# The Study into Cross-Site Request Forgery Attacks within the Framework of Analysis of Software Vulnerabilities

[1] *A.V. Barabanov <ab@cnpo.ru>*
[1] *A.I. Lavrov <mail@cnpo.ru>*
[2] *A.S. Markov <a.markov@bmstu.ru>*
[1] *I.A. Polotnyanschikov <mail@cnpo.ru>*
[2] *V.L. Tsirlov <v.tsirlov@bmstu.ru>*
[1] *NPO Echelon, Elektrozavodskaya street, 24, Moscow, 107023, Russia*
[2] *Bauman MSTU, 2 Baumanskaya street, 5, Moscow,105005 Russia*

**Abstract**. Nowadays, web applications are one of the most popular types of target of evaluation within the framework of the information security certification. The relevance of the study of web applications vulnerabilities during information security certification is due to the fact that web technologies are actively used while producing modern information systems, including information systems critical from the information security point of view, and on the other hand carrying out basic attacks on such information systems do not require violators of high technical competence, since data on typical vulnerabilities and attacks, including the attacking tools are heavily represented in publicly available sources of information, and the information systems themselves are usually available from public communication networks. The paper presents the results of a study of the security of web applications that are target of evaluation within the framework of certification for information security requirements against cross-site requests forgery attacks. The results of systematization and generalization of information about the cross-site requests forgery attacks and security controls used by web application developers are presented. The results of experimental studies of 10 web applications that have passed certification tests against information security requirements are presented. The results of experimental studies have shown that most developers do not pay enough attention to protection from cross-site request forgery attack - 7 out of 10 web applications tested have been vulnerable to this type of attack. Based on the results of processing the results of experimental studies, the distribution of security controls used in web applications and identified vulnerabilities by programming languages were obtained. Recommendations regarding the protection of web applications against cross-site request forgery attack for developers planning to certify their software are formulated.

**Keywords:** information security; software security; analysis of vulnerabilities; web-application; CSRF-attack.

**DOI**: 10.15514/ISPRAS-2017-29(5)-1

**For citation:** Barabanov A.V., Lavrov A.I., Markov A.S., Polotnyanschikov I.A., Tsirlov V.L The study into cross-site request forgery attacks within the framework of analysis of software vulnerabilities. Trudy ISP RAN/Proc. ISP RAS, vol. 29, issue 5, 2017. pp. 7-18. DOI: 10.15514/ISPRAS-2017-29(5)-1

## 1. Introduction

Software created with the use of web-technologies is currently one of the main components in automated control system (ACS) design. The designed ACS are, as a rule, multi-user and can be found on public domain networks (for instance, Internet), which increases the risk of their successful attack. Various procedures (such as certification, independent security audit) are currently used to lower probability of successful attack. They are aimed at identifying vulnerabilities in the software used to design ACS [1, 2].

Software vulnerabilities are analyzed during certification for compliance with the requirements to the protection profiles approved by FSTEC of Russia (Federal Service for Technology and Export Control), which clearly includes requirements of AVA_VAN assurance family “Vulnerability analysis”, and during testing for compliance with the requirements of the technical specifications and classic governing documents of FSTEC of Russia. The procedure for vulnerability analysis recommended by FSTEC of Russia consists in the joint use of approaches specified in the Common Methodology for Information Technology Security Evaluation and ISO/IEC TR 20004 [3]. It should be noted that more specific instructions for the test laboratories (for instance, standard penetration tests) have not yet been developed, which makes this procedure non-determined [4].

The experience of analysis into vulnerabilities of web-applications within the framework of the accredited test laboratory showed that Cross-Site Request Forgery attack, hereinafter – CSRF-attack is currently the most successful attack against targets of evaluation. The main attention of the developers of web-applications, as a rule, is concentrated on implementing measures protecting against attacks like SQL-injections or Cross-site scripting. The situation is aggravated by the fact that measures protecting against CSRF-attacks are still being actively studied, and best practices have not been rigidly registered yet [2, 6].

The goal of this work consisted in developing guidelines for the developers of web-applications, who are planning to certify their solutions as to the information security requirements. The work solves the following tasks to achieve the set goal:

a) Classification and summary of information about CSRF-attacks and measures of protection against them;

b) Consolidation of information about vulnerabilities of web-applications identified within the framework of work of the accredited test laboratories.

