Разработка программного обеспечения моделирования угроз для систем на базе LLM-агентов

О.Р. Лапонина, Р.Н. Костин

Аннотация **– В данной статье описано программное обеспечение, расширяющее возможности проекта OWASP Threat Dragon моделирования угроз для систем на базе LLM-агентов. Добавлены новые компоненты диаграммы, атрибуты новых и существующих в Threat Dragon компонентов, а также возможность автоматической идентификации угроз на основе правил. В качестве эталонной структуры LLM-агента взята структура, определенная OWASP. Разработанное ПО добавляет поддержку компонентов LLM-агентов, позволяет задавать новых 27 атрибутов и автоматически выявляет угрозы по 38 правилам. Правило состоит множества условий, при которых может возникнуть угроза, описания сценария, взятого из документации OWASP, используемых смягчений данной угрозы, которые могут быть проактивными, реактивными и детективными. В правиле также указывается тип угрозы по методологии STRIDE. Диаграмма потоков анализируемой системы строится вручную, атрибуты всех компонент также определяются вручную. Затем на основе диаграммы, атрибутов и правил на диаграмму автоматически добавляются угрозы. При выполнении хотя бы одного условия угроза будет добавлена на диаграмму Threat Dragon для соответствующего компонента. Авторами определен достаточно простой синтаксис правил, что позволяет легко описывать новые угрозы, которые будут затем автоматически добавляться на диаграмму. Разработка получила положительные отзывы от лидеров OWASP Threat Dragon. Она закрывает важный пробел в области безопасности LLM-агентов и предлагает гибкий, расширяемый инструмент для практического применения. В статье также проведен обзор семи методологий моделирования угроз: STRIDE, LINDDUN, PASTA, STRIDE-AI, PLOT4ai, ADMIn и MAESTRO. Проведен обзор программного обеспечения моделирования угроз: OWASP Threat Dragon, OWASP pytm, Microsoft Threat Modeling Tool и ThreatFinderAI.**

Ключевые слова **– LLM-агенты, моделирование угроз, кибербезопасность, безопасность ИИ, агентные системы, OWASP Threat Dragon, автоматическая идентификация угроз, анализ угроз ИИ.**

I. Введение

Агенты на основе больших языковых моделей [1] (LLM, Large Language Models) являются наиболее интересным и многообещающим вариантом применения ИИ. Для каждой новой технологии понимание связанных с ней угроз и разработка мер безопасности занимает время, оставляя больше возможностей для атакующего произвести успешную атаку и остаться незамеченным.

Моделирование угроз [2] – это процесс, который систематизирует разработку безопасных систем. Данный процесс позволяет разрабатывать безопасные системы уже на этапе проектирования или эффективно анализировать существующие.

Применение моделирования угроз к системам на базе LLM-агентов имеет особую важность из-за популярности и новизны направления. Требуется поддерживать необходимый уровень безопасности и учитывать риски в условиях неопределенности поведения данных систем, недостаточной изученности угроз и отсутствия устоявшихся методов безопасности.

II. Цель и задачи работы

Целью данной работы является разработка ПО для моделирования угроз систем на базе LLM-агентов.

Задачи:

1. Изучить процесс моделирования угроз;
2. Изучить структуру LLM-агентов;
3. Изучить угрозы, связанные с LLM-агентами;
4. Провести обзор методологий моделирования угроз;
5. Провести обзор ПО моделирования угроз;
6. Выработать требования для разработки;
7. Разработать ПО для моделирования угроз систем на базе LLM-агентов на основе выработанных требований.

III. Моделирование угроз

Моделирование угроз – это деятельность, направленная на идентификацию и устранение угроз безопасности приложения. В данном процессе формируется модель угроз, которая является структурным представлением всей собранной информации, связанной с безопасностью системы. Модель угроз позволяет взглянуть на

Статья получена 22 мая 2025.
О.Р. Лапонина – МГУ имени М.В. Ломоносова (email: laponina@oit.cmc.msu.ru).
Р.Н. Костин – МГУ им. М.В. Ломоносова (email: rodion.kostin.999@gmail.com)

приложение с точки зрения злоумышленника и «залатать дыры».

Обычно процесс моделирования описывается четырьмя шагами:

1. Построение модели разрабатываемого приложения,
2. Идентификация возможных угроз,
3. Устранение или смягчение выявленных угроз,
4. Анализ безопасности.

Рассмотрим детальнее каждый из шагов.

A. Построение модели системы

Построение модели системы – это фундаментальный этап в процессе моделирования угроз. Его задача – формализовать архитектуру системы таким образом, чтобы можно было выявить потенциальные уязвимости и точки входа для атак. На этом этапе создается техническое описание всех компонентов, связей между ними и условий взаимодействия с внешними средами.

Первый шаг – идентификация компонентов. Описываются все элементы, участвующие в обработке и передаче данных: пользовательские интерфейсы, сервера, базы данных, сторонние сервисы. Каждый компонент сопровождается сведениями о его функциях, уровне доступа, используемых технологиях и степени доверия.

Затем строится диаграмма потоков данных [3] (DFD, Data Flow Diagram, рис. 1), отображающая, как информация перемещается между компонентами. Диаграмма включает компоненты, потоки данных, хранилища, акторов. Каждый поток описывается в виде формата передаваемых данных, используемых протоколов, каналов связи и наличия шифрования, контроля доступа и других свойств.

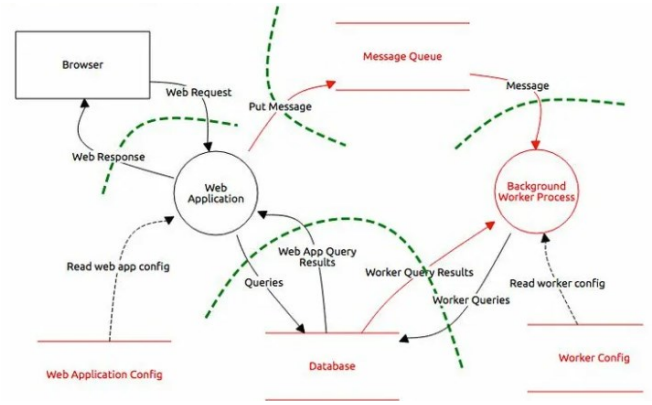

Рисунок 1. DFD с границами доверия в виде зеленых пунктирных линий. Красным цветом помечены элементы, содержащие активные угрозы

Отдельно выделяются границы доверия (рис. 1, зеленые пунктирные линии), пересечение которых требует усиленного контроля. Пример: передача данных из браузера в backend, вызов стороннего API или доступ к внутренним сервисам из публичной сети. Эти переходы всегда должны сопровождаться механизмами аутентификации, авторизации и валидации данных.

Также документируются архитектурные решения: применяемые протоколы, механизмы идентификации и авторизации, политики логирования, мониторинга, бэкапов, шифрования и хранения конфигураций.

В конце получаем модель, детально описывающую систему, которую можно анализировать на угрозы.

B. Идентификация угроз

Идентификация угроз – это процесс, в ходе которого выявляются возможные сценарии атак на систему. Он основан на ранее построенной модели архитектуры и потоков данных. Цель этапа – составить максимально полный перечень потенциальных нарушений безопасности, которые могут привести к компрометации системы или ее компонентов.

Анализ начинается с рассмотрения каждого элемента модели: компонентов, соединений между ними, точек входа и границ доверия. Для каждого из них формулируются гипотетические сценарии нежелательных воздействий. Важно учитывать не только внешние атаки, но и возможные действия со стороны легитимных пользователей, внутренних участников или ошибок в конфигурации.

Угрозы описываются как намеренные или случайные воздействия, способные повлиять на конфиденциальность, целостность, доступность системы. При этом анализ охватывает различные уровни от сетевого до прикладного, от пользовательского интерфейса до хранения и обработки данных. Особое внимание уделяется точкам пересечения с внешней средой, интеграциям и зонам с низким уровнем доверия.

Процесс требует системности: важно рассмотреть каждый компонент, каждое соединение, каждую границу. Даже если угроза кажется маловероятной, она фиксируется для последующего анализа. Такая полнота необходима для качественной оценки рисков и разработки эффективной защиты.

Результатом идентификации является перечень угроз, где каждая описана с указанием задействованных элементов, условий возникновения и возможных последствий. Угрозы из данного списка приоритизуются, например по потенциалу ущерба и вероятности возникновения, а затем данный список направляется на этап разработки мер безопасности, понижающих вероятность возникновения угроз или полностью устраняющих угрозы.

C. Разработка мер безопасности

Разработка мер безопасности – это этап, на котором формулируются конкретные действия, направленные на устранение или снижение рисков, выявленных на предыдущем этапе. Он опирается на список идентифицированных угроз и их оценку, позволяя сосредоточиться на наиболее критичных.

Каждая угроза рассматривается с точки зрения ее предпосылок: уязвимостей, архитектурных слабостей, недостатков в процессах или настройках. Цель – устранить возможность реализации угрозы или, если это невозможно, минимизировать ее влияние на систему. Варианты решений варьируются от технических до организационных.

