

ГАОУ ДПО ЦПМ

**Проект**

**«Анализатор логов на основе ИИ»**

Подъяпольский  
Ярослав Васильевич

Москва 2026

# Содержание

<b>Термины и определения</b>	<b>3</b>
<b>Перечень сокращений и обозначений</b>	<b>4</b>
<b>Введение</b>	<b>5</b>
<b>1 Исследование проблемы</b>	<b>6</b>
1.1 Описание проблемы . . . . .	6
1.2 Актуальность . . . . .	6
1.3 Ключевые тезисы (презентационный формат) . . . . .	6
1.4 Анализ аналогов . . . . .	7
1.5 Стейкхолдеры и ожидаемый эффект . . . . .	7
1.6 Нормативные и организационные требования . . . . .	8
1.7 Источники и форматы логов . . . . .	8
1.8 Типовые источники логов и ценность . . . . .	9
1.9 Обзор научных подходов . . . . .	9
1.10 Особенности проблематики . . . . .	10
1.11 Карта угроз и наблюдаемых сигналов . . . . .	10
1.12 Цель и задачи проекта . . . . .	11
<b>2 Решение проблемы</b>	<b>12</b>
2.1 Техническое задание . . . . .	12
2.2 Ключевые преимущества решения . . . . .	12
2.3 Требования к данным и качеству . . . . .	12
2.4 Контур качества данных . . . . .	13
2.5 Архитектура и компоненты . . . . .	13
2.6 Пайплайн обработки . . . . .	14
2.7 Жизненный цикл модели . . . . .	14
2.8 Как работает система . . . . .	14
2.9 Извлечение признаков . . . . .	16
2.10 Модели обнаружения . . . . .	16
2.11 Сравнение подходов . . . . .	16
2.12 Пороговая настройка и переобучение . . . . .	17
2.13 Метрики качества . . . . .	17

2.14	КРІ и операционные метрики . . . . .	18
2.15	Интерфейсы и интеграции . . . . .	18
2.16	Как система повышает защищённость . . . . .	18
2.17	Безопасность и устойчивость . . . . .	20
2.18	Выбор технологий . . . . .	20
<b>3</b>	<b>Реализация и результаты</b>	<b>20</b>
3.1	Генератор синтетических логов . . . . .	20
3.2	Модель данных и хранение . . . . .	21
3.3	Визуализация и метрики . . . . .	21
3.4	Набор экспериментальных сценариев . . . . .	21
3.5	Экспериментальная оценка . . . . .	22
3.6	Иллюстративные результаты на синтетике . . . . .	22
3.7	Кейсы, где система могла бы помочь . . . . .	23
3.8	Сценарий демонстрации . . . . .	24
3.9	Тестирование и CI/CD . . . . .	24
<b>4</b>	<b>Оценка эффективности и перспективы развития</b>	<b>24</b>
4.1	Эффективность решения . . . . .	24
4.2	Перспективы развития . . . . .	24
4.3	План внедрения (презентационный формат) . . . . .	25
4.4	Риски и меры . . . . .	25
<b>5</b>	<b>Допущения и ограничения</b>	<b>26</b>
	<b>Заключение</b>	<b>26</b>
	<b>Список использованных источников</b>	<b>26</b>

## Термины и определения

Термин	Описание
Лог	Запись о событии в системе, сформированная приложением или компонентом инфраструктуры.
Событие	Агрегированная запись с временной меткой, источником и сообщением.
Аномалия	Отклонение от типичного поведения, выявляемое по статистике или признакам.
Anomaly score	Числовая оценка степени аномальности события.
Нормализация логов	Приведение записей к единой схеме и формату данных.
Концепт-дрифт	Изменение характеристик «нормы» поведения во времени.