## 2. The results of classification and summary of information about CSRF-attacks and relevant security measures

A hacker performing a CSRF-attack makes the web-browser used by the legal user, who has been authenticated in "security measures against " the attacked web-application, send HTTP-request, which is going to be identified by the application as a request received from a legal user, to the web-application.

A possible consequence from a successful CSRF-attack implementation is running of an arbitrary code in the web-application in the name of authenticated user. Thus, the main causes of CSRF-attacks are vulnerabilities in web-applications related to wrong implementation of algorithm of HTTP-request authorization. Success of CSRF-attack is determined by the following factors [7, 8]:

- The browser automatically applies authentication data of the user (for instance, session cookie-files), when sending HTTP-request to the web-application;
- Web-application uses the obtained authentication data to authorize the action required for performance by HTTP-request.

It should be noted that despite difficulties in implementation, there are cases of successful CSRF-attacks of ‘Login’ and ‘Logout’ type on web-applications [1, 9, 10]. The probability of successful ‘stored’ CSRF-attack is higher, because a malicious code is stored on the side of the attacked web-application, and the hacker does not have to make the user (for instance, using methods of social engineering) go to a special resource with a malicious code.

Implementation of the security measures on the client’s side [11-16], represented by plugins/extensions of the browser or additional software (proxy), has significant drawbacks [8] and is currently only of academic interest.

There are suggestion on implementing security measures directly with the browser source code, for instance, using ‘samesite’ properties of the cookie-files, but currently these measures are experimental and are implemented only in certain browsers. Integrated measures (measures implemented jointly by the software code on the client- and the server-sides), as a rule, implement a certain information control policy [6, 17], which contain critical information (for instance, authentication data), between the browser and the web-server. It should be noted that effective implementation of this type of security measures is possible by making changes in the browser source code. Moreover, essential limitations of these security measures are well-known, which does not allow their use as a sole measure of protection.

The most popular security measures against CSRF-attacks are tokens (synchronic tokens or generated using HMAC cryptographic function) that are generated and checked on the web-application side. This security measure is implemented, as a rule, by the web-application itself or the framework. It should be noted that the majority of the most popular frameworks (such as, Ruby on Rails, ASP.NET, Django) implement this measure, which somewhat decreases the workload for the developer of a certain web-application and reduces the number of errors related to implementation of the security algorithm by the developer of the web-application.

The main distinctive feature of the token-based security measures is in the token storage method:

- Generated token may be stored on the web-application side (it is associated with the user session) and it shall be compared with the token received from the web-browser;
- Generated token may be stored on the web-browser side (for instance, in the cookie); when the web-application receives a request from the web-browser, the web-application compares the values of tokens in the cookie and the HTTP-request body.

It should be noted that this measure of the web-application security is used correctly, if it is designed and implemented in a way that HTTP-requests of GET type do not change the server state, and are used only for request of the necessary information. AJAX-requests may be protected with tokens inserted in HTTP-header, or custom HTTP-headers (during implementation of this security measure the web-application only checks availability of the heading in the received request).

The leading specialists in the web-application security recommend using the defense in depth principle, when implementing security measures. Thus, specialists of OWASP community recommend implementing security of the web-application by combining two types of the security measures –HTTP-headers verification and tokens.

In some cases, the developers use three or more security measures for critical information systems (for instance, online banking systems). For example, it can be a combination of tokens, verification of HTTP-header and security measures that require actions from the end user, who performs a critical operation (entry of one-time code/ password).

## 3. Methods and results of the study

The study into the security level of the web-application was carried out in the accredited test laboratory of NPO Echelon (study period: January – November 2016). Brief information about the web-applications that participate in the study is represented in Table 1.