Меры безопасности разрабатываются с учетом архитектуры системы, применяемых технологий, бизнес-требований и допустимого уровня затрат. Они могут включать улучшение контроля доступа, изоляцию компонентов, ограничение прав, внедрение мониторинга, резервного копирования, тестирования, верификации данных. Также учитываются процессы обновлений, управления инцидентами и реагирования на сбои. Защита должна быть многоуровневой: уязвимость на одном уровне не должна автоматически приводить к компрометации всей системы.

Результатом этапа является карта мер безопасности. Это позволяет выстроить прозрачную связь между рисками и действиями по их снижению, а также обеспечить основу для реализации защиты в ходе разработки.

D. Анализ безопасности

Анализ безопасности – заключительный этап моделирования угроз, направленный на оценку полноты и эффективности предложенных мер защиты. Его задача – убедиться, что все значимые угрозы учтены, а система в текущем виде обладает приемлемым уровнем защищенности.

Анализ начинается с сопоставления реализованных или запланированных мер безопасности с выявленными угрозами. Для каждой угрозы проверяется, снижает ли выбранная мера вероятность ее реализации или масштаб возможного ущерба. Если угроза остается без адекватной реакции, она возвращается на этап разработки решений.

Оценивается *остаточный риск* – то есть риск, сохраняющийся после внедрения всех защитных механизмов. При необходимости применяются дополнительные меры или корректируются решения. Такой пересмотр особенно важен для компонентов с высоким уровнем критичности или открытым интерфейсом.

Особое внимание уделяется актуальности угроз и устойчивости системы к изменяющимся условиям: появлению новых уязвимостей, новых интеграций с внешними сервисами.

Результатом анализа является финализированная модель угроз с подтвержденными мерами безопасности. Также формируются выводы о сильных и слабых сторонах архитектуры и рекомендации по улучшению. Такой анализ должен регулярно повторяться при изменениях в системе или появлении новых факторов риска.

IV. LLM-АГЕНТ

Агент на основе больших языковых моделей (LLM) – это программная сущность, способная автономно выполнять задачи с использованием текстовых инструкций, знаний из обучающих данных и внешних инструментов. В отличие от классических алгоритмов, поведение такого агента формируется не жесткими правилами, а генеративной моделью, обученной на масштабных корпусах данных.

Технически LLM-агент состоит из самой LLM-модели, планировщика, памяти и подключаемых инструментов – таких как API, базы данных, другие модели или сервисы. Такой подход позволяет создать систему, которая не просто генерирует текст, а достигает цели за счет выполнения задач на основе своих знаний и инструментов.

A. Структура LLM-агента

Рассмотрим структуру LLM-агентов.

Планировщик

Планировщик в архитектуре LLM-агента отвечает за управление поведением модели в рамках поставленной задачи. Его задача – разбивать цели на подзадачи, выстраивать последовательность действий и принимать решения о том, когда и какие инструменты использовать. Это ключевой компонент, превращающий генеративную модель из реактивного помощника в автономного исполнителя.

Работа планировщика опирается на текущий контекст, включая цели, промежуточные результаты и историю взаимодействия. Он оценивает ситуацию, формулирует следующую логическую операцию и передает ее на выполнение – либо самой языковой модели, либо подключенным внешним средствам (поиску, калькулятору, базе данных и т.п.). Получив результат, он обновляет контекст и переходит к следующему шагу. В более сложных случаях применяется итеративное планирование с пересмотром стратегии на каждом шаге. Это позволяет адаптировать поведение агента к изменяющимся условиям и новым входным данным.

Основная сложность заключается в том, что LLM не обладает внутренней памятью или состоянием. Поэтому планировщик должен явно управлять сохранением контекста: регистрировать факты, фиксировать промежуточные выводы, отслеживать ходы рассуждения. Без этого агент не сможет последовательно выполнять многошаговые задачи и учитывать историю своих действий.

Память

Память LLM-агента – это механизм, необходимый для поддержания последовательного, осознанного поведения. Без нее агент не может помнить предыдущие действия, выводы или цели, что делает невозможным выполнение многошаговых задач.

Память может быть *краткосрочной* и *долгосрочной*. Краткосрочная память – это активный контекст текущего задачи, который позволяет агенту учитывать недавние результаты. Долгосрочная память – это база знаний, накопленных фактов, которая распространяет свое влияние на следующие задачи.

Для реализации памяти используются внешние хранилища и методы выборки релевантной информации. При обновлении памяти агент сохраняет ключевые данные: факты, события, пользовательские предпочтения. При обращении – извлекает нужные элементы, чтобы восстановить контекст. Это требует

механизмов индексирования, семантического поиска и фильтрации.

Память повышает надежность и обучаемость LLM-агента. Она позволяет сохранять предпочтения пользователя, накапливать опыт, отслеживать ошибки, корректировать поведение. В результате агент может работать устойчиво в динамичной среде, где требуется не просто реагировать, а учитывать накопленные знания.

Инструменты

Инструменты позволяют LLM-агенту выходить за пределы внутреннего знания и выполнять действия в реальном мире: делать запросы, проводить вычисления, получать актуальные данные. Без инструментов агент ограничен только тем, что было включено в его обучающую выборку, и не может динамически взаимодействовать с внешней средой.

Инструменты дают агенту доступ к актуальной информации, а также позволяют точно решать задачи, требующие формальных операций – например, математические расчеты, работа с датами, сбора информации из внешнего мира.

Среди наиболее распространенных инструментов – вызов API, чтение базы данных, калькуляторы, файловая система, веб-поиск, выполнение кода. Каждый инструмент представляет собой интерфейс с определенным форматом входа и выхода, который агент использует через текстовые команды.

Архитектура с инструментами легко расширяется: можно подключать новые сервисы под конкретные задачи, настраивать разрешения, добавлять уровни валидации и контроля.

B. Угрозы LLM-агентам

Согласно анализу безопасности LLM-агентов от OWASP [20] с использованием LLM-агентов могут возникнуть 15 угроз, специфичных именно при использовании LLM-агентов:

1. **Отравление памяти**. Агент использует короткую и долговременную памяти, которые изменяет в процессе коммуникации с пользователем. Обе короткая и долговременная памяти влияют на его дальнейшее принятие решений, соответственно отравление памяти может привести к ущербу за счет действий агента.
2. **Неправильное использование инструмента**. Данная угроза случается, когда злоумышленник злоупотребляет инструментом агента, например за счет промпт-инъекций.
3. **Компрометация привилегий**. Агенты могут использовать особые виды авторизации и аутентификации, например основанные на поведении агентов. Таким образом, например, симулируя поведение своего агента определенным образом злоумышленник может получить дополнительные привилегии.
4. **Перегрузка ресурсов**. Агенты используют механизм рассуждений, LLM и инструменты, которые могут потреблять множество ресурсов. Перегрузка может возникнуть, например, за счет зацикливания процесса рассуждений или параллельного запуска множества инструментов.
5. **Каскадные галлюцинации**. Злоумышленник использует уязвимость агента к генерации выглядящих правдивыми галлюцинаций, что может привести к каскаду неверных рассуждений в мульти-агентной системе.
6. **Нарушение намерений и манипулирование целями**. В основе агента лежит декомпозиция целей, поставленных пользователем и разработчиками, установка намерений агента. В процессе коммуникации агента злоумышленник может влиять на ранее установленные намерения и цели, стимулируя нежелательное поведение.
7. **Неправильное и обманчивое поведение**. Следуя своим целям, агент действует неправильно, принося ущерб системе или пользователям.
8. **Отказ в действиях и невозможность отслеживания**. Возникает при невозможности понятия причин действий агентов, например, за счет недостаточного логирования процесса рассуждений.
9. **Подмена личности**. Злоумышленник выдает себя за определенного агента или человека, получая особый доступ в системе.
10. **Когнитивные ограничения человека**. Действия агентов часто валидируют люди. Угроза связана с когнитивными ограничениями людей, что ведет к повышенному количеству ошибок.
11. **Атаки через код**. Агенты могут использовать генерацию кода и его запуск, что может быть использовано злоумышленником для запуска зловредного кода.
12. **Отравление коммуникации агента**. Общение между агентами отравлено, что ведет к неверному их поведению.
13. **Агенты-мошенники в мульти-агентных системах**. Злоумышленник встраивает в мульти-агентную систему агентов, которые производят зловредные действия, скрываясь под легитимными агентами.
14. **Человеческие атаки на мульти-агентные системы**. Злоумышленник производит атаку за счет злоупотребления особенностями мульти-агентных систем, например, делегированием привилегий или доверительными отношениями между агентами.
15. **Манипулирование человеком**. Угроза возникает за счет чрезмерного доверия к выводу агентов, что может привести к поведению человека на основе ложной информации, которая может быть не только галлюцинацией, но и сгенерирована за счет работы злоумышленника.

Таким образом, перед сообществом по кибербезопасности возникают новые вызовы, связанные с появлением новых, ранее не существовавших угроз.

