ИССЛЕДОВАНИЕ АТАК ТИПА «МЕЖСАЙТОВАЯ ПОДДЕЛКА ЗАПРОСОВ»

Барабанов А.В.[1], Лавров А.И.[2], Марков А.С.[3], Полотнянщиков И.А.[4]

Представлено исследование защищённости веб–приложений, являющихся объектами испытаний в рамках сертификации по требованиям безопасности информации, от атак типа «межсайтовая подделка запросов». В работе выполнена систематизация и обобщение сведений о подобных атаках и мерах защиты от них. Представлены результаты исследования, демонстрирующие распределение выявленных уязвимостей по типу разработчика, распределение мер защиты, используемых в веб–приложениях, распределение выявленных уязвимостей по языкам программирования, данные о количестве мер защиты, используемых в исследованных веб–приложениях. По результатам исследования установлено, что в большинстве случаев разработчики веб–приложений не уделяют должного внимания защите от атак типа «межсайтовая подделка запросов». Сформулированы рекомендации разработчикам, планирующим проведение сертификации в отношении своего программного обеспечения.

***Ключевые слова**: защита информации, анализ уязвимостей, веб–приложение, CSRF–атака.*

DOI:10.21581/2311-3456-2016-5-43-50

Введение

По данным Государственного реестра сертифицированных средств защиты информации (система сертификации ФСТЭК России), более половины сертификатов соответствия оформлены на веб-приложения. Это связано в первую очередь с тем, что веб-технологии в настоящее время активно используются при реализации многопользовательских информационных системы с возможностью удаленного взаимодействия (например, Единый портал государственных и муниципальных услуг). Поскольку эти информационные системы, как правило, доступны из сетей общего пользования (сеть «Интернет»), а информация о типовых уязвимостях и атаках на программное обеспечение (ПО), созданное с использованием веб-технологий, в большом объеме представлена в общедоступных источниках информации, вопросы анализа уязвимостей при приведении сертификационных испытаний данного ПО являются актуальными. Анализ уязвимостей ПО выполняется как при сертификации на соответствие требованиям утвержденных ФСТЭК России профилей защиты, в которых в явном виде включены требования семейства доверия AVA «Анализ уязвимостей», так и при испытаниях на соответствие требованиям технических условий и классических руководящих документов ФСТЭК России. Методология анализа уязвимостей, рекомендуемая ФСТЭК России, заключается в совместном использовании подходов, изложенных в Общей методологии оценки (ГОСТ Р ИСО/МЭК 18045) и международном стандарте ISO/IEC TR 20004 [1, 2]. Следует отметить, что более конкретных указаний для испытательных лабораторий (например, в виде типовых тестов проникновения) в настоящее время не существует. При выполнении анализа уязвимостей веб-приложений эксперты испытательных лабораторий, как правило, выполняют тесты, связанные с типовыми атаками на веб-приложения, например, SQL-инъекция, межсайтовая подделка запросов, межсайтовое выполнение скриптов и пр. [3]. Опыт проведения анализа уязвимостей веб-приложений в рамках работы аккредитованной испытательной лаборатории показал, что межсайтовая подделка запросов (Cross-Site Request Forgery, далее по тексту – CSRF-атака) в настоящее время является наиболее успешной атакой, выполняемой в отношении объектов сертификации. Основное внимание разработчиков веб-приложений, как правило, сосредоточено на реализации мер защиты от атак типа SQL-инъекция или межсайтовое выполнение скриптов. Ситуация усугубляется тем, что меры защиты от CSRF-атаки все еще активно изучается, а «лучшие практики» в настоящее время четко не зафиксированы [4].

1 Барабанов Александр Владимирович, кандидат технических наук, ЗАО «НПО «Эшелон», Москва, ab@cnpo.ru

2 Лавров Артём Игоревич, ЗАО «НПО «Эшелон», г. Москва, al@cnpo.ru

3 Марков Алексей Сергеевич, доктор технических наук, старший научный сотрудник, Финансовый университет при Правительстве РФ, ASMarkov@fa.ru

4 Полотнянщиков Иван Александрович, ЗАО «НПО «Эшелон», г. Москва, ipo@cnpo.ru

***Рис.1**. Результаты систематизации информации о CSRF-атаках*

Цель данной работы состояла в выработке рекомендаций для разработчиков веб-приложений, планирующих сертифицировать свои решения по требованиям безопасности информации. Для достижения поставленной цели в работе были решены задачи систематизации и обобщения сведений о CSRF-атаках и мерах защиты от них, выполнено обобщение информации об уязвимостях веб-приложений, выявленных в рамках работы аккредитованной испытательной лаборатории.

