An Examination on Models of Mashup and web Security Threats: Web Application

Narendra M Kandoi[1], Dr. Vilas M Thakare[2]
Department of Computer Science & Engineering
[1] Research Scholar, S.G.B. Amravati University Amravati, Maharashtra, India
[2]Associate Professor, S.G.B. Amravati University Amravati, Maharashtra, India

Abstract- *over the most few years, numerous security researchers proposed to enrich the web stage with more thorough, explanatory establishments. Their objective is designing models which take into consideration an exact thinking on web security issues and creating compelling apparatuses to make the Web a more secure place, relieving at any rate some portion of this weight from the shoulders of web engineers and program merchants. Mashups are significantly more powerful than traditional (binary) software segments. Since mashups are all about combining content from multiple web sites in a highly dynamic fashion, they cannot be built easily with static programming languages that require advance compilation, static type checking and binary files. Data mashups, inverse to the consumer mashups, consolidate comparable sorts of media and data from different sources into a solitary portrayal. The combination of every one of these assets makes another and unmistakable Web benefit that was not initially given by either source. This approach is different from the other dynamic techniques presented in this section because it enforces specific policies for mashup integration; however the policies are still safety properties.*

Keywords- Web mashups, Same-Origin Policy, Web browser,.

I. INTRODUCTION

Web mashups are characterized as Web locales that compose data from more than one website, yet this definition is in tension with the same-inception approach, which prevents such interactions [1]. Many data providers want to publish information for any integrator site to use, but the same-origin policy avoids the integrator Web page from providing XML Http Requests to the data directly [2]. Security depends to some degree on the unwavering quality of the supplier's substance [3]. Be that as it may, mashups can likewise associate powerfully to Web locales not really under the supplier's control, which shows encourage security challenges [4]. Therefore, content providers should secure their servers and validate content, which they don't always do [5]. Cross- origin collaboration inside the program is presently directed by the Same-Origin Policy (SOP). SOP characterizes reports in view of their causes [6]. Records from a similar origin may freely access each other's substance, while such access is refused for reports of various beginnings [7]. Unfortunately, the SOP mechanism ends up being tricky for mashup security. Starting point following in SOP is just fractional and enables content from various sources to exist together under a similar beginning [8]. In browsers today, any third-party content included in a document is considered to have the document's origin regardless of the actual origin of the included content [9]. This turns problematic in a mashup setting, because the third-party content may be freely send to the document's origin [10]. That is, third-party content may not be sent to the document's origin without being declassified by the third-party [11], [12]. The seven challenges are cataloguing, data integrity, making data web-enabled, security and identity, sharing and reusing, trust certificates and version control mechanisms [13]. In mashup improvement there is significantly more concentrate on reusing the substance instead of the execution of a site. While institutionalized arrangements for different substance groups exist, it is frequently shockingly hard to reuse the execution of a site in different settings [14]. Because the current web technologies do not make it easy to specify which parts of the web site are intended to be reusable in other contexts and which not [15] are. In a similar manner, numerous mashups reuse the visual portrayal of sites only, while others reuse the substance independently from its visual portrayal. No well-defined rules or interfaces exist for keeping the content separate from its visual representation [16].

II. MASHUP

Mashups are significantly more powerful than traditional (binary) software segments. Since mashups are all about combining content from multiple web sites in a highly dynamic fashion, they cannot be built easily with static programming languages that require advance compilation, static type checking and binary files [17]. This has created a trend towards more and more dynamic programming languages such as JavaScript, Perl, PHP and Python. Because of the increased focus on content rather than on implementation techniques, the mashup developer base is different from conventional software development projects [18]. The distribution and sharing energy of the Web makes it particularly simple to reuse content in unanticipated, unexpected ways [19].

The history of the web browser as a document viewing environment is apparent when analyzing the restrictions and limitations that web browsers have in the area of networking and security [20].

Many of these limitations date back to the conventions that were established early on in the design and historical evolution of the web browser [21]. Most web applications handle get to utilizing three interrelated security systems:

- Authentication
- Session administration
- Access control [22]

Each of these mechanisms represents a significant area of an application's attack surface, and each is absolutely fundamental to an application's overall security posture [23]. Because of their interdependencies, the overall security provided by the mechanisms is only as strong as the weakest link in the chain. A deformity in any single part may empower an attacker to increase unhindered access to the application's usefulness and data [24].