V. Методологии моделирования угроз

В данном разделе предлагается рассмотреть методологии, разработанные для эффективного моделирования угроз систем – как классических, так и на базе ИИ.

Предлагается рассмотреть их по критериям:

1. Применимость к LLM-агентам,
2. Учет специфики LLM-агентов,
3. Зрелость,
4. Порог входа.

Методология считается применимой к LLM-агентам, если ее можно использовать для систем на базе LLM-агентов без адаптации.

Учет специфики LLM-агентов означает, что методология учитывает особую структуру и поведение LLM-агентов.

«Зрелость» учитывает год создания (при условии, что методология поддерживается и развивается).

«Порог входа» учитывает, насколько легкое использование методологии, учитывая ее комплексность и необходимый уровень понимания.

Результаты анализа представлены в таблице 1.

Таблица 1 – сравнение проанализированных методологий моделирования угроз

Методология	Применим к LLM-агентам	Учет специфики LLM-агентов	Зрелость	Порог входа
STRIDE	±	-	+	+
LINDDUN	±	-	+	±
PASTA	+	-	+	-
STRIDE-AI	±	±	±	±
PLOT4ai	±	±	-	±
ADMIn	±	±	-	±
MAESTRO	+	+	-	-

A. STRIDE

STRIDE [4] – это акроним из категорий угроз: Spoofing (подмена), Tampering (модификация), Repudiation (сокрытие), Information Disclosure (раскрытие информации), Denial of Service (отказ в обслуживании), Elevation of Privilege (повышение привилегий).

Для применения данной методологии необходимо взять модель системы (процессы, хранилища, потоки данных, границы доверия), задаться вопросом «что может пойти не так?» и затем анализировать каждый элемент модели с точки зрения одной из категорий: «может ли произойти здесь подмена данных?», «могут ли повредить данные?» и т.п.

STRIDE помогает в идентификации угроз, эффективно применим к широкому спектру классических систем и охватывает широкое множество угроз за счет достаточно полной категоризации угроз.

STRIDE частично применим к системам на базе LLM-агентов: он полезен для фиксации классических угроз (подмена, подделка, отказ и т.д.) на уровне компонентов и потоков; специфика LLM-агентов в нем не учитывается; методология зрелая (1999 г.); хоть STRIDE эффективно применить может только подготовленный пользователь, тем не менее он достаточно прост в понимании и освоении, поэтому порог входа низкий.

B. LINDDUN

LINDDUN [5] – акроним от слов Linking (связывание), Identifying (идентифицирование), Non-repudiation (невозможность отказа), Detecting (детектирование), Data Disclosure (раскрытие данных), Unawareness (неосознанность), Non-compliance (несоблюдение). Это фреймворк для идентификации угроз приватности данных.

Существует три варианта LINDDUN: Go, Pro, Maestro. Инструменты включают в себя карточки угроз, деревья с угрозами, примерами, последствиями, критериями применимости.

Pro – систематичный подход, использующий диаграмму потоков данных и все инструменты LINDDUN.

Maestro – так же систематичный подход, который аналогичен Pro, но использует еще более детальное описание системы, нежели диаграмму потоков данных. Но подробностей нет, так как он еще не вышел в публичное поле.

LINDDUN предлагает несколько подходов для команд с разными потребностями, что является плюсом. Для команд, которым критична приватность, данный фреймворк подходит лучше всего.

Применим к системам на базе LLM-агентов частично, требует адаптации. Он не охватывает специфику LLM-архитектур и поведенческих моделей, особенно в части планирования, инструментов и генерации. Методология достаточно зрелая (2010 год), но для освоения необходимо изучить алгоритм и предлагаемые инструменты, а также уметь их применять, поэтому порог входа средний.

C. PASTA

PASTA [6] (Process for Attack Simulation and Threat Analysis, процесс для симуляции атак и анализа угроз) – это риск-центрированная методология моделирования угроз 2015 года, состоящая из семи этапов.

1 – *определение целей системы и требований*: бизнес-требований, функциональных требований, требований по безопасности.

2 – *описание технического решения*. На данном этапе описывается «что мы защищаем» с технической перспективы. Данный этап включает описание приложения, базы данных, облачного провайдера, сети, операционной системы, шифрование, внешние сервисы.

3 – *декомпозиция приложения*. Обычно используя диаграммы потоков данных.

4 – *анализ угроз*. Понимание множества угроз на основе общего представления о безопасности, отчетах об угрозах, известных векторах атак, библиотек угроз.

5 – *анализ уязвимостей*. Основной целью данного этапа является сопоставление найденных уязвимостей с активами и угрозами.

6 – *анализ атак*. На данном этапе происходит связывание уязвимостей с угрозами и доказывание осуществления угроз.

7 – *анализ рисков и последствий*. Определяются риски, последствия; строится план мер безопасности, план снижения рисков и оценивается остаточный риск. В итоге получаем риск-профиль приложения, последствия рисков для бизнеса, остаточные риски и стратегию понижения рисков.

Достоинствами данной методологии можно являются полнота, глубина анализа, упор на работу с рисками, их последствий и стратегий снижения. Определяются цели и требования системы, использующиеся технические решения, а приложение декомпозируется, что может быть полезно не только для обеспечения безопасности, но и для повышения эффективности разработки, осознанности принятия решений.

PASTA – одна из самых универсальных методологий. Она применима к системам на базе LLM-агентов, но при этом учета специфики LLM-агентов нет. Методология зрелая, но порог входа высокий, для ее использования необходимо обладать значительными компетенциями, так как она предлагает общие шаги без инструментов и знаний.

D. STRIDE-AI

STRIDE-AI [7] – адаптация STRIDE под ИИ системы. в качестве основы используется FMEA методология и диаграмма жизненного цикла машинного обучения для выявления угроз на каждом этапе.

Данная методология заключается в определения активов системы с помощью описанного авторами жизненного цикла машинного обучения. Авторы предлагают переопределение STRIDE свойств (Аутентичность, Целостность и т.д.) специфично для ML систем, чтобы лучше понять, как свойство может быть нарушено. Исследователи определяют шесть типов активов ИИ-систем: данные, модели, артефакты, акторы, инструменты, процессы.

STRIDE-AI адаптирован под специфические угрозы, связанные с машинным обучением, включая атаки на данные, модели и инфраструктуру. Он охватывает весь жизненный цикл ML-систем, позволяя анализировать угрозы на этапах подготовки данных, обучения, развертывания и эксплуатации. Методология совместима с существующими стандартами информационной безопасности (ISO 27001 [22], NIST CSF [23]), что облегчает ее интеграцию в процессы защиты данных.

Но метод требует значительных ресурсов, так как анализ угроз сложен и требует участия экспертов по ML и кибербезопасности. Из-за отсутствия универсального набора защитных мер каждая система требует индивидуального подхода, что усложняет автоматизацию процесса. Новые атаки на ML-модели появляются регулярно, что требует постоянного обновления модели угроз и защитных стратегий. В небольших проектах методология может оказаться избыточной, так как не всегда оправдывает затраты на ее внедрение.

STRIDE-AI расширяет классический STRIDE с учетом особенностей ИИ. Применим к LLM-агентам в части архитектурных и LLM угроз, но не специфичных для LLM-агентов. Методология недостаточно зрелая (разрабатывается с 2021 года), порог входа средний, так как требует освоения методологии.

E. PLOT4ai

PLOT4ai [8] – набор из 86 угроз и методология моделирования угроз, с помощью которой проще моделировать угрозы для ИИ-систем. Данный метод является адаптацией LINDDUN для ИИ-систем.

Угрозы делятся на 8 категорий: *Technique & Processes* (Технологии и процессы), *Accessibility* (Доступность), *Identifiability & Linkability* (Идентифицируемость и установление связей), *Security* (Безопасность), *Safety* (Физическая и психологическая безопасность), *Unawareness* (Неосведомлённость), *Ethics & Human Rights* (Этика и права человека) и *Non-compliance* (Несоблюдение нормативных требований).

PLOT4ai разработан специально для систем с ИИ. Он хорошо учитывает специфику ИИ в целом и частично подходит для LLM-агентов, но только по отношению к LLM, но не к LLM-агентам. Методология не зрела (2023 г.), и имеет средний порог входа, как ее аналог LINDDUN.

F. ADMIn

ADMIn [9] – фреймворк, предлагающий инструменты для описания системы машинного обучения и идентификации угроз, специфичных для таких систем. Это простой процесс из 5 шагов с фиксированной начальной схемой системы и фиксированной таксономией атак.

Шаги следующие:

1. Взять за основу диаграмму процессов разработки ИИ-приложений, предложенную авторами.
2. Удалить не используемые вводы, выводы, процессы.
3. Добавить используемые вводы, выводы, процессы, но не отображенные на диаграмме.
4. Последовательно для каждого процесса и его вводов и выводов добавлять атаки, которые применимы к приложению, ссылаясь на таксономию атак.
5. Если нужна дополнительная информация, соотнесите атаки для каждого ввода, вывода, процесса на диаграмме с STRIDE.

