Сычев Д.И. Классификация основных проблем в безопасности при разработке серверных приложений // Международный журнал информационных технологий и энергоэффективности. – 2023. –
Т. 8 № 3(29) с. 9–16

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004

КЛАССИФИКАЦИЯ ОСНОВНЫХ ПРОБЛЕМ В БЕЗОПАСНОСТИ ПРИ РАЗРАБОТКЕ СЕРВЕРНЫХ ПРИЛОЖЕНИЙ

Сычев Д.И.
Санкт-Петербургский государственный университет телекоммуникаций имени профессора М.А. Бонч-Бруевича, Санкт-Петербург, Россия (193232, г. Санкт-Петербург, пр. Большевиков, 22, к. 1), e-mail: s.denis_2001@mail.ru

Серверные приложения являются критически важными компонентами современной разработки программного обеспечения, поскольку они отвечают за управление конфиденциальными данными, выполнение сложной бизнес-логики и интеграцию с внешними службами. Однако, поскольку частота и серьезность кибератак продолжают расти, важно понимать распространенные уязвимости и угрозы для серверных приложений, чтобы защитить их от атак. В этой статье рассмотрены некоторые наиболее распространенные уязвимости и угрозы для серверных приложений и обсудим способы их устранения.

Ключевые слова: программное обеспечение, серверные приложения, информационная безопасность.

TAXONOMY OF THE MAIN SECURITY PROBLEMS IN THE DEVELOPMENT OF BACKEND APPLICATIONS

Sychev D.I.
St. Petersburg State University of Telecommunications named after Professor M.A. Bonch-Bruevich, St. Petersburg, Russia (193232, St. Petersburg, Bolshevikov Ave., 22, room 1), e-mail: s.denis_2001@mail.ru

Backend applications are critical components of modern software development, as they are responsible for managing sensitive data, executing complex business logic, and integrating with external services. However, as the frequency and severity of cyber attacks continue to rise, it's essential to understand the common vulnerabilities and threats to backend applications to protect them against attacks. In this article, we'll explore some of the most common vulnerabilities and threats to backend applications and discuss how to mitigate them

Keywords: backend, software, development, security.

Введение

Серверные приложения играют важную роль в современном мире, особенно с ростом популярности веб и мобильных программ. Серверные приложения зачастую хранят и обрабатывают конфиденциальную информацию пользователей, такую как персональные данные, данные о финансовых операциях и другую приватную информацию. Если эти сведения будут похищены, они могут быть использованы злоумышленниками в разных целях, например, для компрометации пользователя или финансового мошенничества.

Защита серверных приложений помогает предотвратить несанкционированный доступ, взлом и другие нарушения безопасности, которые могут привести к потере или повреждению данных. Во многих отраслях и юрисдикциях действуют правила, требующие защиты определенных типов данных. Неспособность защитить эти данные может привести к юридическим и финансовым санкциям. А сам факт нарушения безопасности может нанести ущерб репутации организации и подорвать доверие клиентов.

1. Атаки с использованием инъекции

Инъекционные атаки — это тип кибератак, нацеленных на серверные приложения. При инъекционной атаке злоумышленник отправляет вредоносные данные в серверное приложение, чтобы использовать уязвимость в коде приложения. Это может позволить злоумышленнику выполнить несанкционированные команды или получить доступ к конфиденциальным данным, что приведет к их краже или полной потере, повреждению или другим негативным последствиям.

Одним из распространенных типов атак путем инъекции является SQL-инъекция. При подобном типе атаки злоумышленник отправляет вредоносный SQL запрос серверному приложению, которое на своей стороне использует SQL для взаимодействия с базой данных. Затем вредоносный код может быть запущен приложением, что позволит злоумышленнику получить доступ к данным в базе данных или изменить их. Это может привести к краже или модификации данных или позволить злоумышленнику получить контроль над всем приложением или базовой системой.