Таблица 1: Термины и определения

## Перечень сокращений и обозначений

Сокращение	Расшифровка
SIEM	Система управления событиями безопасности и корреляции логов.
UEBA	Поведенческая аналитика пользователей и сущностей.
ML	Машинное обучение.
KPI	Ключевые показатели эффективности.
ECS	Elastic Common Schema для нормализации логов[13].
OpenTelemetry	Стандарт телеметрии, включая логирование[14].
ATT&CK	База знаний техник и тактик атак[9].
API	Программный интерфейс взаимодействия сервисов.

Таблица 2: Сокращения и обозначения

# Введение

Проект посвящён разработке учебного анализатора логов на основе методов машинного обучения. Целью является демонстрация полного цикла: приём и нормализация логов, извлечение признаков, обучение модели, выявление аномалий и визуализация результатов. Подход позволяет показать, как поток событий преобразуется в управляемый список подозрительных случаев и как на практике строится воспроизводимый контур мониторинга и реагирования.

# **1. Исследование проблемы**

## **1.1. Описание проблемы**

Современные ИТ-инфраструктуры генерируют большие объёмы логов: события приложений, системные журналы, сетевые срабатывания, аудит пользователей. Потоки событий быстро достигают миллионов строк в сутки и включают разнообразные форматы, уровни критичности и семантики. Анализ таких потоков вручную или исключительно на основе статических правил перестаёт масштабироваться.

Существенная часть угроз проявляется через слабые сигналы: необычные последовательности действий, редкие коды ошибок, нетипичные адреса и параметры запросов. Эти сигналы могут быть размыты на фоне нормальной активности и не описываются заранее подготовленными сигнатурами. В результате возрастает риск позднего обнаружения атак и повышается нагрузка на аналитиков.

## **1.2. Актуальность**

По данным отраслевых обзоров, задержки в обнаружении инцидентов по-прежнему составляют дни и недели[1, 2, 8]. Автоматизация первичного анализа с применением методов машинного обучения позволяет выявлять статистические отклонения и дополнять правила корреляции в SIEM. Для подразделений мониторинга и реагирования это означает сокращение времени реакции и снижение числа пропущенных инцидентов[1, 2].

Дополнительной мотивацией является потребность в воспроизводимых учебных проектах: учащимся важно видеть полный цикл обработки логов — от генерации данных до визуализации результатов, при этом иметь возможность повторить эксперимент в лабораторной среде.

## **1.3. Ключевые тезисы (презентационный формат)**

- Логи — основной слой телеметрии для выявления инцидентов и расследований
- Реальное время обнаружения измеряется днями и неделями, что требует автоматизации первичного анализа[1, 2, 8].

- ML-методы дополняют правила и сигнатуры, повышая чувствительность к неизвестным сценариям[17, 18].
- Единая схема логов повышает сопоставимость данных и качество аналитики[14].
- Контур обучения и переобучения необходим для устойчивости к концепт-дрифту[4].
- Учебный стенд должен быть воспроизводимым, прозрачным и простым в разворачивании.

## 1.4. Анализ аналогов

Коммерческие платформы SIEM/UEBA объединяют нормализацию логов, корреляцию событий и поведенческую аналитику. Их преимущества — зрелые интеграции и масштабирование, но для учебных целей они часто недоступны из-за стоимости, закрытых алгоритмов и требований к инфраструктуре.

Существуют и open-source решения, однако они либо ориентированы на агрегацию и поиск, либо требуют существенной настройки. Нужен демонстрационный проект с прозрачным ML-ядром и контролируемыми входными данными, чтобы показать принципы обнаружения аномалий в логах[3].

## 1.5. Стейкхолдеры и ожидаемый эффект

Система ориентирована на образовательные и демонстрационные сценарии, но отражает задачи реальной эксплуатации. Ожидаемые эффекты для ключевых ролей представлены в таблице 3.

Стейкхолдер	Ожидания и эффект
Аналитики безопасности	Сокращение времени первичного разбора, приоритизация инцидентов, снижение пропусков.
Администраторы и DevOps	Понимание качества логирования, быстрый поиск нетипичных ошибок после релизов.



