ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ

им. проф. М. А. БОНЧ-БРУЕВИЧА»

**ИССЛЕДОВАНИЕ МЕТОДОВ И СРЕДСТВ БЕЗОПАСНОСТИ WEB-САЙТОВ**

Пояснительная записка к курсовому проекту

по дисциплине

«Масштабирование компьютерных сетей»

Направление 09.03.02 – Информационные системы

и технологии

Факультет Информационных систем и технологий

Кафедра Безопасности информационных систем

Курс 4

Семестр 7

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Исполнитель

Проверил ст. преп. каф. БИС студ. гр. ИСТ-731

\_\_\_\_\_\_\_\_\_\_\_ Гвоздков И.В. \_\_\_\_\_\_\_\_\_\_\_Коханчик И.М.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2020 г.

2020

Содержание

[**Введение** 3](#_Toc67941917)

[**1 Распространенные уязвимости безопасности веб-сайтов** 5](#_Toc67941918)

[**1.1 Sql-инъекции** 8](#_Toc67941919)

[**1.2 Межсайтовый скриптинг (XSS)** 13](#_Toc67941920)

[**1.3 Уязвимости системы управления содержимым сайта с открытым исходным кодом** 18](#_Toc67941921)

[**1.4 Уязвимости периметральной сети** 20](#_Toc67941922)

[**1.5 Подделка межсайтовых запросов (CSRF)** 21](#_Toc67941923)

[**1.2.6 Уязвимости TLS, SSL** 25](#_Toc67941924)

[**1.2.7 DoS атаки** 27](#_Toc67941925)

[**2 Выбор средств и методов решения проблемы** 29](#_Toc67941926)

[**2.1 Структура безопасности web-сайта** 30](#_Toc67941927)

[**2.2 Методики оценки рисков информационной безопасности** 33](#_Toc67941928)

[**2.2.1 Количественный метод оценки рисков информационной безопасности** 33](#_Toc67941929)

[**2.2.2 Качественный метод оценки рисков информационной безопасности** 36](#_Toc67941930)

[**2.2.3 Выбор метода оценки рисков** 38](#_Toc67941931)

[**2.3 Организация методов и средств защиты web-сайтов** 39](#_Toc67941932)

[**2.3.1 Выбор средств защиты web-сайта** 40](#_Toc67941933)

[**2.3.2 Принимаемые меры защиты web-ресурсов** 42](#_Toc67941934)

[**Заключение** 45](#_Toc67941935)

[**Список использованной литературы** 46](#_Toc67941936)

# **Введение**

С ростом популярности и вседоступности интернета возросло количество web-сайтов и web-сервисов. Вместе с этим число пользователей этих интернет ресурсов возросло в несколько тысяч раз. Большое количество людей пользуются десятками web-сайтами ежедневно. Web-сервисы постоянно собирают информацию о своих пользователях: от персональных данных при регистрации до информации о действиях пользователя на сайте.

В интернете множество web-сайтов и web-ресурсов, предлагающих пользователям различные услуги. Для получения этих услуг сайт запрашивает регистрацию. При регистрации пользователи вводят свои личные данные, соглашаясь с условиями конфиденциальности данных, которые предлагает web-сервис. Помимо личных пользовательских данных, web-сайт располагает и иной важной информацией. Например, хранение пользовательских действий посредством cookie-файлов, хранение паролей от личных кабинетов пользователей, информация, которую сайт публикует на своем пространстве.

От защиты этой информации зависит доверие пользователей к сайту, конфиденциальность личных данных, доступность пользователей к личному кабинету, беспрерывное функционирование сайта.

Актуальность организации комплекса мер по защите информации web-ресурсов обусловлена тем, что в наше время сайты и web-сервисы часто подвергаются различным интернет-атакам, взломам личных кабинетов пользователей, взломам баз данных сайта и кражи персональных данных пользователей. Из-за неправомерных действий хакеров администраторы web-сайтов могут столкнуться со следующими последствиями: приостановление работы web-сервиса, кража личных данных пользователей из баз данных, публикация злоумышленниками неприемлемого контента на интернет пространстве web-сайта, кража информации о действиях пользователей на сайте и прочее.

Существует также большое количество различных средств и методов защиты web-ресурсов: от рекомендаций по написанию кода до специальных программных средств. Чтобы эффективно применять их для защиты web-сайта, нужно понимать и представлять какое средство и ка работает.

Таким образом, защита информации web-ресурсов является важной и актуальной темой. Необходимо защищать сайты от возможных атак, которые могут привести к прерыванию работы сайта, взломов личных кабинетов пользователей и администраторов сайта, взлома базы данных, кражи персональных данных пользователей.

# **1 Распространенные уязвимости безопасности веб-сайтов**

Существует большое множество уязвимостей web-сайтов, есть более распространенные, которые злоумышленники используют чаще всего. От подобных угроз стоит обезопасить web-сайт в первую очередь.

Уязвимость - это недостаток в приложении или устройстве, который может быть использован злоумышленниками. Злоумышленники могут использовать уязвимость для достижения такой цели, как кража конфиденциальной информации, взломать систему, сделав ее недоступной (в сценарии отказа в обслуживании), или повредить данные.

По степени тяжести уязвимости делятся на высокую степень серьезности, среднюю и низкую.

Высокая степень серьезности указывает на то, что злоумышленник может полностью нарушить конфиденциальность, целостность или доступность системы без специального доступа, взаимодействия с пользователем или обстоятельств, которые находятся вне контроля злоумышленника.

Уровень средней степени тяжести указывает на то, что злоумышленник может частично нарушить конфиденциальность, целостность или доступность целевой системы. Им может потребоваться специальный доступ, взаимодействие с пользователем или обстоятельства, не зависящие от злоумышленника. Такие уязвимости могут использоваться вместе с другими уязвимостями для эскалации атаки.

Низкая степень серьезности означает, что злоумышленник может ограниченным образом нарушить конфиденциальность, целостность или доступность целевой системы. Им нужен специальный доступ, взаимодействие с пользователем или обстоятельства, не зависящие от злоумышленника. Для расширения атаки такие уязвимости необходимо использовать вместе с другими уязвимостями.

Компания тестирования безопасности web-приложений Acunetix ежегодно проводит сканирование безопасности многих приложений и публикуется данные о популярных уязвимостях на своем сайте.

За 2020 год было отсканировано на выявление уязвимостей безопасности порядка 5000 web-сайтов и web-приложений [6]. Статистика обнаружения уязвимостей высокой степени тяжести приведена в таблице 2.

Таблица 2 – Уязвимости высокой степени тяжести

|  |  |
| --- | --- |
| **Уязвимость** | **Процент обнаружения данного вида уязвимости среди web-сайтов** |
| Remote Code Execution | 3,03% |
| SQL-инъекция | 7,94% |
| XSS | 24,52% |
| Уязвимые библиотеки JavaScript | 24,06% |
| Слабые пароли | 0,43% |
| Уязвимости WordPress | 23,81% |
| Подделка запросов на стороне сервера | 1% |
| Уязвимости периметральной сети (SSH) | 13,86% |
| Уязвимости периметральной сети (FTP) | 6,76% |
| Уязвимости периметральной сети (DNS) | 1,48% |

На рисунке 3 представлена диаграмма распространенности уязвимостей высокой степени тяжести среди проанализированных web-сервисов. По диаграмме наглядно видно, что web-сайты чаще всего имеют следующие уязвимости:

- XSS;

- уязвимые библиотеки JS;

- уязвимости системы управления содержимым сайта с открытым исходным кодом;

- уязвимости SSH;

- SQL-инъекции.

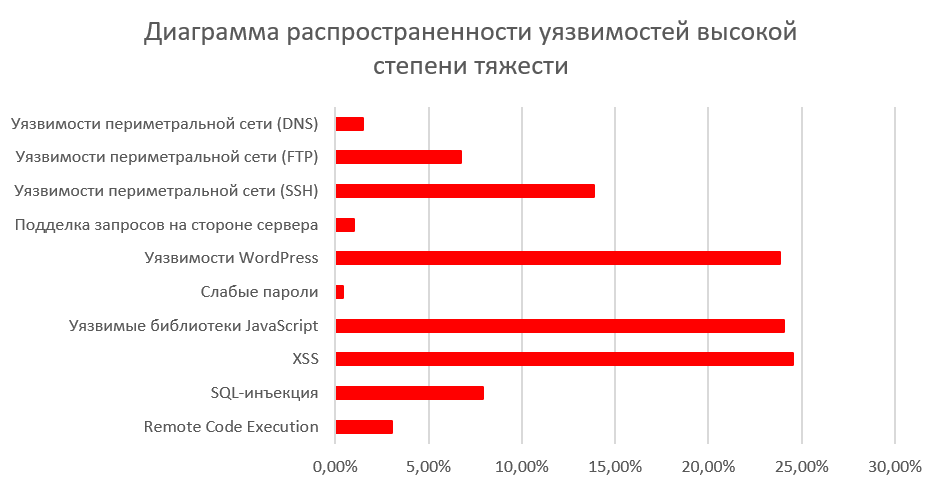


Рисунок 3 - Диаграмма распространенности уязвимостей высокой степени тяжести

Помимо статистики уязвимостей высокой степени тяжести, есть статистика уязвимостей средней степени тяжести. Среди изученных 5000 web-сайтов и web-приложений была выявлена следующая статистика уязвимостей средней степени тяжести, представленная в таблице 3.

