DOI 10.37539/2949-1991.2025.24.1.027
УДК 004.056

Дмитриев Дмитрий Валерьевич,
к.т.н., доцент кафедры “Информатика и Системы Управления”
Нижегородский государственный технический университет
им. Р.Е. Алексеева, Нижний Новгород, Россия

Вайнбаум Денис Алексеевич, магистрант,
Нижегородский государственный технический университет
им. Р.Е. Алексеева, Нижний Новгород, Россия

Исаев Максим Александрович, магистрант,
Нижегородский государственный технический университет
им. Р.Е. Алексеева, Нижний Новгород, Россия

Мельников Роман Васильевич, магистрант,
Нижегородский государственный технический университет
им. Р.Е. Алексеева, Нижний Новгород, Россия

ОЦЕНКА МЕХАНИЗМОВ УПРАВЛЕНИЯ И РАЗГРАНИЧЕНИЯ ПРАВ ДОСТУПА PYTHON BACKEND-ПРИЛОЖЕНИЙ

Аннотация: Статья посвящена различным методам и решениям авторизации в python-приложениях. Анализируются уязвимости в системах для управления доступами в Python. Рассматриваются основные модели управления доступом. На основе предложенных критериев выбирается подходящая библиотека и метод авторизации, что поможет выбрать оптимальный вариант в зависимости от требований приложения.

Ключевые слова: авторизация, безопасность, уязвимости безопасности, атаки на приложения, защита от атак, оценка безопасности.

Введение

Ограничение доступа к данным является одним из методов обеспечения безопасности в приложениях. Однако, несмотря на меры по управлению правами доступа, существуют значительные риски, такие как уязвимости в методах авторизации, которые могут быть использованы злоумышленниками для несанкционированного доступа. Исследования, включая данные OWASP Top-10 и компаний Positive Technologies и Kaspersky, показывают, что уязвимость «Недостатки контроля доступа» остается одной из самых критичных. В 2019 году 37% тестируемых систем [1] имели возможность несанкционированного доступа, и 16% сайтов позволяли злоумышленникам получить полный контроль. В 2020-2021 годах эта уязвимость занимала первое место [2] в антирейтингах.

Уязвимости в правах доступа присутствуют в приложениях на любом языке программирования, и Python-приложения часто сталкиваются с этими проблемами, что обусловлено популярностью данной технологии. Сложные системы с множеством функций, особенно уязвимы к проблемам безопасности прав доступа [3]. Одним из типов таких масштабных приложений являются социальные сети. Крупные платформы стремятся к высокому уровню безопасности, но стартапы, выходящие на рынок, часто подвержены большим уязвимостям, включая проблемы с правами доступа.

Существующие проблемы безопасности прав доступа

Уязвимости авторизации представляют серьезную угрозу безопасности приложений, так как могут привести к несанкционированному доступу к конфиденциальным данным и функциям. Причины возникновения [4] проблем контроля прав доступа можно разделить на четыре группы:

1. отсутствие защиты компонентов системы;
2. проблемы проектирования и реализации систем авторизации;
3. уязвимости инфраструктуры;
4. проблемы поддержки системы авторизации.

Отсутствие защиты включает незащищенные API-методы [5] и фоновые задачи. Проблемы проектирования охватывают неправильный выбор модели прав доступа, некорректную настройку политики безопасности и ошибки в обработке пользовательских данных [6]. Уязвимости инфраструктуры связаны с неправильными настройками серверов и использованием устаревших компонентов. Проблемы поддержки включают отсутствие механизмов аудита, управления списками контроля доступа и регулярной ревизии прав [6].

Для предотвращения уязвимостей необходимо тщательно проектировать систему авторизации, правильно настраивать политики безопасности, регулярно обновлять компоненты и проводить аудит прав доступа.

Модели прав доступа

Политика управления доступом определяет правила взаимодействия компонентов защиты информации. Существуют статические и динамические модели доступа. Основные модели включают мандатную, дискреционную, ролевую и атрибутную. Мандатная модель (MAC) регулирует доступ на основе меток безопасности пользователей и ресурсов [7]. Она может быть иерархической или неиерархической, обеспечивая высокий уровень защиты, но сложна в администрировании. Дискреционная модель (ДУД) позволяет владельцам ресурсов определять права доступа, часто используя списки контроля доступа (ACL) [7]. Она гибкая, но может стать сложной в управлении при большом количестве пользователей. Ролевая модель (RBAC) предоставляет доступ на основе ролей пользователей [7]. Она упрощает администрирование и позволяет реализовать принципы "минимальной привилегии" и "разделения обязанностей". Атрибутная модель (ABAC) использует комбинацию атрибутов [8] пользователя, ресурса, действия и контекста для принятия решений о доступе. Она обеспечивает гранулярный контроль, но требует тщательной подготовки. Некоторые динамические модели, например, риск-ориентированные, могут быть основаны на атрибутивной модели, позволяя создавать комплексные механизмы управления доступом для масштабных информационных систем.