Преимуществом данного подхода является простота алгоритма. Достаточно описать схему по предложенному методу, а затем сопоставить угрозы из таксономии с элементами, причем дополнительно угрозы сопоставлены со STRIDE методологией. Описание начальной схемы достаточно обще, что позволяет описать большинство систем. Таксономия достаточно удобно составлена с достаточно полным перечнем угроз.

Но нет подключения к базам угроз. Для использования требуются специалисты по кибербезопасности. Могут появляться новые угрозы, новая информация и нужно адаптировать модель угроз – ADMIn не предоставляет алгоритма для таких ситуаций.

ADMIn хорошо применим для начального базового моделирования угроз, но для более детальной проработки с постоянным обновлением ADMIn не

подходит. Хотя описание, построенное на 1 этапе в принципе полезно, его можно использовать и в будущем.

Методология немного применима к системам с LLM-агентами, так как учитывает структуру и поведение машинного обучения и может быть адаптирована под агентов. Но методология не учитывает специфику LLM-агентов. Методология не зрелая (2023 г.) и имеет средний порог входа.

G. MAESTRO

MAESTRO [10] – полноценный фреймворк моделирования угроз, разработанный специально для агентских систем.

MAETRO предлагает декомпозировать рассматриваемую систему на 7 уровней по 7-ми уровневую архитектуру агентской системы [11]:

1. Фундаментальные модели
2. Оперирование данными
3. Агентские фреймворки
4. Развертывание и инфраструктура
5. Оценка и мониторинг
6. Безопасность и соответствие требованиям
7. Агентская экосистема

Каждый уровень использует меньший уровень Исключением является 6 уровень безопасности, который пронизывает все уровни, включая наибольший 7 уровень.

Уровень 1 (фундаментальные модели) – ядро, на котором базируется агент. Им может выступать LLM или любые другие формы ИИ.

Уровень 2 (оперирование данными) – уровень обработки, подготовки, хранения данных для ИИ агентов. Включает базы данных, хранилища векторов, RAG и т.п.

Уровень 3 (агентские фреймворки) – охватывает фреймворки для построения агентских систем и другие агентские фреймворки.

Уровень 4 (развертывание и инфраструктура) – описывает инфраструктуру, на которой запущена агентская система, и архитектуру развертывания.

Уровень 5 (оценка и мониторинг) – уровень оценки и мониторинга деятельности агентов, который может включать отслеживание производительности, определение аномалий, логирование действий.

Уровень 6 (безопасность и соответствие требованиям) – уровень, который связан со всеми остальными, включающий компоненты безопасности и контроля требований. Включает в себя агентов, которые используются для безопасности системы.

Уровень 7 (агентская экосистема) – среда, с которой взаимодействуют агенты. В нее могут входить пользователи, бизнес-приложения, сервисы интеграции, платформы, сторонние агенты.

Далее проводится идентификация уровневых угроз. Цель атаки и уязвимость находятся на одном уровне. Для каждого уровня MAESTRO определяет достаточно длинные списки угрозы, длиной от 6 до 13, которые являются вспомогательным средством для идентификации угроз в описываемой системе.

Затем проводится идентификация кросс-уровневых угроз. Цель атаки и уязвимость находятся на разных уровнях.

Далее проводится оценка рисков. Автор предлагает приоритизировать угрозы, используя матрицу рисков. Требуется посчитать вероятность возникновения и последствия каждой угрозы и перемножить показатели.

Далее проводится планирование митигаций (мер безопасности). На вход подается приоритизированный список угроз. Реализуются митигации уровневых, кросс-уровневых и ИИ-специфичных угроз.

Наконец проводится реализация митигаций и мониторинг новых угроз, обновление модели угроз с развитием системы.

Достоинством данного подхода является специфичность для агентских систем, которые являются новым веянием на рынке. Широта и глубина описания системы и поверхности угроз (суммарно 50 угроз) предоставляют хорошую базу для глубокой проработки безопасности.

Но структура кажется громоздкой, особенно для маленьких систем. Подход требует хорошего понимания области и самого фреймворка. Фреймворк не подключен к актуальной базе угроз, таким образом новые угрозы необходимо вручную проверять и самим сопоставлять с нужным уровнем, что может вызвать сложности в неоднозначных ситуациях. Фреймворк достаточно молод и недостаточно протестирован, поэтому его эффективность еще не доказана.

MAESTRO – самая комплексная из современных методологий, специально ориентированных на сложные системы LLM-агентов. Она учитывает генеративное поведение, память, использование инструментов и последовательность действий, причем на разных уровнях и к тому же межуровневые угрозы. Но методология новая (2023г.) и требует серьезной подготовки из-за комплексности, поэтому порог входа высокий.

VI. Инструменты моделирования угроз

В данном разделе предлагается рассмотреть инструменты (программное обеспечение), которые упрощают процесс моделирования угроз, с точки зрения следующих критериев:

1. Поддержка компонент LLM-агентов,
2. Автоматизация идентификации угроз,
3. Задание атрибутов компонентам,
4. Графический интерфейс,
5. Открытый исходный код,
6. Кросс-платформенность,
7. Низкий порог входа.

Критерий «поддержки компонент LLM-агентов» отражает наличие средств для использования компонент LLM-агентов в описании системы, таких как сами ядро агентов, LLM с функцией Function Calling, инструменты LLM-агента и прочие.

Критерий «автоматизации идентификации угроз» показывает, насколько инструмент может автоматизировать идентификацию угроз. Это особенно полезно при динамичном появлении новых угроз,

быстром изменении системы, сложности ручной работы или недостаточной компетенции разработчиков в области безопасности.

«Задание атрибутов компонентам» важно для более детального описания системы, которые полезно при анализе: будь то ручном или автоматическом.

«Графический интерфейс» упрощает пользование программным обеспечением, понижая порог входа для его использования.

Наличие «открытого исходного кода» позволяет адаптировать инструмент под задачи, расширять функциональность, внедрять поддержку LLM или собственные категории угроз.

«Кросс-платформенность» важна для легкой интеграции ПО в процесс разработки. Наличие данного критерия позволяет избежать требования по наличию той или иной операционной системы, устройства и подобных ограничений.

«Низкий порог входа» отражает простоту пользования инструментом, необходимость наличия особых знаний и навыков для его использования.

Таблица 2 – сравнение проанализированных инструментов моделирования угроз

ПО	1	2	3	4	5	6	7
OWASP Threat Dragon	-	±	+	+	+	+	+
OWASP pytm	-	+	+	-	+	+	±
Microsoft Threat Modeling Tool	-	+	+	+	-	-	+
ThreatFinderAI	±	±	-	+	-	+	+

A. OWASP Threat Dragon

OWASP Threat Dragon [12] – это инструмент для моделирования угроз с открытым исходным кодом, разработанный для упрощения процесса выявления и анализа потенциальных угроз в приложениях.

Описание системы происходит с помощью диаграммы потоков данных.

Основные функции включают в себя создание диаграмм потоков данных (DFD), проставления угроз в элементах диаграммы, оценку рисков и рекомендации по устранению уязвимостей. Инструмент доступен как в виде настольного приложения, так и в веб-версии. OWASP Threat Dragon также автоматизирует процесс документирования угроз, что снижает риск пропустить важные аспекты безопасности на ранних стадиях разработки.

Также есть функционал для аналитической отчетности: общая и детальная информация.

В детальном виде угрозы сгруппированы по компонентам со всеми полями, которые у них есть: от приоритета, статуса до мер безопасности, необходимых к реализации. Этого достаточно для приоритизации работы и оценки ее эффективности.

Преимущества OWASP Threat Dragon включают его простоту и доступность благодаря открытой лицензии. Кроме того, его совместимость с DevSecOps позволяет использовать Threat Dragon на ранних стадиях разработки.

Среди недостатков можно отметить ограниченные возможности для моделирования сложных сценариев в крупных и многоуровневых системах. Также инструмент может быть недостаточно гибким для специфических требований отдельных компаний.

Он не поддерживает добавление компонент специфичных LLM-агентам. Однако он имеет открытый исходный код. Уровень автоматизации низкий: все элементы модели и угрозы вводятся вручную, с авто-идентификацией угроз в зачаточном состоянии. Зато есть возможность задавать атрибуты компонентам, имеется графический интерфейс, поддерживается кросс-платформенность. Для пользования инструментов не нужно особых навыков, что формирует низкий порог входа.

B. OWASP pytm

OWASP pytm [13] – это инструмент для моделирования угроз с открытым исходным кодом, созданный для автоматизации процесса создания и анализа моделей угроз с использованием Python. Вместо визуальных диаграмм, как в других инструментах, pytm позволяет описывать систему и ее компоненты программным способом. Основные функции pytm включают описание архитектурных элементов, оценку рисков и автоматическую генерацию диаграмм потоков данных (DFD), что упрощает анализ потенциальных угроз и уязвимостей.

