УДК 004.492.3

Обнаружение XSS-уязвимостей на основе анализа полной карты веб-приложения

Носиров З. А., Ажмухамедов И. М.

Постановка задачи. *Большинство программ, осуществляющие поиск XSS-уязвимостей, обладают рядом недостатков. Эти недостатки связаны с особенностями построения веб-приложений. Существующие программы детектирования XSS-уязвимостей осуществляют поиск только в открытой части веб-ресурса, что негативно сказывается на уровне защищенности веб-ресурса. Потому что уязвимость может находиться в закрытой части веб-ресурса, которая доступна авторизованным пользователям.* ***Целью работы*** *является повышение эффективности защиты веб-приложения от XSS-атак путем разработки программного обеспечения, осуществляющего поиск XSS-уязвимостей на основе анализа полной карты веб-приложения.* ***Новизна****. Элементами новизны разработанного решения является применение методики, основанной на последовательном применении наиболее эффективных алгоритмов для обнаружения различных типов XSS-уязвимостей. Также к элементам новизны стоит отнести поиск XSS-уязвимостей с предварительной авторизацией на тестируемом веб-приложении для создания полной карты веб-ресурса.* ***Результат.*** *Использование разработанного программного обеспечения по детектированию XSS-уязвимостей повысит эффективность защиты веб-приложения.* ***Практическая значимость.*** *Разработанная программа упрощает процесс тестирования веб-приложения. Благодаря функционалу формирования отчета с рекомендациями по устранению найденных XSS-уязвимостей делает возможным использование программы пользователями, не владеющими основами информационной безопасности.*

Ключевые слова*: межсайтовый скриптинг, XSS-атака, внедрение кода, XSS-уязвимость, скриптинг, вредоносный код.*

Введение

Обеспечение информационной безопасности (ИБ) вычислительных систем является одной из приоритетных задач, решаемых любой организацией, в хозяйственной деятельности которой применяются алгоритмы сбора, обработки, хранения, передачи информации. Множества угроз ИБ стали возможны благодаря широкому распространению сети Интернет.

При этом десять лет назад большинство веб-приложений были статическими и не имели интерактивных интерфейсов взаимодействия с пользователями. В них почти не было уязвимостей, которые могли бы быть использованы нарушителями. Поэтому многие разработчики игнорировали вопросы безопасности веб-приложений. Однако на сегодняшний день существует большое число динамических веб-сайтов с множеством новых технологий, которые используются в веб-браузерах. Данные технологии позволяют подключать к веб-

Библиографическая ссылка на статью:
Носиров З. А., Ажмухамедов И. М. Обнаружение XSS-уязвимостей на основе анализа полной карты веб-приложения // Системы управления, связи и безопасности. 2018. № 1. С. 78–94. URL: http://sccs.intelgr.com/archive/2018-01/03-Nosirov.pdf

Reference for citation:
Nosirov Z. A., Azhmuhamedov I. M. Detection of XSS-vulnerabilities based on the analysis of a complete map of the web-application. *Systems of Control, Communication and Security*, 2018, no. 1, pp. 78–94. Available at: http://sccs.intelgr.com/archive/2018-01/03-Nosirov.pdf (in Russian).

приложениям различные модули, которые усиливают взаимодействие посетителей с веб-ресурсом (например, доски объявлений, формы обратной связи и т.д.).

Однако эти новшества имеют и отрицательную сторону. Технологии, функционирующие в динамических веб-сайтах, обеспечивают хорошую платформу нарушителям для проведения компьютерных атак вида «SQL-Injection», «XSS» и т.д. С помощью внедренного кода нарушитель может получить несанкционированный доступ к данным авторизации пользователей и, выдавая себя за них, совершать противоправные действия, как на локальных компьютерах пользователей, так и в сетевом оборудовании компании, меняя конфигурацию сети и программного обеспечения. Отсутствие должных мер по соблюдению правил и норм информационной безопасности приводит к появлению угроз, которые можно реализовать с помощью компьютерных атак, эксплуатирующих уязвимости, связанные с внедрением вредоносного кода. Одним из таких видов компьютерных атак является межсайтовый скриптинг, в англоязычной литературе называемый – XSS (cross Site Scripting, x – используется в данной аббревиатуре для краткости, с – не используется, чтобы избежать путаницы с CSS) [1]. Часто уязвимость, позволяющую реализовать данный тип компьютерной атаки, также называют XSS.

Актуальность

По версии OWASP (открытого проекта обеспечения безопасности веб-приложений), межсайтовый скриптинг является одним из самых распространенных видов компьютерных атак [2]. Об этом же свидетельствуют и результаты исследования компании Positive Technologies (рис. 1) [3].

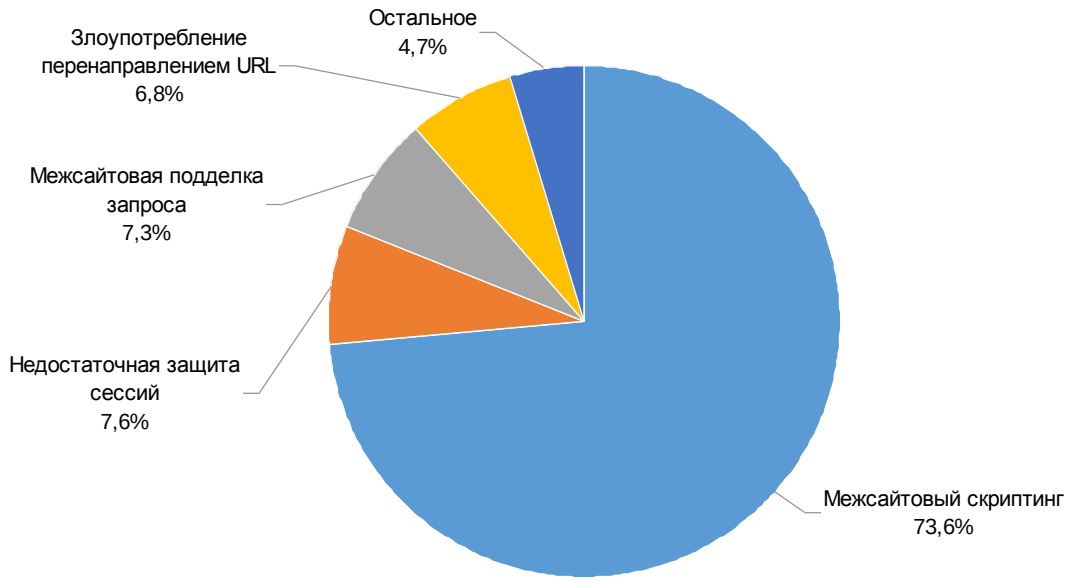

Рис. 1. Результаты исследования Positive Technologies

Виды атак эксплуатирующие XSS-уязвимости

«Межсайтовый скриптинг» – это компьютерная атака, заключающаяся во внедрении вредоносного кода в параметры веб-страницы, отправляемые веб-браузеру пользователя. Внедряемый код похож на компьютерную атаку вида «SQL-injection» [4] и может быть использован различными способами [5]. XSS принято классифицировать по двум критериям: по вектору и способу воздействия (рис. 2).

