УДК 004.056.53

АНАЛИЗ УЯЗВИМОСТЕЙ ВЕБ-ПРИЛОЖЕНИЙ И ОСНОВНЫХ МЕТОДОВ РЕАЛИЗАЦИИ КИБЕРАТАК

Журавель К.С., студент гр. 6301, III курс
Нижневартовский государственный университет
г. Нижневартовск

Аннотация. В данной статье проанализирована роль протокола HTTP в архитектуре веб-приложений и описано, каким образом элементы HTTP-запроса могут использоваться злоумышленниками в качестве векторов атаки. Представлена классификация уязвимостей веб-приложений на основе рейтинга OWASP Top 10. Особое внимание уделяется рассмотрению наиболее распространённых методов кибератак, включая контрабанду HTTP-запросов, SQL-инъекции, XSS-атаки, CSRF-атаки, включение локальных файлов (LFI), DDoS-атаки, DNS-туннелирование, атаки с использованием LLM. Подчеркивается актуальность проблемы обеспечения безопасности веб-приложений и необходимость применения комплексных методов, направленных на их защиту.

Ключевые слова: веб-приложения, контрабанда HTTP-запросов, SQL-инъекции, XSS-атаки, CSRF-атаки, включение локальных файлов (LFI), DDoS-атаки, DNS-туннелирование, атаки с использованием LLM.

Введение

Современное информационное общество характеризуется стремительным ростом количества пользователей сети Интернет и, как следствие, увеличением объема данных, обрабатываемых веб-приложениями, которые стали ключевым элементом информационной инфраструктуры организаций, государственных учреждений и частных пользователей. Через веб-интерфейсы осуществляется доступ к банковским сервисам, государственным порталам, медицинским системам, корпоративным базам данных и другим критически важным ресурсам [1, с. 106].

Одновременно с развитием веб-технологий постоянно совершенствуются и методы кибератак. Злоумышленники активно используют уязвимости программного обеспечения, ошибки проектирования, неправильные настройки безопасности и недостатки протоколов взаимодействия с целью получения конфиденциальной информации, нарушения доступности сервисов. Актуальность проблемы обеспечения безопасности веб-приложений подтверждается статистическими данными, согласно которым уязвимости веб-приложений продолжают занимать лидирующие позиции в общем ландшафте киберугроз [2].

Целью данной статьи является анализ основных уязвимостей веб-приложений и наиболее распространённых методов реализации кибератак с учетом современных тенденций развития информационных технологий и появления новых угроз, связанных с внедрением технологий искусственного интеллекта.

Протокол HTTP и его роль в архитектуре веб-приложений

Базовым протоколом, обеспечивающим взаимодействие между клиентом (веб-браузером) и сервером, является протокол передачи гипертекста (HTTP, HyperText Transfer Protocol). Архитектура веб-приложений предполагает обмен HTTP-сообщениями, которые подразделяются на запросы (HTTP requests) и ответы (HTTP responses).

Структура HTTP-запроса включает метод (GET, POST, PUT, DELETE и др.), URI, заголовки (headers) и, при необходимости, тело сообщения. Каждый из этих элементов может стать вектором для кибератаки, если приложение некорректно его обрабатывает. Злоумышленники часто манипулируют этими полями для обхода механизмов защиты, что приводит к таким последствиям, как искажение логики работы приложения или несанкционированный доступ к данным [3, с. 307].

Классификация уязвимостей веб-приложений

Одним из наиболее авторитетных и распространённых источников в области безопасности веб-приложений и прикладной безопасности в целом является международное сообщество Open Web Application Security Project (OWASP), которое определило рейтинг уязвимостей веб-приложений [4, с. 87]:

— Нарушение контроля доступа (Broken Access Control)

— Криптографические сбои (Cryptographic Failures)

— Инъекции (Injection)

— Небезопасный дизайн (Insecure Design)

— Неправильная настройка безопасности (Security Misconfiguration)

— Уязвимые и устаревшие компоненты (Vulnerable and Outdated Components)

— Сбои идентификации и аутентификации (Identification and Authentication Failures)

— Сбои целостности ПО и данных (Software and Data Integrity Failures)

— Сбои логирования и мониторинга безопасности (Security Logging and Monitoring Failures)

— Подделка серверных запросов (Server-Side Request Forgery (SSRF))

Методы реализации кибератак

1. Контрабанда HTTP-запросов

Контрабанда HTTP-запросов представляет собой метод вмешательства в поток HTTP-запросов, который использует уязвимости в том, как веб-сервер или прокси-серверы обрабатывают последовательность поступающих запро-

сов, что позволяет злоумышленнику ввести вредоносный запрос, который может быть неправильно интерпретирован сервером и привести к нарушению работы всей системы [5, с. 520].

