A Taxonomy Study of XSS Vulnerabilities

Asian Journal of Information Technology 16 (2-5): 169-177, 2017
ISSN: 1682-3915
© Medwell Journals, 2017

A Taxonomy Study of XSS Vulnerabilities

Nayeem Khan, Johari Abdullah and Adnan Shahid Khan
Faculty of Computer Science and Information Technology,
Universitiy Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia

Abstract: As the demand of using internet and our dependence on web application to perform our daily activities is increasing every day, protecting web application from getting attacked by cyber criminal's becomes imperative. The most common type of attack on web applications is XSS. XSS is considered as a major growing web security threat. XSS attacks are the scripts that are embedded in a web page and are executed at the victim's machine. The objective of this study is to perform a literature review on the studies conducted on prevention and detection of XSS vulnerability. Results suggest that research in this field is going on very actively but no study provided full solution to this problem. More focus should be given to hybrid techniques and techniques using probabilistic model for detection of XSS vulnerability.

Key words: Taxanomy study, cross site scripting, web security, web application vulnurability, threat, Malaysia

INTRODUCTION

Cross Site Scripting (XSS) has been around since 199's as one of the major security vulnerability. OWASP (Open Web Application Security Project) has ranked XSS on 2nd among top ten web security vulnerabilities affecting users through web applications. As reported by Web hacking incident database for 2013, XSS vulnerabilities accounted 43% of all reported vulnerabilities. Statistics clearly demand the need to prevent and detect such types of security vulnerability. XSS is a type of injection attack that allows an attacker to send malicious scripts to end-users through a web browser in order to achieve the purpose of attack by stealing victim's confidential data such as password, cookie, sessions, hijacking browser, redirecting users to malicious sites and take unauthorized actions without user's knowledge. The attacker can be an internal or external user or even an administrator that has the ability to send malicious code to the interpreter in the web browser or to the target server. OWSAP has categorized XSS vulnerabilities into three types reflected, stored and DOM based. Stored XSS occurs when malicious Java script is stored on the target server in database, guest book's message forum's, etc., the malicious scripts gets executed when the user visits the malicious site thereby passing the privileges of the user to the attacker which then takes illegitimate actions without user permission. In reflected XSS the attacker injects the malicious code into the server. The injected code is reflected back to attacker in the form of error message or search result which may include some or part of inputs provided to the server as a request. Then reflected XSS attacks are sent to target victim through email or links embedded on the web pages to steal the confidential or takes control over the victim's computer. Document Object Model (DOM) is another kind of XSS vulnerability which occurs due to the inappropriate handling of the objects in a web page. In DOM based XSS entire tainted data flow from source to sink takes place in the web browser. The source of XSS can be any HTML element or the web page's URL while any method with sensitive call which can cause malicious code to be executed is the sink (Klein, 2005). The purpose of the taxonomy study is to show the current scenario of the research on XSS and to highlight the area where more research needs to done.

MATERIALS AND METHODS

The research questions which were addresses in this study (Fig. 1) are as:

- Q1: How much research has been done towards detection and prevention of XSS since its existence?
- Q2: What are the existing approaches used to solve this problem?
- Q3: What are the weaknesses and strengths of existing approaches?
- Q4: Which area should be focused for better results?

Literature to conduct this study was collected from following online databases:

Corresponding Author: Nayeem Khan, Faculty of Computer Science and Information Technology, Universitiy Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia

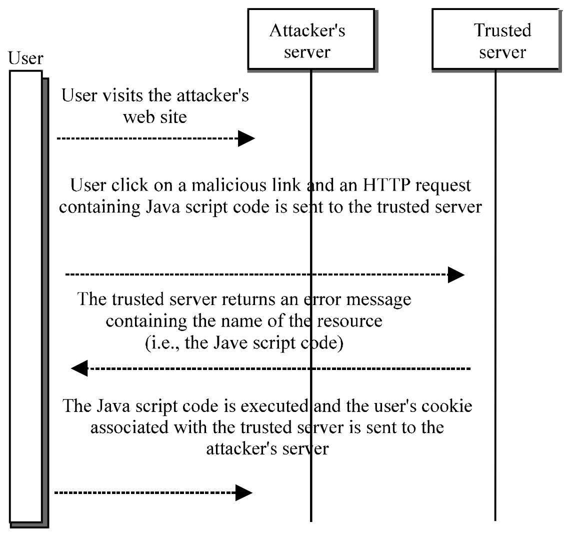
User
Attacker's server
Trusted server
User visits the attacker's web site
User click on a malicious link and an HTTP request containing Java script code is sent to the trusted server
The trusted server returns an error message containing the name of the resource (i.e., the Jave script code)
The Java script code is executed and the user's cookie associated with the trusted server is sent to the attacker's server

Fig. 1: A typical cross-site scripting scenario (Kirda *et al*., 2006)