*Table I. Brief information about the study objects*

| Software identifier | Programming language | Type of developer | Level of measures for secure software development implementation (maturity level) |
|---|---|---|---|
| Software No. 1 | PHP | Russian | 2 |

| | | | |
|---|---|---|---|
| Software No. 2 | Java | Foreign | 5 |
| Software No. 3 | PHP | Russian | 1 |
| Software No. 4 | Java | Foreign | 5 |
| Software No. 5 | C# | Russian | 4 |
| Software No. 6 | Java | Russian | 1 |
| Software No. 7 | C# | Russian | 1 |
| Software No. 8 | PHP | Russian | 1 |
| Software No. 9 | Ruby | Russian | 3 |
| Software No. 10 | Ruby | Russian | 3 |

Level of measures for secure software development implementation (maturity level) was assessed by the expert method with account of the scope of measures implemented by the developer of measures from the basic set of measures for developing secure software suggested in the National Standard GOST R 56939-2016 Information Protection. Secure Software Development. General Requirements. [4, 18]: 1 - not one measure is implemented, 2 - less than 20% of measures is implemented, 3 – from 20% to 40% of measures is implemented, 4 - from 40% to 60% of measures is implemented, 5 - from 60% to 80% of measures is implemented, 6 - over 80% of measures is implemented.

Vulnerabilities were analysed using standard tests developed with account of recommendations and CAPEC resource. Below is the general sequence of the performed tests.

1) Analysis of parts of web-applications (pages), which allow changing the state of the web-application (creating/ changing/ deleting user accounts, protected information, other information etc.).

2) Study of the requests to the identified parts of web-applications: transmission of the requests from the web-browser to the web-application with further interception and analysis of the request structure. The expert analyses the intercepted request and defines the type of security measure against CSRF-attack on a specific page.

3) Generating a mock HTTP-request, which is saved as an HTML-file on the local computer and is opened in the web-browser, provided that there is a session authenticated by the target of evaluation (web-application).

4) If the analysis of intercepted request (cl. 2) revealed security measures against CSRF-attacks, the following actions shall be additionally taken:

a) when tokens are used as a security measure:

- analysis of URL for a presence of token in a plain text;
- sending a request without a token;
- sending a request with an altered token;

- sending a request using one token for various user accounts;
- an attempt to guess /select a token;

b) when using verification of the HTTP-headers as a security measure:

- sending a request with altered HTTP Referer (originally a misspelling of «referrer»)/Origin fields;
- sending a request without HTTP Referrer/Origin fields.

The tests were performed using the following software: BurpSuite software, Scaner-VS software. The average time spent on testing of one web-application by one expert of the test laboratory is 8 hours.

The results of the study are specified below.

1) CSRF-attacks were successful in 70% of cases – 7 out of 10 analysed web-applications turned out to be vulnerable.

2) The majority of CSRF-attacks were successful in relation to web-applications developed in Russia. It should be noted that the only CSRF-attack that was successful in relation to the foreign web-application was that of "Logout" type, and the experts of the test laboratory failed to develop an attack vector that implements information security threat. Only one web-application initially did not have any security measures against CSRF-attacks. The other vulnerable web-applications had security measures based on verification of HTTP-headers or token (Figure 1).

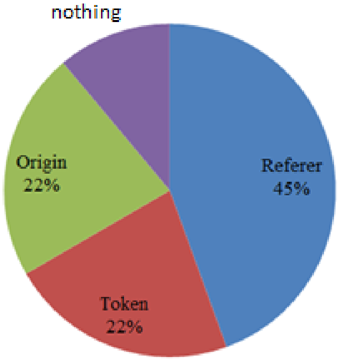

*Fig. 1. Distribution of protection measures used in vulnerable web-applications*

3) It has been established that web-applications written in PHP have a few more vulnerabilities that results in successful CSRF-attacks (Figure 2) [20].

4) The developers upgraded vulnerable web-applications using security measures based on tokens in all cases.

5) In the majority of cases the upgraded web-application and web-applications, where the vulnerability has not been identified, used a combination of several security measures against CSRF-attacks.

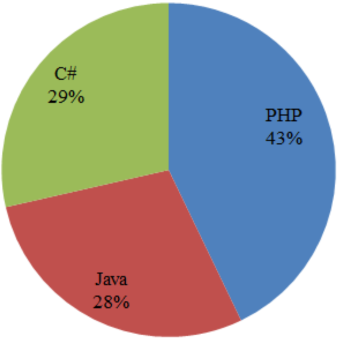

*Fig. 2. Distribution of identified vulnerabilities as to the programming language*

6) The average time required for the web-application developer to correct vulnerability is 3 weeks.

7) One of the results of the study was a deduced empirical rule, in accordance with which the number of vulnerabilities identified in the software is in inverse proportion to the maturity level of the secure software development processes implemented by the developer.