Другим типом инъекционной атаки является инъекция определенной команды. При атаке с инъекцией команд злоумышленник отправляет вредоносный код серверному приложению, которое взаимодействует с операционной системой. Затем злоумышленник может выполнять несанкционированные действия в системе, что позволяет ему получить доступ к конфиденциальным данным или получить контроль над самой системой. [1]

Для защиты от подобного типа инъекций серверные приложения должны реализовывать меры безопасности, такие как проверка ввода пользователя и дополнительная проверка на “чистоту” запроса. Проверка ввода подразумевает под собой обработку данных, которые пользователь отправляет на сервер в, на наличие ожидаемых значений и форматов, а “чистка” ввода включает форматирование или нейтрализацию потенциально вредоносных данных. Кроме того, параметризованные запросы и хранимые процедуры могут помочь предотвратить атаки путем внедрения кода SQL.

Также важно постоянно обновлять внутренние приложения с помощью последних исправлений и обновлений безопасности, поскольку поставщики программного обеспечения часто обнаруживают и устраняют уязвимости, которые могут быть использованы путем инъекционных атак. Регулярное тестирование безопасности и аудит также могут помочь выявить и устранить уязвимости, прежде чем они смогут быть использованы злоумышленниками.

Атаки путем инъекции кода представляют собой серьезную угрозу безопасности серверных приложений. Организации должны внедрять меры безопасности, такие как проверка входных данных и их форматирование, а также поддерживать свои приложения в актуальном состоянии с помощью последних правок и обновлений безопасности. Регулярное

тестирование безопасности и аудит также могут помочь выявить и устранить уязвимости, прежде чем они смогут быть использованы злоумышленниками.

2. Межсайтовый скриптинг

Межсайтовый скриптинг (XSS) — это еще один тип кибератак, нацеленных на серверные приложения. При атаке XSS злоумышленник внедряет вредоносный код на веб-страницу или в приложение, которое затем выполняется браузером жертвы при доступе к странице или приложению.

XSS-атаки могут иметь серьезные последствия, включая кражу конфиденциальных данных, таких как пароли и номера кредитных карт, перехват сеансов пользователей и распространение вредоносных программ среди других пользователей. Серверные приложения особенно уязвимы для XSS-атак, поскольку они часто генерируют динамический контент, который отправляется в браузеры пользователей. [2]

Существует несколько типов XSS-атак, в том числе:

- Неперсистентныая XSS атака: при неперсистентной XSS-атаке злоумышленник отправляет жертве ссылку, содержащую вредоносный скрипт. Когда жертва нажимает на ссылку, в ее браузере запускается сценарий, позволяющий злоумышленнику украсть конфиденциальные данные или захватить их сеанс.
- Stored XSS: при stored XSS-атаке злоумышленник внедряет вредоносный скрипт в веб-приложение, которое затем сохраняется на сервере. Когда другие пользователи получают доступ к приложению, скрипт выполняется их браузером, что позволяет злоумышленнику украсть их данные или захватить их сеансы.
- XSS на основе DOM: при атаке XSS на основе DOM злоумышленник внедряет вредоносный скрипт в веб-приложение, которое затем выполняется браузером жертвы. В отличие от других типов XSS-атак, скрипт не хранится на сервере, что затрудняет его обнаружение и предотвращение.

Для защиты от XSS-атак серверные приложения должны реализовывать несколько мер безопасности, в том числе:

- Проверка ввода. Проверка ввода включает проверку ввода пользователя на наличие ожидаемых значений и форматов. Проверяя и форматируя пользовательский ввод, серверные приложения могут помешать злоумышленникам внедрить вредоносный код в веб-страницы или приложения.
- Шифрование вывода: шифрование вывода включает в себя кодирование специальных символов в пользовательском вводе, чтобы они не выполнялись браузерами. Это не позволяет злоумышленникам внедрять вредоносные скрипты в веб-страницы или приложения.
- Политика безопасности контента (CSP): CSP — это стандарт безопасности, который позволяет веб-разработчикам указывать, какие источники контента разрешены для загрузки веб-страницей. Внедрив CSP, серверные приложения могут предотвратить загрузку злоумышленниками вредоносных сценариев или содержимого на веб-страницы или в приложения.