Таблица 3 - Уязвимости высокой степени тяжести

|  |  |
| --- | --- |
| **Уязвимость** | **Процент обнаружения данного вида уязвимости среди web-сайтов** |
| DoS | 10,88% |
| CSRF | 35,9% |
| Внедрение заголовка хоста | 2,5% |
| Уязвимости TLS / SSL | 36,17% |

На рисунке 4 представлена диаграмма распространенности уязвимостей средней степени тяжести среди проанализированных web-сервисов. Среди уязвимостей средней степени тяжести наиболее распространены следующие:

- уязвимости TLS / SSL;

- подделка межсайтовых запросов (CSRF);

- DoS атаки.



Рисунок 4 - Диаграмма распространенности уязвимостей средней степени тяжести

В первую очередь стоит обезопасить свой web-ресурс от популярных уязвимостей высокой степени тяжести, так как они могут стать легкими мишенями для злоумышленников. Далее рассмотрим распространенные уязвимости высокой и средней степеней тяжести, способы их обнаружения и способы предотвращения атак с их использованием.

## **1.1 Sql-инъекции**

SQL-инъекция - это тип уязвимости безопасности web-приложений, при которой злоумышленник пытается использовать код приложения для доступа или повреждения содержимого базы данных. В случае успеха это позволяет злоумышленнику создавать, читать, обновлять, изменять или удалять данные, хранящиеся во внутренней базе данных. SQL-инъекция - один из наиболее распространенных типов уязвимостей безопасности веб-приложений .

Атака SQL-инъекцией заключается в использовании технологии атаки вставки вредоносных символов. Атаки с использованием SQL-инъекций осуществляются путем внедрения вредоносного кода в уязвимый SQL-запрос.

Успешная атака изменит запрос к базе данных таким образом, что он вернет информацию, желаемую для злоумышленника, вместо информации, которую ожидал получить web-сайт. SQL-инъекции могут даже изменять или добавлять вредоносную информацию в базу данных.

SQL-инъекции обычно выполняются через форму ввода на web-странице. Эти формы ввода часто можно найти в таких функциях, как поля поиска, поля форм и параметры URL.

Для того чтобы выполнить атаку с использованием SQL-инъекции, злоумышленники должны идентифицировать уязвимости на web-странице. После обнаружения цели злоумышленники создают вредоносные полезные нагрузки и отправляют свой входной контент для выполнения вредоносных команд.

Также существуют автоматизированные программы для выполнения SQL-инъекций. Так что в некоторых случаях злоумышленники могут просто использовать автоматизированную программу, для этого им предоставить URL-адрес целевого web-сайта, чтобы получить украденные данные от жертвы.

На рисунках №, № представлен пример уязвимости для SQL-инъекции.

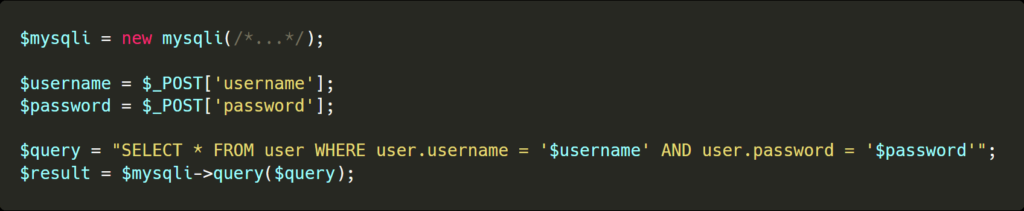


Рисунок 5 - Пример неверного запроса

В этом примере запрос создается путем объединения указанных пользователем значений (имя пользователя и пароль) непосредственно в запрос. Это позволяет злоумышленнику легко избежать кавычек и ввести больше операций SQL. Имея следующее имя пользователя «admin 'или true -» злоумышленник может войти в систему с учетной записью по своему выбору.

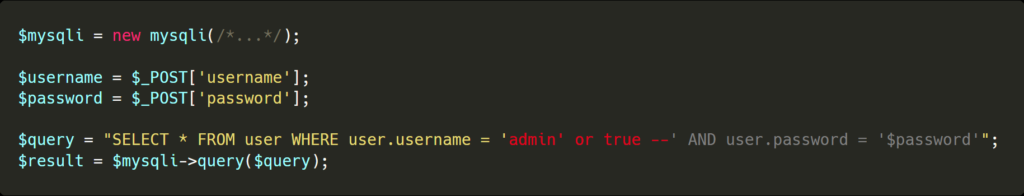


Рисунок 6 - Пример SQL-инъекции

Существует несколько тип атак с использованием SQL-инъекций, они представлены на рисунке 7 [5].

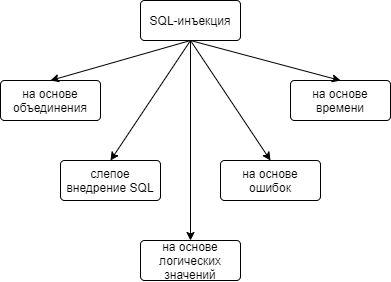


Рисунок 7 - Типы атак с использованием SQL-инъекций

Инъекция SQL на основе объединения является самым популярным методом, применяемый злоумышленниками. Этот метод внедрения позволяет злоумышленникам извлекать данные из базы данных, расширяя результаты исходного запроса. Он использует оператор UNION SQL для объединения двух операторов SELECT в один результат, а затем возвращает его как часть ответа.

Слепое внедрение SQL, как правило, более сложное и трудное для выполнения, чем другие разновидности инъекций. Злоумышленники выполняют слепые инъекции SQL, когда от цели поступают общие сообщения об ошибках. Слепые SQL-инъекции отличаются от обычных SQL-инъекций способом получения информации из базы данных. В этом методе злоумышленники запрашивают в базе данных истинные или ложные вопросы, а затем определяют ответ на основе ответа, а также времени, необходимого для получения ответа сервера при использовании его с атаками на основе времени.

SQL-инъекция на основе логических значений заменяет логику и условия запроса на свои собственные. Он используется в запросах разрешений или аутентификации, где они обманывают базу данных, заставляя думать, что у них повышенный уровень доступа или правильные учетные данные.

При внедрении SQL-кода на основе ошибок злоумышленники используют ошибки базы данных на web-странице, которые были вызваны несанкционированными входными данными. Во время атаки этот метод использует сообщения об ошибках для возврата полных результатов запроса и раскрытия конфиденциальной информации из базы данных. Этот метод также можно использовать для определения уязвимости web-сайта или веб-приложения и получения дополнительной информации для реструктуризации вредоносных запросов.

Внедрение SQL на основе времени работает с использованием операций, выполнение которых занимает много времени - часто несколько секунд. Внедрения SQL на основе времени обычно используются при определении наличия уязвимостей на web-сайте, а также в сочетании с логическими методами во время инъекций слепого SQL.

Таким образом, при использовании злоумышленником SQL-инъекций, администраторы web-сайта могут столкнуться со следующими последствиями:

- злоумышленник может внедрить вредоносный контент в уязвимые поля;

- конфиденциальные данные пользователей могут быть прочитаны из базы данных;

- данные базы данных можно изменить (вставить, обновить, удалить);

- операции администрирования могут выполняться с базой данных.

SQL-инъекции трудно обнаружить. В отличие от межсайтовых сценариев, удаленного внедрения кода и других типов заражений, SQL-инъекции представляют собой уязвимости, которые не оставляют следов на сервере. Вместо этого эксплойт выполняет подлинные запросы к базе данных. В результате большинство атак обнаруживается после того, как злоумышленник использовал уязвимость для выполнения злонамеренных действий или получил административный доступ. Принимая меры предосторожности и активно отслеживая базу данных и запросы к ней, можно определить, выполняет ли злоумышленник вредоносные инъекции на web-сайт.

Самый простой способ защитить web-сайт от SQL-инъекций это поддерживать программное обеспечение и компоненты в актуальном состоянии, но существует и ряд методов, которые можно использовать для предотвращения уязвимостей, связанных с внедрением SQL-кода.

Первое, что можно сделать, это использовать подготовленные операторы с параметризованными запросами. Подготовленные операторы используются, чтобы гарантировать, что ни одна из динамических переменных, которые нужны в запросе, не сможет выйти из своей позиции. Предварительно определяется основной запрос, а после него указываются аргументы и их типы. Поскольку запрос знает тип ожидаемых данных, то он точно знает, как интегрировать их в запрос, не вызывая проблем.

Также рекомендуется использовать хранимые процедуры. Хранимые процедуры – это частые операции SQL, которые хранятся в самой базе данных и различаются только своими аргументами. Хранимые процедуры значительно усложняют злоумышленникам выполнение своего вредоносного SQL-кода, поскольку его нельзя динамически вставлять в запросы.