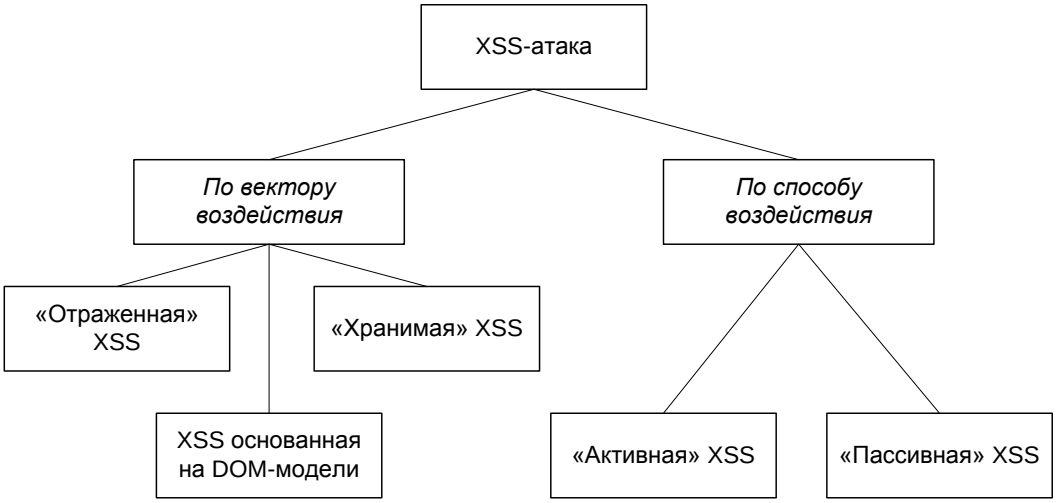

Рис. 2. Классификация XSS-атак

По способу воздействия XSS-атака может быть:

- «активная» XSS — не требующая каких-либо лишних действий со стороны пользователя с точки зрения функционала веб-приложения;
- «пассивная» XSS — срабатывающая при выполнении пользователем определённого действия («клик» или наведение указателя мыши и т.п.).

По вектору воздействия XSS-атаки подразделяются на следующие виды:

- «отраженная» (reflected) XSS;
- «хранимая» (stored) XSS;
- «XSS основанная на DOM-модели».

Рассмотрим более подробно класс XSS-атак по вектору воздействия.

«Отраженная» XSS-атака

«Отраженная» XSS-атака является наиболее распространенной, и уязвимость, которую эксплуатирует данный вид компьютерной атаки, легка в детектировании. При «отраженной» XSS-атаке передача кода серверу и возврат его к пользователю осуществляется в рамках одного HTTP-запроса. Процесс проведения межсайтового скриптинга с использованием «отраженной» XSS-уязвимости сложен. Это связано с тем, что вредоносный код необходимо внедрить в URL адрес, затем URL с внедренным вредоносным кодом отправить пользователю, чтобы он перешел по ссылке для запуска вредоносного кода. Однако сложность реализации не является препятствием для нарушителей.

При проведении «базовой отраженной» XSS-атаки не преследуется цель кражи конфиденциальной информации. Суть данной компьютерной атаки заключается в том, что при переходе на веб-сайт с «отраженной» XSS-уязвимостью, выводится предупреждающее окно. Это результат выполнения скриптового кода. Ссылка для перехода на веб-страницу с «базовой отраженной» XSS-уязвимостью выглядит примерно так:

```
http://site.ru/<script>alert(“XSS успешно
выполнен”)</script>).
```

Если нарушитель отправит данный URL своей жертве в таком виде, то скорее всего продвинутому пользователю веб-адрес покажется странным, и он не перейдет по данной ссылке. Поэтому одним из методов сокрытия вредоносной ссылки является использование кодирования RFC 1738 [6]. Как видно из фрагмента кода, приведенного ниже, после кодирования вредоносного URL скриптовый код становится трудно распознаваемым:

```
URL-адрес с XSS:
http://site.ru/<script>alert('XSS')<script>
URL-адрес с XSS после кодирования:
http:%3a%2f%2fsite.ru%2f%3cscript%3ealert(%27XSS%27)%3cs
cript%3e
```

Нарушители используют «отраженную» XSS-уязвимость с другими компьютерными атаками, чтобы повысить эффект от эксплуатации уязвимости. В частности, появился новый вид компьютерной атаки, так называемый «кликджекинг» (от англ. Clickjacking – «угон клика»). Его суть заключается в том, что пользователь, совершая переход на легитимную страницу, на самом деле производит переход по ссылке, сформированной нарушителем [7].

Особенности «кликджекинга»:

- кража ссылки происходит, когда пользователь нажимает кнопку, сформированную злоумышленником, либо переходит по ссылке;
- нет необходимости в инъецируемых скриптах, что делает атаку более простой в реализации;
- атака основана на Dynamic HTML;
- для захвата ссылки нарушитель всего лишь должен контролировать некоторую часть веб-сайта.

В свою очередь, благодаря «кликджекингу», появился другой вид компьютерной атаки, который сочетает в себе JavaScript и тег Iframe. Специалисты по информационной безопасности назвали эту компьютерную атаку «Cross-Frame Scripting (XFS)». Эта атака, которая сочетает в себе комбинацию вредоносного JavaScript-кода (JS-код) с HTML тегом Iframe, который загружает легитимную страницу для кражи пользовательских данных. XFS-атака обычно бывает успешной только в сочетании с социальной инженерией.

Рассмотрим пример реализации XFS-атаки [8]. Нарушитель, применяя методы социальной инженерии, убеждает пользователя перейти на специально сформированную страницу. После перехода на страницу нарушителя в веб-браузер пользователя подгружаются вредоносный JavaScript-код (JS-код) и Iframe. Когда пользователь вводит учетные данные в Iframe, указывающий на легитимный сайт, вредоносный JS-код крадет нажатия клавиш.