- ACM digital library
- Springer link
- IEEE explore
- Science direct
- Micro Soft academic
- Google scholar

Following key words were used as searching the articles related to our study:

- XSS
- Cross-site scripting
- Cross-site scripting vulnerability
- Cross-site scripting attack
- Cross-site scripting detection
- Cross-site scripting prevention
- Web application vulnerabilities

Classification of XSS attacks on location: Cross site scripting vulnerability and attacks can either occur at client side, server side or at both places. Figure 2 shows taxonomy of XSS based on location. Location of XSS Vulnerability and attacks is defined by the place where the detection technique resides. Client side detection methods looks for XSS vulnerabilities and attacks in the client's browser environment and server side implements the methods to detect XSS vulnerabilities and attacks on the server side or on proxy server. Whereas the hybrid tools research in combination with both client side and server side and often communicate and coordinate with each other.

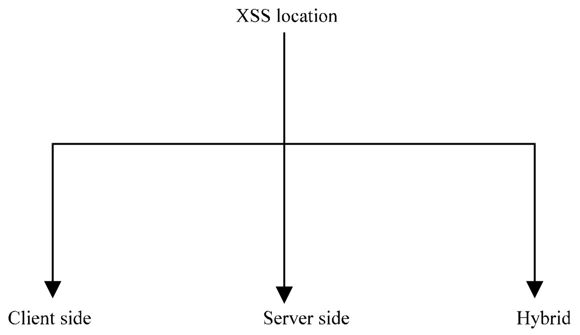

Fig. 2: Taxonomy of XSS based on location

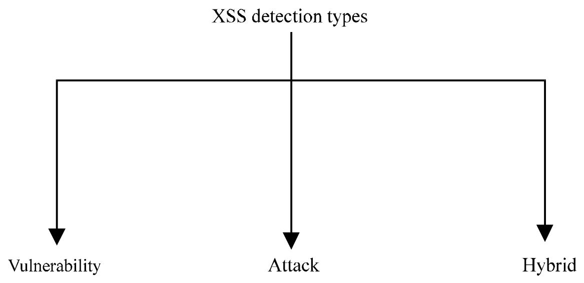

Fig. 3: Taxonomy of XSS detection types

Classification of XSS on detection type: Cross site scripting can be classified into vulnerability detection, attack detection or hybrid. Figure 3 shows the taxonomy of XSS detection types. The primary goal of XSS Vulnerability detection is to detect the vulnerability in the source code and inform the developer about the vulnerability in order to remove it. A vulnerability which is known without the knowledge of the vendor and being exploited by attackers before a vendor fix it is called zero day vulnerability. XSS vulnerability can be code segment or query. XSS vulnerability detection methods determine the presence of susceptibilities in a code segment or query.

The goal of XSS attack detection is to prevent the attack by blocking, preventing attack by modifying the attack containing code. XSS attacks are caused due to the malicious scripts which gives unauthorised access to the resources. XSS attacks must be detected at runtime.

Review of XSS detection techeniques: Several techniques employed by different researchers for detection of XSS are discussed as (Fig. 4).

Program analysis: Static analysis involves reviewing, testing and examining the source code or byte code of an application without executing the application to find the faults. Static analysis allows analyzing the data flow,

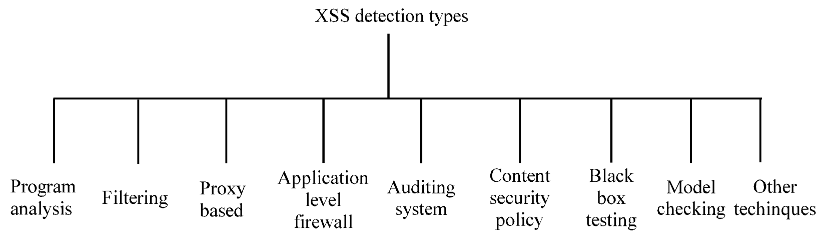

Fig. 4: Various XSS detection techniques

checking the syntax and verifying if the states of application are finite (Perez *et al*., 2011). The process of analyzing the properties and derived properties of a running program that hold more than one execution of running program through program instrumentation is called dynamic analysis (Ball, 1999). Dynamic taint analysis and forward symbolic execution are most commonly used techniques in detection of XSS. Predefined taint sources which are affected in processing are run and observed by dynamic taint analysis (Schwartz *et al*., 2010). Program's which run under symbolic execution with symbolic inputs rather than inputs of concrete nature which maintains a path on every execution of a branch instruction with encoded constraints as input that reach program point (Schwartz *et al*., 2010).