Для предотвращения SQL атаки следует проверять ввод разрешенного списка. Нужно выполнить проверку списка разрешений, чтобы проверить вводимые пользователем данные по сравнению с существующим набором известных, утвержденных и определенных вводов. Если полученные данные не соответствуют заданным значениям, то они отклоняются, защищая web-сайт от вредоносных SQL-инъекций в процессе.

Для защиты от данного типа уязвимости можно использовать брандмауэр web-приложения. Фильтруя потенциально опасные web-запросы, брандмауэры web-приложений могут перехватывать и предотвращать SQL-инъекции. Брандмауэр веб-приложений представляет собой совокупность мониторов и фильтров, предназначенных для обнаружения и блокирования сетевых атак на web-ресурс.

Если web-сайт был взломан с помощью SQL-инъекции, то нужно следовать алгоритму, представленному на рисунке 8.

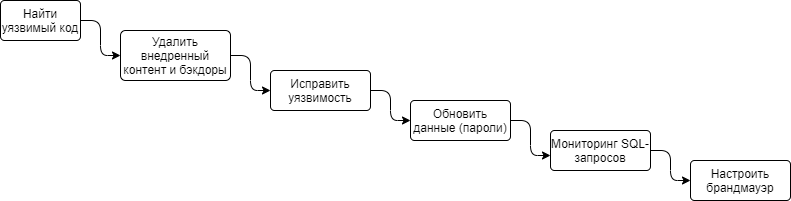


Рисунок 8 - Алгоритм действий при обнаружении взлома с помощью SQL-инъекции

## **1.2 Межсайтовый скриптинг (XSS)**

Cross-site Scripting (XSS) – это атака путем внедрения кода на стороне клиента. Злоумышленник стремится выполнить вредоносные сценарии в web-браузере жертвы, включив вредоносный код в легитимную web-страницу сайта. Атака происходит, когда жертва посещает web-страницу, которая выполняет вредоносный код. Web-страница становится средством доставки вредоносного сценария в браузер пользователя. Обычно для атак с использованием межсайтовых сценариев используются форумы, доски объявлений и web-страницы, на которых можно оставлять комментарии [7].

XSS-атаки возможны в VBScript, ActiveX, Flash и даже CSS. Однако, они наиболее распространены в JavaScript, прежде всего потому, что JavaScript является фундаментальным для большинства браузеров.

Межсайтовые сценарии также могут использоваться для искажения web-сайта вместо таргетинга на пользователя. Злоумышленник может использовать внедренные скрипты для изменения содержимого web-сайта или даже перенаправить браузер на другую web-страницу, содержащую вредоносный код.

Уязвимости XSS считаются менее опасными, чем, например, уязвимости SQLi. Большинство web-браузеров запускают JavaScript в строго контролируемой среде. JavaScript имеет ограниченный доступ к операционной системе пользователя и файлам пользователя. Однако, JavaScript может быть опасен при неправильном использовании как части вредоносного контента:

- вредоносный JavaScript имеет доступ ко всем объектам, к которым имеет доступ остальная часть web-страницы. Это включает доступ к файлам cookie пользователя. Файлы cookie часто используются для хранения токенов сеанса. Если злоумышленник сможет получить файл cookie сеанса пользователя, он может выдать себя за этого пользователя, выполнить действия от имени пользователя и получить доступ к конфиденциальным данным пользователя.

- JavaScript может читать DOM браузера и вносить в него произвольные изменения. Это возможно только на странице, где запущен JavaScript.

- JavaScript может использовать XMLHttpRequest объект для отправки HTTP-запросов с произвольным содержимым в произвольные места назначения.

- JavaScript в современных браузерах может использовать API HTML5. Например, он может получить доступ к геолокации пользователя, веб-камере, микрофону и даже к определенным файлам из файловой системы пользователя. Для большинства этих API требуется согласие пользователя, но злоумышленник может использовать социальную инженерию, чтобы обойти это ограничение.

Вышеупомянутые возможности в сочетании с социальной инженерией, позволяет преступникам проводить сложные атаки, включая кражу файлов cookie, установку троянов, фишинг и кражу личных данных. Уязвимости XSS предоставляют прекрасную основу для эскалации атак до более серьезных. Межсайтовый скриптинг также может использоваться в сочетании с другими типами атак, например, подделкой межсайтовых запросов (CSRF) .

Существует три типа атак межсайтового скриптинга, они представлены на рисунке 9.

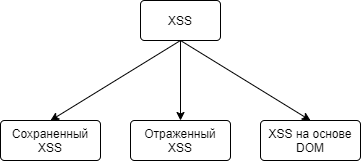


Рисунок 9 - Типы атак межсайтового скриптинга

Самый опасный тип XSS - это Stored XSS (сохраненный/постоянный XSS). Злоумышленник использует Stored XSS для внедрения вредоносного содержимого (называемого полезной нагрузкой), чаще всего кода JavaScript, в целевое web-приложение. Если входная проверка отсутствует, этот вредоносный код постоянно хранится целевым приложением, например, в базе данных. Когда жертва открывает уязвимую web-страницу в браузере, полезная нагрузка XSS-атаки передается браузеру жертвы как часть HTML-кода (точно так же, как и законный комментарий). Это означает, что жертвы в конечном итоге выполнят вредоносный сценарий после просмотра страницы в их браузере.

Второй и наиболее распространенный тип XSS - это Reflected XSS (отраженный/непостоянный XSS). В этом случае полезная нагрузка злоумышленника должна быть частью запроса, отправляемого на web-сервер. Затем он отражается обратно таким образом, что ответ HTTP включает полезную нагрузку из запроса HTTP. Злоумышленники используют вредоносные ссылки, фишинговые электронные письма и другие методы социальной инженерии, чтобы побудить жертву сделать запрос на сервер. Отраженная полезная нагрузка XSS затем выполняется в браузере пользователя. Отраженный XSS не является постоянной атакой, поэтому злоумышленник должен доставить полезную нагрузку каждой жертве. Эти атаки часто совершаются через социальные сети.

Межсайтовый скриптинг на основе DOM происходит, когда web-приложение записывает вводимые пользователем данные в объектную модель документа (DOM), затем считывает данные из DOM и выполняет их в браузере. Часто вредоносный код не отправляется на сервер, что затрудняет его обнаружение брандмауэрами web-приложений (WAF) и инженерами по безопасности, которые анализируют журналы сервера. [8]

XSS-атака состоит из двух этапов:

- чтобы запустить вредоносный код JavaScript в браузере жертвы, злоумышленник должен сначала найти способ внедрить вредоносный код (полезную нагрузку) на web-страницу, которую посещает жертва.

- после этого жертва должна посетить web-страницу с вредоносным кодом. Если атака направлена на конкретных жертв, злоумышленник может использовать социальную инженерию или фишинг, чтобы отправить жертве вредоносный URL-адрес.

Для реализации внедрения вредоносного кода уязвимый web-сайт должен напрямую включать пользовательский ввод на свои страницы. Затем злоумышленник может вставить вредоносную строку, которая будет использоваться на web-странице и обрабатываться браузером жертвы как исходный код. Существуют также варианты XSS-атак, когда злоумышленник заманивает пользователя посетить URL-адрес с помощью социальной инженерии, а полезная нагрузка является частью ссылки, по которой нажимает пользователь.

Пример уязвимого кода для XSS атаки представлен на рисунке №.



Рисунок 10 - Код, который выводит последний комментарий

Приведенный выше сценарий берет последний комментарий из базы данных и включает его в HTML-страницу. Предполагается, что напечатанный комментарий состоит только из текста и не содержит HTML-тегов или другого кода. Он уязвим для XSS, потому что злоумышленник может отправить комментарий, содержащий вредоносные данные. Пример такого комментария представлен на рисунке 11.

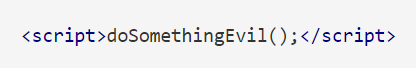


Рисунок 11 - Пример вредоносного кода

Когда страница загружается в браузере жертвы, запускается вредоносный скрипт злоумышленника.

Также преступники часто используют XSS для кражи файлов cookie. Это позволяет им выдавать себя за жертву. Злоумышленник может отправить файл cookie на свой сервер разными способами. Например выполнить клиентский скрипт в браузере жертвы, пример которого представлен на рисунке 12.



Рисунок 12 - Пример вредоносного кода для кражи cookie файлов

Злоумышленники используют различные методы для эксплуатации уязвимостей web-сайтов. В результате не существует единой стратегии для снижения риска атаки с использованием межсайтовых сценариев.

Концепция межсайтового сценария основана на том, что небезопасный пользовательский ввод напрямую отображается на web-странице. Если вводимые пользователем данные должным образом дезинфицированы, атаки с использованием межсайтовых сценариев будут невозможны. Есть несколько способов гарантировать, что пользовательский ввод не cможет быть экранирован на web-сайте.

Во-первых, как и в случае с SQLi, следует ограничить ввод пользователя определенным списком разрешений. Такая практика гарантирует, что на сервер отправляются только известные и безопасные значения.

