TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

Security of Web Browser: A Study on Attacks and Their Defences

Arshey M[1], Dr. Angel Viji .K.S[2]

[1] Research Scholar, Dept of Computer Science & Engineering,Noorul Islam University.
[2] Associate Professor, Dept of Computer Science & Engineering, College of Engineering, Kidangoor.
Email: arshi.sm@gmail.com[1]&ksangelviji@gmail.com[2]

Article Info
Volume 83
Page Number: 7951 - 7961
Publication Issue:
May-June 2020

Article History
Article Received: *19 November 2019*
Revised: *27 January 2020*
Accepted: *24 February 2020*
Publication: *18 May 2020*

Abstract

Today, the internet with its flexible nature and time-saving features have influenced and made all age-groups to depend on it, which has exponentially elevated internet usage. Web browsers (WB) are the utmost significant end-user applications for browsing, presenting and retrieving Internet resources. Now, WB like Apple Safari, Microsoft Internet Explorer, along with Mozilla Firefox is installed on almost all computers. There are countless attacks on WB. Since WB is utilized so frequently, it is essential to configure them securely. The security of such WB has turned into a prime issue in current years. More research has to be performed in the area of WB security attacks. Hence, there stands a requirement to discuss the protection against security attacks for implementing a new effectual defense mechanism. The principal object of this article is to review all the research directions in defense against WB attacks.

I. INTRODUCTION

Internet-centric social networking turns into a necessary part for every single person. Security becomes an important issue since the attacks frequency against the systems is rising rapidly [1]. Internet WB comprises several entry points and could consequently be attacked as of multiple sources. Because of the rapid advancement of modern computers and network infrastructure and the dependence on the Internet, web applications (WA) that are accessed via WB have to turn into a chief objective for cyber criminals [2].

Prior to conferring the WB attacks, it is imperative to know the base of how the internet has progressed over time. The 1st generation WAs have extremely little ability to meet the user needs. To close this gap, the WAs have advanced to offer services for instance searching, uploading and posting. Commons Gateway Interfaces (CGI) protocol gave a path to a user for interacting with the web pages via presenting data into forms [3]. The safety of an application thereby relies on the server as well as on browser-centered units, for instance JavaScript (JS) and cookies. This is hazardous, concerning the pervasiveness of web susceptibilities like Cross-Site Requests Forgery (CSRF), open redirectors (phishing) and Cross-Site Scripting (XSS) even on major websites [4].

1.1 Web Browser & Their Attacks

Basically, a WB stands as a software or software application program that is utilized for accessing information as of WWW [5]. Conventionally, while a user browses the Internet, their interactions with a browser is documented which is termed as public browsing mode. By attacking WB, the attackers could get access to the surfers' classified data, encompassing passwords and surfing habits. The attackers could do this since the WB always leaves caches, cookies, and browsing histories on the users' computers [6].

Some WB, including Chrome, Firefox, Internet Explorer, Safari, and Opera, provide extension features that let the user modify the browser

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

behavior. xWAs namely, e-banking, social networking sites, e-commerce, video-sharing sites, and blogs provide users with the ability to perceive and access information content, and even to contribute their own Web content, show their creativity as well as share information and knowledge. Such applications are modeled to be implemented inside a WB which is a mediator betwixt applications and users. Many diverse WAs could be simultaneously executed within a WB.

Some applications could offer users with helpful functionality, whilst others could be malicious with the only intention to compromise security [7]. Due to the instigation of the Web as well as the websites, the attackers discover several different attacks for harming the websites' confidentiality. Disparate web-browser attacks are proffered in fig 1.