2. SQL-инъекции

Несмотря на многолетнюю историю изучения, атаки с использованием SQL-инъекций продолжают занимать лидирующие позиции в статистике взломов (OWASP Top 10), оставаясь серьезной угрозой для веб-приложений.

SQL-инъекция – это техника внедрения вредоносного SQL-кода в параметры запроса (например, формы поиска, авторизации или обратной связи), которые впоследствии передаются на сервер баз данных. Целью является изменение логики исходного SQL-запроса для получения, изменения или удаления конфиденциальных данных, хранящихся в базе данных.

Например, ввод конструкции «' OR '1'='1» в поле пароля может обмануть систему и подтвердить авторизацию без знания реального пароля. Атака происходит через любые пользовательские поля, где приложение строит SQL-запрос путем простой конкатенации (соединения) строк с введенными данными [6, с. 127].

3. Межсайтовый скриптинг (XSS)

Межсайтовый скриптинг (XSS, Cross-Site Scripting) – это уязвимость, позволяющая внедрить вредоносный исполняемый код (JavaScript, VBScript) на веб-страницу. Это происходит из-за того, что приложение не проверяет и не очищает должным образом пользовательский ввод перед тем, как встроить его в HTML-код страницы. Для обхода систем защиты (WAF) злоумышленники могут использовать различные методы кодирования полезной нагрузки (Payload Encoding) или манипуляции с атрибутами тегов.

Целью межсайтового скриптинга может быть кража учетных записей, перенаправление пользователя на фишинговый сайт, демонстрация провокационного контента от имени пользователя и т.д.

Существует три основных типа XSS-атак:

1. Отраженные (непостоянные) XSS возникают, когда вредоносный скрипт является частью HTTP-запроса (например, в параметрах URL) и сразу же «отражается» в ответе сервера. Для успешной атаки требуется, чтобы жертва перешла по специально сформированной ссылке.

2. Сохраненные (постоянные) XSS являются наиболее опасными, так как при данном типе атак вредоносный код загружается на сервер (например, к комментариям к посту или в поле профиля) и отображается перед всеми пользователями, просматривающими страницу.

3. DOM-based XSS – это атака на стороне клиента, при которой JavaScript-код сайта берет данные из URL или других источников и вставляет их в DOM-структуру документа [6, с. 128].

4. Межсайтовая подделка запросов (CSRF)

Атаки с подделкой межсайтовых запросов (CSRF-атаки, Cross-Site Request Forgery) представляют собой серьёзную угрозу для веб-приложений,

позволяя злоумышленникам выполнять несанкционированные действия от имени пользователей.

Для этого злоумышленник создает вредоносный сайт или страницу, содержащую скрытую форму или JavaScript-код, который автоматически отправляет запрос (например, на смену email или перевод денег) на целевой сайт. Когда аутентифицированный пользователь заходит на страницу злоумышленника, его браузер отправляет этот запрос вместе с cookie на целевой сайт. Сервер, получив валидный запрос, идентифицирует его как легитимный и выполняет. То есть принцип работы CSRF базируется на том, что сайт не проверяет, откуда пришел запрос, а доверяет данным сессии в cookie [7, с. 115].

5. Включение локальных файлов (LFI)

Включение локальных файлов (LFI, Local File Inclusion) – это метод атаки, который возникает при использовании PHP-функций include(), require(), include_once() и require_once() с пользовательским вводом без валидации. Злоумышленник может применять техники обхода каналов (directory traversal), такие как ../../etc/passwd, для доступа к файлам за пределами веб-корня. Это позволяет читать системные файлы (например, /etc/passwd), конфигурации или исходный код приложения.