Существует еще один вид XSS-атаки, эксплуатирующая «отраженную» XSS-уязвимость веб-приложения, так называемый «Redirection XSS». Это вид XSS-атаки, в котором вредоносный код подгружается с ненадежного источника и внедряется в различные скрипты веб-приложения. Компьютерная атака вида «Redirection XSS» выглядит следующим образом:

- данные поступают в веб-ресурс с ненадежного источника в виде веб-запроса;
- полученные данные «вклиниваются» в динамический контент веб-приложения, который отправляется веб-пользователю без проверки на наличие вредоносного кода.

Вредоносный контент, отправляемый в веб-браузер пользователя, часто принимает вид JS-кода, но может также включать HTML, Flash или любой другой тип кода, который может быть обработан веб-браузером.

Компьютерные атаки, использующие «хранимые» XSS-уязвимости, являются, по сравнению с «отраженными» XSS, более опасными, так как они могут быть использованы на протяжении длительного времени. «Хранимые» XSS-атаки в основном используются для кражи сессионной информации (cookies или другой информации о сеансе). Также возможно использовать данную компьютерную атаку для перенаправления пользователя на веб-ресурс нарушителя или выполнять другие вредоносные операции на ПК пользователя.

«Хранимая» XSS-атака

Рассмотрим две наиболее опасные компьютерные атаки, эксплуатирующие «хранимую» XSS-уязвимость веб-приложения.

Кража куков (от англ. Stealing cookies). Как известно, cookies – это небольшой фрагмент данных, который хранится в веб-браузере пользователя и включает в себя пользовательские данные и много полезной информации, используемой для распознавания пользователя во время доступа к веб-приложению. Cookies генерируются на стороне веб-сервера и отправляется веб-браузеру пользователя для сохранения ее в локальном хранилище веб-браузера. При последующем обращении пользователя к веб-приложению, сохраненные cookies будут отправлены веб-серверу для распознавания пользователя и выдачи доступа к веб-ресурсу, основываясь на информации, записанной в cookies [9].

Иногда нарушители для совершения долгосрочной компьютерной атаки используют XSS-уязвимость для перенаправления пользователей на загрузку троянского коня. Специалисты по ИБ называют данную компьютерную атаку «XSS Based Trojan Horse», в переводе с английского означает «XSS атака с внедрением троянского коня» [10].

При проведении данного вида компьютерной атаки нарушителю необходимо, чтобы пользователь скачал троянскую программу. Для этого нарушитель использует «хранимую» XSS-уязвимость веб-приложения. В область веб-приложения с «хранимой» XSS-уязвимостью внедряется вредоносный код для скачивания троянской программы. Любой, кто посетит эту страницу, автоматически перейдет по ссылке с вредоносным JS-кодом. JS-код обработается брау-

зером пользователя и в фоновом режиме загрузит оставшуюся часть вредоносного кода на компьютер пользователя. При этом жертва не заметит атаку, так как используется Iframe для создания дочернего окна, в настройках которого прописан нулевой размер.

«XSS-атака, основанная на DOM-модели»

Следующая компьютерная атака эксплуатирует XSS-уязвимость в DOM-модели, возникающей на стороне пользователя во время обработки данных внутри JavaScript сценария. Данный тип XSS-атаки получил такое название, поскольку реализуется через DOM (Document Object Model), не зависящий от платформы и языка программный интерфейс, позволяющий программам и сценариям получать доступ к содержимому HTML и XML-документов, а также изменять содержимое, структуру и оформление таких документов [11]. При некорректной фильтрации возможно модифицировать DOM атакуемого сайта и добиться выполнения JS-кода в контексте атакуемого сайта.

Рассмотрим более подробно механизм проведения «DOM XSS». Данный вид XSS-атаки использует доверенный, управляемый сервером скрипт, высылаемый пользователю веб-приложения. Например, JS-код осуществляющий проверку работоспособности формы перед отправкой серверу заполненных данных пользователем. Скрипт обрабатывает введенные данные, затем вставляет их обратно в веб-страницу (например, с Dynamic HTML). Это дает возможность реализовать «DOM XSS» т.е. «вклинить» вредоносный код в сценарий JS-кода.

Существующее ПО для обнаружения XSS-уязвимостей

Современные средства защиты информации предлагают широкий спектр алгоритмов и программного обеспечения, способных предотвращать различные угрозы информационной безопасности. Большинство средств обнаружения уязвимостей громоздкие, обладают избыточным функционалом, который может замедлить работу вычислительной системы и увеличить потребление ресурсов. А простые решения «заточенные» только на поиск XSS-уязвимостей также обладают недостатками. Недостатки программ для детектирования XSS связаны с особенностями построения веб-приложений. Большинство веб-приложений предусматривает авторизацию на веб-ресурсе для увеличения привилегий пользователя. То есть авторизованным пользователям будет доступен больший функционал веб-ресурса, чем неавторизованным.

В существующих решениях по детектированию XSS-уязвимостей (таких как XSSF, XenoTix, Wapiti, XSpider (MAX-Patrol), Nemesida Scanner, Acunetix Online Web Security Scanner и др.) поиск осуществляется только в открытой (не требующей авторизации) части веб-сайта [12-15]. Это является существенным недостатком, так как XSS-уязвимость может находиться в недоступной для поиска части веб-ресурса.

Постановка задачи

Исходя из выше изложенного, можно утверждать, что разработка программного обеспечения, осуществляющего поиск XSS-уязвимостей на основе

анализа полной карты веб-приложения, является весьма актуальной задачей. Это и явилось целью данного исследования.

Решение задачи

Для достижения поставленной цели была использована методика, основанная на последовательном применении наиболее эффективных алгоритмов для обнаружения различных типов XSS-уязвимостей. В ходе разработки программного обеспечения использовался язык программирования Delphi.

Функции, реализуемые разработанным программным обеспечением:

- детектирование всех видов XSS-уязвимостей («отраженная XSS», «хранимая XSS» и «XSS основанная на DOM-модели»);
- предварительная авторизация в веб-приложении и хранение кука;
- составление списка всех внутренних URI веб-приложения;
- создание отчета о найденных уязвимостях;
- формирование рекомендаций по найденным уязвимостям.

На (рис. 3) представлена блок-схема, иллюстрирующая алгоритм поиска «отраженных» XSS. Так как данный вид уязвимостей проявляется только при отправке форм, алгоритм провоцирует их отправку методом POST, включая в отправляемые элементы значения и получая ответ в виде HTML сообщения.