Во-вторых, нужно ограничивать HTML во входных данных. Если используется на странице контент, созданный пользователями, нужно убедиться, что он не приведет к созданию HTML-контента путем замены небезопасных символов соответствующими объектами. Сущности имеют тот же вид, что и обычные символы, но не могут использоваться для создания HTML.

В-третьих, для предотвращения XSS атак следует использовать флаги HTTPOnly для файлов cookie. Сеансовые файлы cookie - это механизм, который позволяет web-сайту распознавать пользователя между запросами, и злоумышленники часто крадут сеансы администратора, отфильтровывая свои файлы cookie. После кражи cookie злоумышленники могут войти в свою учетную запись без учетных данных или авторизованного доступа. Для запрета чтения JavaScript содержимого файла cookie можно использовать файлы cookie HttpOnly, что затруднит злоумышленнику захват сеанса.

Также можно использовать WAF для защиты от атак межсайтового скриптинга.

Если использование XSS не удалось предотвратить, то нужно предпринять следующие шаги, для устранения последствий [9]:

- найти уязвимый код;

- удалить вредоносный контент и бэкдоры;

- исправить уязвимость;

- обновить учетные данные;

- настроить WAF.

## **1.3 Уязвимости системы управления содержимым сайта с открытым исходным кодом**

Уязвимости раскрытия исходного кода имеют две проблемы. Если код является открытым, то злоумышленнику проще реализовывать поиск уязвимостей в коде. Злоумышленник может также найти другую важную и конфиденциальную информацию, такую как учетные данные или ключи API, используемые разработчиком для интеграции с внутренними или внешними службами.

В случае открытого исходного кода злоумышленник может проверить компоненты и версии компонентов, используемые для создания web-приложения. Это помогает злоумышленнику разрабатывать атаки, нацеленные на известные уязвимости в этих версиях компонентов.

Злоумышленник также может использовать раскрытие кода для поиска уязвимостей LFI. Анализируя, как построена часть решения, злоумышленники могут угадать всю файловую структуру компонента. Затем они могут использовать это для доступа к файлам конфигурации, которые содержат учетные данные для внутренних баз данных [6].

Для предотвращения данных проблем следует не раскрывать какой-либо исходный код.

Также уязвимости системы управления содержимым сайта с открытым исходным кодом касаются программного обеспечения, которое позволяет управлять контентом сайта: создавать и публиковать записи, размещать виджеты, менять дизайн, редактировать различные элементы, добавлять функционал. Это программы Content Management System (CMS) такие, как WordPress, Joomla!, Drupal.

По оценкам компании Acunetix, на 1 января 2020 года более 35% всех web-сайтов работают на WordPress [6].

Когда дело доходит до безопасности WordPress, есть три компонента: ядро WordPress, темы пользовательского интерфейса и плагины функциональности.

Сообщество разработчиков, создающее ядро WordPress, сильное и зрелое. Обнаруженные уязвимости немедленно исследуются и быстро исправляются. На данный момент WordPress выполняет автоматические обновления для обновлений безопасности (приращение младших номеров версий) и отправляет уведомления системному администратору об успешных и неудачных обновлениях.

Иная ситуация с плагинами и темами. Любой автор может использовать эти механизмы для добавления функциональности в WordPress. Безопасность и качество этих дополнений значительно различаются. Чем популярнее становится дополнение, тем больше риск для безопасности. К сожалению, когда злоумышленник обнаруживает эксплойт, он может иногда атаковать даже десятки тысяч установок WordPress, использующих уязвимый плагин или тему.

Joomla! и Drupal также являются системами CMS с большим количеством пользователей, но они не так популярны, как WordPress. Joomla! и Drupal имеют надстройки, расширяющие их функциональность. Как и в случае с WordPress, ядро поддерживается надежной группой разработчиков и участников, а надстройки с большей вероятностью содержат уязвимости.

Для предотвращения атак, связанных с CMS, следует не раскрывать исходный код и не использовать популярные системы управления содержимым сайта.

Если предотвратить взлом не удалась, алгоритм действий по устранению последствий схож с XSS уязвимостями:

- сделать исходный код закрытым;

- удалить вредоносный контент и бэкдоры;

- обновить учетные данные;

- обновлять CMS для повышения безопасности.

## **1.4 Уязвимости** **периметральной сети**

Каждая локальная сеть защищена от внешнего мира (Интернета) с помощью периферийных устройств или устройств периметра. Они предоставляют такие функции и услуги, как маршрутизация, NAT / PAT, VPN и межсетевой экран. Серверы, такие как web-серверы, почтовые серверы, DNS-серверы, также часто расположены по периметру локальной сети и доступны из Интернета.

Такие устройства и службы нужно регулярно обслуживать для обновления их операционных систем и программного обеспечения, иначе могут появиться уязвимости. Уязвимости также могут появиться при неправильной настройке устройства или службы.

Многие из этих сервисов сейчас перемещаются из внутренних сетей в облако. Следовательно, может быть трудно отличить услугу LAN, услугу WAN и услугу периметра. Однако, независимо от местоположения службы, если критические сетевые элементы имеют уязвимости или неправильно настроены, они могут раскрыть важные данные и потенциально позволить злоумышленнику обойти аутентификацию.

Большинство уязвимостей, связанных с периметральной сетью, приходится на сети SSH и FTP. Ключи SSH защищают доступ к ресурсам. По мере роста компании, а соответственно и web-ресурса, растет и количество используемых ключей SSH, и это может вызвать некоторые проблемы. Например, может быть затруднено отслеживание большого количество ключей. Часто случается, что организации создают новые ключи, не удаляя старые.

Часто компании используют одни и те же ключи для многих служб, что является очень плохой практикой. Это затрудняет изменение или отзыв ключей, и ситуация становится еще хуже, если ключи встроены во внутренние программные системы. В результате ключи становятся статичными и не меняются регулярно. Это дает возможности злоумышленникам для проникновения. Большинство из этих уязвимостей имеют низкую степень серьезности или связаны с неправильной конфигурацией.

## **1.5 Подделка межсайтовых запросов (CSRF)**

Уязвимости подделки межсайтовых запросов (CSRF) возникают, когда web-сервер получает неавторизованный запрос от доверенного браузера. Запросы браузера, отправленные на web-сервер, могут включать файлы cookie сеанса пользователя - это почти всегда происходит, если пользователь уже вошел на сайт.

Злоумышленник может создать вредоносную ссылку, которая позволяет ему выполнить определенное действие, например, перевести деньги с онлайн-банковского счета пользователя на другой счет. Злоумышленник может разместить эту ссылку на контролируемом им web-сайте и убедить пользователя перейти по этой ссылке, по средствам социальной инженерии. Пользователь переходит по ссылке и отправляет запрос на сервер. Поскольку пользователь уже вошел в систему, сервер выполняет действие, используя его учетную запись [6].

Подделка межсайтовых запросов, также известная как CSRF, Sea Surf или XSRF, представляет собой атаку, при которой злоумышленник обманом заставляет жертву выполнять действия от своего имени. Воздействие атаки зависит от уровня разрешений, которыми обладает жертва. Такие атаки используют тот факт, что web-сайт полностью доверяет пользователю, когда он может подтвердить, что пользователи действительно является тем, кем они себя называют. CSRF допускает только изменение состояния, поэтому злоумышленник не может получить содержимое ответа HTTP.

Атака с подделкой межсайтового запроса состоит из двух этапов. Первый - обманом заставить жертву нажать на ссылку или загрузить страницу. Обычно это делается с помощью социальной инженерии и вредоносных ссылок. Второй этап состоит из отправки обработанного, законно выглядящего запроса из браузера жертвы на web-сайт. Запрос отправляется со значениями, выбранными злоумышленником, включая любые файлы cookie, которые жертва связала с этим web-сайтом. Таким образом, web-сайт знает, что эта жертва может выполнять определенные действия на web-сайте. Любой запрос, отправленный с этими учетными данными HTTP или файлами cookie, будет считаться законным, даже если жертва отправит запрос по команде злоумышленника.

Когда на web-сайт делается запрос, браузер жертвы проверяет, есть ли у него какие-либо файлы cookie, связанные с источником этого web-сайта и которые необходимо отправить с HTTP-запросом. В таком случае эти файлы cookie включаются во все запросы, отправляемые на этот web-сайт. Значение cookie обычно содержит данные аутентификации, и такие файлы cookie представляют сеанс пользователя. Это сделано для того, чтобы предоставить пользователю беспроблемную работу, поэтому от него не требуется повторная аутентификация для каждой страницы, которую он посещает. Если web-сайт одобряет файл cookie сеанса и считает, что сеанс пользователя все еще действителен, злоумышленник может использовать CSRF для отправки запросов, как если бы их отправляла жертва. web-сайт не может отличить запросы, отправленные злоумышленником, и запросы, отправленные жертвой, поскольку запросы всегда отправляются из браузера жертвы с их собственным файлом cookie. Атака CSRF просто использует тот факт, что браузер автоматически отправляет cookie на web-сайт с каждым запросом.