Filtering: Filtering is a decision making procedure with predefined set of rules that either accepts or rejects the input. The mechanism for filtering is based on the string comparison of strings between incoming and outgoing request and responses for detection of XSS attack payload. The primary goal of filtering mechanism is to find the perilous tags such as <img>, <script>, etc., inside the response received from server in HTTP form as PUSH and POST methods in order to prevent execution of injected Java script code. WebKit browser engine uses a filter for XSS known as XSS Auditor. XSS Auditor is also used by browsers such as Google Chrome and apple safari. Microsoft's internet explorer version-8 onwards has inbuilt XSS filter. The advantage of filtering is that it widely used schemes as a first level of defense against XSS attacks which filters the web contents containing any suspicious scripts. Sometimes Filtering becomes feeble in preventing various instances of XSS attacks especially when input provided by the user is content-rich HTML (Bisht and Venkatakrishnan, 2008).

The widely used and easy to implement is black list filtering. Problem with black list filtering is that it provides defense against know threats only. Many approaches for evading blacklist have been proposed. Mitre and NVD has pointed towards the vulnerabilities caused due to the poor implementation of black list filtering. Many studies conclude that White list filtering which has been proven as an efficient mechanism for protection against viruses and malwares. Whitelist filtering is more difficult to implement than blacklist filtering and challenging to maintain a suitable and acceptable list of inputs that a system can accept.

Proxy-based: Proxy based techniques are responsible for analyzing all the HTTP and HTTPS data between client and server by performing scanning to check any special HTML character or any encoding sequence before the code gets executed in web browser with the intent to attack. The advantage of implementing proxy based approach is that it is attractive in various situations because of its greater flexibility.

Application-level firewall: To avoid leakage of sensitive information and deny bad requests by using connection rule, application-level firewalls are used which perform the analysis on source code of web page to check for URL's that might lead to the attack. WAFs are designed to protect web applications/servers from web-based attacks that intrusion prevention systems cannot prevent. Application-level firewalls can be network or hosts based but are responsible to monitor traffic to and from web applications to servers and vice versa. The original difference is in the level of ability to anatomize the layer 7 web application logic.

Auditing system: Auditing monitor execution of Java script code and compare the operations against high-level policies to detect malicious behavior. Mozilla web browser has an inbuilt auditing system. Auditing system monitors execution of Java script code (Hallaraker and Vugna, 2005). With the help on distinct intrusion detection techniques to detect malicious behaviors the observed operations in high level policies are compared (Jim *et al*., 2007).

Black box testing: The black box approach for detection of XSS is only limited in monitoring the web request and responses without concerning the information stored in the database (Li *et al*., 2012). There is no role of source code. Black box testing approach for DOM-based XSS is not usually performed since access to the source code is always handy and there is no need to assign it to client side for executed.

Content security policy: Content Security Policy (CSP) helps to prevent XSS attacks. CSP are the recommendations of the W3C working group on Web application security. CSP is activated by a client's browser when the X-content-security-policy HTTP header is provided in a HTTP response. The contents of the policy that will be enforced by browser may either be contained in a header or a text file with same origin (Stamm *et al*., 2010).

Latest trends on prevention and detection of XSS: In the current study we have identified 38 studies which have used various techniques for detection and prevention of XSS, Table 1 depicts that research in this field is growing every year and is going on very actively. Many techniques for prevention or detection of XSS have being proposed depending on the location for mitigating XSS. The locations are either client side or server side or on both sides (Hybrid). Figure 1 shows that 50% studies have been carried out at server side, 34% on client side and 10% on hybrid location. While the number of technique's have been proposed, about 57% studied techniques focus on detection of XSS while 37% have worked on prevention of XSS and 10% studies on detection and prevention (Hybrid) of XSS as shown in Fig. 2 researcher have researched either on the prevention or detection of XSS attacks and XSS vulnerability. Figure 3 shows that 71% studied have been performed on mitigating of XSS attack, 23% on XSS vulnerability and 10% on both (Fig. 5-7).