Систематизация и обобщение сведений о CSRF-атаках и мерах защиты от них

При выполнении CSRF-атаки нарушитель вынуждает веб-браузер, используемый аутентифицированным в веб-приложении пользователем, отправить незаметно для него HTTP-запрос веб-приложению, который будет обработан веб-приложением как легальный. Возможным последствием от успешной CSRF-атаки является выполнение произвольного кода на стороне веб-приложения от имени аутентифицированного пользователя. Таким образом, основной причиной CSRF-атак является наличие уязвимостей в веб-приложениях, связанных с неверной реализацией алгоритма авторизации HTTP-запросов. Успешность CSRF-атаки обусловлена следующими факторами [5]: (1) браузер автоматически прикладывает аутентификационные данные (например, сессионные cookie-файлы) при отправке HTTP-запроса веб-приложению; (2) веб-приложение использует полученные аутентификационные данные для авторизации действия, требуемого к выполнению HTTP-запросом.

В ходе проведенного исследования был выполнен анализ, систематизация и обобщение информации о CSRF-атаках и мерах защиты от них. Были получены классификации CSRF-атак (рис. 1) и мер защиты от них (рис. 2) по различным признакам.

Следует отметить, что, несмотря на сложность в реализации, в настоящее время известны факты успешных CSRF-атак типа «Login» и «Logout» на веб-приложения [6]. Вероятность успешного выполнения CSRF-атаки типа «сохраненная» выше, поскольку вредоносный код хранится на атакуемом веб-сайте, и нарушителю не требуется вынуждать пользователя (например, с использованием методов социальной инженерии) переходит на специальный ресурс с вредоносным кодом.

Реализация мер защиты на стороне клиента [7, 8], выполняемая в форме плагинов/расширений браузера или дополнительного ПО (прокси), обладает существенными недостатками [6] и в настоящее время представляет только академический интерес. Известны предложения по реализации мер защиты непосредственно программным кодом браузера, например использование свойства «samesite» cookie-файлов[5], но в настоящее время эти меры являются экспериментальными и реализованы только в некоторых браузерах. Комбинированные меры (меры, реализуемые совместно программным кодом на стороне клиента и сервера), как правило, реализуют ту или иную политику контроля информационных потоков [5, 9], содержащих критичную информацию (на-