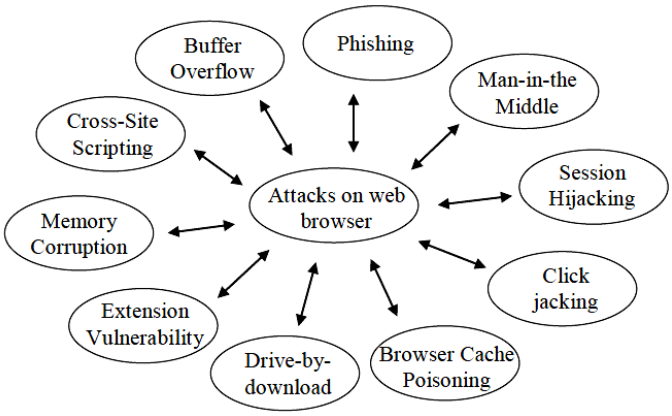

Figure 1: Types of attacks in web-browser

XSS is the prime threat for the WAs. The inquiry is done to find an effectual as well as practical means for assessing the programs of web- centric applications and also for eradicating the hazard [8]. Defending from XSS attacks are clearly explained in [9]. The central defense as of malware attacks/phishing is centered upon blacklists that are utilized via browsers to recognize if the demanded URL was proclaimed as malicious. The industries and academe are now working hard to build solutions to beat phishing threats.

Today, drive-by download (DBD) attacks stands as the utmost general methodologies to propagate malware. Such attacks characteristically utilize memory corruption susceptibilities in browser plug-ins as well as WB for executing shellcode, and consequently, gain the control of a user's computer. If a program endeavors for writing countless data to a buffer of fixed length or a memory, then overflow of the buffer occur is called as buffer overflow attack. Supplementary data could overwrite prevailing data in memory addresses subsequent to the buffer excluding when a program comprises adequate bounds checking to eliminate or flag data when so much is sent to a memory buffer.

Session-Hijacking attack gains illegal access to a Web Server at the network level along with application level. A clickjacking attack is termed as a UI re-dressing attack since it tricks users to click the image or button that would run concealed malicious script as of attacker site. This sort of attack scams victims by creating a malevolent page in order that it they click on a component of a disparate page that is slightly visible or invisible. Via pilfering the victim's clicks, attackers could compel the sufferer to do an unintentional act that is beneficial for them (for instance, start online money transactions) [10]. Section 2 proffers a detailed review of various proposed defending methodologies against some pre-listed WB attacks.

II. LITERATURE SURVEY

Nikos Virvilis et.al [11] scrutinized the protection level that was proffered by eminent WBs on iOS, Android as well as desktop (Windows) platforms, in opposition to a big collection of phishing in addition to malevolent URL. The outcomes exhibited that utmost browser specifically those meant for mobile gadgets which proffer restricted protection from such hazards. As an outcome, countermeasure was suggested and assessed, which could be utilized to notably ameliorate the protection level proffered to the users, despite a web platform they were utilizing.

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

KanpataSudhakara Rao et.al [12] introduced an XBuster for defending from the XSS filter. Most existing client-side solutions do not defend from all attack vectors like a partial script and HTML injections. Also, an XSS may inadvertently facilitate clickjacking attacks. XBuster was therefore designed to defend from all known clickjacking and XSS attacks. A framework of XBuster was employed as a Firefox extension. As a part of HTTP request processing, XBuster split each parameter into JS and HTML contexts. It, then, searched for a match of each H context in the HTTP response. Likewise, each J context was searched for a substring match with each input to the JS engine. Only the H and J contexts that exceed a specific threshold length were searched for. The latter was a crucial design parameter that affected the rates of false negatives and even false positives (FPs). Finally, the efficacy of XBuster on various XSS vulnerable sites was tested.

Chao Zhang et.al [13] put forward a solution JITScope for averting the JIT-compiled code as of being exploited. Via exploiting this code, attackers could bypass all current defenses. JITScope infused a common CFI policy on the considered code and statically compiled code, and also infused the policy " " on the considered code. Thereby, it rendered high protection for this code, and it could beat the utmost control-flow hi-jacking attack. Experimentations evinced that the aforesaid solution had a considerable performance over-head. Additionally, it could be employed in reality to shield opposed to instantaneous exploits.

Hyung-Jin Mun et.al [14] suggested an approach to aid a user to confirm whether a destination could be a file or web document via creating a short URL service. The aforesaid service transmutes a longer URL with details into a brief sort of URL as well as transmits the information for accessing the page with essential data. And, its providers observed the threat of the aimed URL page and then make a decision whether to proffer service or not. Via observing the alteration of web-document, it gauged as well as assessed the hazard of the webpage, in addition, ascertained whether to block the service as per the threshold that averts DBD via the short URL.