При определённых условиях (например, если логи сервера содержат PHP-код или используются обходы вроде PHP-фильтров), LFI может привести к выполнению произвольного кода [3, с. 307].

6. DDoS-атаки

В отличие от атак, нацеленных на кражу данных, DDoS-атаки (Distributed Denial of Service) направлены на перегрузку ресурсов (CPU, память, bandwidth). Это приводит к тому, что сервис становится недоступным для легитимных пользователей.

Отказ сервера в обслуживании может быть критичным не только для коммерческих компаний, но и для государственных порталов, систем здравоохранения или финансовой инфраструктуры.

Ключевое отличие DDoS от DoS – распределенный характер атаки. При DDoS вместо одного компьютера используется ботнет (сеть устройств), что усиливает мощность атаки и маскирует ее под легитимный трафик. При этом для запуска такой атаки злоумышленнику не нужен доступ к серверу жертвы, а генерируемый трафик на начальном этапе может быть неотличим от обычного всплеска активности пользователей.

DDoS-атаки классифицируют по уровням модели OSI (Open Systems Interconnection):

1. Атаки прикладного уровня (L7). Тип атаки – HTTP-флуд, при котором ботнет отправляет тысячи запросов на скачивание большого файла или выполнение поиска, загружая процессор и базу данных сервера. Целью данного типа атак являются конкретные функции веб-приложения. Сложность защиты заключается в необходимости отличать «плохие» запросы от «хороших», так как и те, и другие выглядят легитимно.

2. Атаки транспортного уровня (L4). Тип атаки – SYN Flood, использует уязвимость в процедуре «тройного рукопожатия» TCP, отправляя множество запросов на соединение (SYN-пакетов), но не завершая его. Сервер резервирует ресурсы для каждого полуоткрытого соединения, и их пул быстро исчерпывается, блокируя доступ для новых легитимных пользователей.

3. Атаки сетевого уровня (L3). Типы атаки – UDP-флуд и ICMP-флуд. Они являются самыми грубыми, но эффективными типами атак, суть которых заключается в генерации трафика такой мощности (измеряется в гигабитах или терабитах в секунду), при которой полностью забивается интернет-канал жертвы. Целью данных атак является вытеснение полезного трафика [8].

7. Атаки с использованием DNS-туннелирования

DNS (Domain Name System) – это критически важная распределенная база данных, выполняющая функцию «телефонной книги интернета». Её основная задача – преобразовывать удобные для человека доменные имена (например, example.com) в IP-адреса, необходимые для установления сетевого соединения. Изначально созданная как надежный и незаменимый сервис, архитектура DNS практически не изменилась с момента своего создания, что делает её привлекательной мишенью для злоумышленников. Помимо основной функции, DNS используется для балансировки нагрузки и других задач [9, с. 91].

DNS-туннелирование – метод реализации кибератак, при котором злоумышленники используют инфраструктуру протокола DNS для передачи данных, скрывая трафик в DNS-запросах и ответах.

Алгоритм DNS-туннелирования:

1. Клиент формирует DNS-запрос, в котором часть доменного имени содержит закодированные данные.

2. Запрос отправляется на DNS-сервер, контролируемый злоумышленником или администратором туннеля.

3. Сервер извлекает данные из запроса.

4. В DNS-ответе сервер может передать обратно закодированную информацию. Таким образом создаётся скрытый канал связи, позволяющий передавать данные между системами.

DNS-туннелирование наиболее часто используется для обхода сетевых ограничений, эксфильтрации данных, управления вредоносным ПО, сокрытия сетевой активности.

8. Атаки с использованием больших языковых моделей (LLM)

В то время как SQL-инъекции и XSS-атаки атакуют уязвимости в коде, LLM (Large Language Models) атакуют через то, что является их основной функцией – понимание и выполнение инструкций на естественном языке, заставляя игнорировать системные ограничения через манипуляцию контекстом. Это создаёт новый класс угроз, где атака маскируется под обычный диалог, что затрудняет ее обнаружение.

