DOI 10.37539/2949-1991.2025.29.6.016
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

ОЦЕНКА СИСТЕМ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PYTHON

Аннотация. В статье исследуются возможности систем имитационного моделирования для анализа уязвимостей в управлении доступом на Python. Проводится сравнение библиотек по ключевым критериям: тип моделирования, масштабируемость и визуализация. На основе анализа выбирается оптимальный инструмент для моделирования различных сценариев авторизации

Ключевые слова: Имитационное моделирование, системы моделирования, авторизация, уязвимости безопасности, атаки на приложения, защита от атак.

Введение

При исследовании механизмов авторизации веб-сервисов необходимо применять инструменты, позволяющие имитировать пользователей. Для тестирования авторизации можно применять системы имитационного моделирования (СИМ). В общем, СИМ, представляют собой программные платформы для создания и анализа виртуальных моделей, имитирующих реальные или вымышленные процессы и явления [1]. С их помощью можно безопасно исследовать поведение систем в различных условиях, не прибегая к дорогостоящим и рискованным экспериментам в реальности. К преимуществам использования СИМ можно отнести следующее:

- анализ новых систем и процессов в условиях, максимально приближенных к реальным;
- наблюдение за долгосрочным поведением системы в разных сценариях;
- корректировка параметров и условий для изучения влияния на систему;
- проведение экспериментов без риска повреждения реальных объектов;
- обнаружение скрытых проблем и разработка решений;
- предсказание будущего поведения системы и разработка оптимальных стратегий управления.

СИМ находят применение в прогнозировании экономических трендов, оптимизации бизнес-процессов, управлении транспортными потоками, планировании производства и

многих других областях. Рассмотрим несколько систем имитационного моделирования, подходящих для серверных приложения, реализованных на языке программирования Python.

Существующие системы моделирования на языке python

В качестве первой системы рассмотрим пакет, основанный на Python, SimPy [2]. Он предоставляет гибкие инструменты для моделирования систем авторизации, позволяя воспроизводить взаимодействие пользователей с защищенными ресурсами в дискретно-событийной парадигме. В рамках модели каждый пользователь представляется в виде отдельного процесса, который инициирует запросы к API-эндпоинтам с заданными интервалами. Эндпоинты моделируются как ресурсы с ограниченной емкостью (например, максимальное число одновременных запросов) и политиками доступа, основанными на иерархии ролей (RBAC). Для проверки прав доступа вводится механизм валидации, который анализирует роль пользователя и сравнивает её с требуемым уровнем привилегий для конкретного метода (GET, POST, PUT, DELETE). Если проверка не пройдена, процесс пользователя получает отказ, что фиксируется в статистике системы для дальнейшего анализа уязвимостей.

Моделирование уязвимостей безопасности реализуется через преднамеренное нарушение политик доступа в рамках пользовательских процессов. Например, для воспроизведения горизонтального обхода пользователь может случайным образом подменять идентификатор ресурса в запросе, пытаясь получить доступ к данным другого пользователя с аналогичной ролью. Вертикальный обход имитируется путём отправки запросов к эндпоинтам, требующим более высоких привилегий, без соответствующих прав. Отсутствие проверки прав доступа моделируется через условное отключение механизма валидации для части запросов. Статистика успешных и неудачных попыток позволяет количественно оценить устойчивость системы к каждому типу угроз. Результаты моделирования визуализируются в виде графиков (рисунок 1) и таблиц, демонстрирующих частоту нарушений и эффективность применяемых механизмов защиты.

