ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Volume 1, Issue 10, December 2013**

# Hunting Malicious Attacks in Social Networks

S.Thirunavukkarasu[1], Dr.K.P.Kaliyamurthie[2]

Assistant Professor, Dept of IT, Bharath University, Chennai, TamilNadu, India

Professor & Head, Dept of IT, Bharath University, Chennai, TamilNadu, India

**ABSTRACT:** The Rapid growth of internet resulted in feature rich and dynamic web applications. This increase in features also introduced completely under estimated attack vectors. Cross site scripting attacks, SQL Injection and malicious file execution are the most dominant classes of web vulnerabilities reported by OWASP 2011. These attacks make use of vulnerabilities in the code of web applications, resulting in serious consequences, such as theft of cookies, passwords and other personal credentials. It is caused by scripts, which do not sanitize user input. Several server-side counter measures for XSS attacks do exist, but such techniques have not been universally applied, because of their deployment overhead. The existing client-side solutions degrade the performance of client's system resulting in a poor web surfing experience. This paper presents automata-based symbolic string analyses called XHunter for automatic verification of string manipulating programs we compute the pre and post conditions of common string functions using deterministic finite automata (DFAs).Experiment result shows that this approach finds large number of malicious attacks in web application.

**KEYWORDS:** Social Networks, Scripting Attacks, XSS attacks, Malicious Attacks, Hacker, Crackers.

## I. INTRODUCTION

As the Growth of Internet increases rapidly the attack vectors also increases rapidly they can degrade the performance, by the process of Hacking. Hacking means finding out weaknesses in a computer or computer network and exploiting them, though the term can also refer to someone with an advanced understanding of computers and computer networks. Hackers may be motivated by a multitude of reasons, such as profit, protest, or challenge. The subculture that has evolved around hackers is often referred to as the computer underground but it is now an open community. While other uses of the word hacker exist that are not related to computer security, they are rarely used in mainstream context. They are subject to the long standing hacker definition controversy about the true meaning of the term hacker. In this controversy, the term hacker is reclaimed by computer programmers who argue that someone breaking into computers is better called a cracker, not making a difference between computer criminals (black hats) and computer security experts (white hats). Some white hat hackers claim that they also deserve the title hacker and that only black hats should be called crackers.

### Cyber Security

Web Applications have become one of the most important means of information communication between various kinds of users and service providers. CERT (Computer Emergency Response Team) published an advisory on newly identified security vulnerability affecting all web applications. There are three known variants of cross site scripting: Reflected a page reflects user supplied data directly back to the user. Stored takes malicious data, stores it in a file, a database, or other back end system, and then at a later stage, displays the data to the user, unfiltered. DOM injection - the site's JavaScript code and variables are manipulated rather than HTML elements. Attacks are usually implemented in JavaScript, which is a powerful scripting language.

Using JavaScript allows attackers to manipulate any feature of the rendered page, including adding new elements, manipulating any aspect of the internal DOM tree, and deleting or changing the page format. Initially it was possible that a browser window could steal data from another browser window when more than one browser windows were open simultaneously. To allow user-side customization of Web information, Cookies were implemented. They are pieces of information generated by a Web server and stored in the user's computer, ready for future access. Cookies are embedded in the HTML information flowing back and forth between the user's computer and the servers. Since the

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

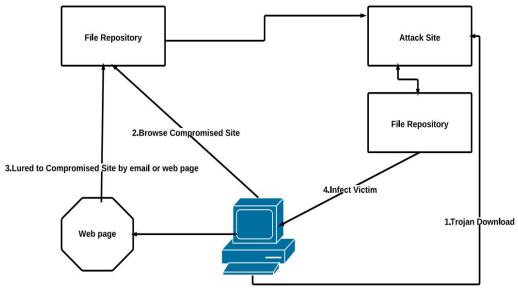
International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Volume 1, Issue 10, December 2013**

information in the cookies is easily accessible, the attackers popped them through cross site scripting, and used them to hijack sessions, and compromise accounts.