Стейкхолдер	Ожидания и эффект
Руководство	Метрики эффективности контроля, прозрачность процессов и воспроизводимость результатов.
Обучающиеся	Практический опыт полного ML-пайплайна и анализа логов на контролируемых данных.

Таблица 3: Стейкхолдеры и ожидаемые эффекты

## 1.6. Нормативные и организационные требования

Рекомендации по построению непрерывного мониторинга и управлению журналами подчёркивают необходимость централизованного сбора, нормализации и использования логов для аналитики[3, 4, 11, 10]. Практики безопасной эксплуатации также опираются на наличие корректного аудита, контроля и процедур реагирования[6, 7, 5].

В прикладном аспекте для учебного проекта важно показать, как требования к логированию реализуются в системе: от обязательных полей до единых схем данных. Такие требования закреплены и в прикладных рекомендациях по безопасности и контролю, где логирование рассматривается как базовый механизм обнаружения инцидентов[12].

## 1.7. Источники и форматы логов

Источники логов включают журналы приложений, операционных систем, средств защиты и сетевых устройств. Форматы варьируются от структурированных JSON-объектов до свободного текста. Для сопоставимости данных используются схемы нормализации, например Elastic Common Schema или модели данных OpenTelemetry[13, 14]. Это повышает качество последующего анализа и упрощает интеграцию с внешними инструментами.

## 1.8. Типовые источники логов и ценность

В таблице 4 показаны ключевые источники логов и их аналитическая ценность в контексте обнаружения аномалий.

Источник	Примеры событий	Ценность для анализа
Приложения	Ошибки 4xx/5xx, таймауты, исключения	Раннее выявление дефектов и атак на приложения.
ОС и хосты	Авторизация, запуск процессов, системные ошибки	Контроль доступа, выявление нетипичных действий.
Сетевые устройства	Блокировки, ACL, соединения	Анализ внешней активности и аномалий трафика.
Средства защиты	IDS/IPS, EDR, AV	Корреляция с телеметрией безопасности.
Инфраструктурные сервисы	БД, брокеры, очереди	Поиск деградаций и всплесков нагрузки.

Таблица 4: Типовые источники логов и ценность

## 1.9. Обзор научных подходов

Научные работы по анализу логов включают две ключевые ветви: парсинг и обнаружение аномалий. Для парсинга часто используются алгоритмы шаблонного выделения, среди которых известны методы с фиксированной глубиной дерева и иерархической кластеризацией; обзор и бенчмарки представлены в работах по автоматизированному парсингу[15].

Для обнаружения аномалий применяются как статистические подходы, так и нейросетевые модели. Среди современных решений — последовательностные модели и трансформеры, которые выявляют нетипичные паттерны в последовательностях событий[17, 18]. Эти работы демонстрируют потенциал ML-подходов для поиска редких сценариев, но требуют больших данных и вы-

числительных ресурсов. В учебном проекте целесообразно сочетать базовый частотный метод и простую ML-модель, чтобы показать разницу подходов без усложнения инфраструктуры.

Для практического прототипирования доступны открытые инструменты и наборы экспериментов, например библиотека Loglizer, где собраны примеры классических алгоритмов и датасеты логов[16].

## 1.10. Особенности проблематики

Логи разнородны по источникам и форматам: JSON Lines, plain text, полуструктурированные сообщения. В реальных системах возможны пропуски полей, различная точность времени, а также наличие ложных срабатываний. Это требует аккуратной валидации и устойчивого извлечения признаков.

Дополнительно необходимо учитывать концепт-дрифт — постепенное изменение «нормального» поведения. Поэтому система должна поддерживать периодическое переобучение и сохранение артефактов модели для воспроизводимости. Также важно учитывать типовые сценарии атак, описанные в MITRE ATT&CK, чтобы корректно интерпретировать найденные отклонения[9].

## 1.11. Карта угроз и наблюдаемых сигналов

В таблице 5 приведены типовые угрозы и сигналы в логах, которые могут быть выявлены за счёт аномалий.