## 4. Recommendations to developers on increasing the security level of web-applications

Based on the results of the study the following recommendations were provided for the developers of web-applications that are planning to hold certification tests as to information safety requirements.

1) It is advisable that the developers implement measures for secure software development in the software lifecycle processes. At the very least, it is recommended to implement measures related to testing penetration of web-application prior to their submission to the test laboratory. To minimize time for such testing, the developers should generate sets of standard tests, which may be developed with account of guidelines represented in the works [17, 19]. The developers are advised against limiting their tests to the standard test only, and are recommended to run additional tests aimed at performing CSRF-attacks, like ‘Login’ and ‘Logout’, and verify that the selected security measure is correctly implemented.

2) The developers are recommended using the defense in depth principle – combine two or more security measures (as a rule, verification of token and HTTP-headers), when implementing security measures against CSRF-attacks in the web-application.

3) When implementing security measures against CSRF-attacks in the web-application, the developers are, first of all, recommended to use security measures

that are already implemented in the operational environment, for instance, frameworks.

## *5. Conclusions*

This work consisted in the study into security of web-applications, which are the test targets within the framework of certification as to information security requirements, against cross-site request forgery attacks. The result showed that the majority of the developers (around 70%) do not pay due attention to implementing security measures against such attacks. Resulting from the study, we defined recommendations for the developers, the main of them being recommendations on the use of defense in depth principle and the use of token-based security measures that had already been implemented by the framework developers.  We deduced empirical rule, in accordance with which the number of vulnerabilities identified in the software is in inverse proportion to the maturity level of the secure software development processes implemented by the developer. Further studies are intended into the issues of the web-application protection against SQL-injection attacks and cross-site scripting attack and defining general guidelines for the developers of web-applications, who are planning certification.

## References

[1]. H. Selim, S. Tayeb, Y. Kim, J. Zhan, and M. Pirouz. Vulnerability Analysis of Iframe Attacks on Websites. In Proceedings of the The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016, Data Science 2016 (MISNC, SI, DS 2016). ACM, New York, NY, USA, Article 45, pp. 1-6, August 2016. DOI: 10.1145/2955129.2955180.

[2]. W. Du, K. Jayaraman, X. Tan, T. Luo, and S. Chapin. Position paper: why are there so many vulnerabilities in web applications? In Proceedings of the 2011 New Security Paradigms Workshop (NSPW '11). ACM, New York, NY, USA, pp. 83-94. 2011. DOI: 10.1145/2073276.2073285.

[3]. A. Barabanov, A. Markov, A. Fadin, V. Tsirlov, I. Shakhalov. Synthesis of Secure Software Development Controls. In Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russia, September 8-10, 2015). SIN '15. ACM, New York, NY, USA, pp. 93-97. 2015. DOI: 10.1145/2799979.2799998.

[4]. A.V. Barabanov, A.S. Markov, V.L. Tsirlov. Methodological Framework for Analysis and Synthesis of a Set of Secure Software Development Controls. Journal of Theoretical and Applied Information Technology. 2016. V. 88. No 1, pp. 77-88.

[5]. N. Jovanovic, E. Kirda, and C. Kruegel. Preventing cross site request forgery attacks. In the IEEE International Conference on Security and Privacy for Emerging Areas in Communication Networks (Securecomm) , pp. 1–10, September 2006.

[6]. A. Czeskis, A. Moshchuk, T. Kohno, and H.J. Wang. Lightweight server support for browser-based CSRF protection. In Proceedings of the 22nd international conference on World Wide Web (WWW '13). ACM, New York, NY, USA, 2013, pp. 273-284. DOI: 10.1145/2488388.2488413.

[7]. K. Jayaraman, P. G. Talaga, G. Lewandowski, S.J. Chapin, and M. Hafiz. Modeling user interactions for (fun and) profit: preventing request forgery attacks on web applications. In Proceedings of the 16th Conference on Pattern Languages of Programs (PLoP '09). ACM, New York, NY, USA, Article 16, pp. 1-9. August 2009. DOI: 10.1145/1943226.1943246.

[8]. A. Barth, C. Jackson, and J.C. Mitchell. Robust defenses for cross-site request forgery. In Proceedings of the 15th ACM conference on Computer and communications security (CCS '08). ACM, New York, NY, USA, pp. 75-88. October 2008. DOI: 10.1145/1455770.1455782.