Shashank Gupta et.al [15] elucidated some performances problems in the prevailing defensive solutions of the JS injection attacks (example: XSS attacks). Furthermore, a comparison for such prevailing solutions was done centered on certain helpful metrics. Grounded on the recognized performance issues, an automatic detection system that scanned the innumerable feasible sites' locations meant for JS injection susceptibility was suggested. This detection system, initially, scanned the website for determining the injection localities. Next, it infused the malevolent XSS vectors on those injection spots. Finally, it took an input as the listing of disparate XSSs used in the secondary phase and scanned the susceptible WA for finding those attacks. The detection competency of their automatic scheme was assessed on a real-globe PHP-WA (BlogIt). Furthermore, the attained outcomes were propitious.

Gerardo Canfora et.al [16] recommended and verified a collection of features extorted as of the structure and content of web pages that could well be utilized as pointers of web security breaches. The extorted features were utilized for developing a predictor centered upon '5' machine learning (ML) algorithms that was deployed to categorize unrecognized WAs. The conducting tests evinced that the recommended collection of features could precisely classify malevolent sites with 0.84-precision and 0.89-recall during the best scenario.

XSS Attack: XSS is the leading class of web susceptibility. An XSS attack takes advantage of the inappropriately coded WAs for transmitting malevolent scripts to users' WB aimed at implementation. [17] Some defending XSS attacks on WAs are discussed in detail. Table-1 lists and expounds various existent protection mechanisms as of XSS attacks.

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

BhawnaMewara et.al [18] assessed '3' browsers – a) Internet Explorer 11, b) Mozilla Firefox 27 as well as c) Google Chrome 32 aimed at reflected XSS attack in opposition to disparate sort of susceptibility. None of those could fully defend from all feasible sorts of XSS-vulnerabilities. Furthermore, Firefox once installed with an append termed XSS-Me was found to be widely utilized to test the XSS vulnerabilities. Experiential outcomes evinced that this client-side solution could shield in opposition to maximal susceptibility than other browsers. It was ascertained to be propitious on the off-chance that this add-on was incorporated within the browser rather being infused as an addition.

Shashank Gupta et.al [19] propounded a robust prototype termed as XSS-SAFE (XSS Secured WA Framework) that was a server-side automatic prototype for the recognition as well as alleviation of XSS attacks. XSS-SAFE was planned centered upon the notion of infusing the JS features as well as developed a notion of infusing the sanitization schedules in a source code of JS to recognize and alleviate the malevolent infused XSS vectors. The JS feature contents and generated rules were repeatedly infused and also sanitization routines were inserted for discovering the XSS attacks. Their approach was estimated on '5' live JSP (JavaServer Pages) programs. The outcomes specified that XSS-SAFE recognized as well as alleviated majority of the formerly recognized and also unrecognized XSS with minimal FPs, lower runtime overhead, as well as 'Zero' false-negative rates.

MisganawTadesseGebre et.al [20] suggested a server-side upload filter which examined file contents uploaded to a server to shield in opposition to Content-Sniffing XSS attacks. Aimed at efficacy, some HTML attributes and elements were selected that might facilitate attackers to automatically implement malevolent code, as well as for precise detection, a collection of usual expressions was attained as of the series of these chosen HTML attributes as well as elements. Via utilizing a customary signature – rather than just a string - to view on a file, their detection framework became utmost accurate.

Table 1: Comparative study on defense against XSS attacks

Author name	Technique/ Implemented algorithm	Advantage	Disadvantage
Angelo Eduardo Nunan*et.al* [21]	ML	Focused on features extorted as of web document content and URL. The suggested features brought extremely accurate classification of malevolent pages.	Experiments performed were restricted to '2' classifiers as well as manipulation of its parameters.
Christopher M. Frenz*et.al* [22]	Intrusion Detection Systems (IDSs)	The IDS captured the probable client-side contents on a Web page	Much more meticulous testing needed to be employed to the suggested

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

		as well as hashed the content and rendered good results.	IDS to perceive how fine it spotted XSS attacks as of the breadth of the entire feasible XSS attacks on a variety of disparate Web pages.
Peter Wurzinger*et.al*[23]	SWAP	Reverse proxy averted each malevolent response as of being transmitted to the client, and thereby effectually inhibited the attack done on the client's browser.	SWAP mightn't be apt for a higher-level performance Web service. Established performance overhead.

