DOI 10.37539/2949-1991.2026.36.1.004
УДК 004.056

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

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

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

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

СРАВНИТЕЛЬНЫЙ АНАЛИЗ БИБЛИОТЕККОНТРОЛЯ ДОСТУПА НА PYTHON ДЛЯ SAAS-СЕРВИСОВ НА ОСНОВЕ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Аннотация. В статье представлены методика и результаты сравнительного анализа библиотек контроля доступа Casbin, Vakt и Balrog для SaaS-сервисов. Анализ основан на имитационном моделировании с использованием платформы Mesa, где агенты воспроизводили действия злоумышленников, пользователей и администраторов. Ключевыми критериями оценки стали безопасность (устойчивость и ущерб), производительность под нагрузкой и сложность управления политиками. Тестирование в две итерации показало, что все библиотеки при корректной настройке достигают максимальной безопасности, однако лучший баланс характеристик продемонстрировала библиотека Vakt, что делает её рекомендуемым выбором для динамичных SaaS-приложений.

Ключевые слова: Контроль доступа, авторизация, SaaS, Python, Casbin, Vakt, Balrog, имитационное моделирование, Mesa, безопасность, производительность.

Введение

Актуальной задачей при разработке SaaS-сервисов является выбор библиотеки контроля доступа, обеспечивающей не только безопасность, но и высокую производительность при управляемых затратах на администрирование. Для проведения комплексного сравнения таких библиотек на языке Python (Casbin, Vakt, Balrog) в условиях, приближенных к реальным, была применена методика имитационного моделирования.

В качестве инструмента моделирования выбрана платформа Mesa, позволяющая создавать агентные модели. В рамках исследования была разработана методология тестирования, включающая три ключевых критерия: «Уязвимости» (оценивается коэффициентом успешных атак и потенциальным ущербом), «Производительность» (время ответа под нагрузкой) и «Сложность управления» (количество шагов для изменения политик). Для каждого критерия в среде Mesa создавались специализированные агенты: «Злоумышленник», «Пользователь» и «Администратор». Тестирование каждой библиотеки проводилось в две итерации: на первой выявлялись уязвимости в базовой конфигурации, на

второй – оценивались итоговые показатели после устранения найденных недостатков. Данный подход позволил получить количественную оценку пригодности библиотек для использования в высоконагруженных SaaS-приложениях.

Методология тестирования

Для проведения сравнительного анализа была разработана концепция тестирования, интегрирующая выбранные библиотеки в единую тестовую среду на основе Mesa. Критерии оценки были разделены на три группы, каждой из которых соответствовал отдельный тип агента-имитатора.

1. Критерий «Уязвимости»: Оценивает устойчивость политик доступа к логическим атакам и потенциальный ущерб от успешной компрометации. Данный критерий разделен на две составляющие:

а) Устойчивость: Количественно измеряется коэффициентом успешности атак (K), рассчитываемым по формуле 1

$$K = \frac{k}{n} * 100\%, \quad (1)$$

где k – количество успешных попыток несанкционированного доступа;

n – общее количество попыток.

б) Ущерб: Оценивает качественные последствия атаки. Общий ущерб (D_{total}) рассчитывается по формуле (2) как сумма нормированных оценок ущерба от каждой успешной атаки (d_i).

$$D_{total} = \sum_{i=1}^{n} d_i, \quad (2)$$

где D_{total} – количественная метрика общего ущерба;

n – количество удавшихся атак в рассматриваемом сценарии;

d_i – количественная оценка ущерба от *i-й* успешной атаки.

Количественная метрика ущерба d_i для каждой атаки принимает значения на нормированной числовой шкале: $d_i \in [1,10] \subset R$. В качестве агента для этого критерия использовался класс **Злоумышленник**.

2. Критерий «Производительность»: Цель – оценка поведения системы под высокой нагрузкой, имитирующей DDoS-атаку. Количественной метрикой выступило время ответа защищенного endpoint'a. Агент – **Пользователь**.

3. Критерий «Сложность управления»: Цель – измерение трудоемкости настройки и изменения правил доступа. Метрики: количество шагов и количество правил, необходимых для защиты endpoint'a. Агент – **Администратор**. Общая схема тестирования представлена на Рисунке 1.

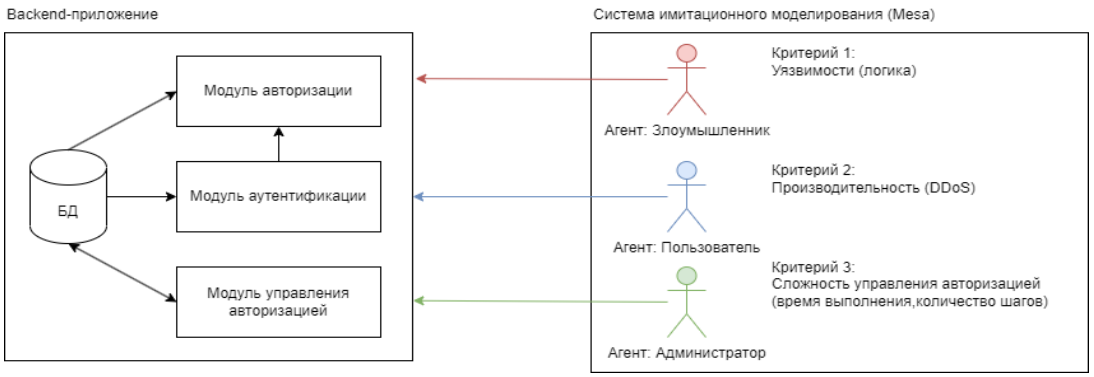

Рисунок 1. Общая схема методологии тестирования библиотек контроля доступа.