5 https://tools.ietf.org/html/draft-west-first-party-cookies-07

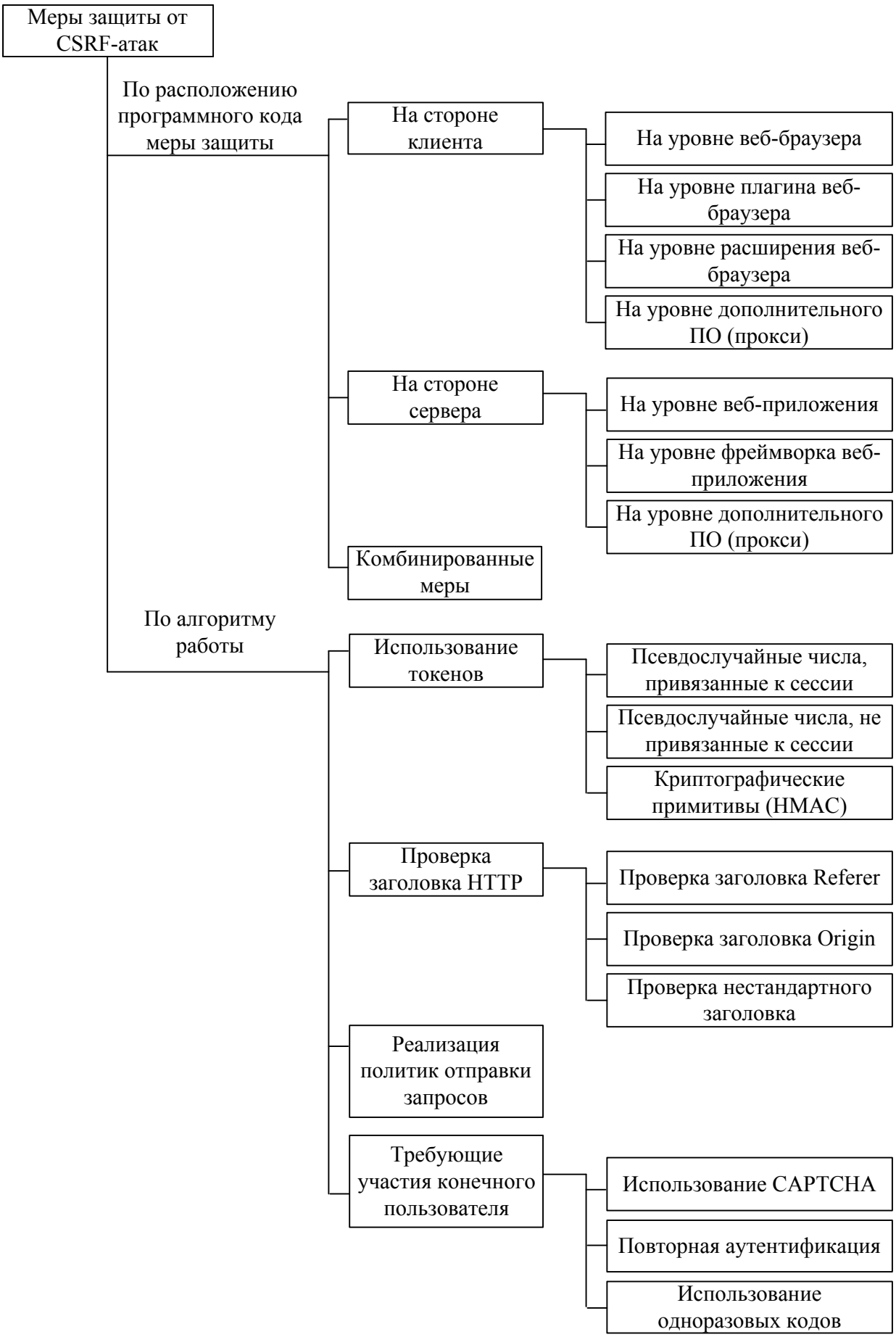

***Рис.2**. Результаты систематизации информации о мерах защиты от CSRF-атак*

пример, аутентификационные данные), между браузером и веб-сервером. Стоит отметить, что эффективная реализация мер защиты данного типа возможна при внесении изменений в программный код браузера. Кроме этого известны принципиальные ограничения мер защиты данного типа, что не позволяет использовать их в качестве единственной меры защиты.

Наиболее популярной мерой защиты от CSRF-атак является использование токенов (синхронных или генерируемых с использованием HMAC), генерируемых и проверяемых на стороне сервера. Эта мера защиты реализуется, как правило, самим веб-приложением или фреймворком. Следует отметить, что большинство популярных фреймворков (например, Ruby on Rails, ASP.NET, Django) реализуют данную меру, что несколько снижает нагрузку на разработчика конкретного веб-приложения и уменьшает количество ошибок, связанных с самостоятельной реализацией алгоритма защиты разработчиком веб-приложения.

Вместе с тем следует отметить, что ведущими специалистами в области безопасности веб-приложений рекомендуется использовать принцип эшелонированной обороны при внедрении мер защиты. Так, специалисты сообщества OWASP рекомендуют реализовывать защиту веб-приложения путем сочетания двух типов мер защиты – проверки HTTP-заголовка и использования токенов. В ряде случаев при реализации критичных информационных систем (например, систем банковского обслуживания) разработчики используют три и более меры защиты. Например, может использоваться сочетание токенов, проверки заголовков и меры защиты, требующей действия от конечного пользователя, выполняющего критичную операцию (ввод одноразового кода/пароля).