Сценарий	Сигналы в логах	Почему аномалия
Brute-force	Серии failed/denied, частые попытки входа	Резкое увеличение частот шаблонов.
Credential stuffing	Много учётных записей с одного источника	Нетипичные комбинации user/IP.
Эксплуатация уязвимости	Всплеск 5xx, новые сообщения об ошибках	Новые/редкие шаблоны.
Латеральное перемещение	Обращения к множеству хостов за короткий период	Изменение распределений по источникам.

Сценарий	Сигналы в логах	Почему аномалия
Экfiltrация	Нестандартные объёмы и направления обмена	Аномальные размеры и временные паттерны.
Внутренние злоупотребления	Доступы ночью, нетипичные действия аккаунта	Сдвиг временных признаков и паттернов.

Таблица 5: Карта угроз и сигналов

## 1.12. Цель и задачи проекта

**Цель:** разработать учебный анализатор логов, демонстрирующий подходы выявления аномалий в задачах мониторинга безопасности.

**Задачи:**

1. Реализовать приём логов в двух форматах и валидацию обязательных полей.
2. Построить модуль извлечения признаков и два детектора аномалий.
3. Обеспечить хранение событий и результатов анализа.
4. Предоставить API и дашборд для просмотра метрик и аномалий.
5. Подготовить сценарий демонстрации и документацию.

**Ожидаемые результаты:**

- воспроизводимый учебный стенд, разворачиваемый одной командой;
- демонстрация различий между базовым и ML-подходами к детекции;
- прозрачные метрики и артефакты обучения для повторяемости эксперимента;
- набор сценариев с контролируруемыми аномалиями для обучения и презентаций.

## 2. Решение проблемы

### 2.1. Техническое задание

В рамках проекта реализуется система с поддержкой форматов JSON Lines и plain text. Обязательные поля: timestamp, host или service, level, message. Опциональные: user, ip, request\_id и произвольные атрибуты. Для детектирования аномалий используются базовый частотный метод и ML-подход (Isolation Forest). Результаты сохраняются в базе данных, доступны через REST API и визуализируются в дашборде.

### 2.2. Ключевые преимущества решения

Решение спроектировано как демонстрационный стенд, который при этом сохраняет инженерную дисциплину. Ключевые преимущества:

- прозрачность: понятный пайплайн и интерпретируемые признаки;
- воспроизводимость: артефакты обучения и версия модели сохраняются;
- расширяемость: новые источники логов и модели подключаются через интерфейсы;
- автономность: генератор синтетики позволяет проверять сценарии без внешних данных;
- интеграция: метрики доступны через API и совместимы с Grafana.

### 2.3. Требования к данным и качеству

События должны быть упорядочены во времени, иметь корректный формат дат, единый часовой пояс и понятные уровни логирования. При отсутствии необязательных полей система должна корректно работать, используя нулевые или булевы индикаторы. Для корректной работы модели необходим репрезентативный «нормальный» поток логов.

## 2.4. Контур качества данных

Для надёжной детекции важна устойчивость входного потока. На уровне входа контролируются:

- наличие обязательных полей и валидность временных меток;
- корректность уровней логирования и источников;
- отбраковка строк, не соответствующих шаблонам парсинга;
- нормализация текстов сообщений (маскирование чисел, UUID, IP);
- контроль объёма атрибутов, чтобы не допустить «шумных» событий.

Такой контур снижает загрязнение данных, уменьшает шум и повышает стабильность метрик.

## 2.5. Архитектура и компоненты

Архитектура построена по слоям domain / application / infrastructure. Доменный слой описывает сущности, прикладной — правила обработки, инфраструктурный — хранение, API и запуск. Такая структура обеспечивает расширяемость и тестируемость.