Пришедшее сообщение затем анализируется на наличие уязвимости следующим образом: если пришедший JS-код устанавливает значение, хранящееся в объектной модели документа в истинное значение (document. vulnerable=true), то страница помечается как содержащая потенциальную угрозу соответствующего типа, иначе страница помечается как безопасная и не добавляется в итоговый список уязвимостей.

Для поиска XSS-уязвимостей, основанных на использовании объектной модели документа – DOM, используется алгоритм, блок-схема которого показана на (рис. 4).

В ходе выполнения алгоритма осуществляется анализ кода страницы на наличие скриптов, спрятанных в HTML тегах. После нахождения содержимого всех скриптов на странице в найденных данных осуществляется поиск вызовов методов объектной модели документа таких как:

- запись чистого HTML;
- прямая модификация модели документа (в том числе события Dynamic HTML);
- прямое выполнение скриптов.

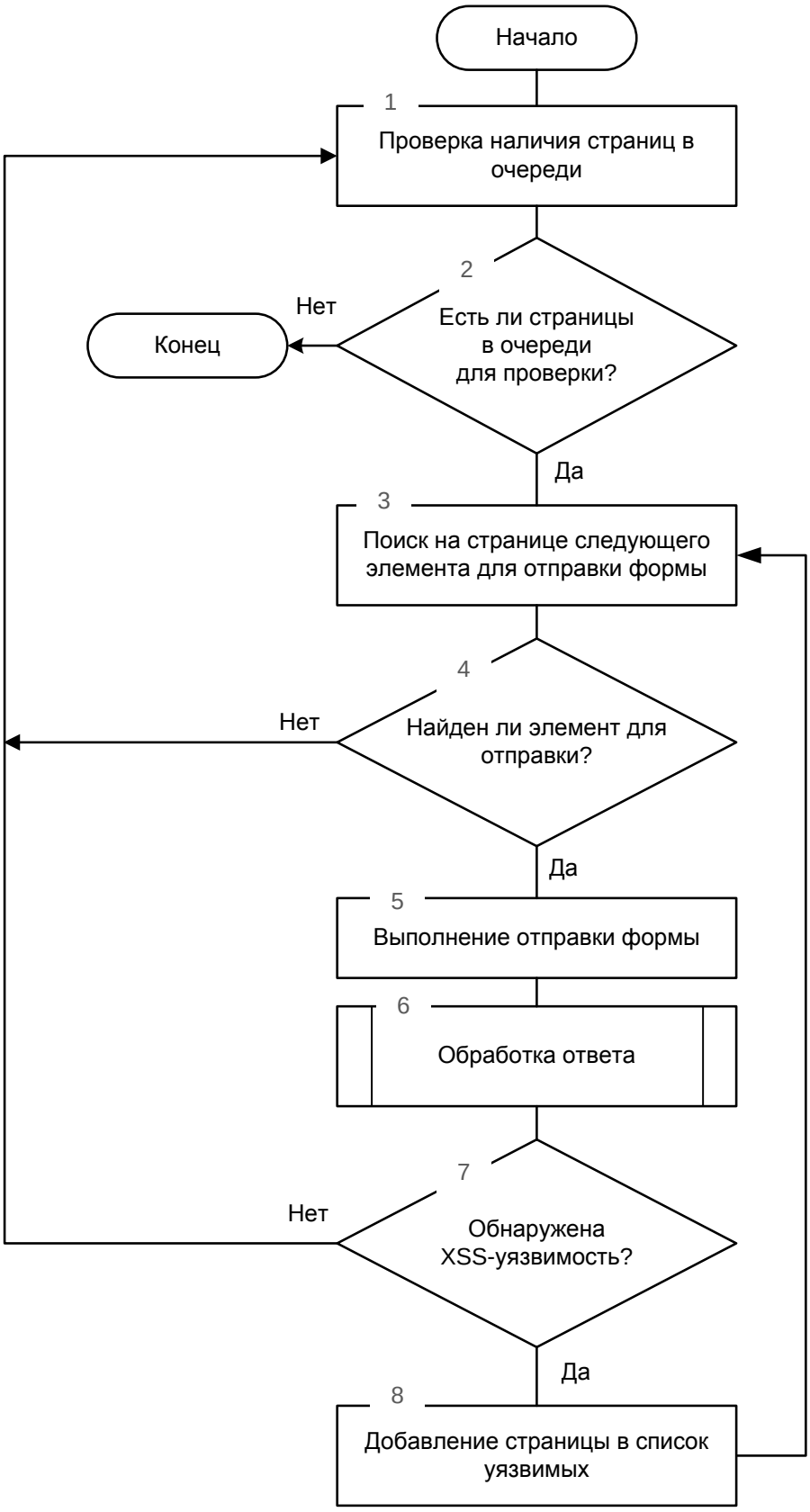

Рис. 3. Блок-схема алгоритма поиска «отраженных» XSS-уязвимостей

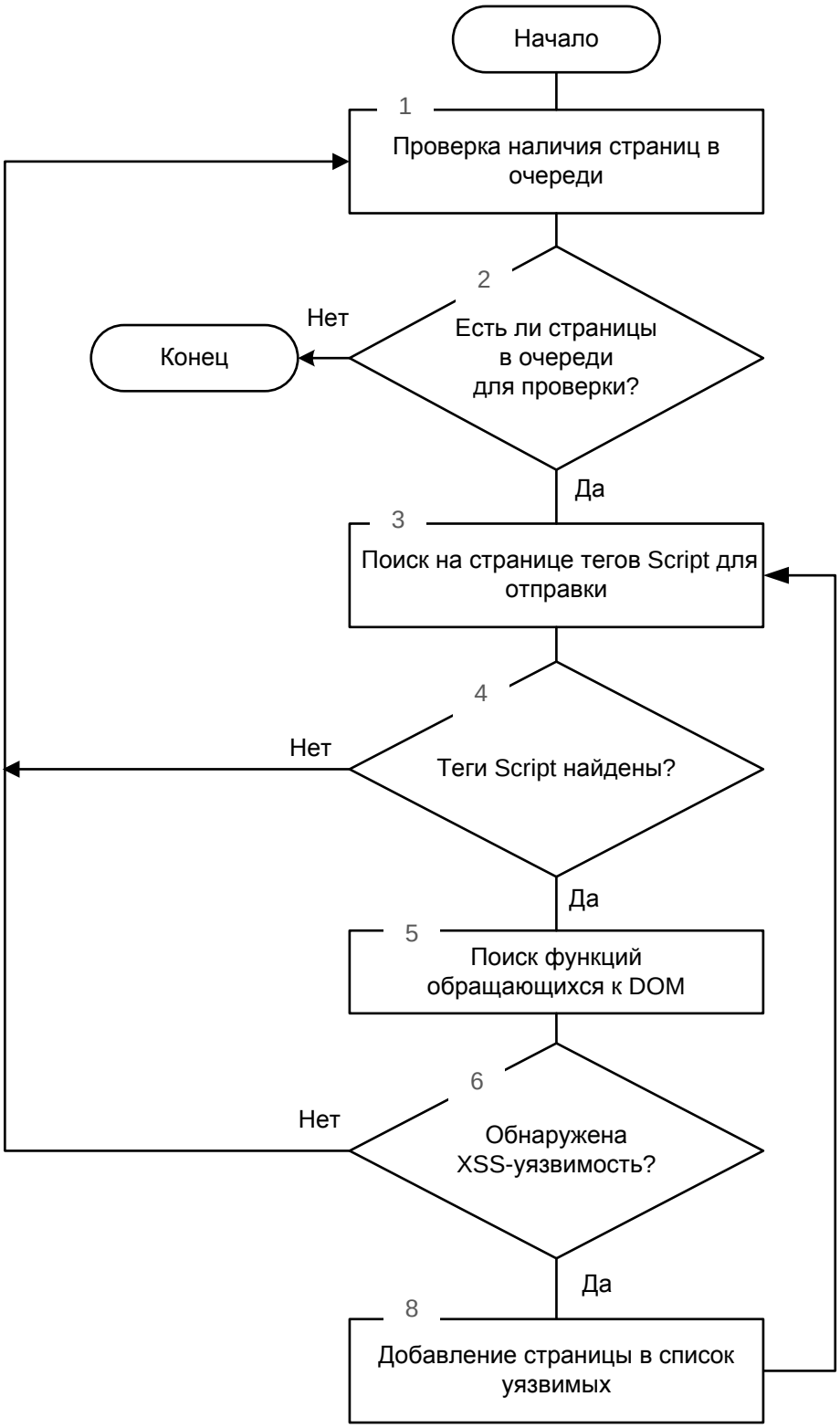

Рис. 4. Блок-схема алгоритма поиска «XSS-уязвимостей, эксплуатирующих DOM»