Over the most few years, numerous security researchers proposed to enrich the web stage with more thorough, explanatory establishments. Their objective is designing models which take into consideration an exact thinking on web security issues and creating compelling apparatuses to make the Web a more secure place, relieving at any rate some portion of this weight from the shoulders of web engineers and program merchants. One natural question is whether formal methods have been successful in this field or whether they can only be considered a theoretical exercise as of now: practical applications are important to showcase the effectiveness of formal methods at dealing with the problems mentioned above and encourage the web security community to integrate efforts with the formal methods community. Hence, the main goal of this research is to design such a unique security framework based on Aspect Oriented Programming (AOP) that even if the developer had not considered security as a one of the component of web application in the beginning, at a later stage it should be free (safe) from major vulnerabilities and attacks [24].

III. TYPES OF MASHUP

There are many sorts of mashup, for example, business mashups, purchaser mashups and information mashups. The most well-known sort of mashup is the shopper mashup, which is gone for the overall population [25].

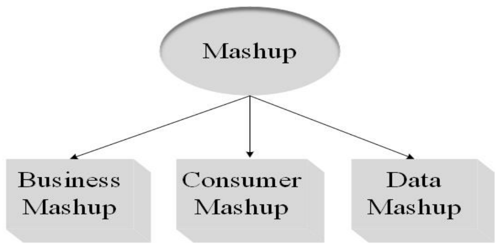

Fig. 1 Types of Mashup

Business (or enterprise) mashups characterize applications that join their own assets, application and information, with other outer Web administrations. They focus information into a solitary introduction and take into account synergistic activity among organizations and engineers. This functions admirably for a coordinated advancement venture, which requires joint effort between the engineers and client (or client intermediary, normally an item administrator) for characterizing and actualizing the business necessities. Undertaking mashups are secure, outwardly rich Web applications that uncover noteworthy data from differing inner and outside data sources.
Consumer mashups consolidate information from various open sources in the program and arrange it through a straightforward program UI. (e.g.: Wikipedia vision joins Google Map and a Wikipedia API).
Data mashups, inverse to the consumer mashups, consolidate comparable sorts of media and data from different sources into a solitary portrayal. The combination of every one of these assets makes another and unmistakable Web benefit that was not initially given by either source.

IV. DIFFERENT SECURITY MODELS OF MASHUP

With the development of the web, the new sort of web applications showed up: mashups. These applications include the content from multiple sources, for example a housing rental website combines the information about the houses and maps them to Google maps. The inclusion of the remote content is usually implemented by the use of frames that separate this content from the main page. Inside, the program executes the Document Object Model (DOM) that is a tree portrayal of the got WebPages.

a) Web Jail: Security architecture for mashups

The client-side security engineering that enables minimum benefit combination of parts into a web mashup based on aspect weaving while the security policies are specified for every iframe of the page. The language of the security policy is relatively simple and is similar to the Content Security Policy (CSP). The security arrangement determines a self-characterized white list for each classification of APIs. For instance, "extcomm: [google .com, youtube .com]" implies that outside correspondence are just permitted to the given spaces.

b) Security policies

The arrangement is another characteristic of an iframe in a mashup, which implies that a mashup integrator can force confinements on the conduct of untrusted outsider components. In the iframe policy, particular security-sensitive events can be fully enabled, fully disabled or enabled only for a self-defined whitelist [25].

c) ***Formal guarantees***

No formal guarantees are provided.

This approach is different from the other dynamic techniques presented in this section because it enforces specific policies for mashup integration, however the policies are still safety properties: the JavaScript programs are not allowed to invoke the APIs that contradict the security policy.

The security of web mashups is an active field of research that is strongly related to JavaScript security. The recent sandboxing libraries techniques described above contribute to this area, however, they are sometimes not practical due to the fact that static analysis only covers a subset of JavaScript. A recent contribution to this field that presented Mashic Compiler. The mashup consists of an integrator code and the gadgets to be added. Some of the researches considered the gadgets that are added by a <script> tag. For this situation, the device and the integrator would get appointed a similar cause and thus, if the contraption is non-considerate, it can break the security of the mashup [25].

Given the contraptions code and the incorporating code, Mashic assembles the integrating code such that every device and the integrator keep running in their own particular iframe. A little library is additionally added to every contraption, which are generally unmodified. It enables the integrators to compose secure mashups where security is accomplished by means of the Same Origin Policy. The compiled code is proportional to the first code, when the contraptions are kind. The meaning of an benign gadget is a novel thought that is characterized through an decorated semantics.

