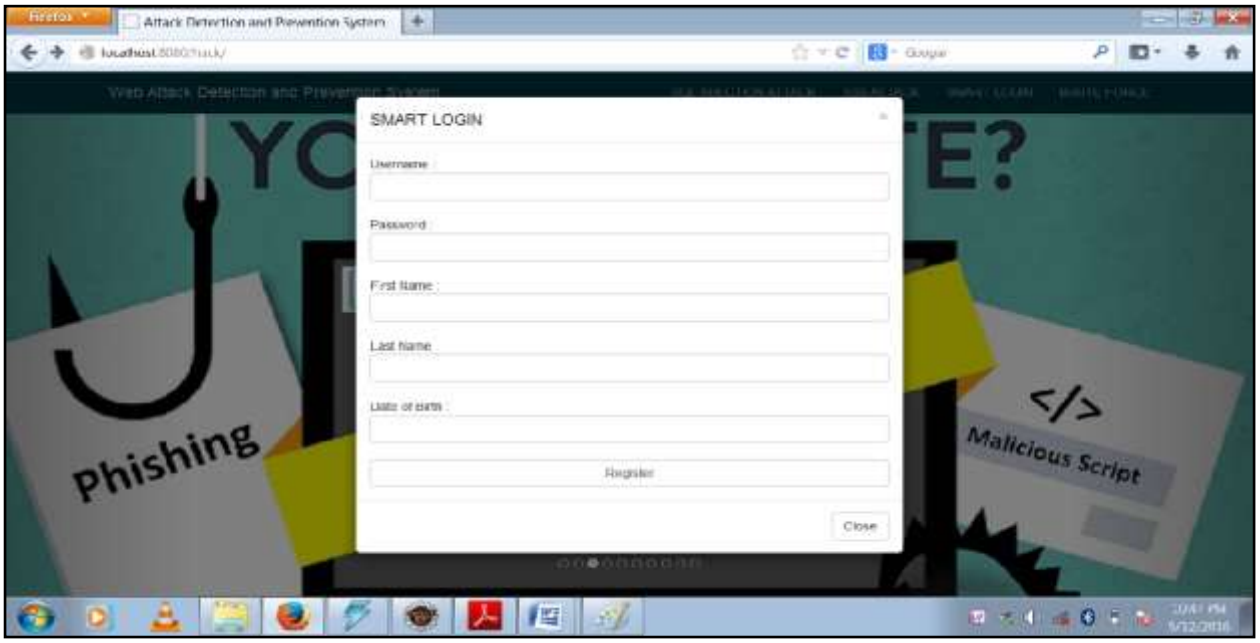


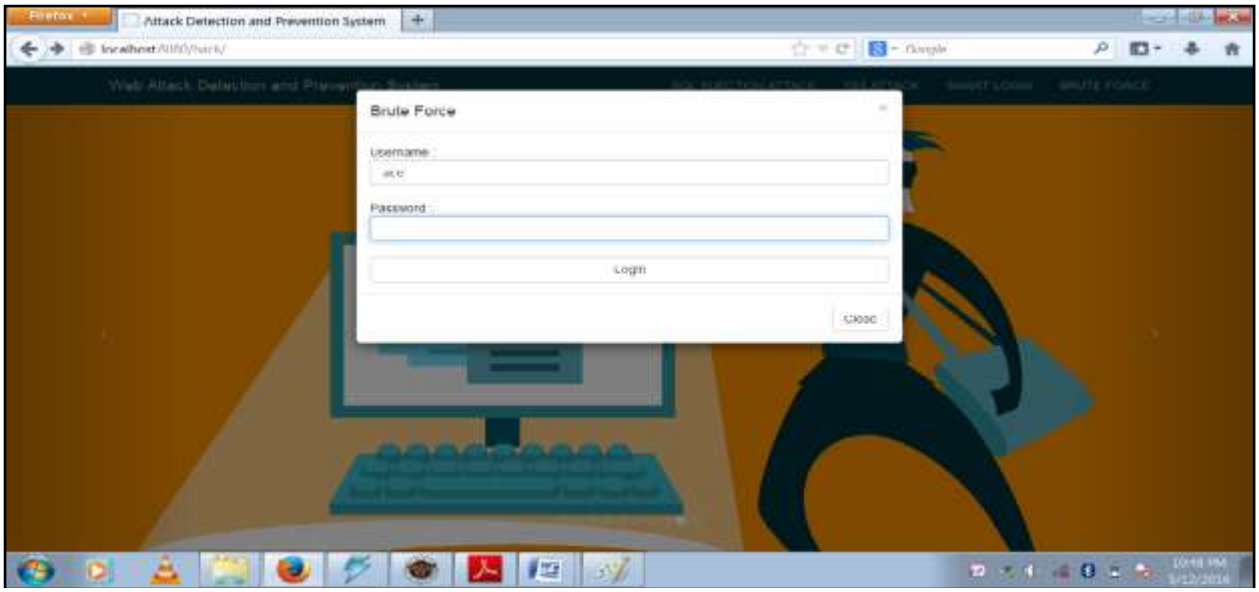
 ISSN: 2454-4248
06 - 11

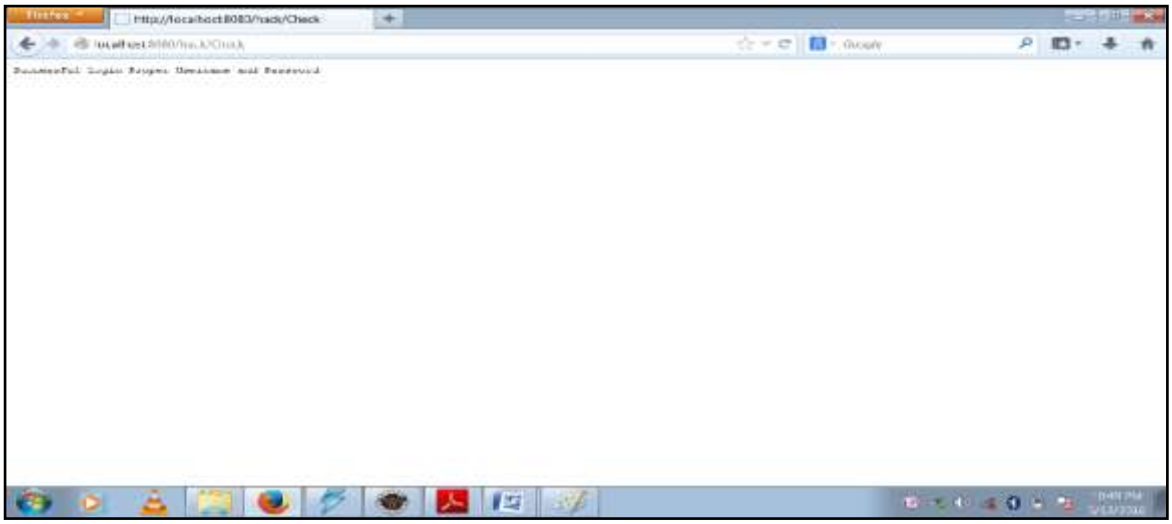




 ISSN: 2454-4248
06 - 11







5. CONCLUSION

The SQL-injection attacks are tremendously dangerous in association to other types of web based attacks for the reason that here the end result is data manipulation.SQL injection holes can be easily exploit by a technique called SQL injection attacks. This proposed integrated approach is an effort to add some more security measures to databases to avoid SQL injection attack.

REFFERENCES

[1] Jose Fonseca, Marco Vieira, and Henrique Madeira "Evaluation of Web Security MechanismsUsing Vulnerability & Attack Injection"-IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 11, NO. 5, SEPTEMBER/OCTOBER 2014.

[2] D. Avresky, J. Arlat, J.C. Laprie, and Y. Crouzet, "Fault Injection for Formal Testing of Fault Tolerance," IEEE Trans. Reliability, vol. 45, no. 3, pp. 443-455, Sept. 2011

[3] J. Arlat, A. Costes, Y. Crouzet, J.-C. Laprie, and D. Powell, "Fault Injection and Dependability Evaluation of Fault-Tolerant Systems," IEEE Trans. Computers, vol. 42, no. 8, pp. 913-923, Aug. 2011.