Для реализации поиска «хранимых» уязвимостей используется алгоритм, блок схема которого показана на (рис. 5). Работа данного алгоритма имеет свои особенности, так как, в отличие от отраженных XSS, «хранимые» уязвимости являются следствием сохранения скрипта в базу данных. Данная операция должна осуществляться с помощью предварительного POST-запроса, чтобы не

позволить вредоносному коду внести изменения в базу данных. Алгоритм во многом схож с алгоритмом поиска отраженных XSS-уязвимостей, за исключением того, что необходимо производить отправку формы и ждать ответа от сервера.

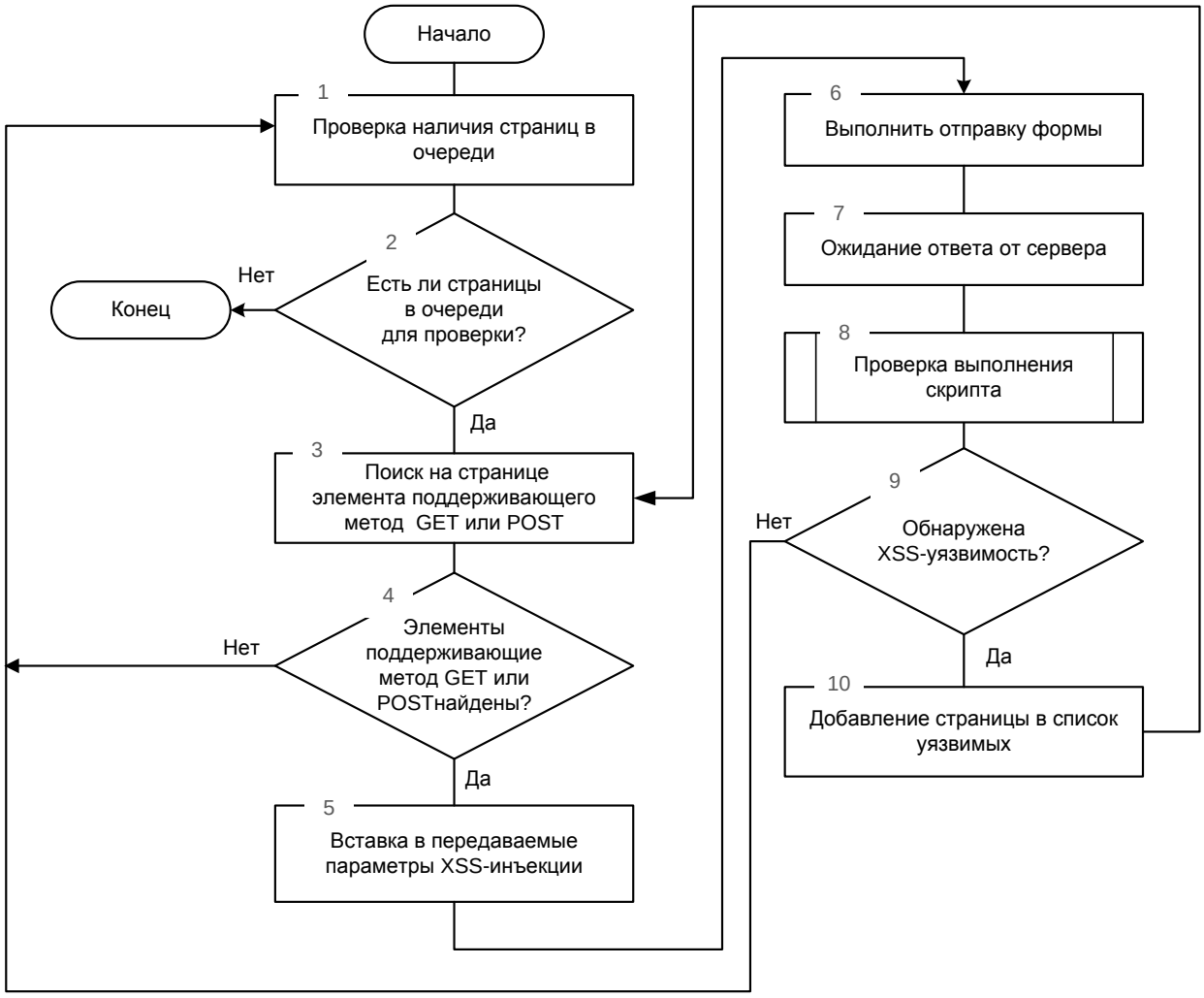

Рис. 5. Блок схема алгоритма поиска «хранимых» XSS-уязвимостей

В случае если скрипт выполнился, форма и XSS-инъекция запоминается в специальную структуру вместе с прочей информацией о найденной уязвимости. При выполнении скрипта продолжение поиска на данной странице не осуществляется, поскольку при отправке формы будет осуществляться выполнение скрипта, находящегося в базе данных веб-приложения. Поэтому данную проверку необходимо запускать повторно после устранения уязвимости. Укрупненная блок-схема разработанной программы показана на (рис. 6).