Преимущества pytm заключаются в его программируемом подходе к моделированию угроз, что дает разработчикам больше гибкости в создании моделей. Это также делает pytm легко интегрируемым в процессы DevOps и CI/CD, позволяя автоматизировать создание и анализ моделей угроз на разных стадиях разработки. К тому же бесплатный с открытой лицензией.

Но его функционал ограничен для моделирования сложных систем или специфичных требований крупных компаний.

Генерация отчетов происходит по шаблону, который можно редактировать, таким образом их можно настроить под свои нужды.

OWASP pytm ориентирован на автоматизацию идентификации угроз и генерацию отчетов через код Python. Специфику LLM-агентов не учитывает. Исходный код открыт. Возможно задавать собственные атрибуты. Не имеет графического интерфейса, что повышает порог входа. Поддерживается кросс-платформенность.

C. Microsoft Threat Modeling Tool

Microsoft Threat Modeling Tool [14] – это инструмент для моделирования угроз, разработанный Microsoft, который помогает разработчикам и специалистам по безопасности идентифицировать и анализировать потенциальные угрозы в архитектуре приложения. Он использует DFD для визуализации компонентов системы и их взаимодействий, что упрощает выявление возможных точек уязвимости. Инструмент автоматически генерирует угрозы на основе шаблонов

STRIDE, помогая оценить риски и предложить возможные меры по их снижению.

Инструмент предлагает 2 режима: описательный и аналитический.

Недостатками можно назвать его ограниченную гибкость при моделировании более сложных систем. Microsoft Threat Modeling Tool также менее полезен для организаций, не использующих продукты Microsoft, так как его интеграция и функционал сильно завязаны на этой экосистеме. Кроме того, несмотря на свою простоту, инструмент может уступать другим решениям в плане масштабируемости, гибкости настройки и автоматизации процессов для более крупных организаций с высокими требованиями к безопасности.

Функционал не поддерживает LLM-специфику. Исходный код закрыт. Но присутствует автоматическая генерация угроз и отчетности, а также добавление пользовательских атрибутов. Не поддерживает кросс-платформенность, так как разработан специально для Windows. Низкий порог входа за счет простого интуитивно понятного интерфейса.

D. ThreatFinderAI

ThreatFinderAI [15] – инструмент моделирования угроз, разработанный исследователями из Цюриха, является попыткой создания приложения для моделирования угроз ИИ-специфичных приложений.

Threat Dragon предлагает диаграмму потоков данных (DFD) для описания систем. Стандартная диаграмма включает следующие компоненты:

1. Акторы
2. Процессы,
3. Хранилища данных,
4. Потоки данных.

Использование разделено на 5 этапов:

1. Создание сценариев угроз,
2. Описание модели системы,
3. Анализ угроз,
4. Управление мерами безопасности,
5. Оценка остаточных рисков.

Инструмент подключен к различным базам угроз (ENISA [17], OWASP AI Exchange [18], MITRE ATLAS [19]) и для каждого типа предлагает множество рекомендованных к рассмотрению угроз, которые пользователь вручную просматривает и добавляет в модель угроз.

Инструмент предлагает множество мер безопасности для каждой угрозы, которые так же нужно просмотреть вручную и выбрать, какие будут применены. Меры безопасности взяты из OWASP AI Exchange.

Инструмент предлагает большое множество, часто нерелевантных угроз, текстовое описание которых нужно анализировать вручную. Генерация отчетности отсутствует.

Инструмент разработан специально для моделирования систем на базе ИИ, но не для систем на базе LLM-агентов. Имеется автоматическая генерация рекомендаций угроз и мер безопасности, но остается значительная ручная работа по их отбору, в том числе из-за большого количества нерелевантных рекомендаций. Исходный код исследователи не предоставляют. Задавать пользовательские атрибуты нельзя. Но имеется графический интерфейс, ПО обеспечивает кросс-платформенность и понятный интуитивный интерфейс за счет чего низкий порог входа.

VII. Требования к ПО моделирования угроз

Для ПО моделирования угроз для систем на базе LLM-агентов были разработаны следующие требования:

1. Поддержка компонент LLM-агентов.
2. Автоматическая идентификация угроз, специфичных для LLM-агентов.
3. Расширяемая база угроз, отделенная от логики.
4. Открытый исходный код.
5. Графический интерфейс;
6. Кроссплатформенность.

Для эффективного моделирования угроз необходимо поддерживать полноценную агентскую архитектуру. Это включает в себя LLM, память для сохранения контекста, планировщик и внешние инструменты. Такая архитектура позволит реалистично представлять поведение агента и точно идентифицировать угрозы.

Автоматическая идентификация угроз важна для быстрого и эффективного моделирования угроз. Для LLM-агентский систем особенно важно идентифицировать специфичные угрозы для LLM-агентов.

Важным требованием является наличие расширяемой базы специфичных атак, обусловленных использованием LLM-агентов. Это включает, например, атаки через инструменты агента.

Открытый исходный код инструмента позволит вносить улучшения, а также способствует стандартизации подходов к безопасности в ИИ-системах.

Графический интерфейс и кроссплатформенность важны с точки зрения удобства и масштаба использования ПО.

VIII. Разработанное ПО моделирования угроз

Авторами разработано ПО моделирования угроз для систем, использующих LLM-агентов.

Данное ПО является расширением проекта OWASP Threat Dragon, который использует диаграмму потоков данных DFD.

OWASP Threat Dragon был выбран в качестве основы благодаря своей открытости, графическому интерфейсу, кроссплатформенности и невысокому порогу входа. Однако в исходной версии отсутствовали возможности моделирования систем, основанных на LLM-агентах. Целью данной работы является реализация в Threat Dragon возможности автоматического определения угроз, характерных для систем на базе LLM-агентов.

Разработанное ПО дополняет Threat Dragon следующими возможностями:

1. Двумя новыми компонентами диаграммы: «LLM-агент» и «Инструмент»;
2. Модулем автоматической идентификации угроз;
3. Расширяемым множеством правил угроз, специфичных для LLM-агентов.

Диаграмма потоков анализируемой системы строится вручную, атрибуты всех компонент также определяются вручную. Затем на основе диаграммы, атрибутов и правил на диаграмму автоматически добавляются угрозы. Пример диаграммы показан на рис. 2.

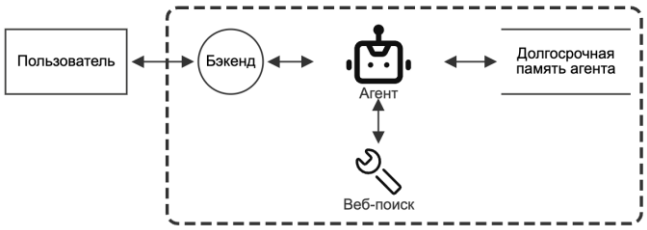

Рисунок 2. Пример диаграммы Threat Dragon с новыми компонентами

A. Описание систем с LLM-агентами

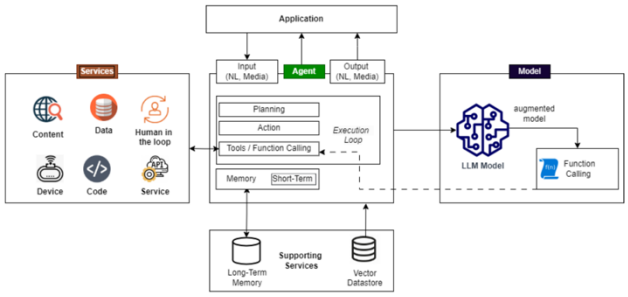

Рисунок 3. Структура LLM-агента от OWASP [20]

В качестве эталонной структуры LLM-агента взята структура, определенная OWASP [20] (рис. 3).

Компонент «LLM-агент» состоит из ядра LLM-агента («Agent» на рис. 3), и аугментированной LLM («Model» на рис. 3), которая участвует в рассуждениях и использовании инструментов.

Компонент «Инструмент» описывает инструменты LLM-агента («Services» на рис. 3), которые агент использует во время своей работы. Хотя они и похожи на обычные функции системы, но они обладают специфичными свойствами и их может использовать только LLM-агент.

Для описания памяти, с которой взаимодействует агент («Supporting Services» на рис. 3), было принято решение использовать компонент «Store», включенный в стандартный набор компонентов Threat Dragon. Данный компонент может быть использован как для описания долговременной памяти агента, так и для внешних баз данных, к которым обращается агент.

B. Правила идентификации угроз

В рекомендациях OWASP [20] перечислены 54 сценария атак, сгруппированные по типам угроз.

Для каждой угрозы в [20] описаны сценарии атак, которые реализуют эти угрозы. Для каждой угрозы в [20] определено до 5 сценариев.

Авторами был проведен анализ каждого сценария с целью выработать правила возникновения угроз. На основе данных сценариев было разработано 38 правил угроз и добавлено 27 новых атрибутов, что является основой для автоматической идентификации угроз.

Автоматическая идентификация угроз реализована на основе правил. Если на созданной пользователем диаграмма DFD существует компонент, удовлетворяющий всем параметрам какого-либо правила, то на диаграмму автоматически добавляется соответствующая угроза.