To allow user-side customization of Web information, Cookies were implemented. They are pieces of information generated by a Web server and stored in the user's computer, ready for future access. Cookies are embedded in the HTML information flowing back and forth between the user's computer and the servers. Since the information in the cookies is easily accessible, the attackers popped them through cross site scripting, used them to hijack sessions, and compromise accounts.




**Web site attack**

The malware has evolved to make increased use of the web. The scope extends further than just malicious scripts embedded in web pages, for example: numerous downloader Trojans use the web as a simple file repository, downloading other malicious files via HTTP. Malicious scripts hosted on attack sites await the visit of vulnerable client browsers before they unleash exploit code in order to infect the victim. Compromised sites provide a convenient mechanism to expose huge number of victims to malicious code. Spammed email messages and enticing web sites are used to lure victims to malicious code. Malware may deliver a traffic redirection payload. Online advertising is a multibillion dollar business nowadays.
Increasing web traffic to a site by directing or referring users provides a mechanism for organizations and individuals to make money through affiliate marketing. The class of applications that integrate with the browser in order to display targeted advertisements is generally referred to as adware. Such software is commonplace today, and is frequently bundled with other applications ("ad-supported software").The web provides the perfect framework for malware authors to blend together the techniques listed above. Today's threats cunningly incorporate spam and web "lures" with exploit scripts to efficiently infect unsuspecting victims. Figure 1.1 provides an overview of some of the key roles the web plays in current malware attacks.

## II. PROBLEM STATEMENT

They are many web vulnerabilities do exist in web application, but according to the Report of OWASP in the Year 2012.They have reported the top most web Vulnerabilities which can effectively causes application to degrade its Performance, they are Cross Site Scripting Attack, SQL Injection Attack and Malicious File Execution Attack. This Research is focused for eliminating those vulnerabilities. The Short Description of those attacks has been illustrated in upcoming sections.

**Cross Site Scripting**
Cross-site scripting (XSS) is a type of computer security vulnerability typically found in Web applications that enables attackers to inject client-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 80% of all security vulnerabilities documented by Symantec as of 2009.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.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Volume 1, Issue 10, December 2013**

There is no single, standardized classification of cross-site scripting flaws, but most experts distinguish between at least two primary flavours non-persistent and persistent XSS. Some sources further divide these two groups into traditional (caused by server-side code flaws) and DOM-based (in client-side code).Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within Win Amp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology.

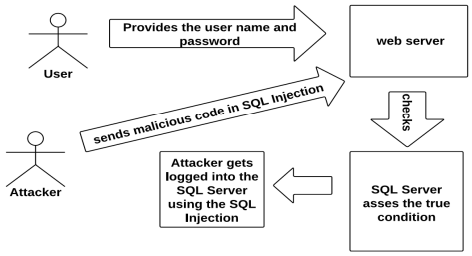
When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. XSS vulnerabilities have been reported and exploited since the 1990s. Prominent sites affected in the past include the social-networking sites Twitter, Facebook, MySpace, and Orkut. In recent years, cross-site scripting flaws surpassed buffer overflows to become the most common publicly reported security vulnerability, with some researchers viewing as many as 68% of websites as likely open to XSS attacks.

A Cross-site scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

**Sql Injection**

SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table or as metadata. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed. The injection process works by prematurely terminating a text string and appending a new command. Because the inserted command may have additional strings appended to it before it is executed, the malefactor terminates the injected string with a comment mark "--". Subsequent text is ignored at execution time.

Using SQL injections, attackers can: Add new data to the database-Could be embarrassing to find yourself selling politically incorrect items on an ecommerce site. Perform an INSERT in the injected SQL, Modify data currently in the database. Could be very costly to have an expensive item suddenly be deeply ‘discounted’. Perform an UPDATE in the injected SQL. Often can gain access to other user’s system capabilities by obtaining their password.