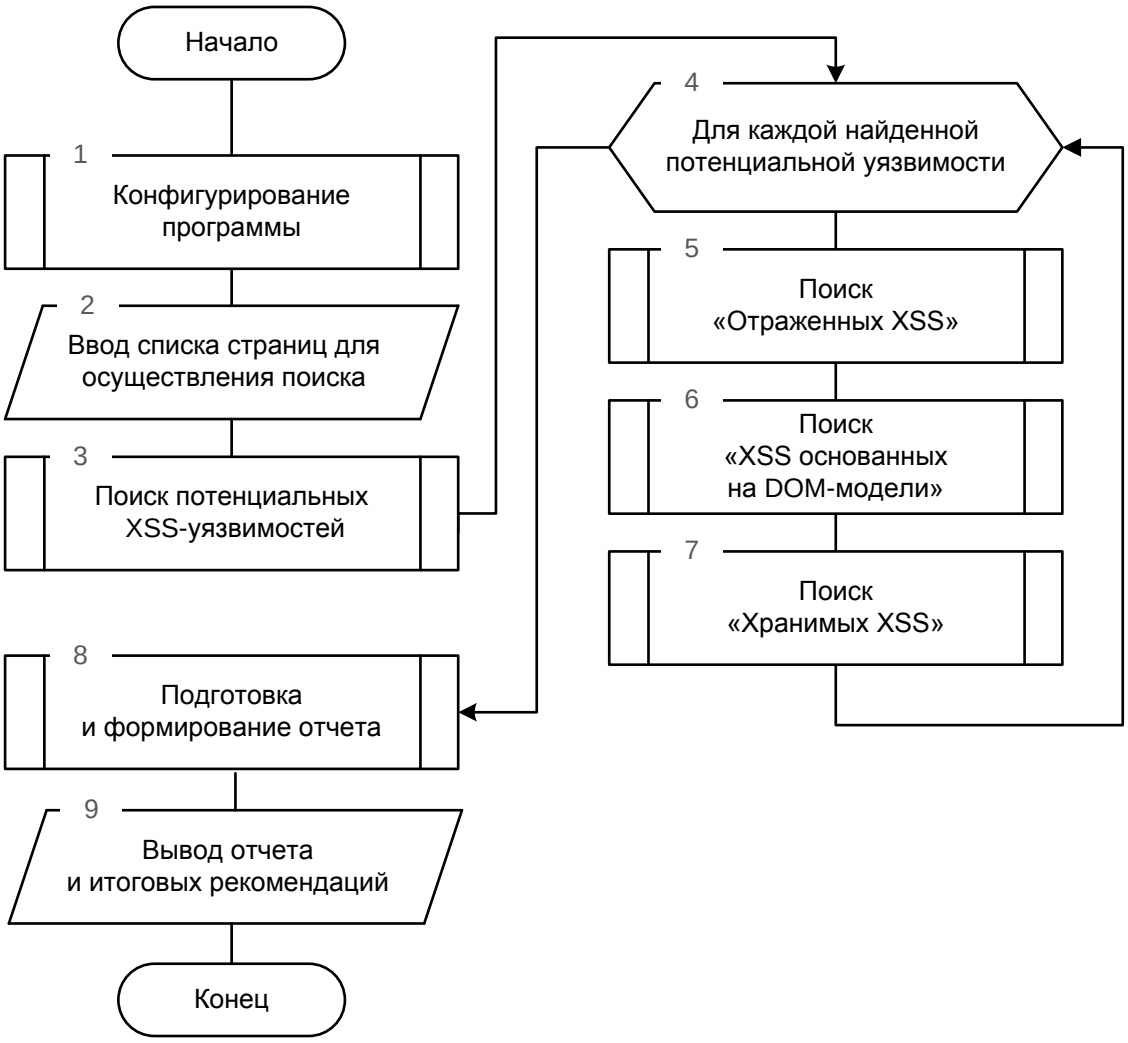

Рис. 6. Блок-схема разработанной программы детектирования XSS-уязвимостей

В разделе конфигурации программы вводятся данные для авторизации, если это необходимо. Также можно осуществить поиск определенного вида XSS-уязвимостей для экономии времени. После завершения поиска и при наличии записей во внутренней структуре, хранящей найденные угрозы и их описание, будет сформирован отчет о найденных уязвимостях. На (рис. 7) представлен фрагмент сформированного отчета.

Отчет предоставляет веб-разработчику полную информацию о тестируемом веб-приложении, а также выдает рекомендации по устранению каждой найденной уязвимости, что значительно облегчает работу по их устранению. Выдача рекомендаций является одним из преимуществ разработанной программы, по сравнению с аналогами.

Отчет по выполнению задачи поиска XSS уязвимостей

Результаты по каждой странице

№	Дата регистрации	Название	URL	Просмот.	Для проверки	Проверен.
1	22.01.2018 15:04:46		http://localhost:7081/basic/web/index.php?r=request%2Fview&id=33	+	+	
2	22.01.2018 15:04:47		http://localhost:7081/basic/web/index.php?r=request%2Fupdate&id=33	+	+	
3	22.01.2018 15:04:47		http://localhost:7081/basic/web/index.php?r=request%2Fdelete&id=33	+	+	
4	22.01.2018 15:04:46	ID	http://localhost:7081/basic/web/index.php?r=request%2Findex&sort=id	+	+	
		Обнаруженные уязвимости				
		1	Инъекция с помощью изображения путем встраивания протокола javascript			
		2	Полное отсутствие фильтрации на сервере			
		Всего обнаружено на странице: 2				
5	22.01.2018 15:04:41	Авторасписание	http://localhost:7081/	+	+	
6	22.01.2018 15:04:41	Выход (admin)	http://localhost:7081/basic/web/index.php?r=site%2Flogout	+	+	
7	22.01.2018 15:04:41	Главная	http://localhost:7081/basic/web/	+	+	
8	22.01.2018 15:04:46	Добавить заявку	http://localhost:7081/basic/web/index.php?r=request%2Fcreate	+	+	
9	22.01.2018 15:04:41	Заявки	http://localhost:7081/basic/web/index.php?r=request%2Findex	+	+	
10	22.01.2018 15:04:54	Заявки	http://localhost:7081/basic/web/index.php?r=request%2Findex&sort=-id	+	+	