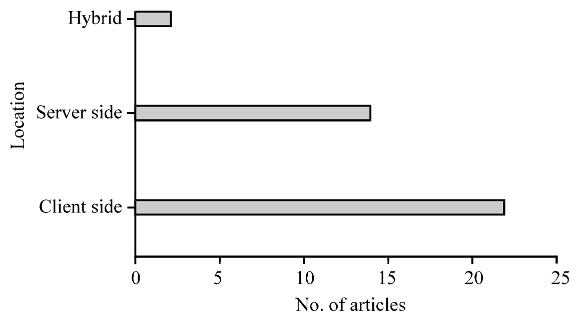

Fig. 5: Distribution of articles based on location

Table 1: Summary of literature review

Titles	Researcher/Years	Detection types	Location	Detection/ prevention	Techniques	Results
An approach for cross site scripting detection and removal based on genetic algorithms	Hydara *et al.* (2014)	V	H	D	Genetic Algorithm	Genetic algorithm-based approach for XSS detection and removal. Algorithm still under study
An efficient detecting mechanism for cross-site script attacks in the cloud	Kan *et al.* (2014)	A		D	Proposes framework	Results show higher accuracy rate and lower impact on performance of applications
Detecting cross site scripting vulnerabilities introduced by HTML5	Dong *et al.* (2014)	V	S	D	Systematic analysis, black-box testing, filtering	Result shows that tool can efficiently detect XSS vulnerability introduced by HTML5
Preventing client side XSS with rewrite based dynamic information flow	Xiao *et al.* (2014)	A	C	P	Dynamic information flow	Solution can accurately mark and trace the information flow of sensitive data and detect the irregular accesses of sensitive data from untrusted domains
Preventing cross-site scripting with script-free HTML	Seffernick *et al.* (2014)	A	S	H	Context-free grammar	Server-side tool is implemented to detect the presence of a potential XSS attack
XSS peeker: a systematic analysis of cross-site scripting vulnerability scanners	Bazzolia *et al.* (2014)	V	S	D	Black-box scanners	Discuss the results of a detailed and systematic study on 6 black-box web scanners both proprietary and open source, Results show variance in payloads
A tale of the weaknesses of current client-side XSS Filtering	Sebastian *et al.* (2013)	A	C	S	Filtering	Successfully bypasses the XSS filter on first try in over 80% 80% of all vulnerable web applications
Analysis and prevention for cross-site scripting attack based on encoding	Lan *et al.* (2013)	A	S	P	Encoding	Upper information leaked is 92.29% and rate of the false positive is about 0.319%

Table 1: Continue

Titles	Researcher/Years	Detection types	Location	Detection/ prevention	Techniques	Results
BOSF: By-Owner Script Filtering	Kim *et al.* (2013)	A	S	P	Filtering	Effective solution with minimal performance overheads
Detecting cross-site scripting vulnerability using concolic testing	Ruse and Basu (2013)	H	H	H	Static analysis runtime monitoring	Efficient method, can easily identify conditional copy vulnerabilities
Detection of cross-site scripting attack under multiple scenarios	Das *et al.* (2015)	A	H	D	Validation	Satisfactory results obtained on 4 attack scenarios
Path sensitive static analysis of web applications for remote code execution ulnerability detection	Zheng and Zhang (2013)	V	C	D	Static analysis	Results shows technique is very effective, producing fewer false positives compared to alternative techniques
SWAP: mitigating XSS Attacks using a reverse proxy	Wurzinge *et al.* (2013)	A	S	H	Proxy interceptor	Experiments shows the effect of SWAP to successfully detect XSS
Cross site scripting attacks detection algorithmbased on the appearance position of characters	Matsuda *et al.* (2012)	A	S	D	Appearance/frequency of Characters	About 99.5% successful on attack test samples and 97.5% successful on normal test sample
DESERVE: a framework for detecting program security vulnerability exploitations	Mohosina and Zulkernine (2012)	V		D	Buffer overflow	The evaluation results indicate that the approach can effectively detect attacks with reasonable overhead in terms of execution time
Enemy of the state:a state-aware black-box web vulnerability scanner	Adoupe *et al.* 2012	V	S	D	Black-Box	Evaluation shows black-box scanner is effective and can also find vulnerabilities which other scanner could not find
Integrated approach to prevent SQL injection attack and reflectedcross site scripting attack	Sharma *et al.* (2012)	A	S	H	Security query, Sanitizer	The results show that approach is simple and effective to prevent XSS and SQL injection
PathCutter: Severing the self-propagation path of XSS JavaScript worms in social web networks	Cao *et al.* (2012)	A	H	P	Content blocking	Evaluation shows that rendering overhead is <4% and memory overhead for one additional view is less than 1%
An advanced web attack detection and prevention tool	Kapodistria *et al.* (2011)	A	S	H	Penetration Testing	Proposed new tool to prevents attacks and compare proposed tool with dot defender
BIXSAN: browser Independent XSS Sanitizer for prevention of XSS attacks	Chandra and Selvakumar (2011)	A	C	P	Filtering	Results show approach reduces the anomalous behaviour of browser with 100% sanitization rate
Automatic XSS detection and Snort signatures/ACLs generation by the means of a cloud-basedhoneypot system	Jacob (2011)	A	S	D	Honeypot	Snorts Signatures which is able to detect 64% of the same set of XSS attacks
L-WMxD: Lexical based Webmail XSS discoverer	Tang *et al.* (2011)	V	S	D	XSS fuzzer	Successful in finding vulnerabilities in 21 Webmail services out of 26 rea-world Webmail applications
S2XS2: A server side approach to automatically detect XSS attacks	Shahriar and Zulkernine (2011)	A	S	D	Automated framework	Results show that false positive rates vary between zero and 5.2% with negligible runtime overhead
An execution-flow based method for detecting cross-site scripting attacks	Zhang *et al.* (2010)	A	C	D	Execution-flow analysis	Result shows that it it protects against a variety of XSS attacks and has an acceptable performance overhead
Auditing the defense against cross site scripting in web applications	Shar and Tan (2010)	A	S	D	Code auditing input validation and filtering	Evaluation shows feasibility and effectiveness of the proposed approach Client-side
cross-site scripting protection	Kirda *et al.* (2009)	A	C	P	Web proxy, rule generation	Effective in protecting information leakage