Phishing: This attack supports dynamic user interactions. In phishing, the attackers consistently search for novel creative ways for tricking users to believe that their activities engage a genuine email or website. Phishers encompass more skills at counterfeiting sites to look similar to the preferred destination, and also encompassing graphics and logos on the phishing e-mails. [24] Disparate existent defense mechanisms from phishing attacks (PAs) have been delineated below:

Mohamed Alsharnouby et.al [25] made a scrutiny to appraise whether ameliorated WB safety indicators and elevated consciousness of phishing have brought a user enhanced competency for protecting oneself from those attacks. Moreover, the participants were proffered innumerable sites and told to recognize the phishing sites. Eye-tracking was utilized to attain targeted quantitative information wherein visual cues attract users as they ascertained the legality of sites. The outcomes evinced that users productively spotted only fifty-three percent of phishing sites even when geared up to recognize it and that they normally expend a little time staring at safety indicators contrasted to site contents while doing appraisals. Nevertheless, it was perceived that stare time on chrome components correlated to elevated capability for spotting phishing.

Jema David Ndibwile et.al [26] proffered UnPhishMe, an effectual mobile application framework that exploits a specific flaw of phishing sites: these sites accepted any sort of data (input) for verification. A mobile gadget user generates a false login account and credentials with UnPhishMe. UnPhishMe ascertained whether the present login page jumps to different web page subsequent to authentication. The efficacy of UnPhishMe was gauged by performing user experimentation on android mediums and also its memory, detection accurateness, as well as CPU performance, were validated. The outcomes evinced that UnPhishMe effectually aided the users in recognizing PAs with 96% accurateness and utilized a smaller computational power.

Xun Dong et.al [27] elucidated a framework - examination of the users' behaviors-to spot phishing sites. It was corroborated to be an accurate framework and conferred how it was modeled and executed to be difficult to resolve, and encompass delineated its exceptional strength in shielding users as of phishing breaches. UBPD -user-behavior centric phishing detection- was not modeled to

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

substitute existent methodologies. Instead, it must be employed to complement other methodologies, to offer better general protection. This filled a notable gap in present anti-phishing technology's ability.

Mahmood Moghimi et.al [28] suggested a rule-centered method to spot PAs in internet banking. It utilized '2' feature sets that were suggested to ascertain the webpage identity. This suggested feature sets encompassed '4' aspects to assess the page resources, as well as '4' facets to recognize the accessing protocols of page resource components. Estimated string matching algorithms were utilized to ascertain the association betwixt the URL and a web page's content in their 1st suggested feature set. Their suggested features are autonomous as of 3rd-party services like WB history and/or search engine result. The algorithm of the Support vector machine (SVM) was employed for categorizing web pages. The experimentations signified that the suggested model could spot phishing pages on internet banking with accurateness of 0.860% false-negative and 99.140% true positive alarm. They extorted the concealed knowledge as of the suggested SVM design by adopting an associated methodology. The extorted rules were embedded into a browser extension termed PhishDetector for making the suggested framework highly functional as well as simple to utilize. Assessing the employed browser extension specified that it could find PAs in internet banking with maximal reliability and accuracy.

Table 2: Comparative study on defense against Phishing attacks

Author name	Technique/ Implemented Algorithm	Advantages	Disadvantages
HuuHieu Nguyen and Duc Thai Nguyen [29]	ML	This system could detect and categorize phishing sites with 98.80% accurateness.	Though it rendered high classification accurateness for spotting phishing websites, the process is complex.
Tuan Anh Pham *et.al* [30]	Genetic Programming	Generated more accurate phishing prediction models when contrasted to other ML algorithms and also produced models with lower FNR.	Some of the evaluation parameters affect GP performance.
RouthuSrinivasa Rao *et.al*[31]	Computer vision technique	SURF detector extorted unique key point features for recognizing similarity degree betwixt authorized site and distrustful site. It could also spot Zero-day	This methodology failed when a major portion of the phishing site is substituted with ads or the overall image style was changed.