[9]. M. Zhou, P. Bisht, and V.N. Venkatakrishnan. Strengthening XSRF defenses for legacy web applications using whitebox analysis and transformation. In Proceedings of the 6th international conference on Information systems security (ICISS'10), pp. 96-110. 2010.

[10]. E. Shernan, H. Carter, D. Tian, P. Traynor, and K. Butler. More Guidelines Than Rules: CSRF Vulnerabilities from Noncompliant OAuth 2.0 Implementations. In Proceedings of the 12th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2015), pp. 239-260, June 2015. DOI: 10.1007/978-3-319-20550-2_13

[11]. H. Shahriar and M. Zulkernine. Client-Side Detection of Cross-Site Request Forgery Attacks. In Proceedings of the 2010 IEEE 21st International Symposium on Software Reliability Engineering (ISSRE '10). IEEE Computer Society, Washington, DC, USA, pp. 358-367. November 2010. DOI: 10.1109/ISSRE.2010.12.

[12]. P.D. Ryck, L. Desmet, T. Heyman, F. Piessens, and W. Joosen. CsFire: transparent client-side mitigation of malicious cross-domain requests. In Proceedings of the Second international conference on Engineering Secure Software and Systems (ESSoS'10), pp. 18-34. 2010. DOI: 10.1007/978-3-642-11747-3_2.

[13]. R. Pelizzi and R. Sekar. A server- and browser-transparent CSRF defense for web 2.0 applications. In Proceedings of the 27th Annual Computer Security Applications Conference (ACSAC '11). ACM, New York, NY, USA, pp. 257-266. December 2011. DOI: 10.1145/2076732.2076768.

[14]. L. Xing, Y. Zhang, and S. Chen. A client-based and server-enhanced defense mechanism for cross-site request forgery. In Proceedings of the 13th international conference on Recent advances in intrusion detection (RAID'10), pp. 484-485. 2010.

[15]. N. Gelernter and A. Herzberg. Cross-Site Search Attacks. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15). ACM, New York, NY, USA, pp. 1394-1405. October 2015. DOI: 10.1145/2810103.2813688.

[16]. E. Z. Yang, D. Stefan, J. Mitchell, D. Mazières, P. Marchenko, and B. Karp. Toward principled browser security. In Proceedings of the 14th USENIX conference on Hot Topics in Operating Systems (HotOS'13). USENIX Association, Berkeley, CA, USA, pp. 17-17. 2013.

[17]. W. Maes, T. Heyman, L. Desmet, and W. Joosen. Browser protection against cross-site request forgery. In Proceedings of the first ACM workshop on Secure execution of untrusted code (SecuCode '09). ACM, New York, NY, USA, pp. 3-10. November 2009. DOI: 10.1145/1655077.1655081.

[18]. A. Barabanov, A. Markov, V. Tsirlov. Procedure for substantiated development of measures to design secure software for automated process control systems. In Proceedings of the International Siberian Conference on Control and Communications, SIBCON 2016, IEEE, 1-4. June 2016. DOI: 10.1109/SIBCON.2016.7491660.

[19]. X. Li and Y.Xue. A survey on server-side approaches to securing web applications. ACM Comput. Surv., 46, 4, Article 54 (March 2014), 29 pages. April 2014. DOI: 10.1145/2541315

[20]. A.S. Markov, V.L. Tsirlov. Experience in identifying vulnerabilities in foreign software products. Voprosy kiberbezopasnosti [Cybersecurity Issues]. 2013. No 1(1), pp. 42-48. (In Russian).

# Исследование атак типа «Cross-Site Request Forgery» в рамках проведения анализа уязвимостей веб-приложений

[1] *А.В. Барабанов < ab@cnpo.ru>*
[1] *А.И. Лавров < mail@cnpo.ru>*
[2] *А.С. Марков < a.markov@bmstu.ru>*
[1] *И.А. Полотнянщиков < mail@cnpo.ru>*
[2] *В.Л. Цирлов <v.tsirlov@bmstu.ru>*
[1] *НПО «Эшелон», 107023, Россия, г. Москва, ул. Электрозаводская, д.24*
[2] *МГТУ им. Н.Э. Баумана,*
*105005, Россия, г. Москва, 2-я Бауманская ул., д. 5, стр. 1*