Table 1: Continue

Titles	Researcher/Years	Detection types	Location	Detection/ prevention	Techniques	Results
Code-injection attacks in browsers supporting policies	Elias	A	C	P	Browser enforced policies	New policy scheme for web browsers
Document structure integrity: a robust basis for cross-site scripting defense	Nadijia *et al.* (2009)	A	C	P	Document structure Integrity	Results shows this approch thwarts over 98% of the attacks with no false postives
XSSDS: server-side detection of cross-site scripting attacks	Johns *et al.* (2008)	A	S	D	Proto typical implementation	Approach can reliably detect XSS with a tolerable false positive rate
XSS-GUARD: precise dynamic prevention ofcross-site scripting attacks	Bisht and Venkatakrishnan (2008)	A	S	P	Framework	Discuss reasons for the failure of filtering mechanisms in gaurding XSS attacks
Cross-site scripting prevention with dynamic data tainting and static analysis	Philipp	A	C	P	Dynamic data tainting and static analysis	Evaluation shows that this approach has small number of false positives and is practically feasible
Defeating script injection attacks with browser enforced embedded policies	Jim *et al.* (2007)	A	C	P	Browser-enforced embedded policies	This approach requires fewer changes in web applications and browsers Results show that performance overhead is low and deployment is practical
Prevention of cross-site scripting attackson current web applications	Garcia-Alfaro and Navarro (2007)	A	S	P	Survey on prevention solutions	Surveys current approaches for prevention of XSS. The advantages and limitations of each proposal are also discussed
Pixy: a static analysis tool for detecting web application vulnerabilities (technical report)	Nenad *et al.* (2006)	V	C	D	Data flow using N-analysis	Proposed technique is able to detect taint-style vulnerabilities automatically; False positive rate is around 50%
A multi-model approach to the detection of web-based attacks	Kruegel *et al.* (2008)	A	S	D	anomaly detection	This approach uses different anomaly detection techniques to detect attacks against web servers and web-based applications
Detecting malicious javascript code in mozilla	Hallaraker and Vigna (2005)	A	C	D	High-level policies	Evaluation show substantial overhead with respect to the execution time
A proposal and implementation of automatic detection/ collection systemfor cross-site scripting vulnerability	Ismail *et al.* (2004)	V	C	D	Request/response change mode	The proposed approach and techniques described in this study is useful in identifying web application security problems

V = Vulnerability, S = Server, P = Prevention, A = Attack, C = Client, D = Detection, H = Hybrid