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

		PAs.	Higher computation and accuracy costs.
SattarSeifollahi*et. al* [32]	Optimization centric Clustering algorithms	Produced highly meaningful as well as effortlessly explicable clusters in the phishing data sets when contrasted to the existent k-means clustering algorithm. Optimization centric algorithms specified the existence of well-divided clusters in the phishing e-mails data-set.	Optimization of the clustering algorithm with feature selection algorithm requirements made the process a bit complex.

Drive-by-Download attacks: A user wrongly accessing a compromised site is normally taken to another site (i.e., adversary) and is compelled to download malwares subsequent to being subjugated. Furthermore, the adversary pilfers the user's details via utilizing data-leaking malware and might as well attempt for compromising public websites possessed by means of individual users via mimicking the website supervisor utilizing the purloined credentials. Those websites finally turned into landing sites intended for DBD malware infection.

Amir Javed et.al [33] targeted to do a work that developed approaches to recognize malevolent URLs in OSNs attempting to resolve the issue. It was done to estimate that the URLs were possibly to be malevolent in seconds of entering the interaction - prior to the DBD could execute its payload. An ML model was built for utilizing machine action data and tweet metadata, for moving past the post-execution categorization of those URLs as malevolent and for predicting a URL would be malevolent with 99.0% F-measure (utilizing 10-folds cross-validation) in addition to 83.30% (utilizing a hidden testset) at 1s into the interface with the URL. Thereby, a base was proffered as of which for disconnecting the link to the server prior to a DBD has finished and also proactively blocking as well as averting an attack, instead of repairing and reacting later.

Mitsuaki Akiyama et.al [34] recommended a monitoring scheme termed Honey Circulator centered upon a honeytoken that keenly leaks bait credentials as well as draws in adversaries to the decoy server which performs similar to a compromised web content management (CWCM) scheme. For attaining effectual counter-measures to an attack cycle comprising DBD, compromised websites, and credential leakage, the main focus was given on tracing the utilizations of bait credentials exposed by means of malware as well as observing the acts of adversaries in a CWCM. For recursively examining the DBD phases on a web-centered attack cycle, the recommended scheme involved gathering malware, disseminating bait details, observing untrustworthy access, as well as examining compromised web content. It could instantaneously find out unrecognized malevolent entities with no massive web crawling on account of the direct observation of CWCM system. This recommended system facilitated continual and also stable

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

scrutinizing for concerning '1' year. Additionally, roughly all the discovered malevolent websites hadn't been formerly reported in public blacklists.

Manoj Cherukuri et.al [35] suggested a light-weight approach utilizing kernel machines aimed at the recognition of shellcodes (SCs) utilized in DBD. Since the SCs were given in web pages as JS strings, the suggested approach's efficacy was studied with around 9850 SCs as well as 10000 JS strings gathered as of the wild. Their examination evinced that the trained SVMs categorized with an accurateness of over ninety-nine. The performance shown by the approach was contrasted to an emulation centered approach as well as perceived that this approach showed better accuracies with thirty-three percent of the time encompassed by means of the emulation centered approach.

Manuel Egele et.al [36] recommended a framework to counter DBD that contingents upon x86-instruction emulation for recognizing JS string buffers encompassing SC. The detection was incorporated into a browser, as well as executed prior the control was sent to the SC, thereby, effectually forestalling the DBD. The solution sustained top-level performance via averting unessential emulator invocations, whilst making sure that each buffer along with possible SC was validated. A prototype of their system was implemented and estimated around thousands of malevolent as well as genuine websites. The outcomes expounded that the system executed precise detection with no FP.