- The devices only learn what is being sent to them by the integrator.
- The contraptions may only interact with the integrator by replying to its messages, along these lines they can't straightforwardly change the load of the integrator [25].

d) The Mashic compiler Model

The Mashic compiler enhances mashup security. There are two approaches to incorporate contraptions in a mashup:

- Using HTML script tags: for this situation, the contraption is specifically implanted in the incorporating site page and acquires the cause of the last mentioned. This suggests the device keeps running with similar benefits of the integrator;
- Using HTML iframe labels: for this situation, the contraption is stacked in a confined domain and jelly its own origin; henceforth as far as possible its capacities on the incorporating site page. The communications between the contraption and the integrator are restricted to message passing.

Unfortunately, web designers ordinarily forfeit security for programming accommodation and implement mashup by making utilization of script tags. The Mashic compiler takes in input a current mashup and produces a protected mashup in view of iframe labels and message passing. The paper presents two formal outcomes: an accuracy result, demonstrating that the yield of the Mashic compiler is equivalent to the first mashup when the embedded device is " benign " and a security result, proving confidentiality and integrity properties for the compiled mashup.

e) The Yoshi Hama's Browser Model

The model formalizes the browser using a big-step operational semantics, covering the evaluation of client-side scripts, the presence of multiple browser windows, the DOM, cookies and HTTP requests. The model incorporates a few non-trivial highlights of real web browsers, similar to document content that may reference external resources (for example, <img> and <script> tags), DOM transformation operations, an eval build for dynamic code assessment, top notch capacities, and event handlers. Unfortunately, the formalism was only explained by a couple of inference rules showing how one may give huge advance semantics for a web browser, yet it isn't thorough or finishes enough to be usable in formal verifications.

The browser model utilizes data stream marks for fine-grained get to control on sites, focusing on the protected combination of substance from various, commonly distrusting websites (mashup security). In this view, sites characterize sets of marks with get to control properties and join these names to segments of their HTML documents. The names are then naturally spread by the web program and followed on singular DOM nodes and script variables to uphold get to control checks. Verifiable data streams where privileged insights are spilled by the execution of contingent program branches relying upon private data.

V. WEB SECURITY THREATS

There are numerous dangers related with web perusing and web applications, including phishing, drive-by downloads, blog spam, account takeover, and snap extortion. Albeit some of these dangers spin around misusing execution vulnerabilities, (for example, memory wellbeing blunders in programs or deceiving the client), we center, in this paper, on routes in which an attacker can manhandle web usefulness that exists by plan [26]. For instance, a HTML frame component gives a vindictive web a chance to webpage create GET and POST solicitations to discretionary sites, prompting security dangers like cross-website request for fraud (CSRF). Sites utilize various diverse procedures to protect themselves against CSRF, yet we do not have a deductively thorough approach for concentrate these resistances. By figuring a precise model of the web, we can assess the security of these safeguards and decide how they associate with augmentations to the web platform [27].

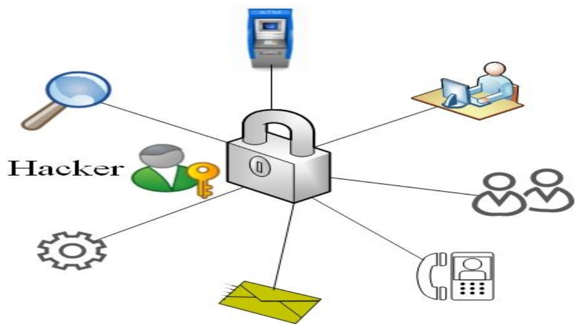

Fig.2 Web Security Threat

Mashup security issues are somewhat different from normal web app security issues

- Authentication to various backend administrations with various certifications, confirmation protocols
- Authorization to various backend administrations requiring qualities from divergent sources
- Bridging point-to-point protocol security systems, for example, SSL
- Extending consistence principles and controls out to the cloud
- Understanding the ramifications of information being utilized as a part of new ways

Web application security concerns are partitioned into two essential classifications: Physical security and semantic security. The main class plans to cover issues, for example, secure and dependable information trade. This gathering of security concerns can profit by existing techniques and methodologies of Web 1.0 to deal with the security and trust issues [28]. The semantic security worries then again, handle the data partaking in a more elevated amount by misusing the authoritative arrangements so as to portray the common information in a PC procedure capable way.