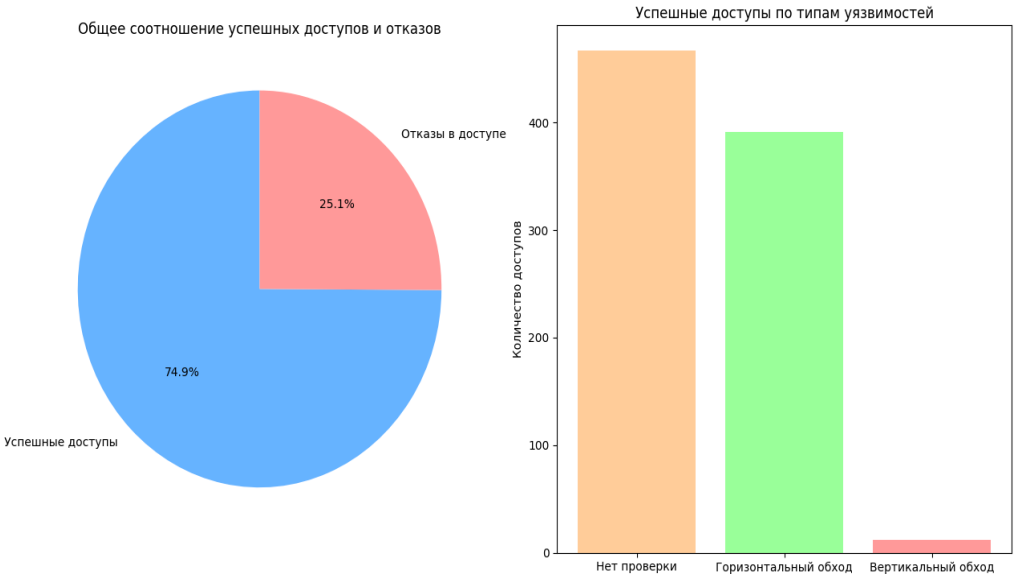

Рисунок 1. – Соотношения результатов симуляции СИМ SimPy

Второй рассматриваемой системой является Mesa – платформа агентного моделирования на Python, предназначенная для анализа сложных адаптивных систем и emergent behavior (возникновения системных свойств, не присущих отдельным агентам) [3]. В отличие от процессно-ориентированных инструментов (SimPy, AnyLogic), Mesa фокусируется

на взаимодействии агентов, каждый из которых обладает собственными атрибутами и поведенческими правилами. Среда моделирования может быть представлена в виде сетки, графа или иной структуры, что позволяет анализировать влияние локальных взаимодействий на глобальную динамику системы.

В рамках исследования разработана агентно-ориентированная модель системы авторизации, где пользователи различных ролей (гость, участник, администратор и др.) взаимодействуют с API-эндпоинтами, включая защищенные и уязвимые точки доступа. Сбор статистики (рисунок 2) включает частоту успешных/неудачных запросов, распределение уязвимостей среди ролей и динамику эксплуатации слабых мест системы. Это позволяет количественно оценить устойчивость механизмов авторизации к различным типам атак.

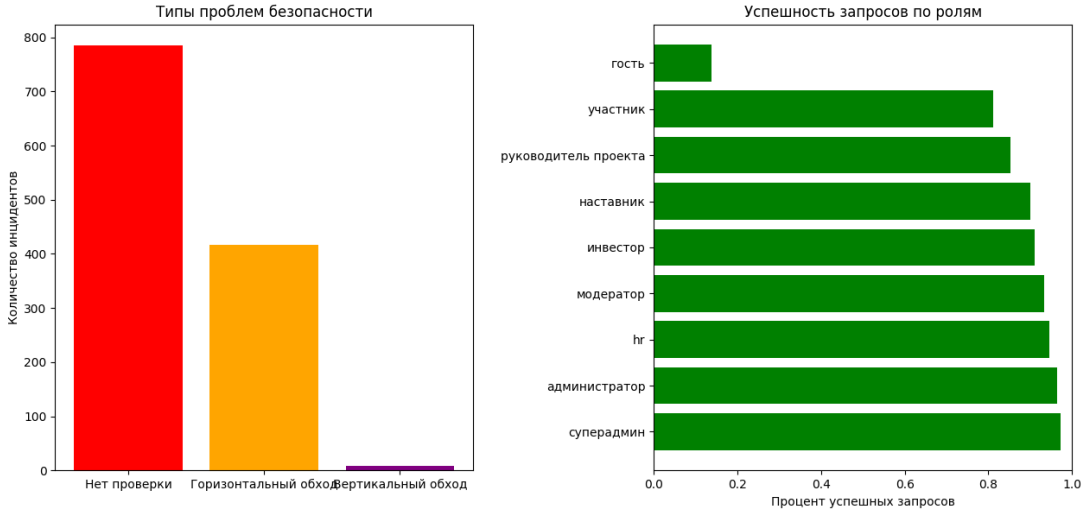

Рисунок 2. – Графические результаты симуляции библиотеки Mesa

Третьей рассматриваемой системой является PyCX – это Python-фреймворк для агентного моделирования, ориентированный на изучение сложных систем и эмерджентного поведения [4]. В отличие от SimPy, он предлагает упрощенный синтаксис для быстрого прототипирования, поддерживая визуализацию в реальном времени, что делает его удобным для образовательных и исследовательских задач в социальных науках, биологии и экономике. Однако PyCX уступает Mesa в масштабируемости и больше подходит для академического использования, чем для промышленного применения. В рамках данного исследования с его помощью реализована агентно-ориентированная модель системы авторизации API, где пользователи различных ролей взаимодействуют с эндпоинтами, классифицированными по типам уязвимостей: отсутствие проверки прав, горизонтальный и вертикальный обход привилегий. Модель, основанная на трех ключевых функциях (инициализация, наблюдение, обновление), собирает статистику успешных и неудачных запросов, визуализируя динамику атак для оценки эффективности механизмов контроля доступа.