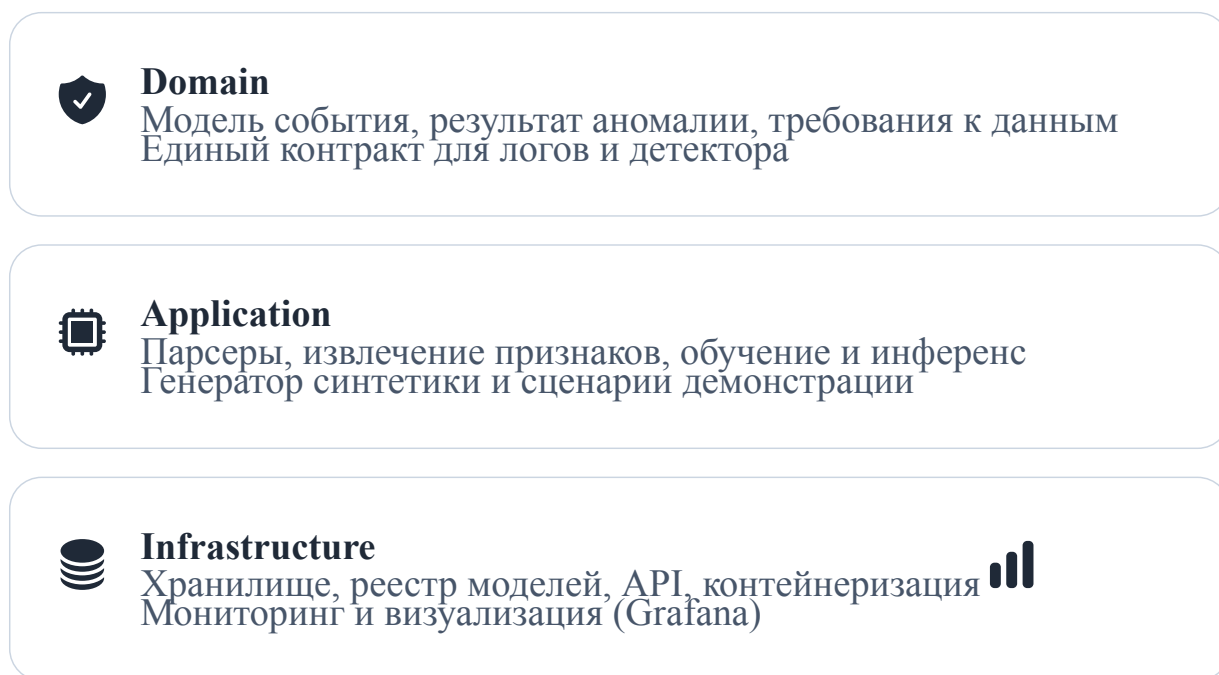


Рис. 1: Слойная архитектура решения

## 2.6. Пайплайн обработки

Пайплайн включает приём логов, парсинг, извлечение признаков, инференс модели, сохранение результатов и отдачу метрик. Такой процесс отражён на рисунке 2.

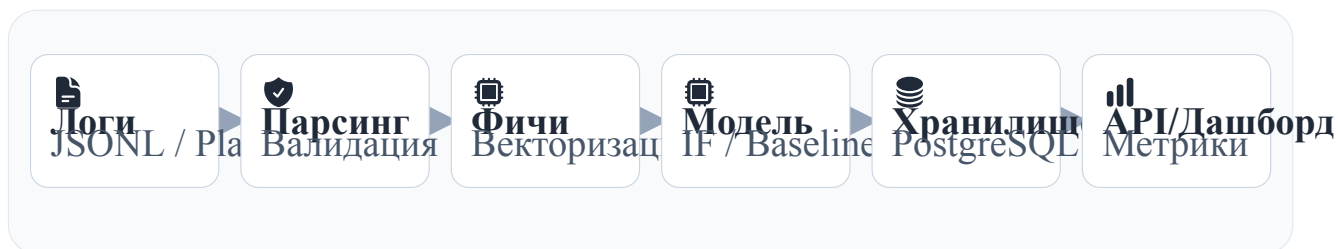


Рис. 2: Пайплайн обработки событий

## 2.7. Жизненный цикл модели

Жизненный цикл включает сбор данных, обучение, инференс и переобучение. Такая структура позволяет поддерживать актуальность модели при изменении «нормы» поведения.