Рис. 7. Фрагмент сформированного отчета

Сравнение с аналогами

Для оценки разработанного программного обеспечения было проведено его сравнение по основным характеристикам с аналогичными решениями (XSSF, XenoTix, Wapiti, XSpider (MAX-Patrol), Nemesida Scanner, Acunetix Online Web Security Scanner). Результаты сравнения программ, осуществляющие поиск XSS-уязвимостей представлены в таблице 1. Также разработанное ПО и перечисленные аналогичные решения были протестированы на специальном интернет ресурсе, предназначенный для тренировок и тестирования по поиску XSS-уязвимостей (http://www.insecurelabs.org) таблица 2.

Результаты тестирования и сравнения свидетельствуют о том, что разработанное ПО имеет ряд преимуществ перед существующими продуктами, а именно:

- осуществление поиска XSS в закрытой части веб-ресурса;
- нахождение большего количества XSS по сравнению с аналогичными решениями;
- минимально затрачиваемое время нахождения XSS-уязвимостей;
- формирование рекомендаций по устранению обнаруженных уязвимостей.

Перечисленные преимущества делает разработанную программу конкурентоспособным продуктом, направленным на детектирование XSS-уязвимостей.

Таблица 1 – Результаты сравнения программ, осуществляющие поиск XSS-уязвимостей

Характеристики	XSSF	XenoTix	Wapiti	XSpider	Nemesida Scanner	Acunetix Web Security Scanner	Разработанное ПО
Авторизация в системе	–	–	–	–	–	–	+
Поиск хранимых XSS	+	+	+	+	+	+	+
Поиск отраженных XSS	+	+	+	+	+	+	+
Поиск DOM XSS	+	–	–	–	+	+	+
Наличие ГИП	+	+	–	+	–	+	+
Выдача рекомендаций	–	–	–	+	–	+	+

Таблица 2 – Результаты тестирования программ интернет ресурсе, предназначенном для тестирования поиска XSS-уязвимостей

Наименование показателя	XSSF	XenoTix	Wapiti	XSpider	Nemesida Scanner	Acunetix Web Security Scanner	Разработанное ПО
Количество найденных уязвимостей / общее количество	6/8	5/8	6/8	8/8	7/8	8/8	8/8
Общее затраченное время на поиск (сек.)	40	49	37	35	33	30	28
Среднее время нахождения одной уязвимости (сек.)	6,67	9,8	6,17	4,37	4,71	3,75	3,5

Выводы

Разработанные алгоритмы, реализованные в виде программы, позволяют повысить эффективность защиты веб-приложения от XSS-атак. Они значительно упрощает процесс тестирования веб-ресурса разработчиком, благодаря функционалу формирования отчета с рекомендациями по устранению найденных XSS-уязвимостей.

Литература

1. Элхади А. М. Полное пособие по межсайтовому скриптингу // SecurityLab.ru [Электронный ресурс]. 11.12.2012. – URL: https://www.securitylab.ru/analytics/432835.php?R=1 (дата обращения 15.01.2018).

2. OWASP Top 10 - 2017 The Ten Most Critical Web Application Security Risks // OWASP the free and open software security community [Электронный ресурс]. 2017. – URL: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf (дата обращения 20.01.2018).

3. Positive Research. Уязвимости веб-приложений: пора анализировать исходный код // Positive Research Center [Электронный ресурс]. 08.08.2017 – URL: http://www.blog.ptsecurity.ru/2017/08/web-attacks.html. (дата обращения 20.01.2018).

4. Евтеев Д. SQL Injection от А до Я // Positive Technologies [Электронный ресурс]. 11.10.2014. – URL: https://www.ptsecurity.com/upload/corporate/ru-ru/analytics/PT-devteev-Advanced-SQL-Injection.pdf (дата обращения 21.02.2018).

5. Types of XSS: Stored XSS, Reflected XSS and DOM-based XSS // Acunetix Blog [Электронный ресурс]. 17.01.2018. – URL: https://www.acunetix.com/websitesecurity/xss/ (дата обращения 22.01.2018).

6. Berners-Lee T. Uniform Resource Locators // RFC 1738 – IETF [Электронный ресурс]. – URL: https://www.ietf.org/rfc/rfc1738.txt (дата обращения 23.01.2018).

7. Do Son. Clickjacking Attack // Penetration Testing Security Training Share [Электронный ресурс] 27.07.2017. – URL: https://securityonline.info/clickjacking-attack/ (дата обращения 24.01.2018).

8. Cross-frame-scripting // OWASP the free and open software security community [Электронный ресурс]. 22.06.2017. – URL: https://www.owasp.org/index.php/Cross_Frame_Scripting (дата обращения 24.01.2018).

9. Cross-site-scripting // OWASP the free and open software security community [Электронный ресурс]. 11.04.2009. – URL: https://www.owasp.org/index.php/Cross-site-scripting (дата обращения 25.01.2018).

10. Fogie S., Grossman J., Hansen R., Rager A., Petkov P. XSS attacks: Cross Site Scripting exploits and defense – Seth Fogie, Oxford: Elsevier Limited, 2007. – 448 p.

11. DOM Based XSS // OWASP the free and open software security community [Электронный ресурс]. 22.06.2017. – URL: https://www.owasp.org/index.php/DOM_Based_XSS (дата обращения 24.01.2018).

12. Джатана Н., Агравал А., Собти К. Пост-эксплуатация XSS: продвинутые методы и способы защиты // SecurityLab.ru [Электронный ресурс]. 12.05.2013. – URL: https://www.securitylab.ru/analytics/440187.php (дата обращения 30.01.2018).

13. XSpider // Positive Technologies [Электронный ресурс]. 12.05.2013. – URL: https://www.ptsecurity.com/ru-ru/products/xspider/ (дата обращения 20.02.2018).

14. Nemesida Scanner // PENTESTIT [Электронный ресурс]. 03.01.2017. – URL: https://www.pentestit.ru/nemesida-scanner/ (дата обращения 20.02.2018).