XSS-атаки представляют собой серьезную угрозу безопасности серверных приложений. Организации должны внедрять меры безопасности, такие как проверка ввода, кодирование

вывода и политики безопасности контента, чтобы предотвратить успешные атаки XSS. Регулярное тестирование безопасности и аудит также могут помочь выявить и устранить уязвимости, прежде чем они смогут быть использованы злоумышленниками.

3. Небезопасное криптографическое хранилище

Небезопасное криптографическое хранилище — это тип уязвимости кибербезопасности, которая может повлиять на серверные приложения. Когда серверное приложение хранит конфиденциальную информацию, такую как пароли или номера кредитных карт, оно должно использовать безопасный метод для шифрования и хеширования данных, чтобы предотвратить несанкционированный доступ. Однако, если используемый метод шифрования и хеширования является слабым или ошибочным, злоумышленники могут легко получить доступ к конфиденциальной информации.[3]

Есть несколько основных уязвимостей, которые приводят к небезопасному состоянию криптографического хранилища, в их числе:

- Слабые алгоритмы шифрования и хеширования. Если приложение использует слабый алгоритм шифрования и хеширования, злоумышленник может легко взломать шифрование или хэш и получить доступ к конфиденциальным данным.
- Плохое управление ключами: если приложение не хранить ключи шифрования добстаточно надежно, злоумышленник может украсть ключ и получить доступ к зашифрованным данным.
- Хранение паролей в виде обычного текста. Если приложение хранит пароли в виде обычного текста, злоумышленник, получивший доступ к базе данных или файловой системе приложения, может легко прочитать и использовать пароли.

Для защиты криптографического хранилища серверные приложения должны, в обязательном порядке, реализовывать такие меры безопасности, как:

- Надежные алгоритмы шифрования и хеширования. Серверные приложения должны использовать надежные алгоритмы шифрования и хеширования, такие как AES и SHA-256, для шифрования и хеширования конфиденциальных данных.
- Безопасное управление ключами. Ключами шифрования следует управлять безопасно, используя такие методы, как чередование ключей, разделение ключей и разделение ключей, чтобы предотвратить несанкционированный доступ.
- *Salt*-хеширование: при хешировании паролей серверные приложения должны использовать *salt*-хэширование, которое добавляет случайное значение к каждому паролю перед его хешированием. Это затрудняет взлом хэша злоумышленниками с помощью предварительно вычисленных радужных таблиц.
- Разделение данных. Конфиденциальные данные следует отделять и защищать с помощью таких методов, как шифрование, контроль доступа и разделение обязанностей, чтобы предотвратить несанкционированный доступ.

Незащищенное криптографическое хранилище является серьезной уязвимостью, которая может повлиять на серверные приложения. Для защиты от этой уязвимости организациям следует применять меры безопасности, такие как надежные алгоритмы шифрования и хеширования, безопасное управление ключами, *salt* хеширование и разделение

данных. Регулярное тестирование безопасности и аудит также могут помочь выявить и устранить уязвимости, прежде чем они смогут быть использованы злоумышленниками. [3]

4. Неполное логирование и мониторинг

Неполное ведение логов и мониторинг — это тип уязвимости кибербезопасности, когда приложение не регистрирует или не отслеживает события должным образом, затрудняя обнаружение инцидентов безопасности и реагирования на них. Злоумышленники могут использовать эту уязвимость, чтобы получить доступ к конфиденциальным данным или нарушить работу приложения, не будучи обнаруженными.

Можно выделить несколько основных инструментов, пренебрежение которыми со стороны разработчиков, не позволит своевременно выявить утечки и взломы в безопасности:

- Неполное ведение логов. Если приложение регистрирует только ограниченный набор событий, оно может пропустить важные события безопасности, которые могут указывать на взлом или атаку.
- Неэффективный анализ логов: даже если приложение регистрирует все соответствующие события, если эти логи не анализируются эффективно, инциденты безопасности могут остаться незамеченными.
- Отсутствие предупреждений: если приложение не отправляет предупреждения при возникновении событий безопасности, администраторы могут не знать об инцидентах безопасности, пока не станет слишком поздно.