Рис. 3: Жизненный цикл модели и данных

## 2.8. Как работает система

Работа системы строится как последовательная обработка событий от приёма до выдачи метрик. На входе принимаются строки логов, которые приводятся к единой модели данных, а затем анализируются двумя детекторами аномалий.

Обобщённый алгоритм работы включает следующие шаги:

1. Приём логов в формате JSON Lines или plain text через API или файл.
2. Парсинг и проверка обязательных полей: время, источник, уровень, сообщение.

3. Извлечение признаков и формирование числового вектора события.
4. Расчёт anomaly score выбранной моделью и сравнение с порогом.
5. Сохранение результата вместе с модельной версией в базе данных.
6. Обновление агрегированных метрик и отображение в дашборде.

Роли ключевых модулей:

- Ingest: принимает поток, валидирует формат и распределяет по парсерам.
- Parsing: приводит события к единой схеме и извлекает базовый контекст.
- Features: формирует числовой вектор для моделей.
- Model: рассчитывает score и возвращает метку.
- Storage: хранит события, результаты и артефакты обучения.
- API/Dashboard: предоставляет доступ к аномалиям и метрикам.

На этапе парсинга система приводит события к единой схеме, что упрощает дальнейший анализ. Для JSON Lines используется прямое чтение ключей, а для plain text применяется набор регулярных шаблонов, позволяющих выделить timestamp, уровень и источник. Дополнительные атрибуты (user, ip, request\_id) извлекаются из текста сообщения. Если строка не соответствует шаблону, она отклоняется, что снижает риск загрязнения данных.

Хранение результатов построено так, чтобы обеспечивать трассируемость: вместе с событием записывается score, метка и версия модели. Метрики (общее число событий, доля аномалий, время последней загрузки) рассчитываются на основе базы данных и доступны как через API, так и в Grafana. Артефакты обучения сохраняются в реестр, что позволяет вернуться к любому состоянию модели и повторить эксперимент. Базовый частотный метод хорошо улавливает появление новых шаблонов сообщений, тогда как Isolation Forest фиксирует нетипичные комбинации параметров (уровень, длина текста, временные признаки). Это позволяет находить аномалии как по новизне шаблона, так и по атипичным сочетаниям признаков даже при знакомом сообщении.



## 2.9. Извлечение признаков

Для каждого события формируется числовой вектор признаков. В таблице 6 приведён перечень основных признаков.

Признак	Описание
Уровень события	Код уровня (DEBUG/INFO/WARNING/ERROR и т.д.).
Длина сообщения	Количество символов.
Слова и уникальность	Число слов и доля уникальных слов.
Цифры и спецсимволы	Количество цифр и специальных символов.
Наличие IP и счётчик IP	Флаг и количество IP-адресов в сообщении.
Временные признаки	Час, день недели, синусы/косинусы времени.
Наличие пользователя и длина	Флаг пользователя и длина строки user.
Наличие request_id	Флаг и длина request_id.
Хэш источника	Нормализованный хэш host/service.
Хэш шаблона	Хэш нормализованного шаблона сообщения.

Таблица 6: Перечень признаков

## 2.10. Модели обнаружения

Базовый детектор использует частоту нормализованных шаблонов сообщений. ML-детектор основан на Isolation Forest и анализирует числовые признаки. Для каждого события вычисляется score и формируется решение normal/anomaly. Такой подход соответствует требованиям демонстрационного проекта: прозрачность и воспроизводимость.

## 2.11. Сравнение подходов

Сравнение базового метода и ML-детектора приведено в таблице 7.