**Аннотация.** Веб-приложения являются одним из наиболее распространенных типов объектов исследования в рамках работы системы сертификации средств защиты информации. Актуальность исследования уязвимостей в веб-приложениях в рамках сертификации по требованиям безопасности информации обусловлена тем, что веб-технологии, с одной стороны, активно используются при реализации современных информационных систем, в том числе критичных с точки зрения информационной безопасности, а, с другой стороны, проведение базовых атак на подобные информационные системы не требуют от нарушителей высокой технической компетентности, поскольку данные о типовых уязвимостях и атаках, включая инструментальные средства проведения атак, в большом объеме представлены в общедоступных источниках информации, а сами информационные системы, как правило, доступны из сетей связи общего пользования. В работе представлены результаты исследования защищённости веб-приложений, являющихся объектами испытаний в рамках сертификации по требованиям безопасности информации, от атак типа «межсайтовая подделка запросов». Приведены результаты систематизации и обобщения сведений об атаке типа «межсайтовая подделка запросов» и мерах защиты, используемых разработчиками веб-приложений. Представлены результаты экспериментальных исследований 10 веб-приложений, которые проходили сертификационные испытания по требованиям безопасности информации. Результаты экспериментальных исследований показали, что большинство разработчиков не уделяют должного внимания защите от межсайтовой подделки запросов – 7 из 10 исследованных веб-приложений оказали уязвимыми к данному типу атаки. По результатам обработки результатов экспериментальных исследований получены распределения мер защиты, используемых в веб-приложениях, и выявленных уязвимостей по языкам программирования. Сформулированы рекомендации в части защиты веб-приложений от межсайтовой подделки запросов для разработчиков, планирующих проведение сертификации своего программного обеспечения.

**Ключевые слова:** информационная безопасность; безопасное программное обеспечение; анализ уязвимостей; веб-приложение; межсайтовая подделка запроса.

**DOI:** 10.15514/ISPRAS-2017-29(5)-1

**Для цитирования:** Барабанов А.В., Лавров А.И., Марков А.С., Полотнянщиков И.А., Цирлов В.Л. Исследование атак типа «Cross-Site Request Forgery» в рамках проведения анализа уязвимостей веб-приложений. Труды ИСП РАН, том 29, вып. 5, 2017 г., стр. 7-18 (на английском языке). DOI: 10.15514/ISPRAS-2017-29(5)-1

## Список литературы

[1]. H. Selim, S. Tayeb, Y. Kim, J. Zhan, and M. Pirouz. Vulnerability Analysis of Iframe Attacks on Websites. In Proceedings of the The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016, Data Science 2016 (MISNC, SI, DS 2016). ACM, New York, NY, USA, Article 45, pp. 1-6, August 2016. DOI: 10.1145/2955129.2955180.

[2]. W. Du, K. Jayaraman, X. Tan, T. Luo, and S. Chapin. Position paper: why are there so many vulnerabilities in web applications? In Proceedings of the 2011 New Security Paradigms Workshop (NSPW '11). ACM, New York, NY, USA, pp. 83-94. 2011. DOI: 10.1145/2073276.2073285.

[3]. A. Barabanov, A. Markov, A. Fadin, V. Tsirlov, I. Shakhalov. Synthesis of Secure Software Development Controls. In Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russia, September 8-10, 2015). SIN '15. ACM, New York, NY, USA, pp. 93-97. 2015. DOI: 10.1145/2799979.2799998.

[4]. A.V. Barabanov, A.S. Markov, V.L. Tsirlov. Methodological Framework for Analysis and Synthesis of a Set of Secure Software Development Controls. Journal of Theoretical and Applied Information Technology. 2016. V. 88. No 1, pp. 77-88.

[5]. N. Jovanovic, E. Kirda, and C. Kruegel. Preventing cross site request forgery attacks. In the IEEE International Conference on Security and Privacy for Emerging Areas in Communication Networks (Securecomm) , pp. 1–10, September 2006.

[6]. A. Czeskis, A. Moshchuk, T. Kohno, and H.J. Wang. Lightweight server support for browser-based CSRF protection. In Proceedings of the 22nd international conference on World Wide Web (WWW '13). ACM, New York, NY, USA, 2013, pp. 273-284. DOI: 10.1145/2488388.2488413.