ClickJacking: Clickjacking could be lessened in countless ways. Browser Add-on/Browser centered, Website code/script, and Website + Browser Codes are some existing clickjacking mitigation approaches. A characteristic clickjacking attack utilizes '2' nested iframes for cropping as well as positioning an element as of an aimed website. The internal iframe comprises the target page and have to be huge enough for displaying it, in order that the part wherein the user would click could be seen devoid of scrolling. The smaller outer iframe works as a window on the page which is loaded into the internal iframe. But for the user interface re-dressing attack, only the outer iframe must be huge enough for displaying the aimed part. [37]

Jawwad A. Shamsi et.al [38] proffered a framework to counteract clickjacking. As a solution, user feedback was employed to make dynamic white together with black lists and conquered limits given by former solutions. Clicksafe, a browser-centered tool to render augmented reliability as well as security in opposition to clickjacking attacks was suggested. Clicksafe was grounded on '3' major units. The detection unit spotted malevolent units on a webpage that forward users to outside links. The alleviation mode proffered the interception of user clicks as well as the educated cautions to the users who could after that decide to carry on or discontinue. Clicksafe as well integrates a feedback mode that captures the users' actions, transmutes them to ratings and permitted forthcoming interactions to be highly noted. Clicksafe was pre-dominant when contrasted to other identical tools as the detection as well as mitigation was centered upon an inclusive framework that utilized the detection of malevolent web units and embedding user feedback. The click safe's mechanism was explained its performance was delineated, and its potential was highlighted in rendering safety against clickjacking to countless users.

Extension Vulnerabilities: Browser extensions alter the core browser user experiences by modifying the browser's user interface and also by interacting with web sites.

Sruthi Bandhakavi et.al [39] proffered VEX which was a framework intended for emphasizing possible security susceptibility in WB extensions via employing static information-flow scrutiny for the JS code utilized to execute extensions. Innumerable patterns of flows and unsafe programming practices were delineated that might cause privilege elevation in Firefox browser extensions. Also, VEX examined

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

those extensions meant for flow patterns utilizing context- and flow-sensitive, high-precision static scrutiny. Thousands of browser extensions are examined, in addition VEX ascertained '6' exploitable vulnerabilities, '3' of which were formerly unrecognized. VEX as well found countless instances of bad programming practices that might direct to security susceptibility. It was evinced that on contrasted to present Mozilla extension review tools, VEX highly diminished the human load for manually vetting extensions whilst searching for key sorts of hazardous flows.

III. CONCLUSION

Over the past few decennia, the number of the user utilizing the WAs has been augmented. Though no single tool could guard the host of probable browser attacks, a defense-in-depth methodology could help to secure WBs. Browser centered attacks originate as of malicious websites. With no security patches, the WBs are susceptible to disparate sorts of attacks. Manifold web security threats and their proposed defenses have been clearly discussed. This review work would help researchers to implement an effectual defense mechanism in opposition to WB attacks. For future work, grounded on the current research problems discussed above, it is necessary to explore optimization centered computer vision and ML approaches that make the process simple and proffers effective results.

REFERENCES

[1] Patil Shital Satish and R. K. Chavan, "Web Browser Security: Different Attacks Detection and Prevention Techniques", International Journal of Computer Applications, Vol. 170, No. 9, pp. 36-41, 2017.

[2] Nayeem Khan, Johari Abdullah and Adnan Shahid Khan, "Defending Malicious Script Attacks Using Machine Learning Classifiers", Wireless Communications and Mobile Computing 2017.

[3] Mrunali P. Pathak, NidaKausar Khan and Tejashree C. Tantak, "Novel Approach To Detect and Prevent Web Attacks", International Research Journal of Engineering and Technology (IRJET), Vol. 3, No. 5, pp. 504-510, 2016.

[4] Chetan Bansal, KarthikeyanBhargavan, Antoine Delignat-Lavaud, and Sergio Maffeis, "Keys to the cloud: formal analysis and concrete attacks on encrypted web storage", In International Conference on Principles of Security and Trust, pp. 126-146, Springer, Berlin, Heidelberg, 2013.

[5] Dhruwajita Devi, DhrubajyotiPathak, and Sukumar Nandi, "Vulnerabilities in Web Browsers", Indian Institute of Technology, Guwahati, India 2010.

[6] Fu-Hau Hsu, Min-Hao Wu, Yi-Wen Chang and Shiuh-Jeng Wang, "Web security in a windows system as Privacy Defender in private browsing mode", Multimedia Tools and Applications Vol. 74, No. 5, pp. 1667-1688, 2015.