Подделка межсайтовых запросов будет эффективна только в том случае, если жертва аутентифицирована. Это означает, что для успешной атаки жертва должна войти в систему. Поскольку CSRF-атаки используются для обхода процесса аутентификации, могут быть некоторые элементы, на которые эти атаки не влияют, даже если они не защищены от них, например общедоступный контент. Например, общедоступная контактная форма на web-сайте защищена от CSRF. Такие HTML-формы не требуют от жертвы каких-либо привилегий для отправки формы. CSRF применяется только к ситуациям, когда жертва может выполнять действия, доступные не всем.

Самый популярный метод предотвращения подделки межсайтовых запросов – использование токена вызова, который связан с конкретным пользователем и отправляется как скрытое значение в каждой форме изменения состояния в web-приложении. Этот токен, называемый токеном анти-CSRF (часто сокращенно CSRF-токеном), работает следующим образом:

- web-сервер генерирует токен и сохраняет его;

- токен статически установлен как скрытое поле формы;

- форма отправлена пользователем;

- токен включен в данные POST-запроса;

- приложение сравнивает токен, сгенерированный и сохраненный приложением, с токеном, отправленным в запросе;

- если эти токены совпадают, запрос действителен;

- если эти токены не совпадают, запрос недействителен и отклоняется.

Этот метод защиты CSRF называется шаблоном токена синхронизатора. Он защищает форму от атак с подделкой межсайтовых запросов, поскольку злоумышленнику также необходимо угадать токен, чтобы успешно обманом заставить жертву отправить действительный запрос. Токен также должен стать недействительным через некоторое время и после выхода пользователя из системы [10].

Чтобы механизм защиты от CSRF был эффективным, он должен быть криптографически безопасным. Токен нелегко угадать, поэтому он не может быть сгенерирован на основе предсказуемого шаблона. Также рекомендуется использовать параметры защиты от CSRF в популярных фреймворках, таких как AngularJS, и воздерживаться от создания собственных механизмов. Это позволяет избежать ошибок и ускоряет и упрощает внедрение.

Атаки CSRF возможны только потому, что файлы cookie всегда отправляются с любыми запросами, которые отправляются в конкретный источник, связанный с этим файлом cookie. Можно установить флаг для файла cookie, который превращает его в файл cookie того же сайта. Файл cookie того же сайта - это файл cookie, который может быть отправлен только в том случае, если запрос сделан из источника, связанного с файлом cookie (не междоменным). Считается, что cookie и источник запроса имеют одно и то же происхождение, если протокол и хост (но не IP-адрес) одинаковы для обоих. В настоящее время файлы cookie на том же сайте лучше подходят в качестве дополнительного уровня защиты из-за этого ограничения. Следовательно, вы должны использовать их только вместе с другими механизмами защиты от CSRF [10].

Файлы cookie по своей сути уязвимы для CSRF, потому что они автоматически отправляются с каждым запросом. Это позволяет злоумышленникам легко создавать вредоносные запросы, ведущие к CSRF. Хотя злоумышленник не может получить тело ответа или сам файл cookie, он может выполнять действия с повышенными правами жертвы. Воздействие уязвимости CSRF связано с привилегиями жертвы. Хотя получение конфиденциальной информации не является основной целью CSRF-атаки, изменения состояния могут отрицательно повлиять на используемое web-приложение.

Проверить, уязвим ли web-сайт для CSRF, можно с помощью web-сканера уязвимостей, который включает в себя специализированный модуль сканера CSRF.

Для защиты от данной уязвимости следует придерживаться следующих правил:

- поддерживать осведомлённость о рисках, связанных с уязвимостями CSRF;

- использовать токены против CSRF;

- использовать файлы cookie SameSite;

- проводить регулярное сканирование сайта на уязвимость.

## **1.2.6 Уязвимости TLS, SSL**

Secure Sockets Layer (SSL) и Transport Layer Security (TLS) – это протоколы криптографической безопасности. Они используются для обеспечения безопасности сетевого взаимодействия. Их основными целями является обеспечение целостности данных и конфиденциальности связи. SSL и TLS обычно используются web-браузерами для защиты соединений между web-приложениями и web-серверами.

Протоколы SSL / TLS позволяют зашифровать соединение между двумя средами (клиент-сервер). Шифрование позволяет убедиться, что никакая третья сторона не сможет прочитать данные или вмешаться в них. Незашифрованная связь может раскрыть конфиденциальные данные, такие как имена пользователей, пароли, номера кредитных карт и прочее.

Файл cookie, который используется для аутентификации на web-сайте, отправляется в виде обычного текста, и любой, кто перехватывает соединение, может его увидеть. Злоумышленник может использовать эту информацию для входа в панель администрирования нашего сайта. С этого момента возможности злоумышленника резко расширяются. Если обращение к web-сайту с помощью SSL или TLS, злоумышленник, перехватывающий трафик, видит зашифрованную информацию [11]. Пример зашифрованной информации представлен на рисунке 13.



Рисунок 13 - Обращение к web-сайту с помощью SSL или TLS

Протоколы SSL и TLS используют криптографию с открытым ключом. За исключением шифрования, эта технология также используется для аутентификации взаимодействующих сторон. Это означает, что одна или обе стороны точно знают, с кем общаются. Когда устанавливается безопасное соединение, сервер отправляет клиенту свой сертификат SSL или TSL. Затем сертификат проверяется клиентом с помощью доверенного центра сертификации, подтверждая идентичность сервера. Такой сертификат нельзя подделать, поэтому клиент может быть на сто процентов уверен, что общается с нужным сервером.

Старые варианты SSL и TLS уязвимы для многих атак. Злоумышленник, который идентифицирует web-сервер, который все еще использует такие версии (обычно из-за неправильной конфигурации), может взломать или обойти шифрование и получить доступ к информации, которой обмениваются сервер и пользователи.

Криптографические протоколы SSL и TLS, как и любая другая технология, имеют свои недостатки. Все опубликованные и основные уязвимости влияют на более старые версии протокола (версии 1.2 и старше). Так что большинство уязвимостей, связанных с SSL и TLS, предотвращаются использованием протокола новейшей версии.

## **1.2.7 DoS атаки**

Целью DDoS-атаки (Denial of Service) является предотвращение доступа законных пользователей к сайту. Чтобы DDoS-атака была успешной, злоумышленнику необходимо отправить больше запросов, чем сервер-жертва может обработать.

Атака DDoS проверяет пределы ресурсов web-сервера, сети и приложения, отправляя пики поддельного трафика. Некоторые атаки представляют собой просто короткие пакеты вредоносных запросов на уязвимые конечные точки, такие как функции поиска. DDoS-атаки используют целую армию зомби-устройств, называемых ботнетами. Эти ботнеты обычно состоят из взломанных устройств Интернета вещей, web-сайтов и компьютеров.

При запуске DDoS-атаки ботнет атакует цель и истощает ресурсы приложения. Успешная DDoS-атака может помешать пользователям получить доступ к web-сайту или замедлить его настолько, чтобы увеличить показатель отказов, что приведет к финансовым потерям и проблемам с производительностью.

Основной целью злоумышленника, использующего метод атаки типа «отказ в обслуживании» (DoS), нарушить доступность web-сайта:

- web-сайт может медленно реагировать на законные запросы;

- web-сайт можно полностью отключить, что сделает невозможным доступ к нему законных пользователей.

Есть два типа атак вида «отказ в обслуживании»: Dos и DDoS. Атаки отказа в обслуживании (DoS) и атаки распределенного отказа в обслуживании (DDoS) очень похожи. Единственное различие между ними это их масштаб. Единичные DoS-атаки происходят из одного источника, в то время как DDoS-атаки происходят из разных мест, часто подделывая запрос [12].

При любом типе данной атаки злоумышленник использует один или несколько компьютеров. DoS-атаки находятся в нижней части этого спектра, а DDoS-атаки – в верхней части. Очень крупные DDoS-атаки могут охватывать сотни или тысячи систем.

DDoS-атаки не могут украсть информацию о посетителях сайта. Единственная цель DDoS-атаки – перегрузить ресурсы сайта. DDoS-атаки могут использоваться как способ вымогательства и шантажа. Например, владельцев web-сайтов могут попросить заплатить выкуп злоумышленникам, чтобы они остановили DDoS-атаку. DDoS-атаки могут иметь множество других мотивов, включая политическую, хактивистскую, террористическую и деловую конкуренцию.

Признаки DDoS-атак представлены на рисунке 14.