[7]. K. Jayaraman, P. G. Talaga, G. Lewandowski, S.J. Chapin, and M. Hafiz. Modeling user interactions for (fun and) profit: preventing request forgery attacks on web applications. In Proceedings of the 16th Conference on Pattern Languages of Programs (PLoP '09). ACM, New York, NY, USA, Article 16, pp. 1-9. August 2009. DOI: 10.1145/1943226.1943246.

[8]. A. Barth, C. Jackson, and J.C. Mitchell. Robust defenses for cross-site request forgery. In Proceedings of the 15th ACM conference on Computer and communications security (CCS '08). ACM, New York, NY, USA, pp. 75-88. October 2008. DOI: 10.1145/1455770.1455782.

[9]. M. Zhou, P. Bisht, and V.N. Venkatakrishnan. Strengthening XSRF defenses for legacy web applications using whitebox analysis and transformation. In Proceedings of the 6th international conference on Information systems security (ICISS'10), pp. 96-110. 2010.

[10]. E. Shernan, H. Carter, D. Tian, P. Traynor, and K. Butler. More Guidelines Than Rules: CSRF Vulnerabilities from Noncompliant OAuth 2.0 Implementations. In Proceedings of the 12th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2015), pp. 239-260, June 2015. DOI: 10.1007/978-3-319-20550-2_13

[11]. H. Shahriar and M. Zulkernine. Client-Side Detection of Cross-Site Request Forgery Attacks. In Proceedings of the 2010 IEEE 21st International Symposium on Software Reliability Engineering (ISSRE '10). IEEE Computer Society, Washington, DC, USA, pp. 358-367. November 2010. DOI: 10.1109/ISSRE.2010.12.

[12]. P.D. Ryck, L. Desmet, T. Heyman, F. Piessens, and W. Joosen. CsFire: transparent client-side mitigation of malicious cross-domain requests. In Proceedings of the Second international conference on Engineering Secure Software and Systems (ESSoS'10), pp. 18-34. 2010. DOI: 10.1007/978-3-642-11747-3_2.

[13]. R. Pelizzi and R. Sekar. A server- and browser-transparent CSRF defense for web 2.0 applications. In Proceedings of the 27th Annual Computer Security Applications Conference (ACSAC '11). ACM, New York, NY, USA, pp. 257-266. December 2011. DOI: 10.1145/2076732.2076768.

[14]. L. Xing, Y. Zhang, and S. Chen. A client-based and server-enhanced defense mechanism for cross-site request forgery. In Proceedings of the 13th international conference on Recent advances in intrusion detection (RAID'10), pp. 484-485. 2010.

[15]. N. Gelernter and A. Herzberg. Cross-Site Search Attacks. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15). ACM, New York, NY, USA, pp. 1394-1405. October 2015. DOI: 10.1145/2810103.2813688.

[16]. E. Z. Yang, D. Stefan, J. Mitchell, D. Mazières, P. Marchenko, and B. Karp. Toward principled browser security. In Proceedings of the 14th USENIX conference on Hot Topics in Operating Systems (HotOS'13). USENIX Association, Berkeley, CA, USA, pp. 17-17. 2013.

[17]. W. Maes, T. Heyman, L. Desmet, and W. Joosen. Browser protection against cross-site request forgery. In Proceedings of the first ACM workshop on Secure execution of untrusted code (SecuCode '09). ACM, New York, NY, USA, pp. 3-10. November 2009. DOI: 10.1145/1655077.1655081.

[18]. A. Barabanov, A. Markov, V. Tsirlov. Procedure for substantiated development of measures to design secure software for automated process control systems. In Proceedings of the International Siberian Conference on Control and Communications, SIBCON 2016, IEEE, 1-4. June 2016. DOI: 10.1109/SIBCON.2016.7491660.

[19]. X. Li and Y.Xue. A survey on server-side approaches to securing web applications. ACM Comput. Surv., 46, 4, Article 54 (March 2014), 29 pages. April 2014. DOI: 10.1145/2541315

[20]. Марков. А.С., Цирлов В.Л. Опыт выявления уязвимостей в зарубежных программных продуктах. Вопросы кибербезопасности, 2013, № 1 (1), стр. 42-48