Для решения проблемы недостаточного ведения логов и мониторинга серверные приложения должны реализовывать несколько мер безопасности, в том числе [8-9]:

- Эффективный анализ логов: логи следует регулярно анализировать для выявления аномалий и инцидентов безопасности. Это можно сделать вручную или с помощью автоматизированных инструментов, таких как системы управления информацией и событиями безопасности (SIEM).
- Оповещения: внутренние приложения должны отправлять оповещения при возникновении событий безопасности, используя такие механизмы, как уведомления по электронной почте или текстовые сообщения.
- Хранение и резервное копирование. Логи должны храниться в течение соответствующего периода времени, и необходимо регулярно делать резервные копии, чтобы предотвратить потерю данных в случае взлома или атаки.

Если вести некачественное логирование приложения и не уделять должного внимания мониторингу системы, это может привести к серьезной уязвимости, которая может повлиять на серверные приложения. Для защиты от этой уязвимости организациям следует применять меры безопасности, такие как комплексное ведение логирования, эффективный анализ логирования, оповещения, а также хранение и резервное копирование. Регулярное тестирование безопасности и аудит также могут помочь выявить и устранить уязвимости, прежде чем они смогут быть использованы злоумышленниками [5-7].

5. Небезопасные API

API или интерфейсы прикладного программирования представляют собой набор протоколов и инструментов, которые позволяют различным программным приложениям

взаимодействовать друг с другом. Когда API-интерфейсы спроектированы и реализованы небезопасно, они могут быть использованы злоумышленниками для получения несанкционированного доступа к конфиденциальным данным или выполнения вредоносного кода на сервере. [10]

Некоторые распространенные уязвимости, связанные с небезопасным API, включают:

- Недостаточный контроль аутентификации и авторизации: это может позволить злоумышленникам обойти аутентификацию и получить доступ к конфиденциальным данным или функциям.
- Атаки путем внедрения. Злоумышленники могут внедрять вредоносный код или данные через API, которые затем могут выполняться на сервере.
- Недостаточная проверка входных данных: API, которые не проверяют входные данные, могут быть использованы злоумышленниками для выполнения вредоносного кода.
- Небезопасная передача данных: API-интерфейсы, которые передают данные по небезопасным каналам, таким как HTTP, могут быть перехвачены и манипулированы злоумышленниками.

Чтобы снизить эти риски, разработчики должны следовать передовым методам обеспечения безопасности API, в том числе:

- Внедрение строгой проверки подлинности и средств контроля авторизации, таких как многофакторная проверка подлинности и управление доступом на основе ролей.
- Проверка входных данных и использование параметризованных запросов для предотвращения инъекций.
- Использование шифрования и безопасных протоколов, таких как HTTPS, для защиты данных при передаче.
- Регулярное тестирование и аудит API на наличие уязвимостей в системе безопасности.

Следуя этим рекомендациям, разработчики могут обеспечить безопасность своих API и защитить их от несанкционированного доступа и утечки данных.

Вывод

В заключение, защита серверных приложений имеет решающее значение для обеспечения безопасности и целостности данных пользователей и организаций. Недостаточное обеспечение защиты серверных приложений может привести к серьезным уязвимостям безопасности, таким как рассмотрены выше. Эти уязвимости могут быть использованы злоумышленниками для получения несанкционированного доступа к конфиденциальным данным, нарушения бизнес-операций или кражи ценной информации.

Чтобы защититься от этих уязвимостей, организациям следует применять комплексные меры безопасности, включая методы безопасного кодирования, регулярное тестирование и аудит безопасности, строгий контроль доступа и надлежащие методы шифрования и хеширования. Кроме того, мониторинг и логирование всех важных событий, эффективный анализ журналов и оповещение администраторов о уникальных событиях безопасности могут помочь организациям своевременно обнаруживать инциденты безопасности и реагировать на них [4-6].

Следуя этим рекомендациям безопасности и сохраняя бдительность в отношении возникающих угроз, организации могут защитить свои серверные приложения и сохранить доверие своих клиентов.