In any case, Mashup applications, by their tendency, include communication between different page parts. Frequently these parts are stacked from various causes. Cross-inception collaboration inside the program is at present controlled by the alleged Same-Origin Policy (SOP). SOP characterizes reports in view of their starting points [29]. Records from a similar birthplace may uninhibitedly get to each other's content, while such access is denied for archives of various causes. Lamentably, the SOP component ends up being hazardous for mashup security.

There are two sorts of security issues with the mashups one the issues that emerge specifically from the absences of innovation and antagonistic mishandle of them and in other hand the issues that emerge from the inquiries concerning reliability of content. The last one is progressively an issue of standard and it exists due to the idea of mashups. Today there are more than 5386 mashups recorded in http://www.programmableweb.com which is a website that gathers data about mashups and mashup advancements [30].

So there's a genuine downside to expanded dependence on Web applications: they're inalienably shaky and effortlessly traded off. All things considered, Symantec rates 73 percent of Web application vulnerabilities as easy to manhandle. Helpless Web applications not simply put compose structures and devices at significantly more serious hazard; they additionally offer an immediate course to classified client information, for example, account history, charge card numbers and well being records, and to delicate corporate data. This genuine disadvantage can be overwhelmed by actualizing the best possible security worries at run time alongside the improvement of web application [31].

VI. LITERATURE SURVEY

Samiha Ayedet al. [32] proposed a structure to connect the security approaches with the detail and the execution periods of uses characterized for the frameworks. The proposed engineering depends on an AOP to enforce security strategies considering both access and use control inside distributed frameworks. The enforcement of security policies had started by translating the set of security policy rules into an

AOP aware knowledge. Based on the translation, the aspect generation phase taken place. During that phase, generic and abstract patterns were derived on the basis of definition of security policies. The sending of the structure modules, proposed in this paper, considered the progressions that may happen in the security approach amid the application execution. They also presented the implementation as well as the evaluation of our proposition.

M.I.P.Salas*et al.* [33] proposed a new approach to analyze the robustness of Web Services by Fault Injection with WS Inject. This allowed the emulation and generation of attacks; however, the process was delayed and often not automated. In the examination, the Cross-site Scripting (XSS) attack was copied. As per the research cited, this was a fairly frequent attack, whose impacts were much crushed for servers and users of Web Services. The results of the Penetration Testing phase helped to develop the rules for vulnerabilities analysis. Be that as it may, the outcomes got by soap UI demonstrated an large percentage of false positives and false negatives. They likewise confirmed the security provided by WS-Security standard with the extra Security Token against XSS attack. In the two stages, the utilization of WS-Security lessened the quantity of vulnerabilities altogether. Be that as it may, this was enhanced with the utilization of different details. One advantage position of the proposed approach was that it depended on the utilization of a fault injector of universally useful, which was utilized to imitate a few sorts of assaults and created variations of the same, which was typically constrained in the apparatuses regularly utilized for security testing, as the vulnerabilities scanners.

Jose Fonseca et al. [34] proposed a strategy and a model apparatus to assess the web application security components. The approach was based on the possibility that injecting practical vulnerabilities in the web application and attacking them consequently were utilized to assist the evaluation of existing security components and apparatuses in the custom setup situations. The proposed vulnerability and attack injection structure depended on the investigation of an extensive number of vulnerabilities in real web applications to give consistent with life comes about. Notwithstanding the generic methodology, the paper depicted the execution of the Vulnerability and Attack Injector Tool (VAIT) that permitted the automation of the whole procedure. VAIT is utilized to run an arrangement of examinations that showed the attainability and the viability of the proposed system. The tests incorporated the assessment of scope and bogus positives of an interruption location framework for SQL Injection attacks and the evaluation of the adequacy of two best business web application helplessness scanners. The o results showed that the injection of vulnerabilities and attacks was indeed an effective way to evaluate security mechanisms and to point out not only their weaknesses but also ways for their improvement.

Junjie*et al.* [35] proposed the hierarchical Stochastic game nets (SGN) model and analysis methods, included important theorems and corollaries based on which the complicated attack and defense processes were described and the identity, confidentiality, availability and integrity in web services were analyzed and evaluated quantificationally. SGN had a powerful modeling and analyzing ability for the complicated and dynamic game problems, by which the complexity of the security issues of web services were solved properly. A series of simulation results were presented to show that, by applying hierarchical SGN model to describe the attack and defense behaviors in web services, quantifiable results can be successfully obtained for the evaluation of important attributes.