Киберпреступники уже продемонстрировали нестандартные методы кражи данных, делегируя задачи поиска и передачи информации локальным

LLM-ассистентам, встроенным в ОС или браузеры. Вместо того чтобы внедрять сложный и легко обнаруживаемый вредоносный код, они манипулируют запросами к ИИ (искусственному интеллекту), который сам выполняет сканирование файловой системы и передачу данных. Такой подход позволяет кибератаке долгое время оставаться незамеченной, так как антивирусы пока не обучены анализировать и блокировать запросы к легитимным ИИ-сервисам на предмет вредоносности [10, с. 365-366].

Предоставление LLM-ассистентам доступа к файловой системе, почте и другим конфиденциальным данным усиливает риски.

Заключение

Безопасность веб-приложений остаётся одной из наиболее актуальных проблем современной информационной безопасности. К основным методам реализации кибератак можно отнести контрабанду HTTP-запросов, SQL-инъекции, межсайтовый скриптинг (XSS), межсайтовую подделку запросов (CSRF), включение локальных файлов (LFI), DDoS-атаки, атаки с использованием DNS-туннелирования и атаки с использованием больших языковых моделей (LLM). Большинство атак основано на недостаточной проверке пользовательских данных, ошибках проектирования и неправильной конфигурации систем.

Современные тенденции развития информационных технологий, включая интеграцию искусственного интеллекта и автоматизированных сервисов, приводят к появлению новых классов угроз, которые требуют дальнейших исследований и разработки эффективных механизмов защиты.

Таким образом, обеспечение безопасности веб-приложений должно включать комплекс мер, охватывающих этапы проектирования, разработки, тестирования и эксплуатации программных систем. Только системный подход к выявлению и устранению уязвимостей позволит снизить риски компрометации данных и обеспечить устойчивую и безопасную работу веб-сервисов.

Список литературы:

1. Еремин, Е. О. Обзор открытых наборов данных для выявления атак на веб-приложения / Е. О. Еремин // International Journal of Open Information Technologies. – 2024. – Т. 12. – № 3. – С. 106–112.
2. Шатурный, М. В. Анализ актуальных угроз и разработка подходов к защите веб приложений / М. В. Шатурный // Инженерный вестник Дона. – 2024. – № 7 (115).
3. Крылов, И. Д. Определение рисков информационной безопасности типа обхода Web Application Firewall / И. Д. Крылов, И. В. Кича, Д. П. Яковлев, А. А. Жданов, Д. К. Шульга, И. О. Елфимов, Г. В. Беликов, В. А. Селищев // Известия ТулГУ. Технические науки. – 2023. – № 8. – С. 305–309. DOI: 10.24412/2071-6168-2023-8-305-306.

4. Косов, Н. А. Анализ уязвимостей мобильных приложений на Android / Н. А. Косов, И. А. Голубничев // Экономика и качество систем связи. – 2023. – № 1 (27). – С. 84–91.

5. Казарян, К. К. Вредоносные запросы / К. К. Казарян, В. В. Белан // StudNet. – 2022. – № 1. – С. 518–524.

6. Щеглова, Д. Д. Угрозы для веб-сайтов и способы защиты от них / Д. Д. Щеглова // Парадигма. – 2025. – № 8.1. – С. 127–133.

7. Вильховский, Д. Э. Возможности ИИ в сфере кибербезопасности: вопросы обнаружения, предотвращения и реагирования на SQL-инъекции, XSS- и CSRF-атаки / Д. Э. Вильховский // Математические структуры и моделирование. – 2024. – № 4 (72). – С. 111–124. DOI:10.24147/2222-8772.2024.4.111-124.

8. Голубятников, А. О. DDoS-атаки и методы борьбы с ними / А. О. Голубятников // E-Scio. – 2022. – № 10 (73).

9. Москвичев, А. Д. Использование DNS-туннелирования для передачи вредоносного программного обеспечения / А. Д. Москвичев, К. С. Москвичева // Вопросы кибербезопасности. – 2022. – № 4 (50). – С. 91–99. DOI: 10.21681/2311-3456-2022-4-91-99.

10. Тагиев, Р. Н. Новая эра кибератак: как хакеры используют LLM для кражи данных на примере инцидента с пакетом NX / Р. Н. Тагиев // Вестник науки. – 2025. – № 9 (90). – С. 365–370. DOI: 10.24412/2712-8849-2025-990-365-371.