Методы и результаты проведенного исследования

Исследование защищённости веб-приложений проводилось в рамках работы аккредитованной испытательной лаборатории НПО «Эшелон» (период исследования январь-ноябрь 2016 г.). Краткая информация о веб-приложениях, участвующих в исследовании, представлена в таблице 1.

Степень внедрения мер разработки безопасного ПО (уровень зрелости) оценивалась экспертным методом с учетом объема реализованных разработчиком мер из базового набора мер по разработке безопасного ПО, предлагаемого национальным стандартом ГОСТ Р 56939-2016 «Защита информации. Разработка безопасного программного обеспечения. Общие требования» [10-12]: 1 – не реализовано ни одной меры, 2 – реализовано менее 20% мер, 2 – реализовано от 20% до 40% мер, 3 - реализовано от 40% до 60% мер, 4 - реализовано от 60% до 80% мер, 5 - реализовано более 80% мер.

При проведении анализа уязвимостей использовались типовые тесты, разработанные с учетом рекомендаций [9] и ресурса CAPEC. Общая последовательность выполняемых тестов представлена далее по тексту.

Таблица 1.

Краткая информация об объектах исследования

Идентификатор ПО	Язык программирования	Тип разработчика	Степень внедрения мер разработки безопасного ПО (уровень зрелости)
ПО №1	PHP	отечественный	2
ПО №2	Java	зарубежный	5
ПО №3	PHP	отечественный	1
ПО №4	Java	зарубежный	5
ПО №5	C#	отечественный	4
ПО №6	Java	отечественный	1
ПО №7	C#	отечественный	1
ПО №8	PHP	отечественный	1
ПО №9	Ruby	отечественный	3
ПО №10	Ruby	отечественный	3

1. Анализ частей веб-приложения (страниц), которые позволяют изменить состояние веб-приложения (создание\изменение\удаление учетных записей пользователей, защищаемой информации, иной информации и т.д.) [13].

2. Изучение запросов к идентифицированным частям веб-приложения: осуществляется передача запросов от веб-браузера к веб-приложению с последующим перехватом и анализом структуры запросы. В результате анализа перехваченного запроса эксперт должен определить тип защиты от CSRF-атаки на конкретной странице.

3. Формирование поддельного HTTP-запроса, который сохраняется в виде HTML-файла на локальном компьютере и открывается в веб-браузере, при условии наличия аутентифицированной объектом оценки (веб-приложением) сессии.

4. Если при анализе перехваченного запроса (п. 2) были обнаружены меры защиты от CSRF-атак, дополнительно выполняются следующие действия:

а) при использовании в качестве меры защиты токенов:

- анализ URL на наличие токена в открытом виде;
- отправка запроса без токена;
- отправка запроса с измененным токеном;
- отправка запроса с использованием одного токена для разных учётных записей пользователей;
- попытка угадать/подобрать токен;

б) при использовании в качестве меры защиты проверки заголовка HTTP-пакета:

- отправка запроса с измененными полями HTTP Referer/Origin;
- отправка запроса без полей HTTP Referrer/Origin.

При проведении тестов использовались следующие программные средства автоматизации: ПО BurpSuite, ПО «Сканер-ВС». Среднее время проведения испытаний одного веб-приложения одним экспертом испытательной лаборатории составило 8 часов.

В рамках исследования были получены результаты, представленные далее по тексту.

1. CSRF-атака была выполнена успешно в 70% случаев - 7 из 10 проанализированных веб-приложений оказались уязвимыми.

2. Подавляющее большинство успешных CSRF-атак было выполнено в отношении веб-приложений отечественной разработки (рис. 3). Следует отметить, что единственная успешная

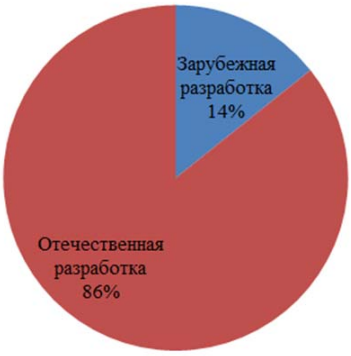

***Рис.3**. Распределение выявленных уязвимостей по типу разработчика*

CSRF-атака, выполненная в отношении веб-приложения от зарубежного разработчика, имела тип «Logout», и экспертам испытательной лаборатории не удалось разработать вектор атаки, реализующий угрозы нарушения безопасности информации. Только в одном веб-приложении изначально не было реализовано ни одной из мер защиты от CSRF-атак. В остальных уязвимых веб-приложениях была реализована защита на основе проверки заголовка HTTP или токена (рис. 4).