Rui Andre Oliveira *et al.* [36] presented an experimental approach that permitted understanding how well a given web service framework was prepared to handle Denial of Service (DoS) attacks. DOS attacks may exact serious harm to the web service co-ops, included monetary and reputation losses It was important that the web service framework was able to provide a secure environment, so that the services were delivered even when facing attacks. The model was based on a set of phases that included the execution of a large number of well-known DoS attacks against the target framework and the classification of the observed behavior. Results showed that four out of the six frameworks tested were vulnerable to at least one type of DoS attacks and indicated that even very popular platforms required urgent security improvements.

VII. CONCLUSION

With the development of the web, the new sort of web applications showed up: mashups. These applications include the content from multiple sources, for example a housing rental website combines the information about the houses and maps them to Google maps. This gathering of security concerns can profit by existing techniques and methodologies of Web to deal with the security and trust issues. The semantic security worries then again, handle the data partaking in a more elevated amount by misusing the authoritative arrangements so as to portray the common information in a PC procedure capable way. In any case, Mashup applications, by their tendency, include communication between different page parts. Frequently these parts are stacked from various causes.

REFERENCES

[1] Wang, Helen J., Xiaofeng Fan, Jon Howell, and Collin Jackson, "Protection and communication abstractions for web browsers in Mashup OS", In ACM SIGOPS Operating Systems Review, Vol. 41, No. 6, pp. 1-16, 2007.

[2] Giffin, Daniel, Amit Levy, Deian Stefan, David Terei, David Mazieres, John Mitchell, and Alejandro Russo, "Hails: Protecting data privacy in untrusted web applications", Journal of Computer Security, pp. 1-35, 2012.

[3] Lawton, George, "Web 2.0 creates security challenges", Computer, Vol. 40, No. 10, 2007.

[4] Liu, Xuanzhe, Yi Hui, Wei Sun, and Haiqi Liang, "Towards service composition based on mashup", In Services IEEE Congress, pp. 332-339, 2007.

[5] Oppliger, Rolf, RuediRytz, and Thomas Holderegger, "Internet banking: Client-side attacks and protection mechanisms", Computer, Vol. 42, No. 6, 2009.

[6] Magazinius, Jonas, AslanAskarov, and Andrei Sabelfeld, "A lattice-based approach to mashup security", In Proceedings of the 5th ACM symposium on information, computer and communications security, pp. 15-23, 2010.

[7] Singh, Kapil, Alexander Moshchuk, Helen J. Wang, and Wenke Lee, "On the incoherencies in web browser access control policies", In Security and Privacy (SP), pp. 463-478, 2010.

[8] Patel, Ahmed, Samaher Al-Janabi, Ibrahim AlShourbaji, and Jens Pedersen, "A novel methodology towards a trusted environment in mashup web applications", Computers & Security, Vol.49, pp. 107-122, 2015.

[9] Oda, Terri, Glenn Wurster, Paul C. van Oorschot, and Anil Somayaji, "SOMA: Mutual approval for included content in web pages", In Proceedings of the 15th ACM conference on Computer and communications security, pp. 89-98, 2008.

[10] Felt, Adrienne, Pieter Hooimeijer, David Evans, and Westley Weimer, "Talking to strangers without taking their candy: isolating proxied content", In Proceedings of the 1st Workshop on Social Network Systems, pp. 25-30, 2008.

[11] Hashimoto, Gilberto Tadayoshi, Pedro Frosi Rosa, Edmo Lopes Filho, and Jayme Tadeu Machado, "A Security Framework to Protect Against Social Networks Services Threats", In Systems and Networks Communications (ICSNC), Fifth International Conference, pp. 189-194, 2010.

[12] Koschmider, Agnes, Victoria Torres, and Vicente Pelechano, "Elucidating the mashup hype: Definition, challenges, methodical guide and tools for mashups", In Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web at WWW, pp. 1-9, 2009.

[13] Kim, D. and Solomon, M.G., “Fundamentals of information systems security”, Jones & Bartlett Learning, 2016.

[14] Taivalsaari, Antero, and TommiMikkonen, "Mashups and modularity: Towards secure and reusable web applications", In Automated Software Engineering-Workshops, 23rd IEEE/ACM International Conference, pp. 25-33, 2008.

[15] Ankolekar, A., Krötzsch, M., Tran, T. and Vrandecic, D., “The two cultures: Mashing up Web 2.0 and the Semantic Web”, In Proceedings of the 16th international conference on World Wide Web, pp. 825-834, 2007.

[16] Bry, François, Sebastian Schaffert, Denny Vrandečić, and KlaraWeiand, "Semantic wikis: Approaches, applications, and perspectives", Reasoning Web, Semantic Technologies for Advanced Query Answering, pp. 329-369, 2012.