Существующие решения авторизации

Авторизация в приложениях часто реализуется с помощью специализированных библиотек. Эти инструменты предоставляют функционал для управления доступом, позволяя разработчикам определять и внедрять политики безопасности. Выбор подходящей библиотеки зависит от специфики проекта и требований безопасности. Ниже рассмотрены популярные Python-библиотеки для авторизации.

Casbin является универсальной библиотекой, поддерживающая ACL, RBAC и ABAC модели [9]. Использует файлы модели и политики, обрабатываемые компонентом Enforcer. Предоставляет широкий функционал, но может быть сложна в настройке. Balrog ориентирована на системы с предопределенными ролями. Роли и разрешения задаются статически в коде, что упрощает тестирование и версионирование. Vakt представляет собой ABAC-инструмент, обеспечивающий гибкое управление доступом на основе атрибутов [10]. Позволяет динамически изменять политики и использует многоуровневое кэширование.

Authoritah позволяет реализовать RBAC с контекстно-зависимыми ролями. Обеспечивает гибкое управление доступом, но может усложнить систему при частом изменении ролей. Access Control и Permission являются простыми библиотеками для базового управления доступом. Access Control работает с правилами, связывающими пользователя, действие и ресурс. Permission разработана специально для Flask.

Выбор механизма авторизации зависит от масштаба и сложности системы. Для простых приложений достаточно ACL (Casbin, Access Control, Permission). Для систем с группами пользователей подходит RBAC (Casbin, Balrog, Authoritah). Сложные бизнес-правила требуют ABAC (Casbin, Vakt, PyABAC). Дополнительными факторами выбора являются:

- популярность решения (размер сообщества);
- наличие механизмов аудита и логирования (Vakt и Casbin);
- возможности управления правами доступа (Casbin);
- производительность (кэширование в Vakt, оптимизация в Casbin);
- наличие механизмов проверки правил на непротиворечивость (отсутствие во всех рассмотренных библиотеках).

Заключение

Большинство существующих причин проблем безопасности прав доступа могут быть устранены применением существующих инструментов авторизации, соответствующих требованиям проектов. Однако рассмотренные решения имеют свои ограничения. Для крупномасштабных информационных систем, таких как социальные сети, наиболее подходящими являются Casbin и Vakt, обеспечивающие гибкую настройку политики безопасности. Однако ни одно решение не обладает всеми необходимыми функциями. Возможные решения этих ограничений включают: комбинирование инструментов, интеграцию одной из библиотек с дополнительной разработкой функций, необходимых проектам, применение динамических моделей, например, риск-ориентированных. Также возможна и полная реализация системы авторизации, однако данный подход может привести к другим угрозам безопасности.

Список литературы:

1. Уязвимости и угрозы веб-приложений в 2019 году [Электронный ресурс]. – Режим доступа: https://www.ptsecurity.com/ru-ru/research/analytics/web-vulnerabilities-2020/ (дата обращения: 16.01.2025).

2. Уязвимости и угрозы веб-приложений в 2020–2021 гг [Электронный ресурс]. – Режим доступа: https://www.ptsecurity.com/ru-ru/research/analytics/web-vulnerabilities-2020-2021/ (дата обращения: 16.01.2025).

3. Wen S. A quantitative security evaluation and analysis model for web applications based on OWASP application security verification standard / S. Wen, B. Katt // Computers & Security. – 2023. – Vol. 135. – Article 103532.

4. Неисправный контроль доступа [Электронный ресурс]. – Режим доступа: https://securelist.ru/top-10-web-app-vulnerabilities/109215/ (дата обращения: 16.01.2025).

5. Bararia A. Systematic Review of Common Web-Application Vulnerabilities / A. Bararia, Ms. V. Choudhary // Interantional journal of scientific research in engineering and management. – 2023. – Vol. 07, No. 01.

6. Уязвимость, связанная с нарушением авторизации [Электронный ресурс]. – Режим доступа: https://knowledge-base.secureflag.com/vulnerabilities/broken_authorization/broken_authorization_vulnerability.html (дата обращения: 17.01.2025).

7. Бопп В. А. Типы моделей разграничения доступа / В. А. Бопп // Известия Тульского государственного университета. Технические науки. – 2020. – № 5. – С. 233-236.

8. Чаус Е. А. Построение обобщенной модели контроля доступа на основе матрицы контроля доступа, ролевой и атрибутной моделей / Е. А. Чаус // Молодой ученый. – 2016. – № 16 (120). – С. 53-56.

9. Документация Casbin [Электронный ресурс]. – Режим доступа: https://casbin.org/docs (дата обращения 17.01.2025).

10. Документация Vakt [Электронный ресурс]. – Режим доступа: https://github.com/kolotaev/vakt (дата обращения 17.01.2025).