Правило состоит множества **условий** (**conditions**), при которых может возникнуть угроза, **описания сценария** из OWASP (**description**) и **смягчений** данной угрозы (**mitigations**), которые предназначены для уменьшения угрозы и могут быть проактивными, реактивными и детективными. В правиле также указывается тип угрозы по методологии STRIDE (**strideType**). При выполнении хотя бы одного условия угроза будет добавлена на диаграмму Threat Dragon для соответствующего компонента.

Правило описывается с использованием атрибутов компонента, атрибутов потоков данных от/к данному компоненту и атрибутов соседних компонентов. Соседними компонентами являются компоненты, с которыми связан рассматриваемый компонент потоками данных.

Условия в правилах идентификации угроз

Условие состоит из следующих полей:

1. **nodeType** – тип компонента (узла) DFD, к которому применимо условие. Указывается в строковом формате.
2. **props** – атрибуты компонента в формате объекта JSON.
3. **neighbours** – соседи компонента. Соседями являются компоненты, которые связаны с целевым потоками данных. Указывается в формате JSON.

Массив neighbours описывает соседей рассматриваемого компонента. Для выполнения рассматриваемого условия все соседи, описанные в neighbours, должны присутствовать на диаграмме.

Описание соседа состоит из следующих полей:

1. **nodeType** – тип соседнего компонента (в строковом формате).
2. **props** – атрибуты соседнего компонента, аналогично атрибутам целевого компонента.
3. **flows** – массив потоков данных, связанных с соседним компонентом.

Поле **flows** состоит из массива, который описывает потоки данных между рассматриваемыми компонентом и соседом. Это условие состоит из следующих полей:

1. **direction** – направление потока. Значениями могут быть “from” (от соседа), “to” (к соседу), “any” (в любом направлении). Поле задается в строковом формате.
2. **props** – атрибуты потока (аналогично целевому компоненту).

C. Атрибуты компонентов

Атрибут доверия (**isTrusted**) был добавлен новым компонентам LLM-агент и Инструмент и существующим

в Threat Dragon компонентам диаграммы Актор, Процесс и Хранилище данных.

Для Хранилища данных определен атрибут (**isLongTermAgentMemory**), указывающий, что память используется как долгосрочная память агента.

Для Процесса определен атрибут (usesHII), указывающий, что в данном процессе используется Human Intervention Interface.

Для Потока данных определены атрибуты наличия промпта пользователя (**hasUserPrompt**) в передаваемых данных и атрибут возможного наличия медиа-файлов в передаваемых данных (**mayContainMedia**).

Больше всего атрибутов определено для LLM-агента:

1. **hasLogging** – логирование поведения агента. Во время работы агент рассуждает, принимает решения, делает выводы, использует инструменты, сохраняет информацию в краткосрочную и долгосрочную память – все это определяет поведение агента, которое может быть логировано.
2. **hasDynamicAdminPrivileges** – атрибут наличия динамически присваиваемых привилегий администратора. Привилегии присваиваются динамически на время при возникновении ситуаций, в которых агенту необходимо временное повышение его прав.
3. **hasMultiDomainAccess** – атрибут наличия мультидоменного доступа, например к ресурсам HR и бухгалтерии. Мультидоменный доступ означает доступ к разного рода бизнес-отделам компании, которые характеризуются разными уровнями прав агента к данным отделам.
4. **canBeRegisteredByUser** – атрибут возможности регистрации агента пользователем системы.
5. **inheritsPrivileges** – атрибут возможности наследования привилегий.
6. **usesAuth** – атрибут использования авторизации для доступа к защищенным ресурсам системы.
7. **hasConstaints** – атрибут наличия установленных разработчиками ограничений агента, которые обычно задаются в системном контексте агента.
8. **usesBehavioralAuth** – атрибут использования поведенческой авторизации для доступа к защищенным ресурсам системы, которая основана на поведении агента.
9. **isSecurity** – атрибут, который обозначает, что агент выполняет некоторые функции безопасности системы, то есть является своего рода защитником системы.
10. **isAuthenticator** – атрибут роли агента, как модуля аутентификации.

Атрибуты инструмента:

1. **isDangerous** – атрибут опасности инструмента. Может ли инструмент привести к летальным последствиям? Например, инструмент управления оружием.
2. **isApi** – атрибут, указывающий, что инструмент является API стороннего ресурса.
3. **usingParameters** – атрибут, указывающий, что инструмент требует ввод параметров для своего функционирования.
4. **isAutomated** – автоматизирован ли инструмент. Например, рассылка письма сразу всем сотрудникам компании.
5. **requiresAdmin** – атрибут требования прав администратора для пользования инструментом.
6. **isResourceIntensive** – атрибут, отражающий, что инструмент использует значительные ресурсы системы.
7. **hasQuota** – атрибут наличия квоты на использования инструмента. Например, биллинг за запрос к ресурсу.
8. **executeAgentGeneratedCode** – атрибут, указывающий, что инструмент умеет запускать сгенерированный произвольный код.

Для каждой угрозы определен наиболее подходящий тип STRIDE, добавлено описание, а также митигации, которые позволят снизить вероятность возникновения угрозы.

D. Пример правила идентификации угроз

Некоторые агенты используют инструменты, которые в свою очередь используют API с квотой за вызовы. Обычно квота исчисляется в количестве запросов в определенный период времени. Разработчики учитывают данный лимит для бесперебойной работы агента, создают хороший запас квоты, и обычно не ожидают ее исчерпания.

С помощью разработанных правил можно описать ситуацию, в которой пользователь, который имеет возможность отправки промпта агенту, использует уязвимость планировщика и вводит агента в состояние многочисленного вызова API-инструмента, обладающего квотой. Данный пример можно изобразить на диаграмме следующим образом (рис. 4):

Рисунок 4. Пример диаграммы Threat Dragon с угрозой опустошения квоты API-инструмента

В правиле описаны следующие условия:

1. Целевая компонента имеет тип *tm.Agent*;
2. Целевая компонента имеет двух соседей:
    a. Сосед с любым типом (*any*), от которого поступает поток данных с пользовательским промптом.
    b. Сосед с типом *tm.Tool*, который имеет 2 атрибута:
        1) Арибут *isApi* в значении true;
        2) Атрибут *hasQuota* в значении true.

После автоматической идентификации угрозы в компоненту агента будет добавлена карточка угрозы (рис. 5).

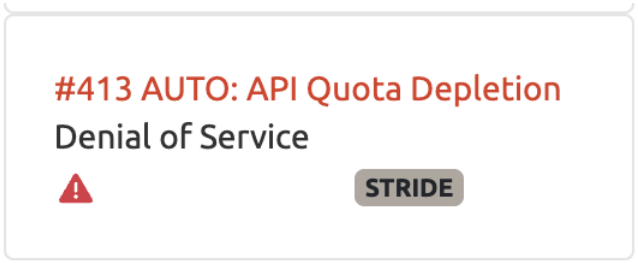

Рисунок 5. Карточка автоматически идентифицированной угрозы API Quota Depletion

Для всех угроз, которые идентифицированы автоматически, добавляется префикс «AUTO:».

Пример открытой карточки угрозы показан на рис. 6.

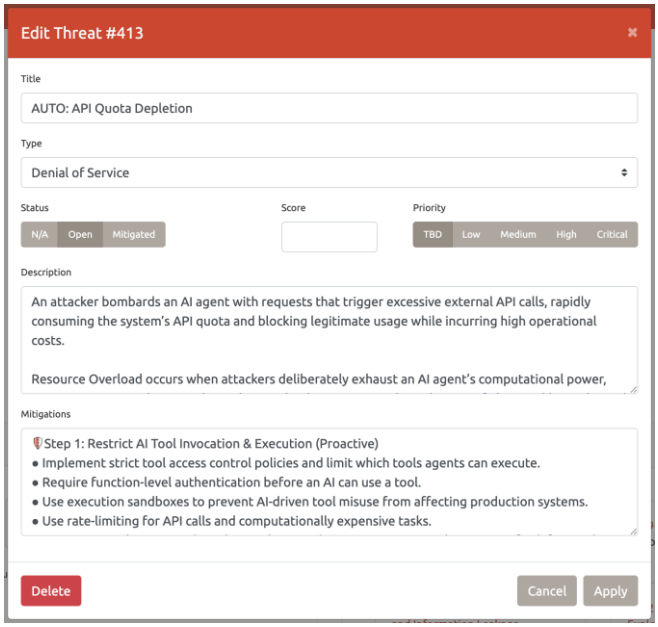

Рисунок 6. Пример открытой карточки автоматически идентифицированной угрозы API Quota Depletion

Авторами определен достаточно простой синтаксис правил, что позволяет легко описывать новые угрозы, которые будут затем автоматически добавляться на диаграмму. Пример синтаксиса приведен на Листинге 1.