Список литературы

1. Krasov A. V., Shterenberg S. I. Methods for building a trusted environment in Unix operating systems based on the implementation of a digital watermark //2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). – IEEE, 2020. – С. 253-257.
2. Сахаров Д. В. и др. Разработка модели обеспечения отказоустойчивости сети передачи данных //Известия высших учебных заведений. Технология легкой промышленности. – 2016. – Т. 34. – №. 4. – С. 14-20.
3. Штеренберг С. И., Данилова Ю. С. Разработка методики внедрения и выявления эффективности siem-системы //Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2020. – №. 3. – С. 40-45.
4. More Security Best Practices for Backend Developers. MAY 4, 2020. John Au-Yeung URL: https://medium.com/swlh/more-security-best-practices-for-backend-developers-c7a41f85bf8e
5. Understanding the Importance and Value of Backend Security. TeskaLabs Blog. URL: https://teskalabs.com/blog/backend-security-importance
6. Writing Secure Code: Practical Strategies and Proven Techniques for Building Secure Applications in a Networked World. Michael H., David L.
7. Gelfand A. M. et al. Development of a model for the distribution of a manifesting code in a secure information system // Modern Science: Actual Problems of Theory and Practice. Series: Natural and technical sciences. – 2018. – no. 8. - pp. 91-97.
8. Krasov A. V. et al. SOFTWARE implementation of intrusion and anomalie prevention in the network infrastructure.
9. Гельфанд А. М., Гвоздев Ю. В., Штеренберг С. И. Исследования недостатков языков высокоуровнего программирования для осуществления скрытого вложения в исполнимые файлы //Актуальные проблемы инфотелекоммуникаций в науке и образовании. – 2015. – С. 295-297.
10. Пестов И. Е. и др. Выявление угроз безопасности информационных систем //Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2017). – 2017. – С. 525-527.
11. Зимин А. Е., Косов Н. А. Обеспечение информационной безопасности в процессе создания и использования программ для ЭВМ //Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2017). – 2017. – С. 343-348Unmanned Aircraft Systems (UAS) Frequently Asked Questions." Federal Aviation Administration, 26 Aug. 2021, www.faa.gov/uas/getting_started/fly_for_fun/frequently_asked_questions/.

References

1. Krasov A. V., Shterenberg S. I. Methods for building a trusted environment in Unix operating systems based on the implementation of a digital watermark //2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). – IEEE, 2020. – pp. 253-257.
2. Sakharov D. V. et al. Development of a model for ensuring the fault tolerance of a data transmission network // Izvestia of higher educational institutions. Light industry technology. 2016. - T. 34. - No. 4. - pp. 14-20.
3. Shterenberg S. I., Danilova Yu. S. Development of a methodology for implementing and identifying the effectiveness of a siem system // Bulletin of the St. Petersburg State University of Technology and Design. Series 1: Natural and technical sciences. – 2020. – no. 3. - pp. 40-45.
4. More Security Best Practices for Backend Developers. MAY 4, 2020. John Au-Yeung URL: https://medium.com/swlh/more-security-best-practices-for-backend-developers-c7a41f85bf8e
5. Understanding the Importance and Value of Backend Security. TeskaLabs Blog. URL: https://teskalabs.com/blog/backend-security-importance
6. Writing Secure Code: Practical Strategies and Proven Techniques for Building Secure Applications in a Networked World. Michael H., David L.
7. Gelfand A. M. et al. Development of a model for the distribution of a manifesting code in a secure information system // Modern Science: Actual Problems of Theory and Practice. Series: Natural and technical sciences. – 2018. – no. 8. - pp. 91-97.
8. Krasov A. V. et al. SOFTWARE IMPLEMENTATION OF INTRUSION AND ANOMALIE PREVENTION IN THE NETWORK INFRASTRUCTURE.
9. Gelfand A. M., Gvozdev Yu. V., Shterenberg S. I. Investigation of the shortcomings of high-level programming languages for the implementation of hidden attachments to executable files // Actual problems of infotelecommunications in science and education. - 2015. - S. 295-297.
10. Pestov I. E. et al. Identification of threats to the security of information systems // Actual problems of infotelecommunications in science and education (APINO 2017). - 2017. - pp. 525-527.
11. Zimin A. E., Kosov N. A. Ensuring information security in the process of creating and using computer programs // Actual problems of infotelecommunications in science and education (APINO 2017). - 2017. - pp. 343-348.