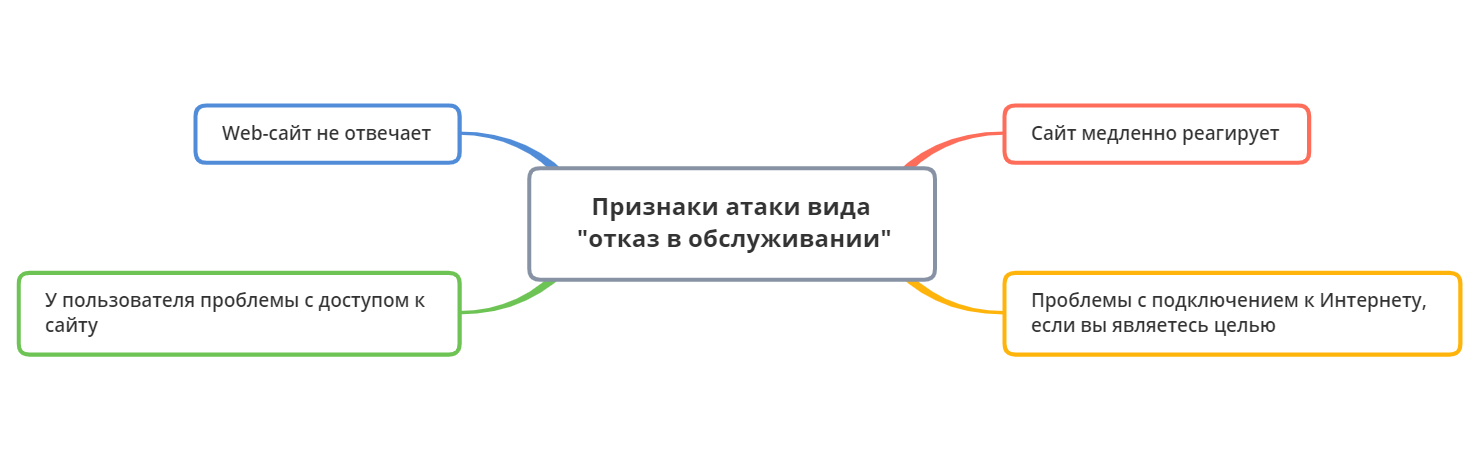


Рисунок 14 - Признаки атаки вида «отказ в обслуживании»

Есть несколько способов предотвратить Dos-атаки на web-сайт. Активация брандмауэра web-приложения для защиты от DDoS-атак – это уровень защиты, который находится между сайтом и получаемым трафиком. Брандмауэры приложений web-сайтов – это специальные брандмауэры приложений для сайтов, которые выходят за рамки метаданных пакетов, передаваемых на сетевом уровне, они сосредотачиваются на передаваемых данных. Брандмауэры приложений были созданы, чтобы понимать типы данных, разрешенные для каждого протокола, например SMTP и HTTP [12].

Большинство атак на web-сайты происходят из определенных стран, таких как Китай, Россия и Турция. Брандмауэр дает возможность заблокировать их взаимодействие (POST) с сайтом. Можно заблокировать некоторые страны, чтобы IP-адреса из этих стран по-прежнему могли просматривать весь контент на web-сайте, но не могли регистрироваться, отправлять комментарии или пытаться войти.

Также очень важно отслеживать трафик сайта, чтобы быть в курсе пиков трафика и DDoS-атак. Резкое увеличение коэффициента исходящего трафика является признаком DDoS-атак. Поэтому стоит пользоваться службами мониторинга трафика.

В настоящее время DDoS-атаки являются наиболее распространенными атаками и затрагивают все организации, независимо от их размера. Любой web-сайт может стать жертвой этих атак, поскольку они в большинстве своем автоматизированы. Ещё DDoS-атаки могут служить отвлечением для других атак, особенно когда злоумышленники разрабатывают атаки, нацеленные на финансовые учреждения. Кража данных может происходить, когда все взоры прикованы к DDoS-атаке. Поэтому, когда сайт подвержен DDoS-атаке, также стоит уделить внимание поиску других атак.

# **2 Выбор средств и методов решения проблемы**

Вопрос с безопасностью и защитой web-сайта нужно решать сразу на этапе разработки web-сайта, иначе к нему придется возвращаться при возникновении проблем на этапе функционирования сайта.

Основная задача защиты сайтов — разработка ресурса, предельно удовлетворяющего требованиям безопасности, либо приведение уже имеющегося сайта к этим требованиям путем анализа использующихся и потенциально опасных уязвимостей с последующем выполнением ряда работ для их устранения в перспективе.

Для разработки комплекса мер безопасности web-сайта нужно сначала оценить риски безопасности, определить основные виду уязвимостей, от которых следует защитить конкретный web-ресурс.

В данном разделе рассмотрим методики оценки рисков информационной безопасности, методы и средства по защите web-сайтов.

## **2.1 Структура безопасности web-сайта**

Безопасность web-сайта – это меры, принимаемые для защиты web-сайта от кибератак, это непрерывный процесс и неотъемлемая часть управления сайтом.

Наличие защищенного web-сервиса так же важно для чьего-либо присутствия в Интернете, как и наличие хоста web-сайта. Если web-сайт взломан и заблокирован, он теряет до 98% своего трафика. Отсутствие безопасного web-сайта может повлечь негативные последствия, такие как, например, утечка данных клиента, что может привести к судебным искам, крупным штрафам и испорченной репутации [2].

Национальный институт стандартов и технологий США (NIST) разработал концепцию кибербезопасности, которая составляет основу принципов безопасности web-сервисов. Структура безопасности web-сайтов включает в себя пять этапов. Структура безопасности представлена на рисунке 15.



Рисунок 15 – Структура безопасности web-сайтов

На этапе идентифицирования проходит инвентаризация и управление активами, они документируются и проверяются. Создается список активов web-сайта, чтобы иметь возможность предпринять шаги для аудита и защиты каждого из них от атак.

Далее, на этапе защиты, организуется защита web-сайта от взлома. Она организуется как и по средствам брандмауэра, так и средствами укрепления web-приложение, то есть конкретными методами для защиты конкретных уязвимостей.

Этап обнаружения включает в себя непрерывный мониторинг. Непрерывный мониторинг – это набор действий, которые позволяют наблюдать за web-средой, осуществляемой с целью получения информации о функционировании web-сайта. Данная концепция относится к внедрению инструментов для мониторинга web-сайта и предупреждения о любых проблемах. Мониторинг должен быть установлен для проверки состояния безопасности:

- записи DNS;

- SSL-сертификаты;

- конфигурации web-сервера;

- обновления приложений;

- доступ пользователей;

- целостности файлов.

Этап реагирования включает в себя анализ и смягчение последствий. Это помогает создать категорию ответа, так как при возникновении инцидента необходимо иметь плана реагирования. Наличие плана реагирования до инцидента сокращает время принятия решений по реагированию, что позволяет смягчить последствия происшествия.

Правильный план реагирования на инциденты включает:

- выбор группы или человека по реагированию на инциденты;

- сообщение об инциденте для проверки результатов;

- смягчение события.

Процесс реагирования на инциденты разбит на четыре основных этапа:

- подготовка и планирование;

- обнаружение и анализ;

- сдерживание, искоренение и восстановление;

- действия после инцидента.

На этапе подготовки и планирования определяются все необходимые инструменты и ресурсы до того, как произойдет инцидент. Это происходит на ряду с предыдущими разделами структуры безопасности. Хостинговые компании играют решающую роль на этом этапе, обеспечивая достаточную безопасность систем, серверов и сетей. Также важно убедиться, что web-разработчик или техническая команда готовы справиться с нарушением безопасности.

Этап обнаружения и анализа включает готовность справиться с любым инцидентом. Требуется быть готовым к любой атаке и знать методы решения возникшей проблемы. Так как вариантов атак большое количество, то в основном определяются более уязвимые компоненты web-сайтов.

Этап сдерживания предполагает, что процесс должен адаптироваться к типу проблемы, обнаруженной на web-сайте, и заранее иметь определенную стратегию, основанную на атаке. Система принятия решений и стратегии являются важной частью этого этапа. Если конкретный файл идентифицируется как 100% вредоносный, то должны быть предприняты действия по его удалению. Если файл содержит частично вредоносный код, следует удалить только этот фрагмент. У каждого сценария должен быть определенный процесс.

Последний этап, но достаточно важный, это этап действия после инцидента. На этом этапе группа реагирования на инциденты должна представить отчет с подробным описанием того, что произошло, какие действия были предприняты и насколько хорошо сработало вмешательство. Данные отчеты следует анализировать и извлекать уроки для принятия мер предотвращения подобных проблем в будущем. Эти действия могут быть простыми, такие как обновление компонента, изменение паролей или добавление брандмауэра web-сайта для предотвращения атак на периферии.

Пятый этап структуры безопасности «Восстановление» наступает после проведения полного анализа всех этапов в случае инцидента. Восстановление также связано с наличием плана резервного копирования для ситуаций, в которых все предыдущие этапы завершились неудачно. Этот процесс также включает в себя исправление слабых мест.

## **2.2 Методики оценки рисков информационной безопасности**

Оценка рисков – это процесс, направленный на выявление рисков кибербезопасности, их источников и способов снижения их до приемлемого уровня риска.

Цель оценки рисков информационной безопасности – определить приоритеты угроз, чтобы правильно распределить время и ресурсы. Оценка рисков определяет ключевые информационные активы, их ценность (качественную или количественную) для организации, а также ее клиентов и партнеров. Методологии оценки рисков информационной безопасности указывают, как определяются риски. Методологии также определяют:

- базовые критерии безопасности: минимальный набор средств защиты для отражения рисков;

- шкалу риска: универсальный способ количественной оценки риска;

- риск-аппетит: уровень риска, который организация готова принять;

- управление рисками на основе сценариев или активов: стратегии по уменьшению ущерба, причиненного определенными инцидентами или который может быть нанесен определенным частям организации.