Листинг 1. Синтаксис правил угроз

```
{
  "id": "TM-001", // Идентификатор угрозы
  "name": "...", // Название угрозы
  "description": "...", // Описание угрозы
  "mitigations": "...", // Меры по смягчению
угрозы
  "strideType": "...", // Тип угрозы по STRIDE
  "conditions": [ // Условия добавления угрозы
на диаграмму
    {
      "nodeType": "tm.Agent", // Тип
рассматриваемого компонента
      "props": {}, // Атрибуты компонента
      "neighbours": [ // Соседи компонента
        {
          "nodeType": "tm.Process", // Тип
компонента-соседа
          "props": {}, // Атрибуты соседа
          "flows": [ // Потоки к/от соседа
            {
              "direction": "from", //
Направление потока
              "props": {} // Атрибуты потока
            }
          ]
        }
      ]
    }
  ]
}
```

IX. Сравнение разработки с рассмотренными решениями

Разработка имеет совершенно новые функции, которых не было в выше рассмотренных решениях:

1. Описывать системы, использующие LLM-агентов;
2. Автоматически идентифицировать угрозы, специфичные для LLM-агентов.

Дополнительно разработка удовлетворяет требованиям удобного, простого и расширяемого приложения, которое использует расширяемую базу угроз на основе правил, имеет графический интерфейс и кросс-платформенно.

В совокупности это дает новые возможности для более быстрого и простого моделирования угроз для систем, построенных на базе новой технологии, LLM-агентов.

X. Отзывы

Разработка уже получила положительные отзывы от технических лидеров OWASP Threat Dragon Джона Гадсдена и Лео Рединга [21].

Джон Гадсден:

- *«This is a great piece of work and certainly boosts Threat Dragon's ability to be useful, and we certainly need to get this feature in»*
- *«An impressive amount of work»*
- *«This is very valuable work and is greatly appreciated»*

Лео Рединг:

- *«This is awesome!»*
- *«Again, this is great and is really appreciated!»*

XI. Заключение

В рамках данного исследования были проанализированы 7 методологий и 4 программного обеспечения моделирования угроз. Для каждой методологии проведена оценка по критериям: применимость к LLM-агентам, учет специфики LLM-агентов, зрелость и порог входа. Программное обеспечение рассмотрено по критериям: наличие специфичных для LLM-агентов компонент, автоматической идентификации угроз, задания пользовательский атрибутов компонентам, наличия графического интерфейса, наличия открытого исходного кода, кроссплатформенность и порог входа. Разработаны требования, новое программное обеспечение и получены положительные отзывы от лидеров OWASP Threat Dragon за ценный вклад в проект.

Анализ показал, что большинство существующих методологий моделирования угроз не подходят для

работы с системами на базе LLM-агентов. Классические подходы вроде STRIDE или LINDDUN не учитывают важные особенности таких систем: память, планирование, вызов внешних инструментов. Даже те методики, которые адаптированы под ИИ, чаще всего не охватывают весь спектр угроз

Из всех рассмотренных подходов только методология MAESTRO достаточно хорошо справляется с описанием угроз для LLM-агентов. Она учитывает разные уровни системы и даже взаимосвязанные угрозы между уровнями. Однако она сложна в применении и требует высокой квалификации от команды, что делает ее трудной для повседневного использования.

Большая часть методологий не ориентирована на ИИ и объемна к применению. Предложенные подходы в виде игр и карточек ограничены как во множестве рассматриваемых угроз, так и в типах угроз: LINDDUN нацелен на обеспечение приватности; PLOT4ai хранит 86 карточек; ADMIn и STRIDE-AI определили свое множество угроз, на которое призывают ориентироваться.

Кроме того, большинство ПО моделирования угроз не умеют автоматически учитывать особенности LLM-агентов. Они не предлагают автоматическую генерацию угроз и плохо подходят для быстрых итераций разработки. Только один инструмент, ThreatFinderAI, частично ориентирован на ИИ, но даже он не покрывает особенности LLM-агентов и не имеет открытого кода для доработки.

Благодарности

Тема статьи, написанной по результатам магистерской диссертации, соответствует направлению “Кибербезопасность” на факультете ВМК МГУ [24]. Как примеры похожих работ см. публикации [25,26].

Как обычно, отмечаем работы В.П. Куприяновского и его многочисленных соавторов, положивших начало цифровой повестке в журнале [27, 28].

Библиография

[1] Luo J., Zhang W., Yuan Y. и др. Large Language Model Agent: A Survey on Methodology, Applications and Challenges [Электронный ресурс] // arXiv, 27 мар. 2025. URL: https://arxiv.org/abs/2503.21460.

[2] Threat Modeling Overview [Электронный ресурс] // OWASP Community Projects. URL: https://owasp.org/www-community/Threat_Modeling#:~:text=Threat%20modeling%20is%20a%20family,of%2C%20threats%20to%20the%20system.

[3] What is a Data Flow Diagram (DFD)? [Электронный ресурс] // IBM Think Blog, 22 ноя. 2024. URL: https://www.ibm.com/think/topics/data-flow-diagram.

[4] Allen-Addy C. Threat Modeling Methodology: STRIDE [Электронный ресурс] // Блог IriusRisk, 29 сен. 2023. URL: https://www.iriusrisk.com/resources-blog/threat-modeling-methodology-stride.

[5] LINDDUN — Privacy Threat Modeling Framework [Электронный ресурс]. URL: https://linddun.org/.

[6] Uceda Vélez T., Morana M. M. Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis. Hoboken (N.J.): Wiley, 2015. 696 p. DOI 10.1002/9781118988374.

[7] Mauri L., Damiani E. Modeling Threats to AI-ML Systems Using STRIDE (Sensors, 2022, Vol. 22, No. 17, Art. 6662) [Электронный ресурс]. URL: https://www.mdpi.com/1424-8220/22/17/6662.

[8] PLOT4ai Library — Practical Library of Threats 4 Artificial Intelligence [Электронный ресурс] // GitHub репозиторий, верс. от 3 нояб. 2022. URL: https://github.com/PLOT4ai/plot4ai-library.

[9] Kumar V., Mayo J., Bahiss K. ADMIn: Attacks on Dataset, Model and Input. A Threat Model for AI Based Software [Электронный ресурс]. arXiv, 15 янв. 2024. URL: https://arxiv.org/abs/2401.07960.

[10] Huang K. Agentic AI Threat Modeling Framework: MAESTRO [Электронный ресурс] // Cloud Security Alliance Blog, 6 фев. 2025. URL: https://cloudsecurityalliance.org/blog/2025/02/06/agentic-ai-threat-modeling-framework-maestro.

[11] Huang K. 7 Layered Agentic AI Reference Architecture [Электронный ресурс] // Medium, 21 дек. 2024. URL: https://kenhuangus.medium.com/7-layered-agentic-ai-reference-architecture-20276f83b7ee.

[12] OWASP Threat Dragon — Project Site [Электронный ресурс]. URL: https://owasp.org/www-project-threat-dragon/.

[13] OWASP pytm — Pythonic Framework for Threat Modeling [Электронный ресурс]. URL: https://owasp.org/www-project-pytm/.

[14] Microsoft Threat Modeling Tool: Release Notes и Загрузка [Электронный ресурс] // Microsoft Learn, 26 окт. 2023. URL: https://learn.microsoft.com/azure/security/develop/threat-modeling-tool-releases.

[15] von der Assen J., Sharif J., Feng C. и др. Asset-Centric Threat Modeling for AI-Based Systems // Proc. IEEE Int. Conf. on Cyber Security and Resilience, 2024. Pp. 437–444. DOI 10.1109/CSR61664.2024.10679445.

[16] draw.io / diagrams.net — Flowchart Maker & Online Diagram Software [Электронный ресурс]. URL: https://app.diagrams.net/.

[17] European Union Agency for Cybersecurity (ENISA), “Artificial Intelligence Cybersecurity Challenges, Threat Landscape for Artificial Intelligence,” 2020.

[18] OWASP Foundation, “AI Exchange,” May 2024, https://owaspai.org/.

[19] The MITRE Corporation, “MITRE ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems),” https://atlas.mitre.org/.

[20] Agentic AI – Threats and Mitigations / OWASP Agentic Security Initiative. – Version 1.0 – February 2025. – URL: https://genai.owasp.org/resource/agentic-ai-threats-and-mitigations/

[21] OWASP/threat-dragon. Pull Request #1261: Automative threat generation for LLM-agent based systems [Электронный ресурс]. URL: https://github.com/OWASP/threat-dragon/pull/1261

[22] ISO/IEC 27001:2022 – Information security, cybersecurity and privacy protection – Information security management systems – Requirements [Электронный ресурс]. – Международная организация по стандартизации (ISO). URL: https://www.iso.org/standard/27001.html

[23] National Institute of Standards and Technology. NIST Cybersecurity Framework 2.0 [Электронный ресурс] / U.S. Department of Commerce. – Gaithersburg, MD, 2024. – (NIST Cybersecurity Framework, CSWP 29). URL: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf

[24] Сухомлин, Владимир Александрович. "Создание профиля" Кибербезопасность и искусственный интеллект" для направления подготовки ФИИТ на основе куррикулумного подхода." Современные информационные технологии и ИТ-образование 17.3 (2021): 724-734.