SQL Injections

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Volume 1, Issue 10, December 2013**

## III. LITERATURE SURVEY

**A.Makridakis and E.Athanasopoulos "Understanding the Behavior of Malicious Applications in Social Networks"**

The World Wide Web has evolved from a collection of static HTML pages to an assortment of Web 2.0 applications. Online social networking in particular is becoming more popular by the day since the establishment of SixDegrees in 1997. Millions of people use social networking web sites daily, such as Facebook, My-Space, Orkut, and LinkedIn. A side-effect of this growth is that possible exploits can turn OSNs into platforms for malicious and illegal activities, like DDoS attacks, privacy violations, disk compromise, and malware propagation. This article shows that social networking web sites have the ideal properties to become attack platforms. They introduced a new term, antisocial networks that refers to distributed systems based on social networking web sites which can be exploited to carry out network attacks. An adversary can take control of a visitor's session by remotely manipulating their browsers through legitimate web control functionality such as image-loading HTML tags, JavaScript instructions, and Java applets. This Paper shows that social networking websites have the ideal properties to become attack platforms.

They introduced the new term; antisocial networks that refer to distributed systems based social networking web sites which can be exploited to carry out network attacks. This paper is study about many attack vectors but this paper doesn't deal about detection/prevention mechanism. This paper studies about the behaviour of malicious applications that causes the remote user to hack legitimate users data.

**Gregory Blanc , Ruo Ando and Youki Kadobayashi "Term Rewriting Deobfuscation for static Client side scripting malware detection"**

Ensuring users with a safe web experience has become a critical problem recently as fraud and privacy infringement on the Internet are becoming current. Web-scripting-based malware is also intensively used to carry out longer-term exploitation such as XSS worms or botnets, and server-side countermeasures are often ineffective against such threats while client-side ones seldom deal with the problem of obfuscation. In order to provide a sounder and more complete analysis, this paper proposes a carry out deobfuscation of web-scripting-language-based malware. This paper, studies the possibility of automating the deobfuscation process using a term rewriting system based on automated deduction. Such static approach intends to evade anti-analysis techniques and unknown obfuscation schemes. With some preliminary experiments in JavaScript, This paper shows evidence that this is actually possible and highlights several challenges needed to tackle in order to implement an effective script-based malware deobfuscator. This approach can be generalized to web scripting languages other than JavaScript such as Action Script or VBScript. Applications encompass script-based malware static analysis or malware distribution website crawling. This paper is included in a wider project that aims to provide a client-based defence against Web 2.0 malware.

Term Rewriting System-The Input from the user is first checked for sinks if sinks exist then the sink data is rewritten. For example<script> alert("hacked");</script> is changed to < script > alert("hacked"); < script >. Two standard Tools are used, Encryption Technique is a Symmetric Encryption standard which convert every plaintext message into ciper text message using mono alphabetic cipher Term Rewriting System takes huge amount of time to rewrite the sinks (malicious data).Mono alphabetic cipher uses fixed substitution over the entire message .To prevent the vulnerability in the network a new tool called Automaton can be used to parse the Malicious sinks, Which is more efficient compared to the above Techniques.

**Automata-based String Analysis**

We use automata-based string analysis techniques that we mentioned above for vulnerability analysis and vulnerability signature generation. Our analysis takes an attack pattern specified as a regular expression and a JSP program as input and 1) identifies if there is any vulnerability based on the given attack pattern, 2) generates a DFA characterizing the set of all user inputs that may exploit the vulnerability. As we have stated earlier, our string analysis framework uses a DFA to represent values that string expressions can take. At each program point, each string variable is associated with a DFA. To determine if a program has any vulnerability, we use a forward reach ability analysis that computes an over-approximation of all possible values that string variables can take at each program point.

Intersecting the results of the forward analysis with the attack pattern gives us the potential attack strings if the program is vulnerable. The backward analysis computes an over-approximation of all possible inputs that can generate those attack strings. The result is a DFA for each user input that corresponds to the vulnerability signature. We will discuss

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Volume 1, Issue 10, December 2013**

how to use vulnerability signatures to generate effective sanitization routines in the next section. Here we focus on how to conduct forward and backward symbolic reach ability analyses with summarization techniques.

**Forward Analysis – Detect vulnerabilities**

Vulnerability analysis is conducted using the dependency graph. The approach identifies the possible values of each node. Each node in the dependency graph is associated with a DFA. DFA accepts an over-approximation of the strings values that the string expression represented by that node can take at run time. Intersecting the DFA for the sink nodes with the DFA for the attack pattern identifies the vulnerabilities.