[4] N. Neves, J. Antunes, M. Correia, P. Ver_ıssimo, and R. Neves, "Using Attack Injection to Discover New Vulnerabilities," Proc. IEEE/IFIP Int'l Conf. Dependable Systems and Networks, 2006.

[5] N. Jovanovic, C. Kruegel, and E. Kirda, "Precise Alias Analysis for Static Detection of Web Application Vulnerabilities," Proc. IEEE Symp. Security Privacy, 2006.

[6] IBM Global Technology Services "IBM Internet Security Systems X-Force 2012 Trend & Risk Report," IBM Corp., Mar. 2013.

[7] The Privacy Rights Clearinghousewww.privacyrights.org/databreach, Accessed 1 May 2013, Apr. 2012.

[8] M. Fossi, et al., "Symantec Report on the Underground Economy, Symantec Security Response," 2008.

[9] D. Powell and R. Stroud, "Conceptual Model and Architecture of MAFTIA," Project MAFTIA, Deliverable D21, 2003.

[10] B. Livshits, "Stanford SecuriBench," suif.stanford.edu/_livshits/ securibench, Accessed 1 May 2013, 2005

[11] D. Powell and R. Stroud, "Conceptual Model and Architecture of MAFTIA," Project MAFTIA, Deliverable D21, 2003.

[12] V. Krsul, "Software Vulnerability Analysis," PhD thesis, Purdue univ.

[13] J. Fonseca and M. Vieira, "Mapping Software Faults with We Security Vulnerabilities," Proc. IEEE/IFIP Int'l. Conf. Dependable Systems and Networks, June 2008.

[14] B. Damele, "Sqlmap: Automatic SQLi Tool," sqlmap.sourceforge. net, Accessed 1 May 2013, 2009.