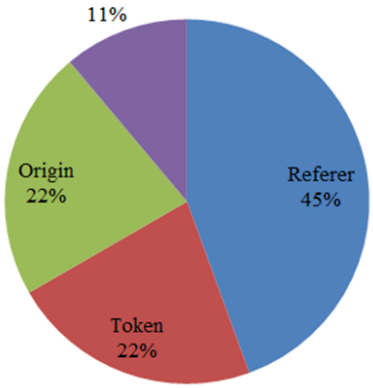

***Рис.4**. Распределение мер защиты, используемых в уязвимых веб-приложениях*

3. Установлено, что уязвимостей, приводящих к успешным CSRF-атакам, немного больше для веб-приложений, написанных на языке программирования PHP (рис. 5).

4. При доработке уязвимых веб-приложений разработчики во всех случаях использовали меры защиты, основанные на токенах.

5. В подавляющем большинстве случаев в доработанных веб-приложениях и веб-приложениях, в которых уязвимость не была выявлена, использовалось сочетание нескольких мер защиты от CSRF-атаки (рис. 6).

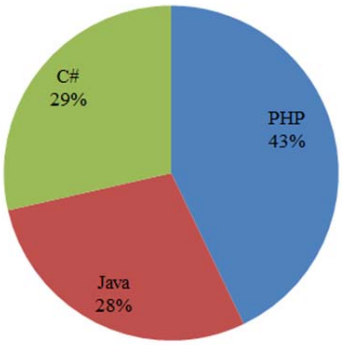

Рис.5. *Распределение выявленных уязвимостей по языкам программирования*

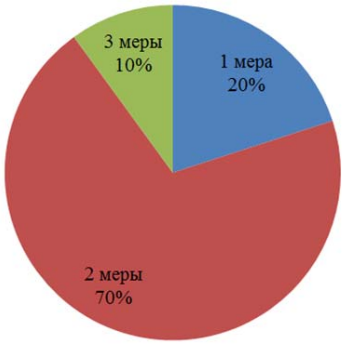

Рис.6*. Данные о количестве мер защиты, используемых в исследованных веб-приложениях*

6. Среднее время исправления уязвимости разработчиком веб-приложения составило 3 недели.

7. В качестве одного из результатов исследований было сформулировано эмпирическое правило, в соответствии с которым число уязвимостей, выявляемых в ПО, обратно пропорционально уровню зрелости процессов разработки безопасного ПО, внедренных у разработчика.

По результатам проведённого исследования были сформулированы следующие рекомендация для разработчиков веб-приложений, планирующих проведение сертификационных испытаний по требованиям безопасности информации.

1. Разработчикам рекомендуется внедрять меры разработки безопасного ПО в процессы жизненного цикла ПО. Как минимум рекомендуется реализовать меры, связанные с тестированием проникновения веб-приложений перед их передачей в испытательную лабораторию. С целью минимизации времени на проведение такого тестирования разработчикам рекомендуется формировать наборы типовых тестов, которые могут быть разработаны с учетом рекомендаций, представленных в работах [3, 9]. При выполнении тестов разработчикам рекомендуется не ограничиваться выполнением стандартного теста, а дополнительно проводить тесты, направленные на выполнение CSRF-атак типа «Login» и «Logout», проверку корректности реализации выбранной меры защиты.

2. При реализации в веб-приложении мер защиты от CSRF-атак разработчикам рекомендуется использовать принцип эшелонированной обороны – сочетать две или более меры защиты (как правило, проверка токена и заголовка HTTP).

3. При реализации в веб-приложении мер защиты от CSRF-атака разработчикам рекомендуется в первую очередь использовать меры защиты, уже реализованные в среде функционирования, например в фреймворках.