[17] Fensel, Dieter, Mick Kerrigan, and Michal Zaremba, "Implementing semantic web services", The SESA Framework 2008.

[18] Daniel, Florian, and Maristella Matera, “Mashups: Concepts, Models and Architectures” Springer, 2014.

[19] Mikkonen, Tommi, ArtoSalminen, and Antero Taivalsaari, "Enabling Global, Dynamic Web-Based Software Reuse--Mashware Revisited", In Software Engineering and Advanced Applications (SEAA), 40th EUROMICRO Conference, pp. 475-478, 2014.

[20] Akhawe, Devdatta, Adam Barth, Peifung E. Lam, John Mitchell, and Dawn Song, "Towards a formal foundation of web security", In Computer Security Foundations Symposium (CSF), 23rd IEEE, pp. 290-304, 2010

[21] Taivalsaari, Antero, TommiMikkonen, Dan Ingalls, and Krzysztof Palacz, "Web browser as an application platform: The lively kernel experience", 2008

[22] Gollmann, Dieter, "Computer security", Wiley Interdisciplinary Reviews: Computational Statistics, Vol. 2, No. 5, pp. 544-554, 2010.

[23] Stuttard, Dafydd, and Marcus Pinto, “The web application hacker's handbook: finding and exploiting security flaws”, John Wiley & Sons, 2011.

[24] Van Eeten, Michel J., and Johannes M. Bauer, "Economics of malware: Security decisions, incentives and externalities", OECD Science, Technology and Industry Working Papers, No. 1, pp. 0-1, 2008.

[25] Di Lorenzo, Giusy, Hakim Hacid, Hye-young Paik, and BoualemBenatallah, "Data integration in mashups", ACM Sigmod Record, Vol. 38, No. 1,pp. 59-66, 2009

[26] Jang-Jaccard, Julian, and Surya Nepal, "A survey of emerging threats in cyber security", Journal of Computer and System Sciences, Vol. 80, No. 5,pp. 973-993, 2014.

[27] Grier, Chris, Shuo Tang, and Samuel T. King, "Designing and implementing the op and op2 web browsers", ACM Transactions on the Web (TWEB), Vol. 5, No. 2,pp. 11, 2011

[28] Papazoglou, Mike P., and Willem-Jan Heuvel, "Service oriented architectures: approaches, technologies and research issues", The VLDB Journal—The International Journal on Very Large Data Bases, Vol. 16, No. 3,pp. 389-415, 2007.

[29] Daniel, Florian, and Maristella Matera,“Mashups: Concepts, Models and Architectures”, Springer, pp. 1-341, 2014.

[30] Costa, Cristiano Andre da, Adenauer Correa Yamin, and Claudio Fernando Resin Geyer, "Toward a general software infrastructure for ubiquitous computing", IEEE pervasive computing: mobile and ubiquitous systems, Los Alamitos, Vol. 7, No. 1, pp. 64-73, 2008.

[31] Guragai, Binod, Nicholas C. Hunt, Marc P. Neri, and Eileen Z. Taylor, "Accounting information systems and ethics research: Review, synthesis, and the future", Journal of Information Systems, Vol. 31, No. 2,pp. 65-81, 2015.

[32] Ayed, Samiha, Muhammad SabirIdrees, Nora Cuppens, and Frederic Cuppens, "Achieving dynamicity in security policies enforcement using aspects", International Journal of Information Security, PP. 1-21, 2017.

[33] Salas, M. I. P., and Eliane Martins, "Security testing methodology for vulnerabilities detection of xss in web services and ws-security", Electronic Notes in Theoretical Computer Science, Vol. 302, pp. 133-154, 2014.

[34] Fonseca, Jose, Marco Vieira, and Henrique Madeira "Evaluation of web security mechanisms using vulnerability & attack injection", IEEE Transactions on Dependable and Secure Computing, Vol. 11, No. 5, pp. 440-453, 2014

[35] Lv, Junjie, Yuanzhuo Wang, Jingyuan Li, Kun Meng, and Chuang Lin, "Security analysis for Web service behaviors based on hierarchical stochastic game model", Chinese Journal of Electronics, Vol. 24, No. 3, pp. 449-454, 2015.

[36] Oliveira, Rui André, NunoLaranjeiro, and Marco Vieira, "Assessing the security of web service frameworks against Denial of Service attacks", Journal of Systems and Software, Vol. 109, pp. 18-31, 2015.