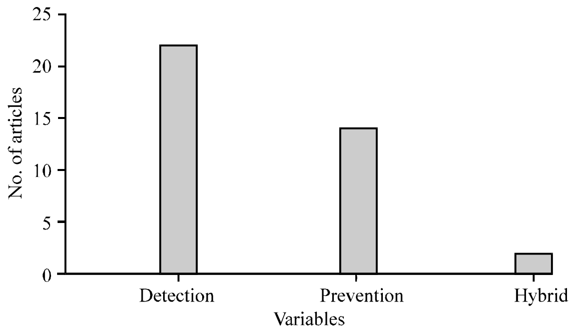

Fig. 6: Distribution of articles based on type

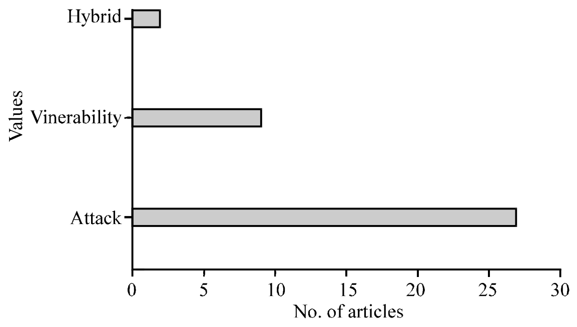

Fig. 7: Distribution of articles based on threat

RESULTS AND DISCUSSION

Despite number of techniques for mitigating XSS have been proposed either at client side or server side, it still remains a threat to users. Thus an efficient approach to mitigate XSS is demanded. Kidra *at al.* (Engin, 2006) proposes Noxes which they claim the first client side web proxy for mitigating XSS that relays all the HTTP requests from browser and serves as an application level firewall. Noxes supports an XSS mitigation mode that significantly reduces the number of connection alert prompts while at the same time providing protection against XSS attacks where the attackers may target sensitive information such as cookies and session IDs (Engin, 2009). The main limitation of Noxes it demands user customized configuration and user interaction during any suspicious even. Voget (Philipp, 2007) proposed a technique which is a combination of static and dynamic analysis for mitigating XSS which aim to identify the information leakage using tainting of input data in the browser. The problem with this technique is that it does not mitigate the damage caused by other types of XSS attacks such as port scanning, web page defacement and browser resource consumption. Hallaraker and Vigna (2005) proposes an auditing system for detection of malicious Java script. The study proposes how JS-engine of browser can be modifies to allow behavior based analysis of java scripts by using intrusion detection mechanism to prevent XSS attack. Bates *et al.* (2010) proposes XSS auditor for achieving high performance and reliability by bringing the interface between browser HTML parser and JS engine. The study shows that regular expression-based filtering systems have severe issues and proposed a superior approach in the form of the XSS auditor which has been adopted by the Web kit browser family (Chrome, Safari, Yandex) (Ben, 2014). Some server side XSS mitigation techniques have also been proposed like XSS- GUARD by Bish and Venkatakrishnan (2008) with the main focus on differencing between benign and malicious code and transforms the server programs such that they produce a shadow web page for each real response page. The reason behind to produce shadow pages is to generate desired set of unauthorized script sequence corresponding to the HTTP response. The limitation of XSS-Guard is that its implementation relies on browser's Java script detection component. Another reason behind the failure in identifying scripts of malicious nature with the purpose to attack web browsers is based on quirks. Jim *et al.* (2007) proposed Browser-Enforced Embedded Policies (BEEP) in which browser is modified such that it is capable of detecting malicious scripts based on security policies sent from server programs and preventing those scripts from being executed. A few studies have been carried on both client and server sides. Keep in view the limitations of client and server side techniques for mitigating XSS. More study need to be done on development of hybrid technique. Hybrid detection type may contain objectives of both vulnerability detection and attack detection. Hybrid detection type is a two-fold scheme containing vulnerability detection phase with the aim to detect exploit and attack detection phase with aim to prevent attack. In addition to hybrid detection approach machine learning classifiers can be used to achieve high detection rate with low false positives and performance overheads.

CONCLUSION

We have conducted a study related to research on XSS. We identified Many studies have been carried out in order to mitigate XSS have been identified. Despite all the efforts form researchers eliminating XSS is still prevalent. Most of the studied techniques either focus on providing client or server side solution. Less attention has been paid into the development of techniques at hybrid. Hybrid detection techniques may contain objectives of both vulnerability detection and attack detection. Probabilistic model can also be used for detection of XSS with high accuracy and low performance overheads.

REFERENCES

Ball, T., 1999. The Concept of Dynamic Analysis. In: Software Engineering-ESEC/FSE 99, Nierstrasz, O. and M. Lemoine (Eds.). Springer, Berlin, Germany, pp: 216-234.

Bates, D., A. Barth and C. Jackson, 2010. Regular expressions considered harmful in client-side XSS filters. Proceedings of the 19th International Conference on World Wide Web, April 26-30, 2010, ACM, Raleigh, North Carolina, ISBN:978-1-60558-799-8, pp: 91-100.

Bazzoli, E., C. Criscione, F. Maggi and S. Zanero, 2014. XSS Peeker: A systematic analysis of cross-site scripting vulnerability scanners. Master Thesis, Cornell University Library, Ithaca, New York, USA.

Bisht, P. and V.N. Venkatakrishnan, 2008. XSS-GUARD: Precise dynamic prevention of cross-site scripting attacks. Proceedings of the International Conference on Detection of Intrusions and Malware and Vulnerability Assessment, July 10-11, 2008, France, pp: 23-43.

Cao, Y., V. Yegneswaran, P.A. Porras and Y. Chen, 2012. Path cutter: Severing the self-propagation path of XSS JavaScript worms in social web networks. Proceeding of the Symposium on Network and Distributed System Security, February 6-9, 2012, DBLP Publisher, San Diego, California, USA., pp: 1-14.

Chandra, V.S. and S. Selvakumar, 2011. Bixsan: Browser independent XSS sanitizer for prevention of XSS attacks. ACM. SIGSOFT Software Eng. Notes, 36: 1-7.

Das, D., U. Sharma and D.K. Bhattacharyya, 2015. Detection of cross-site scripting attack under multiple scenarios. Comput. J., 58: 808-822.

Dong, G., Y. Zhang, X. Wang, P. Wang and L. Liu, 2014. Detecting cross site scripting vulnerabilities introduced by HTML5. Proceedings of the 11th International Joint Conference on Computer Science and Software Engineering (JCSSE), May 14-16, 2014, IEEE, Beijing, China, ISBN:978-1-4799-5822-1, pp: 319-323.