15. Audit Your Web Security with Acunetix Vulnerability Scanner // Acunetix [Электронный ресурс]. 03.01.2017. – URL:https://www.acunetix.com/vulnerability-scanner/ (дата обращения 20.02.2018).

References

1. Elkhadi A. M. Polnoe posobie po mezhsaitovomu skriptingu [Complete cross-site scripting allowance]. *SecurityLab.ru*, 11 December 2012. Available at: https://www.securitylab.ru/analytics/432835.php?R=1 (accessed 15 January 2018) (in Russian).

2. OWASP Top 10 - 2017 The Ten Most Critical Web Application Security Risks. *OWASP the free and open software security community*, 2017. Available at: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf (accessed 20 January 2018).

3. Positive Research. Uiazvimosti veb-prilozhenii: pora analizirovat' iskhodnyi kod [Vulnerability of web applications: it's time to analyze the source code]. *Positive Research Center*, 08 August 2017. Available at: http://www.blog.ptsecurity.ru/2017/08/web-attacks.html. (accessed 20 January 2018) (in Russian).

4. Evteev D. SQL Injection ot A do Я [SQL Injection from A to Я]. *Positive Technologies*, 11 October 2014. Available at: https://www.ptsecurity.com/upload/corporate/ru-ru/analytics/PT-devteev-Advanced-SQL-Injection.pdf (accessed 21 February 2018) (in Russian).

5. Types of XSS: Stored XSS, Reflected XSS and DOM-based XSS. *Acunetix Blog*, 17 January 2018. Available at: https://www.acunetix.com/websitesecurity/xss/ (accessed 23 January 2018).

6. Berners-Lee T. Uniform Resource Locators. *RFC 1738 – IETF*. Available at: https://www.ietf.org/rfc/rfc1738.txt (accessed 09 December 2017).

7. Do Son. Clickjacking Attack. *Penetration Testing Security Training Share*, 27 July 2017. Available at: https://securityonline.info/clickjacking-attack/ (accessed 24 January 2018).

8. Cross-frame-scripting. *OWASP the free and open software security community*, 22 June 2017. Available at: https://www.owasp.org/index.php/Cross_Frame_Scripting (accessed 24 January 2018).

9. Cross-sire-scripting. *OWASP the free and open software security community*, 11 April 2009. Available at: https://www.owasp.org/index.php/Cross-site-scripting (accessed 25 January 2018).

10. Fogie S., Grossman J., Hansen R., Rager A., Petkov P. *XSS attacks: Cross Site Scripting exploits and defense*. Oxford, Elsevier Limited, 2007, 448 p.

11. Dom Based XSS. *OWASP the free and open software security community*, 22 June 2017. Available at: https://www.owasp.org/index.php/DOM_Based_XSS (accessed 24 January 2018).

12. Jatana N., Agrawal A., Sobti K. Post-ekspluatatsiia XSS: prodvinutye metody i sposoby zashchity [Post-operation XSS: advanced methods and protection methods]. *SecurityLab.ru*, 12 May 2013. Available at: https://www.securitylab.ru/analytics/440187.php (accessed 30 January 2018) (in Russian).

13. XSpider. *Positive Technologies*, 12 May 2013. Available at: https://www.ptsecurity.com/ru-ru/products/xspider/ (accessed 20 February 2018) (in Russian).

14. Nemesida Scanner. *PENTESTIT*, 03 January 2017. Available at: https://www.ptsecurity.com/ru-ru/products/mp8/ (accessed 20 February 2018) (in Russian).

15. Audit Your Web Security with Acunetix Vulnerability Scanner. *Acunetix*, 03 January 2017. Available at: https://www.acunetix.com/vulnerability-scanner/ (accessed 20 February 2018).

Статья поступила 09 февраля 2018 г.

Информация об авторах

Носиров Зафаржон Амрулоевич – студент-исследователь кафедры «Информационная безопасность». Астраханский государственный университет. Область научных интересов: информационная безопасность, безопасность «Интернет вещей» (IoT), веб-безопасность, программирование. E-mail: nosirovzafar@outlook.com

Ажмухамедов Искандар Маратович – доктор технических наук, заведующий кафедрой «Информационная безопасность», доцент. Астраханский государственный университет. Область научных интересов: криптографические алгоритмы, анализ и оценка рисков, стеганография, защищенные сети. E-mail: aim_agtu@mail.ru

Адрес: 414056, Россия, г. Астрахань, ул. Татищева, д. 20а.

Detection of XSS-Vulnerabilities Based on Analysis of a Complete Map of a Web-Application

Z. A. Nosirov, I. M. Azhmuhamedov

Purpose: *Most programs that search XSS-vulnerabilities have some of shortcomings. These drawbacks are related to the peculiarities of building web applications. Existing programs for detecting XSS-vulnerabilities perform search only in the open part of the web-resource, which negatively affects the level of security of the web-resource. Because the vulnerability can be in the closed part of the web-resource, which is available to authorized users. The goal of the work is, to increase the effectiveness of protecting the Web-application from XSS-attacks by developing software that searches for XSS-vulnerabilities based on the analysis of the complete map of the web-application.* ***Novelty****. Elements of the novelty of the developed solutions is the application of a technique based on the consistent application of the most effective algorithms for identifying various types of XSS-vulnerabilities. A side from novelty element is the search XSS-vulnerabilities with pre-authorization on the tested web application to create a full map of the web-resource.* ***Results.*** *Using the developed software to detect XSS-vulnerabilities will increase the effectiveness of protecting the Web-application.* ***Practical relevance.*** *The developed program simplifies the process of testing a web application. Due to the functionality of the report, generation with recommendations for the elimination of detected XSS-vulnerabilities makes it possible to use the program by users who do not know the basics of information security.*

Key words*: cross site scripting, XSS-attack, code introduction, XSS-vulnerabilities, scripting, exploit*

Information about Authors

Zafarzhon Amruloevich Nosirov – research-student of the Department of «Information security». Astrakhan State University. Field of research: information security, Security «IoT», web-security, programming. E-mail: nosirovzafar@outlook.com

Iskandar Maratovich Azhmuhamedov – Dr. habil. of Engineering Sciences, Head of the Department of «Information security», Associate Professor. Astrakhan State University. Field of research: cryptography algorithms, analysis and risk assessment, steganography, secure Networks. E-mail: aim_agtu@mail.ru

Address: Russia, 414056, Astrakhan, Tatisheva Str., 20a.