Тестирование каждой библиотеки проводилось в две итерации. На первой итерации использовалась базовая, логически неполная конфигурация правил, что позволило выявить

типичные уязвимости. На второй итерации правила дорабатывались для устранения найденных проблем, после чего проводились повторные замеры всех критериев.

Результаты тестирования и сравнительный анализ

Тестирование библиотек проводилось по единой методологии в две итерации. Первая итерация выявила уязвимости в базовых конфигурациях, вторая – оценила итоговые показатели после их устранения.

1. Библиотека Vakt (ABAC). Исходное правило содержало недостаточные контекстные проверки, что привело к успешным атакам с коэффициентом 16.7% и ущербом 42 единицы. После доработки политик уязвимости были устранены (K=0%, D_{total}=0). Библиотека показала рекордную производительность (стабильный отклик при 1000 агентах) и наилучшую управляемость (4 шага на добавление правила). Итог: Безопасность – 10.0, Производительность – 10.0, Управляемость – 8.34. Общая оценка – 9.45.

2. Библиотека Balrog (RBAC+ABAC). В начальной конфигурации обнаружились проблемы с наследованием ролей (K=25%, D_{total}=58). После упрощения иерархии и добавления приоритетных deny-правил безопасность достигла максимума. Производительность была хорошей, но не оптимальной, а управляемость низкой (7 шагов на добавление ресурса). Итог: Безопасность – 10.0, Производительность – 7.96, Управляемость – 6.08. Общая оценка – 8.01.

3. Библиотека Casbin (RBAC/ABAC/ACL). Базовая политика была уязвима к подмене user_id (K=33.3%, D_{total}=67). Безопасность во второй итерации обеспечена за счет детализированных политик. Casbin показала наивысшую производительность, но катастрофически низкую управляемость (12 шагов на новый сценарий). Итог: Безопасность – 10.0, Производительность – 9.84, Управляемость – 3.30. Общая оценка – 7.71.

Сравнительный анализ. Тестирование подтвердило, что безопасность достигается грамотной конфигурацией, а не выбором модели. Ключевые различия проявились в управляемости и производительности (рисунок 2). **Vakt** показал лучший баланс и рекомендуется для динамичных SaaS-сервисов. **Balrog** – надежное, но менее производительное решение с усложненной конфигурацией. **Casbin** – инструмент для экспертов в стабильных системах, где гибкость и скорость важнее простоты управления.

Интеграция выбранного решения. На основе анализа для интеграции выбран Vakt. Рекомендуется модульная архитектура, разделяющая аутентификацию, авторизацию (ядро на Vakt), управление политиками и аудит. Это обеспечивает масштабируемость, производительность и безопасность.

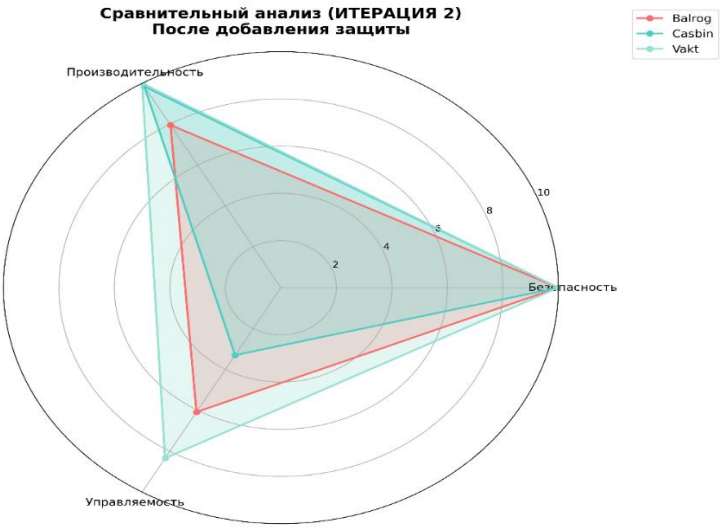

Рисунок 2. Сравнительный анализ библиотек

Заключение

Проведенное исследование с применением имитационного моделирования в среде Mesa позволило выполнить комплексный сравнительный анализ библиотек контроля доступа. Результаты подтвердили, что безопасность является в первую очередь следствием грамотной конфигурации, а не наследуемым свойством инструмента. По итогам оценки по критериям безопасности, производительности и управляемости библиотека Vakt продемонстрировала наилучший баланс и рекомендуется для использования в динамичных SaaS-сервисах. Balrog представляет собой надежное, но менее производительное решение с усложненным управлением, а Casbin применима в стабильных системах, где высочайшая производительность и гибкость оправдывают высокую сложность администрирования.

Список литературы:

1. Документация Mesa [Электронный ресурс]. – Режим доступа: https://mesa.readthedocs.io/latest/ (дата обращения 08.01.2026).

2. Alfadel M. Empirical analysis of security vulnerabilities in Python packages / M. Alfadel, D.E. Costa, E. Shihab // Empirical Software Engineering. – 2023. – Vol. 28, Issue 3.

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. Шуриков, В. В. Сравнительный анализ моделей для анализа поведения нарушителей безопасности при реализации кибератаки / В. В. Шуриков // Информационные технологии и интеллектуальные системы: сборник научных трудов по итогам III ежегодной национальной конференции, Москва, 18–20 марта 2025 года. – Москва: РТУ МИРЭА, 2025. – С. 1234-1238.