Выводы

Проведенное исследование защищённости веб-приложений, являющихся объектами испытаний в рамках сертификации по требованиям безопасности информации, от атак типа «межсайтовая подделка запросов» показало, что большинство разработчиков не уделяют должного вниманию внедрению мер защиты от подобных атак. По результатам исследования сформулированы рекомендации для разработчиков, основными из которых являются рекомендации по использованию эшелонированной обороны и использования уже реализованных разработчиками фреймворков мер защиты. Было сформулировано эмпирическое правило, в соответствии с которым число уязвимостей, выявляемых в ПО, обратно пропорционально уровню зрелости процессов разработки безопасного ПО, внедренных у разработчика.

Рецензент*: Федичев Андрей Валерьевич, кандидат технических наук, доцент, директор Федерального бюджетного учреждения «Научный центр правовой информации» при Министерстве юстиции Российской Федерации. E-mail: fedichev@scli.ru*

Литература

1. Марков А.С., Цирлов В.Л., Барабанов А.В. Методы оценки несоответствия средств защиты информации / Под. ред. А.С.Маркова. - М.: Радио и связь, 2012. 192 с.

2. Барабанов А.В., Евсеев А.Н. Применение международного стандарта для поиска уязвимостей // Безопасные информационные технологии: Сборник трудов Пятой Всероссийской научно-технической конференции. - М., 2015. - С. 50-52.
3. Барабанов А.В., Федичев А.В. Разработка типовой методики анализа уязвимостей в веб приложениях при проведении сертификационных испытаний по требованиям безопасности информации Вопросы кибербезопасности. 2016. № 2 (15). С. 2-8.
4. 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) , pages 1–10, September 2006
5. Alexei Czeskis, Alexander Moshchuk, Tadayoshi Kohno, and Helen J. Wang. 2013. 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, 273-284. DOI: http://dx.doi.org/10.1145/2488388.2488413
6. Adam Barth, Collin Jackson, and John C. Mitchell. 2008. 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, 75-88. DOI=http://dx.doi.org/10.1145/1455770.1455782
7. Hossain Shahriar and Mohammad Zulkernine. 2010. 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, 358-367. DOI=10.1109/ISSRE.2010.12 http://dx.doi.org/10.1109/ISSRE.2010.12
8. Philippe De Ryck, Lieven Desmet, Thomas Heyman, Frank Piessens, and Wouter Joosen. 2010. 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), Fabio Massacci, Dan Wallach, and Nicola Zannone (Eds.). Springer-Verlag, Berlin, Heidelberg, 18-34. DOI=http://dx.doi.org/10.1007/978-3-642-11747-3_2
9. Wim Maes, Thomas Heyman, Lieven Desmet, and Wouter Joosen. 2009. 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, 3-10. DOI=http://dx.doi.org/10.1145/1655077.1655081
10. Барабанов А.В., Марков А.С., Цирлов В.Л. 28 магических мер разработки безопасного программного обеспечения // Вопросы кибербезопасности. 2015. № 5 (13). С. 2-10.
11. Barabanov A.V., Markov A.S., Tsirlov V.L. Methodological Framework for Analysis And Synthesis of a Set of Secure Software Development Controls, Journal of Theoretical and Applied Information Technology. 2016. Vol. 88. No 1, pp. 77-88.
12. Barabanov A., Markov A., Fadin A., Tsirlov V., Shakhalov I. Synthesis of Secure Software Development Controls. In Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russian Federation, September 08-10, 2015). SIN '15. ACM New York, NY, USA, 2015, pp. 93-97. DOI: 10.1145/2799979.2799998.
13. Markov A.S., Fadin A.A., Tsirlov V.L. Multilevel Metamodel for Heuristic Search of Vulnerabilities in the Software Source Code, International Journal of Control Theory and Applications, 2016, Volume 9, Issue No. 30, pp. 313-320.