Существует два метода оценки рисков: количественный и качественный.

### **2.2.1 Количественный метод оценки рисков информационной безопасности**

Количественный анализ риска – это числовая оценка общего воздействия риска на цели проекта, такие как цели по стоимости и графику. Количественная оценка рисков выражается количественно или измеряется в виде определенных чисел, цифр и процентов. Эта методология отвечает на вопросы «Каковы финансовые последствия этого риска?» и «Сколько данных было бы потеряно или скомпрометировано, если бы этот риск был реализован?».

Количественная оценка рисков применяется в ситуациях, когда исследуемые угрозы и связанные с ними риски можно сопоставить с конечными количественными значениями, выраженными в деньгах, процентах, времени, человекоресурсах. Метод позволяет получить конкретные значения объектов оценки риска при реализации угроз информационной безопасности. [14]

Чтобы начать количественный анализ рисков, группа оценки должна сначала определить ключевые активы для организации. Эта методология оценки рисков IT-безопасности включает такие факторы, как IT-оборудование, системы обработки данных и объекты, а также менее очевидные активы, такие как сотрудники, мобильные устройства и сами данные, которые хранятся в системе. После того, как все ключевые активы определены, рассчитывается стоимость каждого в долларах. Для каждого риска определяется, на какие активы он повлияет и какая часть этого актива будет потеряна или скомпрометирована в процентах. Затем берется процент убытков, умноженный на стоимость актива, чтобы получить долларовую сумму убытков для этого конкретного риска.

Этапы проведения количественной оценки рисков информационной безопасности представлены на рисунке 16.

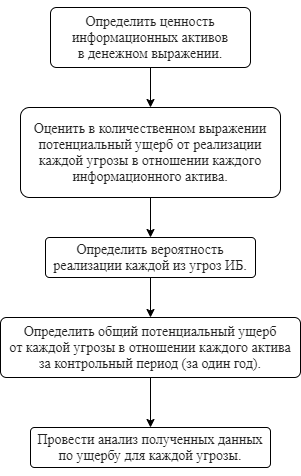


Рисунок 16 - Этапы проведения количественной оценки рисков ИБ

По каждой угрозе необходимо принять решение: принять риск, снизить риск либо перенести риск.

Принять риск — значит осознать его, смириться с его возможностью и продолжить действовать как прежде. Применимо для угроз с малым ущербом и малой вероятностью возникновения.

Снизить риск означает ввести дополнительные меры и средства защиты. То есть провести намеренную работу по снижению риска. При этом необходимо произвести количественную оценку эффективности дополнительных мер и средств защиты. Все затраты, которые несет организация, начиная от закупки средств защиты до ввода в эксплуатацию, не должны превышать размера ущерба от реализации угрозы. [14]

Перенести риск — значит переложить последствия от реализации риска на третье лицо, например с помощью страхования.

В результате количественной оценки рисков должны быть определены:

- ценность активов в денежном выражении;

- полный список всех угроз ИБ с ущербом от разового инцидента по каждой угрозе;

- частота реализации каждой угрозы;

- потенциальный ущерб от каждой угрозы;

- рекомендуемые меры безопасности, контрмеры и действия по каждой угрозе.

Обратной стороной этого подхода, основанного на числах, является то, что он не учитывает влияние на бизнес-функции или то, как производство будет затронуто в различных сценариях риска. При оценке того, как риск повлияет на бизнес-единицы, процессы и репутацию, используется качественный подход.

### **2.2.2 Качественный метод оценки рисков информационной безопасности**

Качественный подход отвечает на вопросы: «Как этот риск повлияет на команду?» и «Как потери повлияют на уровень обслуживания?» Это мнение гораздо более субъективно, чем его количественный аналог, поскольку при проведении оценки учитываются мнения и точки зрения различных заинтересованных сторон организации.

При качественном подходе не используются количественные или денежные выражения для объекта оценки. Вместо этого объекту оценки присваивается показатель, ранжированный по трехбалльной (низкий, средний, высокий), пятибалльной или десятибалльной шкале (0… 10). Для сбора данных при качественной оценке рисков применяются опросы целевых групп, интервьюирование, анкетирование, личные встречи. [14]

Анализ рисков информационной безопасности качественным методом должен проводиться с привлечением сотрудников, имеющих опыт и компетенции в той области, в которой рассматриваются угрозы.

Качественную методологию оценки рисков IT-безопасности, возможно, намного проще выполнить, чем количественный анализ, но она также менее точна. Этот метод обычно включает в себя созыв комитета из делегатов из разных частей компании, чтобы обсудить, как их команды будут затронуты различными рисками. Вместо определения сколько денег потеряется в конкретной ситуации, определяется как эта ситуация повлияет на производительность команды.

Этапы проведения качественной оценки рисков представлены на рисунке 17.

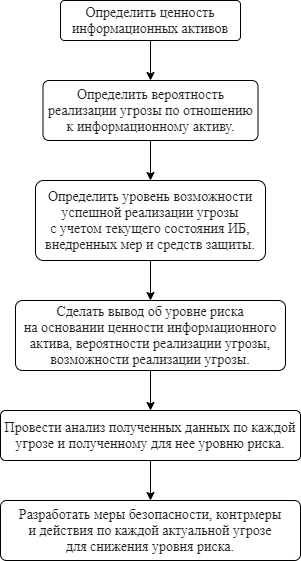


Рисунок 17 - Этапы проведения качественной оценки рисков ИБ

Результатом качественной оценки должен быть отчет о том, какие активы и системы наиболее важны для различных частей организации. Комитет по оценке необязательно будет знать финансовые последствия, если эти системы будут скомпрометированы, но они будут понимать, какие бизнес-единицы будут затронуты и сколько производительности будет потеряно при различных сценариях риска. Кроме того, оценщик должен понимать влияние на репутацию компании и любые соображения по связям с общественностью, если риск будет реализован и станет общеизвестным.

### **2.2.3 Выбор метода оценки рисков**

Целью обоих методов является понимание реальных рисков информационной безопасности организации, определение перечня текущих угроз, выбор эффективных контрмер и средств защиты. Каждый метод оценки риска имеет свои преимущества и недостатки.

Количественный метод обеспечивает наглядное представление в деньгах объектов оценки (ущерба, затрат), но он более трудоемкий и в ряде случаев неприменим.

Качественный метод позволяет выполнить оценку риска быстрее, но оценки и результаты более субъективны и не дают четкого представления о вреде и затратах.

При разработке методологии оценки рисков информационной безопасности для организации важно понимать, что для получения всестороннего представления о процессе управления рисками необходимы как количественный, так и качественный анализ. Процессы управления рисками требуют не только понимания воздействия, но и создания структуры управления рисками, которая устанавливает приемлемый уровень риска для обеспечения функционирования бизнес-операций.

Создание эффективных процессов управления рисками означает определение рисков, которые компания будет принимать, передавать, смягчать и избегать. Иногда меры безопасности не являются рентабельными.

Хотя организации необходимо конкретное и однозначное представление о финансовых последствиях и количестве потерянных данных, ей также необходимо понимать субъективные последствия риска и то, как они могут помешать производству или запятнать репутацию компании. Менеджеры рискуют подорвать бизнес-цели, игнорируя субъективные последствия требований безопасности. Руководители службы безопасности могут потратить время на проведение совещаний для качественной части оценки, в то время как аналитики рассчитывают стоимость активов и определяют финансовые последствия для количественной оценки.

После завершения обеих частей оценки результаты представляются соответствующим заинтересованным сторонам для обсуждения как объективных, так и субъективных воздействий риска. После изучения отчетов высшее руководство и члены правления получат более полное представление о ландшафте рисков компании, что позволит им принимать обоснованные решения по составлению бюджета и распределению ресурсов для их решения.

Таким образом, эффективная методология оценки рисков информационной безопасности будет включать как количественные, так и качественные подходы, чтобы описать точную картину риска. При создании программы оценки рисков стоит рассматривать возможность использования обоих этих методов для защиты наиболее важных активов.

## **2.3 Организация методов и средств защиты web-сайтов**

Средства защиты web-сайтов — программные или аппаратно-программные комплексы, предназначенные для обеспечения защиты web-приложений и различных web-сервисов.

К средствам защиты web-сайтов можно отнести следующие [15]:

- Web Application Firewall (WAF);

- средства анализа web-сайтов на наличие вирусов;

- балансировщики нагрузки на web-приложения;

- средства защиты от DDoS.

Чтобы выбрать средство защиты web-сайтов, следует обращать внимание на его возможности. К наиболее полезным функциям и свойствам средства защиты web-сайтов можно отнести следующие [15]:

- скорость обработки трафика — средства защиты не должны влиять на работу пользователей с web-сайтом;

- блокирование нелегитимных запросов и предупреждение атаки типа «Отказ в обслуживании»;

- работа с наиболее используемыми протоколами http/https и web-sockets;

- обнаружение актуальных уязвимостей и выявление угроз информационной безопасности;

- блокирование атак перебора паролей и сбора информации;

- виртуальный патчинг, который позволяет закрыть уязвимость до того, как код web-приложения будет исправлен;

- минимальный процент ложных срабатываний.