[7] Marin Šilić, JakovKrolo and GoranDelač, "Security vulnerabilities in modern web browser architecture", In MIPRO, 2010 Proceedings of the 33rd International Convention, pp. 1240-1245, IEEE, 2010.

[8] AbdallaWasefMarashdih and ZarulFitriZaaba, "Cross Site Scripting: Removing Approaches in Web Application", Procedia Computer Science, Vol. 124, pp. 647-655, 2017.

[9] LwinKhinShar and HeeBengKuan Tan, "Defending against cross-site scripting attacks", Computer Vol. 45, No. 3, pp. 55-62, 2012.

[10] Marco Balduzzi, Manuel Egele, EnginKirda, DavideBalzarotti and Christopher Kruegel, "A solution for the automated detection of clickjacking attacks", In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, pp. 135-144, ACM, 2010.

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

[11] Nikos Virvilis, AlexiosMylonas, NikolaosTsalis, and DimitrisGritzalis, "Security Busters: Web browser security vs. rogue sites." Computers & Security Vol. 52, pp. 90-105, 2015.

[12] KanpataSudhakara Rao, Naman Jain, Nikhil Limaje, Abhilash Gupta, Mridul Jain and Bernard Menezes, "Two for the price of one: A combined browser defense against XSS and clickjacking", In Computing, Networking and Communications (ICNC), 2016 International Conference on, pp. 1-6, IEEE, 2016.

[13] Chao Zhang, MehrdadNiknami, Kevin Zhijie Chen, Chengyu Song, Zhaofeng Chen and Dawn Song, "JITScope: Protecting web users from control-flow hijacking attacks", In Computer Communications (INFOCOM), 2015 IEEE Conference on, pp. 567-575, IEEE, 2015.

[14] Hyung-Jin Mun, and YongzhenLi,"Secure short URL generation method that recognizes risk of target URL", Wireless Personal Communications Vol. 93, No. 1, pp. 269-283, 2017.

[15] Shashank Gupta and B. B. Gupta, "Automated discovery of javascript code injection attacks in PHP web applications", Procedia Computer Science, Vol. 78, pp. 82-87, 2016.

[16] Gerardo Canfora and Corrado Aaron Visaggio, "A set of features to detect web security threats", Journal of Computer Virology and Hacking Techniques Vol. 12, No. 4, pp. 243-261, 2016.

[17] Hsiu-Chuan Huang, Zhi-Kai Zhang, Hao-Wen Cheng and Shiuhpyng Winston Shieh, "Web Application Security: Threats, Countermeasures, and Pitfalls", Computer, Vol. 50, No. 6, pp. 81-85, 2017.

[18] BhawnaMewara, SheetalBairwa, and Jyoti Gajrani, "Browser's defenses against reflected cross-site scripting attacks", In Signal Propagation and Computer Technology (ICSPCT), 2014 International Conference on, pp. 662-667, IEEE, 2014.

[19] Shashank Gupta, and B. B. Gupta, "XSS-SAFE: a server-side approach to detect and mitigate cross-site scripting (XSS) attacks in JavaScript code", Arabian Journal for Science and Engineering Vol. 41, No. 3, pp. 897-920, 2016.

[20] MisganawTadesseGebre, Kyung-Suk Lhee, and ManPyo Hong, "A robust defense against content-sniffing xss attacks", In Digital Content, Multimedia Technology and its Applications (IDC), 2010 6th International Conference on, pp. 315-320, IEEE, 2010.

[21] Angelo Eduardo Nunan, Eduardo Souto, Eulanda M. dos Santos, and Eduardo Feitosa, "Automatic classification of cross-site scripting in web pages using document-based and URL-based features", In Computers and Communications (ISCC), 2012 IEEE Symposium on, pp. 000702-000707, IEEE, 2012.

[22] Christopher M. Frenz, and Jong P. Yoon, "XSSmon: a perl based IDS for the detection of potential XSS attacks." In Systems, Applications and Technology Conference (LISAT), 2012 IEEE Long Island, pp. 1-4, IEEE, 2012.