Garcia-Alfaro, J. and G. Navarro-Arribas, 2007. Prevention of cross-site scripting attacks on current web applications. Proceedings of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems, November 25-30, 2007, Portugal, pp: 1770-1784.

Hallaraker, O. and G. Vigna, 2005. Detecting malicious javascript code in mozilla. Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems, June 16-20, 2005, IEEE, Santa Barbara, California, USA., ISBN:0-7695-2284-X, pp: 85-94.

Hydara, I., A.B.M. Sultan, H. Zulzalil and N. Admodisastro, 2014. An approach for cross-site scripting detection and removal based on genetic algorithms. Proceedings of the 9th International Conference on Software Engineering Advances ICSEA, October 12-16, 2014, IARIA, Nice, France, ISBN:978-1-61208-367-4, pp: 227-232.

Ismail, O., M. Etoh, Y. Kadobayashi and S. Yamaguchi, 2004. A proposal and implementation of automatic detection-collection system for cross-site scripting vulnerability. Proceedings of the 18th International Conference on Advanced Information Networking and Applications (AINA04), Vol. 1, March 29-31, 2004, IEEE, Japan, ISBN:0-7695-2051-0, pp: 145-151.

Jacob, B., 2011. Automatic XSS detection and Snort signatures-ACLs generation by the means of a cloud-based honeypot system. Ph.D Thesis, Edinburgh Napier University, Dinburgh, Scotland.

Jim, T., N. Swamy and M. Hicks, 2007. Defeating script injection attacks with browser-enforced embedded policies. Proceedings of the 16th International Conference on World Wide Web, May 08-12, 2007, ACM, Banff, Alberta, Canada, ISBN:978-1-59593-654-7, pp: 601-610.

Johns, M., B. Engelmann and J. Posegga, 2008. Xssds: Server-side detection of cross-site scripting attacks. Proceedings of the ACSAC 2008 Annual Conference on Computer Security Applications, December 8-12, 2008, IEEE, Passau, Germany, ISBN:978-0-7695-3447-3, pp: 335-344.

Kan, W., T.Y. Wu, T. Han, C.W. Lin, C.M. Chen and J.S. Pan, 2014. An Efficient Detecting Mechanism for Cross-Site Script Attacks in the Cloud. In: Advanced Technologies, Embedded and Multimedia for Human-centric Computing. Huang, Yueh-Min., C. Han-Chieh , D. Der-Jiunn, J.J.H.P. James.(Ed.). Springer, Netherlands, pp: 663-672.

Kapodistria, H., S. Mitropoulos and C. Douligeris, 2011. An advanced web attack detection and prevention tool. Inf. Manage. Comput. Secur., 19:

Kim, J., K.S. Han, B. Jiang and E.G. Im, 2013. BOSF: BY-owner script filtering. Proceedings of the Third International Conference on Digital Information Processing and Communications, January 30-February 1, 2013, SDIWC, Dubai, UAE., pp: 26-30.

Kirda, E., C. Kruegel, G. Vigna and N. Jovanovic, 2006. Noxes: A client-side solution for mitigating cross-site scripting attacks. Proceedings of the ACM Symposium on Applied Computing, April 23-27, 2006, Dijon, France, pp: 330-337.

Kirda, E., N. Jovanovic, C. Kruegel and G. Vigna, 2009. Client-side cross-site scripting protection. Comput. Secur., 28: 592-604.

Klein, A., 2005. DOM based cross site scripting or XSS of the third kind. Web Appl. Secur. Consortium Articles, 4: 365-372.

Kruegel, C., G. Vigna and W. Robertson, 2005. A multi-model approach to the detection of web-based attacks. Comput. Networks, 48: 717-738.

Lan, D., W.S. Ting, Y. Xing and Z. Wei, 2013. Analysis and prevention for cross-site scripting attack based on encoding. Proceedings of the IEEE 4th International Conference on Electronics Information and Emergency Communication (ICEIEC), November 15-17, 2013, IEEE, New York, USA., ISBN:978-1-4673-4933-8, pp: 102-105.

Li, X., W. Yan and Y. Xue, 2012. SENTINEL: Securing database from logic flaws in web applications. Proceedings of the 2nd ACM Conference on Data and Application Security and Privacy, February 07-09, 2012, ACM, San Antonio, Texas, USA., ISBN:978-1-4503-1091-8, pp: 25-36.

Matsuda, T., D. Koizumi and M. Sonoda, 2012. Cross site scripting attacks detection algorithm based on the appearance position of characters. Proceedings of the 2012 Mosharaka International Conference on Communications Computers and Applications (MIC-CCA), October 12-14, 2012, IEEE, Japan, ISBN:978-1-4673-5230-7, pp: 65-70.

Mohosina, A. and M. Zulkernine, 2012. DESERVE: A framework for detecting program security vulnerability exploitations. Proceedings of the 2012 IEEE Sixth International Conference on Software Security and Reliability (SERE), June 20-22, 2012, IEEE, Kingston, Ontario, Canada, ISBN:978-1-4673-2067-2, pp: 98-107.