### **2.3.1 Выбор средств защиты web-сайта**

После оценки и определения рисков информационной безопасности принимаются конкретные методы и средства по защите web-сайтов. Защиту уязвимостей стоит обеспечивать еще на фазе разработки web-сайта, до ввода его в эксплуатацию. На этапе разработки сайта, написания кода, следует обращать внимание на существующие актуальные уязвимости, и предотвращать возможность использования их злоумышленниками.

Помимо защиты сайта средствами ручного написания кода, существует большое количество специализированных программ для защиты уязвимостей web-сайтов. Некоторые из них нацелены на конкретные уязвимости, другие предоставляют комплекс мер для защиты нескольких уязвимостей web-сайтов. Далее будут рассмотрены популярные и актуальные средства защиты web-сайтов.

Одним из популярных средств является программа Variti. Основными услугами является:

- защита от DDoS-атак;

- защита от ботов (рекламного фрода, перебора паролей, парсинга)

- защита от кражи аккаунтов и личных данных;

- аудит безопасности (нагрузочное и стресс-тестирование, тестирование на проникновение).

В большинстве своем программа специализируется на защите виде угроз, которые основаны на бот-атаках.

Следующим оптимальным средством защиты является программное средство Sucuri. Sucuri Website Security Platform — это сервис, который предоставляет защиту web-сайтов от любых видов угроз. Его работа основывается на перенаправлении трафика через сервера Sucuri, где проверяется каждый поступивший запрос на сайт и блокируется в случае наличия явных подозрений. Такой подход не снижает скорость работы сайта, так как при блокировании выявленного вредоносного запроса, все остальные спокойно проходят дальше. Более того, работоспособность сайта увеличивается за счет того, что запросы поступают на сторонний сервер Sucuri и снижают нагрузку сервера самого сайта. [16]

Брандмауэр web-приложений Sucuri предоставляет защиту от вредоносного программного обеспечения и взлома, от эксплойтов нулевого дня, DDoS-атак, от автоматизированных хакерских инструментов нацеленных на все сайты.

Продукт Infowatch Attack Killer является еще одним средством, обеспечивающим непрерывную безопасность web-сервисов. Программа также обеспечивает защиту от актуальных web-угроз. Основными преимуществами является непрерывная защита ценной информации, снижение репутационных и финансовых рисков, доступность ресурса и защита от DDoS-атак, автоматизация процесса проактивной защиты на основе самообучающихся алгоритмов. [17]

Брандмауэр web-приложений Imperva Incapsula анализирует трафик web-приложений, чтобы остановить атаки и обеспечить бесперебойную работу организации. поэтому получаемый сайтом трафик – это только тот трафик, который нужен организации. Данный брандмауэр представляет собой автоматизированную систему безопасности, которая интегрирует аналитику и снижает риски, создаваемые сторонним кодом.

Помимо использования конкретных программных средств защиты web-сервисов требуется обращать внимание на общие серы по защите web-сайтов

### **2.3.2 Принимаемые меры защиты web-ресурсов**

Вместе с использованием специальных программных средств нужно придерживаться еще нескольких мер по защите web-ресурсов.

Наиболее важная мера безопасности это обновление защитных программных средств. Каждый день бесчисленные web-сайты подвергаются взлому из-за устаревшего и небезопасного программного обеспечения. Большинство атак на web-сайты автоматизированы. Боты постоянно сканируют каждый сайт на предмет каких-либо возможностей эксплуатации. Уже недостаточно обновлять один раз в месяц или даже раз в неделю, потому что боты с большой вероятностью обнаружат уязвимость до того, как она будет исправлена. Особенно это касается сайтов, сделанных с помощью систем управления содержимым сайта с открытым исходным кодом.

Защищенность web-сайта во многом зависит от уровня безопасности доступа, поэтому следует использовать надежные пароли, особенно для администраторов сайта.

Также следует ограничивать доступ и разрешения пользователям. Код web-сайта может не быть целью злоумышленника, но пользователи будут. Следует придерживаться принципа наименьших привилегий. Он основан на принципе, который решает две задачи [2]:

- использование минимального набора привилегий в системе для выполнения действия;

- предоставление этих привилегий только на время, когда необходимо действие.

Предоставление привилегий определенным ролям будет определять, что они могут и что не могут делать. В идеальной системе роль остановит любого, кто попытается выполнить действие, выходящее за рамки того, для чего она предназначена. Важно, чтобы у каждого пользователя было соответствующее разрешение, необходимое для выполнения своей работы. Тщательно определенные роли пользователей и правила доступа ограничат количество возможных ошибок. Это также уменьшает вероятность заражения скомпрометированными учетными записями и может защитить от ущерба, нанесенного мошенническими пользователями.

Если для управления содержимым сайта используется CMS, то следует изменить настройки CMS по умолчанию. Наиболее распространенные атаки на web-сайты полностью автоматизированы. Многие из этих атак полагаются на то, что у пользователей есть только настройки по умолчанию. Это означает, что можно избежать большого количества атак, изменив настройки по умолчанию при установке выбранной CMS. Также следует выбирать и безопасные расширения. При принятии решения, какие расширения использовать, следует обратить внимание на источник расширения, его возраст и количество установок, и на дату последнего обновления.

В случае взлома решающее значение для восстановления web-сайта имеют резервные копии web-сайтов. Хотя это не следует рассматривать как замену решения по безопасности web-сайта, но резервная копия может помочь восстановить поврежденные файлы.

Не стоит забывать и о файлах конфигурации сервера. Файлы конфигурации сервера, которые чаще всего находятся в корневом web-каталоге, очень эффективны. Они позволяют выполнять правила сервера, включая директивы, повышающие безопасность web-сайта. Несколько рекомендаций, которые можно добавить для конкретного web-сервера, представлены ниже [2]:

- предотвратить просмотр каталогов (предотвращает просмотр злоумышленниками содержимого каждого каталога на web-сайте);

- предотвращение хотлинкинга изображений (не позволяет другим веб-сайтам отображать изображения, размещенные на вашем web-сервере);

- защита конфиденциальных файлов (можно установить правила для защиты определенных файлов и папок).

Для безопасности передачи данных требуется пользоваться защищенными сертификатами. Сертификаты SSL используются для шифрования данных при передаче между хостом (web-сервером или брандмауэром) и клиентом (web-браузером). Это помогает гарантировать, что информация будет отправлена ​​на правильный сервер и не будет перехвачена. Важно знать, что сертификаты SSL не защищают web-сайты от атак и взломов. SSL-сертификаты шифруют данные при передаче, но не добавляют защитный слой к самому web-сайту.

Обязательным и необходимым является использование инструментов сканирования и мониторинга уязвимостей web-сайта. Требуется следить за каждым компонентом сайта, чтобы гарантировать целостность web-приложения. Механизмы оповещения могут улучшить время отклика и уменьшить ущерб в случае нарушения. Можно использовать специальные программные средства для сканирования уязвимостей web-сайта и незамедлительно исправлять повреждения.

# **Заключение**

Для обеспечения безопасности web-ресурсов, требуется все время быть очень осторожными и следить за обновлениями в среде безопасности web-сайтов. Со временем модули безопасности устаревают, злоумышленники придумывают новейшие способы проникновения, поэтому организации по защите сайтов усовершенствуют свое программное обеспечение. Обеспечить безопасность web-приложения достаточно сложно. Большинство уязвимостей связано не с тем, какие системы используются, а с тем, как они используются.

Лучший способ повысить безопасность web-ресурсов – внедрить автоматизацию тестирования безопасности в жизненный цикл разработки. Это означает интеграцию сканирования web-уязвимостей со средствами отслеживания проблем, средами непрерывного развертывания и аналогичными инструментами.

Организация для обеспечения безопасности web-сайта должна придерживаться структуры информационной безопасности, реализуя все пять функций: идентификацию, защиту, обнаружение, реагирование и восстановление.

Таким образом, организация сможет максимально обезопасить web-сайт, будет готова верно реагировать на атаки и решать последствия.

# **Список использованной литературы**

1 ГОСТ 7.32-2017. Система стандартов по информации, библиотечному и издательском делу. Отчет о научно-исследовательской работе;

2 Джоел Скембрей, Майк Шема, Секреты хакеров. Безопасность Web-приложений;

3 Электронный ресурс, статья «Безопасность Web-сайтов» https://sucuri.net/

4 Электронный ресурс, статья «Website Vulnerabilities & Threats» https://sucuri.net/guides/website-security/

5 Электронный ресурс, статья «Уязвимости сайтов» https://semantica.in/blog/uyazvimosti-sajtov.html

6 Электронный ресурс, статья «Статистика уязвимостей web-сайтов» https://www.acunetix.com/white-papers/acunetix-web-application-vulnerability-report-2020/

7 Электронный ресурс, статья «Методы оценки рисков» https://kontur.ru/articles/1691

8 Электронный ресурс, статья «Средства защиты» https://www.anti-malware.ru/security/application-security