## IV. CONCLUSION

Large amount of websites are vulnerable to XSS attacks. The proposed solution is found to be very effective by the experimental results. The solution is platform independent and has been implemented on a platform independent browser, so it can be used with other operating systems with a few changes. Cross site scripting vulnerability exists on all the platforms, so it is a big advantage over other solutions. We use automata based string analysis techniques (XHunter) that we mentioned above for vulnerability analysis and vulnerability signature generation. Our analysis takes an attack pattern specified as a regular expression and a JSP program as input and 1) identifies if there is any vulnerability based on the given attack pattern, 2) generates a DFA characterizing the set of all user inputs that may exploit the vulnerability. The solution can be further extended to cover other pernicious vulnerabilities and attacks. It can be implemented as a Common solution to be used in all the web browsers.

Many of the techniques have problems handling attacks that take advantage of poorly-coded stored procedures and cannot handle attacks that disguise themselves using alternate encodings. We present a new Tool called XHunter that can be used to check the correctness of string manipulation operations in web applications XHunter implements automata based approach for automatic verification of string manipulating programs based on symbolic string analysis**.** This Research can be extended by placing XHunter as a Midware that is in between client and server.

## REFERENCES

[1] D.Arulsuju and R.Purushothaman "Hunting Malicious Attacks in Social Networks" at IEEE-ICoAC 2011 ISBN Number 978-1-4673-0671-3/11/$26.00©2011 IEEE.
[2] S.Christey. Vulnerability type distributions in CVE, Oct. 2006. http: //cwe.mitre.org/documents/vuln-trends.html.
[3] R. Cytron, J. Ferrante, B. K. Rosen, M. N. Wegman, and F. K. Zadeck. Efficiently computing static single assignment form and the control dependence graph. Transactions on Programming Languages and Systems, 13(4):451–490, Oct 1991.
[4] J. Foster, M. Fähndrich, and A. Aiken. A theory of type qualifiers. In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), pages 192–203, Atlanta, Georgia, May 1–4, 1999.
[5] J. S. Foster, T. Terauchi, and A. Aiken. Flow-sensitive type qualifiers. In PLDI '02: Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation, pages 1–12, 2002. ACM Press.
[6] C. Gould, Z. Su, and P. Devanbu. Static checking of dynamically generated queries in database applications. In Proceedings of the 25th International Conference on Software Engineering (ICSE), pages 645–654, May 2004.
[7] O. Hallaraker and G. Vigna. Detecting malicious JavaScript code in Mozilla. In ICECCS '05: Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems, pages 85–94, 2005.
[8] K. J. Higgins. Cross-site scripting: Attackers' new favorite flaw, September 2006. http://www.darkreading.com/document.asp?doc_id=103774&WT.svl=news1_1 .
[9] J. E. Hopcroft, R. Motwani, and J. D. Ullman. Introduction to Automata Theory, Languages and Computability. Addison-Wesley, Boston, MA, 2000.
[10] H. Hosoya and B. C. Pierce. Xduce: A typed xml processing language (preliminary report). In Selected papers from the Third International Workshop WebDB 2000 on The World Wide Web and Databases, pages 226–244, London, UK, 2001. Springer-Verlag.
[11] Y.-W. Huang, F. Yu, C. Hang, C.-H. Tsai, D.-T. Lee, and S.-Y. Kuo. Securing web application code by static analysis and runtime protection. In WWW '04: Proceedings of the 13th international conference on World Wide Web, pages 40–52, New York, NY, USA, 2004. ACM Press.
[12] T. Jim, N. Swamy, and M. Hicks. Defeating scripting attacks with browser-enforced embedded policies. In WWW '07: Proceedings of the 16th international conference on World Wide Web, pages 601–610, 2007. ACM.
[13] 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.
[14] N. Jovanovic, C. Kruegel, and E. Kirda. Precise alias analysis for syntactic detection of web application vulnerabilities. In ACM SIGPLAN Workshop on Programming Languages and Analysis for Security, Ottowa, Canada, June 2006.
[15] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic. Noxes: A client-side solution for mitigating cross site scripting attacks. In SAC '06: Proceedings of the 2006 ACM symposium on Applied computing, pages 330–337, 2006. ACM