Nadji, Y., P. Saxena and D. Song, 2009. Document structure integrity: A robust basis for cross-site scripting defense. Proceedings of the Conference on Network and Distributed System Security Symposium, February 11-18, 2009, DBLP Publisher, San Diego, California, USA., pp: 1-20.

Nenad, J., C. Kruegel and E. Kirda, 2006. Pixy: A static analysis tool for detecting web application vulnerabilities. Master Thesis, TU Wien, Vienna, Austria.

Perez, P.M., J. Filipiak and J.M. Sierra, 2011. LAPSE+ Static Analysis Security Software: Vulnerabilities Detection in Java EE Applications. In: Future Information Technology, Park, J.J., L.T. Yang and C. Lee (Eds.). Springer, Berlin, Germany, pp: 148-156.

Ruse, M.E. and S. Basu, 2013. Detecting cross-site scripting vulnerability using concolic testing. Proceedings of the 10th International Conference on Information Technology: New Generations (ITNG), April 15-17, 2013, IEEE, Ames, Iowa, USA., ISBN:978-0-7695-4967-5, pp: 633-638.

Ruse, M.E., 2013. Model checking techniques for vulnerability analysis of Web applications. Ph.D Thesis, Lowa State University, Ames, Iowa.

Schwartz, E.J., T. Avgerinos and D. Brumley, 2010. All you ever wanted to know about dynamic taint analysis and forward symbolic execution (but might have been afraid to ask). Proceedings of the IEEE Symposium on Security and Privacy (SP), May 16-19, 2010, IEEE, Pittsburgh, Pennsylvania, USA., ISBN:978-1-4244-6894-2, pp: 317-331.

Seffernick, M., 2014. Preventing cross-site scripting with script-free HTML. J. Undergraduate Res. Ohio State, 4: 1-7.

Shahriar, H. and M. Zulkernine, 2011. S2XS2: A server side approach to automatically detect XSS attacks. Proceedings of the 2011 IEEE 9th International Conference on Dependable, Autonomic and Secure Computing (DASC), December 12-14, 2011, IEEE, Kingston, Ontario, Canada, ISBN:978-1-4673-0006-3, pp: 7-14.

Shar, L.K. and H.B.K. Tan, 2010. Auditing the defense against cross site scripting in web applications. Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT), July 26-28, 2010, IEEE, New York, USA., ISBN:978-989-8425-18-8, pp: 1-7.

Sharma, P., R. Johari and S.S. Sarma, 2012. Integrated approach to prevent SQL injection attack and reflected cross site scripting attack. Intl. J. Syst. Assur. Eng. Manage., 3: 343-351.

Stamm, S., B. Sterne and G. Markham, 2010. Reining in the web with content security policy. Proceedings of the 19th International Conference on World Wide Web, April 26-30, 2010, ACM, Raleigh, North Carolina, USA., ISBN:978-1-60558-799-8, pp: 921-930.

Stock, B., S. Lekies, T. Mueller, P. Spiegel and M. Johns, 2014. Precise client-side protection against DOM-based cross-site scripting. Proceedings of the 23rd USENIX Conference on Security Symposium, August 20-22, 2014, USENIX, San Diego, California, USA., ISBN:978-1-931971-15-7, pp: 655-670.

Tang, Z., H. Zhu, Z. Cao and S. Zhao, 2011. L-WMxD: Lexical based webmail XSS discoverer. Proceedings of the 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), April 10-15, 2011, IEEE, China, ISBN:978-1-4577-0249-5, pp: 976-981.

Wurzinger, P., C. Platzer, C. Ludl, E. Kirda and C. Kruegel, 2009. SWAP: Mitigating XSS attacks using a reverse proxy. Proceedings of the Workshop on Software Engineering for Secure System, May 19, 2009, Vancouver, BC., pp: 33-39.

Xiao, W., J. Sun, H. Chen and X. Xu, 2014. Preventing client side XSS with rewrite based dynamic information flow. Proceedings of the 2014 6th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), July 13-15, 2014, IEEE, Changsha, China, ISBN:978-1-4799-3845-2, pp: 238-243.

Zhang, Q., H. Chen and J. Sun, 2010. An execution-flow based method for detecting cross-site scripting attacks. Proceedings of the 2nd International Conference on Software Engineering and Data Mining, June 23-25, 2010, Chengdu, China, pp: 160-165.

Zheng, Y. and X. Zhang, 2013. Path sensitive static analysis of web applications for remote code execution vulnerability detection. Proceedings of the 2013 International Conference on Software Engineering, May 18-26, 2013, IEEE Press, San Francisco, California, USA., ISBN:978-1-4673-3076-3, pp: 652-661.