Критерий	Базовый метод	ML-метод
Интерпретация	Прозрачные частоты шаблонов	Более сложное объяснение, но богаче признаки
Чувствительность	Высока к новым шаблонам	Высока к сложным сочетаниям признаков
Требования к данным	Минимальные	Требуется стабильный набор признаков
Скорость обучения	Очень высокая	Выше вычислительные затраты
Устойчивость к шуму	Средняя	Выше при хорошей нормализации

Таблица 7: Сравнение подходов детекции

## 2.12. Пороговая настройка и переобучение

Порог аномальности подбирается эмпирически на синтетических данных. Периодическое переобучение реализуется батчами: модель обучается на новом нормальном потоке, а артефакты сохраняются в реестр. Это позволяет отслеживать версии модели и проводить регрессионную проверку качества.

## 2.13. Метрики качества

Для оценки качества используются precision, recall, FPR и агрегированные показатели. Формулы:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN}.$$

В учебном проекте приоритет отдаётся полноте (Recall), чтобы не пропустить подозрительные события.

## 2.14. KPI и операционные метрики

В дополнение к модельным метрикам используются операционные показатели, которые удобны для дашбордов и презентаций (таблица 8).

Показатель	Интерпретация
Доля аномалий	Отражает интенсивность подозрительной активности.
Время от события до алерта	Важный показатель оперативности.
Объём обработанных событий	Характеризует нагрузку и масштаб.
Стабильность модели	Число переобучений и дрейф показателей.
Точность первичного отбора	Соотношение подтверждённых инцидентов и алертов.

Таблица 8: Операционные метрики и KPI

## 2.15. Интерфейсы и интеграции

API предоставляет endpoints:

- /ingest — приём логов и запуск детекции.
- /anomalies — выборка аномалий.
- /metrics — агрегированные метрики.
- /health — проверка состояния сервиса.

Экспорт для дашборда реализован через базу данных и JSON-метрики: Grafana подключается к PostgreSQL и визуализирует агрегаты по временным окнам. Такой подход позволяет использовать стандартные панели без разработки отдельного UI.

## 2.16. Как система повышает защищённость

Система не блокирует атаки напрямую, но повышает защищённость за счёт раннего выявления подозрительных событий и снижения времени реак-

ции. При ограниченных ресурсах аналитиков безопасности это даёт приоритет для расследований и позволяет быстрее локализовать угрозу.