[25] Laponina, Olga R., and Sergey A. Malakhovsky. "Using the ZAP vulnerability scanner to test web applications." International Journal of Open Information Technologies 5.8 (2017): 18-26.

[26] Osincev, Aleksandr A., and Olga R. Laponina. "Vulnerability testing in web applications external entities XML." International Journal of Open Information Technologies 7.10 (2019): 71-79.

[27] О работах по цифровой экономике / В. П. Куприяновский, Д. Е. Намиот, С. А. Синягов, А. П. Добрынин // Современные информационные технологии и ИТ-образование. – 2016. – Т. 12, № 1. – С. 243-249. – EDN XEQRFJ.

[28] Волков, А. А. О задачах создания эффективной инфраструктуры среды обитания / А. А. Волков, Д. Е. Намиот, М. А. Шнепс-Шнеппе // International Journal of Open Information Technologies. – 2013. – Т. 1, № 7. – С. 1-10. – EDN ROMIZX.

Threat Modeling Software Development for LLM-Agent-Based Systems

O.R. Laponina, R.N. Kostin

Abstract – **This article describes software that extends the capabilities of the OWASP Threat Dragon threat modeling project for LLM agent-based systems. New diagram components, attributes of new and existing components in Threat Dragon, and the ability to automatically identify threats based on rules are added. The structure defined by OWASP is taken as a reference structure of the LLM agent. The developed software adds support for LLM agent components, allows you to specify 27 new attributes, and automatically identifies threats based on 38 rules. A rule consists of a set of conditions under which a threat may occur, a description of a scenario taken from OWASP documentation, and the mitigations used for this threat, which can be proactive, reactive, and detective. The rule also specifies the threat type according to the STRIDE methodology. The flow diagram of the analyzed system is built manually, and the attributes of all components are also defined manually. Then, threats are automatically added to the diagram based on the diagram, attributes, and rules. If at least one condition is met, the threat will be added to the Threat Dragon diagram for the corresponding component. The authors defined a fairly simple syntax for the rules, which makes it easy to describe new threats, which will then be automatically added to the diagram. The development has received positive feedback from OWASP Threat Dragon leaders. It closes an important gap in the field of LLM agent security and offers a flexible, extensible tool for practical use. The article also provides an overview of seven threat modeling methodologies: STRIDE, LINDDUN, PASTA, STRIDE-AI, PLOT4ai, ADMIn and MAESTRO. An overview of threat modeling software is provided: OWASP Threat Dragon, OWASP pytm, Microsoft Threat Modeling Tool and ThreatFinderAI.**

Keywords – **LLM agents, threat modeling, cybersecurity, AI security, agent systems, OWASP Threat Dragon, automatic threat identification, AI threat analysis.**

REFERENCES

[1] Luo J., Zhang W., Yuan Y. i dr. Large Language Model Agent: A Survey on Methodology, Applications and Challenges [Jelektronnyj resurs] // arXiv, 27 mar. 2025. URL: https://arxiv.org/abs/2503.21460.

[2] Threat Modeling Overview [Jelektronnyj resurs] // OWASP Community Projects. URL: https://owasp.org/www-community/Threat_Modeling#:~:text=Threat%20modeling%20is%20a%20family,of%2C%20threats%20to%20the%20system.

[3] What is a Data Flow Diagram (DFD)? [Jelektronnyj resurs] // IBM Think Blog, 22 noja. 2024. URL: https://www.ibm.com/think/topics/data-flow-diagram.

[4] Allen-Addy C. Threat Modeling Methodology: STRIDE [Jelektronnyj resurs] // Blog IriusRisk, 29 sen. 2023. URL: https://www.iriusrisk.com/resources-blog/threat-modeling-methodology-stride.

[5] LINDDUN — Privacy Threat Modeling Framework [Jelektronnyj resurs]. URL: https://linddun.org/.

[6] Uceda Vélez T., Morana M. M. Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis. Hoboken (N.J.): Wiley, 2015. 696 p. DOI 10.1002/9781118988374.

[7] Mauri L., Damiani E. Modeling Threats to AI-ML Systems Using STRIDE (Sensors, 2022, Vol. 22, No. 17, Art. 6662) [Jelektronnyj resurs]. URL: https://www.mdpi.com/1424-8220/22/17/6662.

[8] PLOT4ai Library — Practical Library of Threats 4 Artificial Intelligence [Jelektronnyj resurs] // GitHub repozitorij, vers. ot 3 nojab. 2022. URL: https://github.com/PLOT4ai/plot4ai-library.

[9] Kumar V., Mayo J., Bahiss K. ADMIn: Attacks on Dataset, Model and Input. A Threat Model for AI Based Software [Jelektronnyj resurs]. arXiv, 15 janv. 2024. URL: https://arxiv.org/abs/2401.07960.

[10] Huang K. Agentic AI Threat Modeling Framework: MAESTRO [Jelektronnyj resurs] // Cloud Security Alliance Blog, 6 fev. 2025. URL: https://cloudsecurityalliance.org/blog/2025/02/06/agentic-ai-threat-modeling-framework-maestro.

[11] Huang K. 7 Layered Agentic AI Reference Architecture [Jelektronnyj resurs] // Medium, 21 dek. 2024. URL: https://kenhuangus.medium.com/7-layered-agentic-ai-reference-architecture-20276f83b7ee.

[12] OWASP Threat Dragon — Project Site [Jelektronnyj resurs]. URL: https://owasp.org/www-project-threat-dragon/.

[13] OWASP pytm — Pythonic Framework for Threat Modeling [Jelektronnyj resurs]. URL: https://owasp.org/www-project-pytm/.

[14] Microsoft Threat Modeling Tool: Release Notes i Zagruzka [Jelektronnyj resurs] // Microsoft Learn, 26 okt. 2023. URL: https://learn.microsoft.com/azure/security/develop/threat-modeling-tool-releases.

[15] von der Assen J., Sharif J., Feng C. i dr. Asset-Centric Threat Modeling for AI-Based Systems // Proc. IEEE Int. Conf. on Cyber Security and Resilience, 2024. Pp. 437–444. DOI 10.1109/CSR61664.2024.10679445.

[16] draw.io / diagrams.net — Flowchart Maker & Online Diagram Software [Jelektronnyj resurs]. URL: https://app.diagrams.net/.

[17] European Union Agency for Cybersecurity (ENISA), "Artificial Intelligence Cybersecurity Challenges, Threat Landscape for Artificial Intelligence," 2020.

[18] OWASP Foundation, "AI Exchange," May 2024, https://owaspai.org/.

[19] The MITRE Corporation, "MITRE ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems)," https://atlas.mitre.org/.

[20] Agentic AI – Threats and Mitigations / OWASP Agentic Security Initiative. – Version 1.0 – February 2025. – URL: https://genai.owasp.org/resource/agentic-ai-threats-and-mitigations/

[21] OWASP/threat-dragon. Pull Request #1261: Automative threat generation for LLM-agent based systems [Jelektronnyj resurs]. URL: https://github.com/OWASP/threat-dragon/pull/1261

[22] ISO/IEC 27001:2022 – Information security, cybersecurity and privacy protection – Information security management systems – Requirements [Jelektronnyj resurs]. – Mezhdunarodnaja organizacija po standartizacii (ISO). URL: https://www.iso.org/standard/27001.html

[23] National Institute of Standards and Technology. NIST Cybersecurity Framework 2.0 [Jelektronnyj resurs] / U.S. Department of Commerce. – Gaithersburg, MD, 2024. – (NIST Cybersecurity Framework, CSWP 29). URL: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf

[24] Suhomlin, Vladimir Aleksandrovich. "Sozdanie profilja" Kiberbezopasnost' i iskusstvennyj intellekt" dlja napravlenija podgotovki FIIT na osnove kurrikulumnogo podhoda." Sovremennye informacionnye tehnologii i IT-obrazovanie 17.3 (2021): 724-734.

[25] Laponina, Olga R., and Sergey A. Malakhovsky. "Using the ZAP vulnerability scanner to test web applications." International Journal of Open Information Technologies 5.8 (2017): 18-26.

[26] Osincev, Aleksandr A., and Olga R. Laponina. "Vulnerability testing in web applications external entities XML." International Journal of Open Information Technologies 7.10 (2019): 71-79.
[27] O rabotah po cifrovoj jekonomike / V. P. Kuprijanovskij, D. E. Namiot, S. A. Sinjagov, A. P. Dobrynin // Sovremennye informacionnye tehnologii i IT-obrazovanie. – 2016. – T. 12, # 1. – S. 243-249. – EDN XEQRFJ.
[28] Volkov, A. A. O zadachah sozdanija jeffektivnoj infrastruktury sredy obitanija / A. A. Volkov, D. E. Namiot, M. A. Shneps-Shneppe // International Journal of Open Information Technologies. – 2013. – T. 1, # 7. – S. 1-10. – EDN ROMIZX.