[23] Peter Wurzinger, Christian Platzer, Christian Ludl, EnginKirda, and Christopher Kruegel, "SWAP: Mitigating XSS attacks using a reverse proxy." In Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems, pp. 33-39, IEEE Computer Society, 2009.

[24] Ike Vayansky and Sathish Kumar, "Phishing–challenges and solutions", Computer Fraud & Security, Vol. 2018, No. 1, pp.15-20, 2018.

[25] Mohamed Alsharnouby, FurkanAlaca, and Sonia Chiasson, "Why phishing still works: User strategies for combating phishing attacks", International Journal of Human-Computer Studies Vol. 82, pp. 69-82, 2015.

[26] Jema David Ndibwile, YoukiKadobayashi, and Doudou Fall, "UnPhishMe: Phishing Attack Detection by Deceptive Login Simulation

May – June 2020
ISSN: 0193-4120 Page No. 7951 - 7961

through an Android Mobile App", In Information Security (AsiaJCIS), 2017 12th Asia Joint Conference on, pp. 38-47, IEEE, 2017.

[27] Xun Dong, John A. Clark and Jeremy L. Jacob, "Defending the weakest link: phishing websites detection by analysing user behaviours", Telecommunication systems, Vol. 45, No. 2-3, pp. 215-226, 2010.

[28] Mahmood Moghimi and Ali YazdianVarjani, "New rule-based phishing detection method", Expert systems with applications, Vol. 53, pp. 231-242, 2016.

[29] HuuHieu Nguyen and Duc Thai Nguyen, "Machine learning based phishing web sites detection", In AETA 2015: Recent Advances in Electrical Engineering and Related Sciences, pp. 123-131.,Springer, Cham, 2016.

[30] Tuan Anh Pham, QuangUy Nguyen and XuanHoai Nguyen, "Phishing attacks detection using genetic programming", In Knowledge and Systems Engineering, pp. 185-195, Springer, Cham, 2014.

[31] RouthuSrinivasa Rao, and Syed Taqi Ali, "A computer vision technique to detect phishing attacks", In communication systems and network technologies (CSNT), 2015 fifth international conference on, pp. 596-601, IEEE, 2015.

[32] SattarSeifollahi, AdilBagirov, Robert Layton and Iqbal Gondal, "Optimization based clustering algorithms for authorship analysis of phishing emails", Neural Processing Letters Vol. 46, No. 2, pp. 411-425, 2017.

[33] Amir Javed, Pete Burnap and Omer Rana, "Prediction of drive-by download attacks on Twitter", Information Processing & Management, 2018.

[34] Mitsuaki Akiyama, Takeshi Yagi, Takeo Hariu and YoukiKadobayashi, "HoneyCirculator: distributing credential honeytoken for introspection of web-based attack cycle", International Journal of Information Security, pp. 1-17, 2017.

[35] Manoj Cherukuri, Srinivas Mukkamala, and Dongwan Shin, "Detection of shellcodes in drive-by attacks using kernel machines", Journal of Computer Virology and Hacking Techniques, Vol. 10, No. 3, pp. 189-203, 2014.

[36] Manuel Egele, Peter Wurzinger, Christopher Kruegel, and EnginKirda, "Defending browsers against drive-by downloads: Mitigating heap-spraying code injection attacks", In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 88-106, Springer, Berlin, Heidelberg, 2009.

[37] A. Sankara Narayanan, "Clickjacking vulnerability and countermeasures", International Journal of Applied Information Systems (IJAIS) Foundation of Computer Science FCS, New York, USA Vol. 4, No. 7, pp. 7-10. 2012.

[38] Jawwad A. Shamsi, Sufian Hameed, Waleed Rahman, Farooq Zuberi, Kaiser Altaf and Ammar Amjad, "Clicksafe: Providing security against clickjacking attacks", In High-Assurance Systems Engineering (HASE), 2014 IEEE 15th International Symposium on, pp. 206-210, IEEE, 2014.

[39] Sruthi Bandhakavi, Samuel T. King, Parthasarathy Madhusudan and Marianne Winslett, "VEX: Vetting Browser Extensions for Security Vulnerabilities", In USENIX Security Symposium, Vol. 10, pp. 339-354. 2010.