A STUDY OF CROSS-SITE REQUEST FORGERIES DURING SOFTWARE SECURITY EVALUATION

Barabanov A.V.[6], Lavrov A.I.[7], Markov A.S.[8], Polotnyanschikov I.A.[9]

A research of the web application security against cross–site request forgeries during software security evaluation was conducted. We have provided generalization related to cross–site request forgeries and security controls against that type of attack. We have investigated a set of web application and derived pie charts (e.g., cross–site request forgeries dependencies on programming languages, developer's origin, number of security controls implemented). Recommendation for web application developers related to protection against cross–site request forgeries was proposed.

Keywords*: information security, vulnerability analysis, web–application, CSRF.*

References

1. Markov A.S., Tsirlov V.L., Barabanov A.V. Metody otsenki nesootvetstviya sredstv zashchity informatsii / Pod. red. A.S.Markova. - M.: Radio i svyaz', 2012. 192 p.
2. Barabanov A.V., Evseev A.N. Primenenie mezhdunarodnogo standarta dlya poiska uyazvimostey,Bezopasnye informatsionnye tekhnologii: Sbornik trudov Pyatoy Vserossiyskoy nauchno-tekhnicheskoy konferentsii. - M., 2015, pp. 50-52.

6 Alexander Barabanov, Ph.D, CISSP, CSSLP, NPO Echelon, Moscow, ab@cnpo.ru
7 Artem Lavrov, NPO Echelon, Moscow, al@cnpo.ru
8 Alexey Markov, Doctor of Science (Comp), CISSP, FU, Moscow, ASMarkov@fa.ru
9 Ivan Polotnyanschikov, NPO Echelon, Moscow, ipo@cnpo.ru

3. Barabanov A.V., Fedichev A.B. Razrabotka tipovoy metodiki analiza uyazvimostey v veb prilozheniyakh pri provedenii sertifikatsionnykh ispytaniy po trebovaniyam bezopasnosti informatsii, Voprosy kiberbezopasnosti [Cybersecurity issues], 2016, No 2 (15), pp. 2-8.
4. 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) , pages 1–10, September 2006
5. Alexei Czeskis, Alexander Moshchuk, Tadayoshi Kohno, and Helen J. Wang. 2013. 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, 273-284. DOI: http://dx.doi.org/10.1145/2488388.2488413
6. Adam Barth, Collin Jackson, and John C. Mitchell. 2008. 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, 75-88. DOI=http://dx.doi.org/10.1145/1455770.1455782
7. Hossain Shahriar and Mohammad Zulkernine. 2010. 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, 358-367. DOI=10.1109/ISSRE.2010.12 http://dx.doi.org/10.1109/ISSRE.2010.12
8. Philippe De Ryck, Lieven Desmet, Thomas Heyman, Frank Piessens, and Wouter Joosen. 2010. 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), Fabio Massacci, Dan Wallach, and Nicola Zannone (Eds.). Springer-Verlag, Berlin, Heidelberg, 18-34. DOI=http://dx.doi.org/10.1007/978-3-642-11747-3_2
9. Wim Maes, Thomas Heyman, Lieven Desmet, and Wouter Joosen. 2009. 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, 3-10. DOI=http://dx.doi.org/10.1145/1655077.1655081
10. Barabanov A.V., Markov A.S., Tsirlov V.L. 28 magicheskikh mer razrabotki bezopasnogo programmnogo obespecheniya,Voprosy kiberbezopasnosti [Cybersecurity issues], 2015, No 5 (13), pp. 2-10.
11. Barabanov A.V., Markov A.S., Tsirlov V.L. Methodological Framework for Analysis And Synthesis of a Set of Secure Software Development Controls, Journal of Theoretical and Applied Information Technology. 2016. Vol. 88. No 1, pp. 77-88.
12. Barabanov A., Markov A., Fadin A., Tsirlov V., Shakhalov I. Synthesis of Secure Software Development Controls. In Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russian Federation, September 08-10, 2015). SIN '15. ACM New York, NY, USA, 2015, pp. 93-97. DOI: 10.1145/2799979.2799998.
13. Markov A.S., Fadin A.A., Tsirlov V.L. Multilevel Metamodel for Heuristic Search of Vulnerabilities in the Software Source Code, International Journal of Control Theory and Applications, 2016, Volume 9, Issue No. 30, pp. 313-320.