Заключение

Проведённый анализ позволяет сделать вывод, что выбор системы имитационного моделирования (СИМ) для исследования механизмов авторизации веб-сервисов должен основываться на специфике решаемых задач. Библиотека SimPy демонстрирует оптимальную эффективность при дискретно-событийном моделировании, таком как анализ очередей и обработка запросов, благодаря своей процессно-ориентированной архитектуре. Однако её применение ограничено в контексте многоагентных систем, где требуется моделирование сложных взаимодействий между автономными сущностями.

Напротив, Mesa, как специализированный фреймворк для агентного моделирования, предоставляет расширенные возможности для анализа emergent behavior и взаимодействий между агентами, включая поддержку визуализации и гибких сценариев. Это делает её предпочтительным выбором для исследования систем авторизации, где критически важны моделирование поведения пользователей, злоумышленников и администраторов, а также оценка устойчивости к атакам. Однако следует учитывать, что Mesa требует больше вычислительных ресурсов по сравнению с SimPy, особенно при масштабировании моделей. PyCX, несмотря на свою простоту и удобство для прототипирования и образовательных целей, уступает Mesa в функциональности и масштабируемости, что ограничивает его применение в промышленных и исследовательских проектах, требующих глубокого анализа и высокой точности.

В рамках данного исследования механизмы прав доступа для системы авторизации будут тестироваться по трём ключевым критериям (рисунок 3):

1. Количество выявленных уязвимостей – оценка устойчивости системы к нарушениям логики бизнес-правил.

2. Время работы системы авторизации – анализ производительности системы при применении различных механизмов контроля доступа.

3. Сложность управления – измерение времени, необходимого для применения новой конфигурации модели авторизации, а также количества шагов, требуемых для изменения политики безопасности.

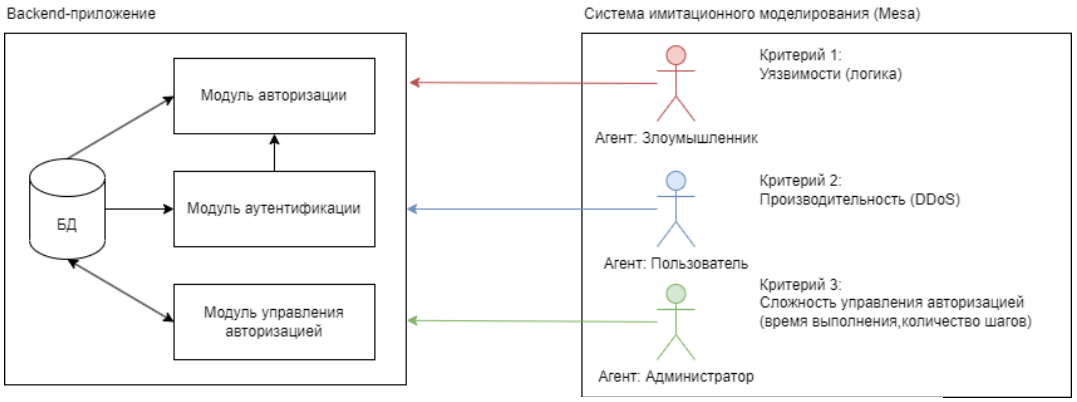

Рисунок 3. – Схема тестирования системы авторизации

Эти критерии позволят не только оценить текущее состояние системы, но и разработать рекомендации по её оптимизации, обеспечивая баланс между безопасностью, производительностью и удобством администрирования.

Список литературы:

1. Banks J. Discrete-Event System Simulation / J. Banks, J. S. Carson II, B. L. Nelson, D. M. Nicol. – 5th ed. – Pearson, 2019. – 640 p.

2. Документация SimPy [Электронный ресурс]. – Режим доступа: https://simpy.readthedocs.io/en/latest/ (дата обращения 29.05.2025).

3. Документация Mesa [Электронный ресурс]. – Режим доступа: https://mesa.readthedocs.io/latest/ (дата обращения 29.05.2025).

4. Документация PyCX [Электронный ресурс]. – Режим доступа: https://github.com/hsayama/PyCX (дата обращения 29.05.2025).