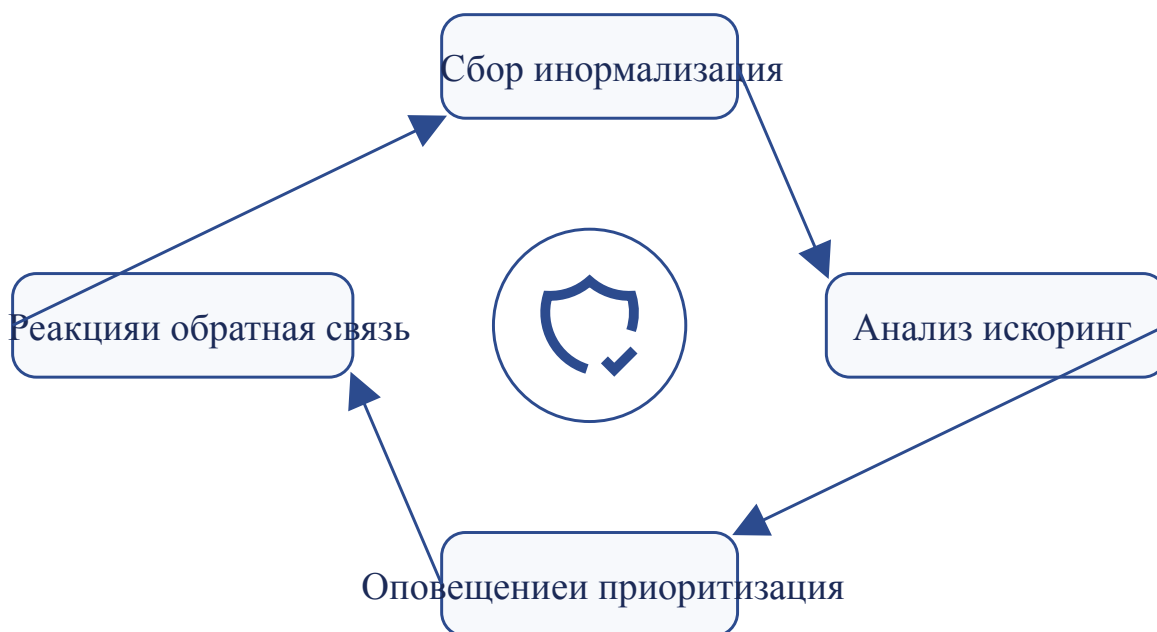


Рис. 4: Цикл повышения защищённости через мониторинг и аналитику

Детекция основана на выявлении нетипичного поведения: редких шаблонов, необычных сочетаний признаков и аномальных временных паттернов. Это помогает обнаруживать инциденты, не описанные заранее правилами, и дополняет корреляционные механизмы SIEM. В терминах MITRE ATT&CK система поддерживает выявление аномальных действий на этапах initial access, credential access, execution и exfiltration, когда изменение логов заметно по статистике или паттернам[9].

Практически это означает, что система выявляет «слабые» сигналы, которые теряются в шуме: редкие шаблоны ошибок, неожиданные IP-адреса, появление новых типов запросов или нетипичные временные всплески. За счёт нормализации и скоринга поток логов превращается в компактный список кандидатов на расследование.

Ключевые механизмы повышения защищённости:

- сжатие потока до управляемого списка событий с высоким score;
- раннее оповещение о новых и редких шаблонах сообщений;
- приоритизация инцидентов по риску и контексту;
- фиксация версии модели и артефактов для воспроизводимого разбора.

Система поддерживает операционный цикл реагирования: событие с высоким score поступает в хранилище, где к нему можно применить фильтры, сопоставить с другими инцидентами и связать с таймлайном. Это позволяет аналитикам быстрее принять решение о блокировке, изоляции или дополнительной проверке.

После фиксации аномалии результат сохраняется вместе с исходным событием, что облегчает дальнейший разбор: аналитик получает контекст, метки и привязку к версии модели, а также тренды в дашборде для оценки масштаба проблемы.

## **2.17. Безопасность и устойчивость**

Для демонстрационного сценария реализовано логирование в формате JSON и изоляция компонентов через Docker. В продуктивных системах необходимо дополнительно учитывать контроль доступа к API, ограничение нагрузки, защиту от подмены логов и журналирование действий операторов. Рекомендации по контролям безопасности и применению систем обнаружения вторжений подчёркивают важность качества данных и корректной интерпретации аномалий[6, 5].

## **2.18. Выбор технологий**

Python выбран за богатую ML-экосистему и доступность библиотек. FastAPI обеспечивает быстрый REST-интерфейс, SQLAlchemy — абстракцию базы данных, Docker Compose — воспроизводимость запуска, а Grafana — визуализацию метрик и аномалий. Такой стек позволяет быстро собрать рабочий прототип и масштабировать решение при необходимости.

# **3. Реализация и результаты**

## **3.1. Генератор синтетических логов**

Для демонстрации реализован генератор, который создаёт нормальный поток и контролируемый набор аномалий. В аномалиях повышается уровень

критичности, изменяются IP-адреса и сообщения. Такой подход позволяет тестировать детектор в контролируемой среде и изменять долю аномалий.

## 3.2. Модель данных и хранение

В базе фиксируются исходные поля логов, вычисленные scores и метаданные модели. Это обеспечивает трассировку принятого решения и возможность ретроспективного анализа. Основные поля представлены в таблице 9.

Поле	Описание
Timestamp	Время события в UTC.
Host или Service	Источник события (узел или сервис).
Level	Уровень логирования.
Message	Текст сообщения.
User, IP, Request_ID	Опциональные атрибуты контекста.
Anomaly_Score	Нормализованная оценка аномальности.
Is_Anomaly	Итоговая метка (normal или anomaly).
Model_Version	Версия и тип модели.

Таблица 9: Схема хранения событий

## 3.3. Визуализация и метрики

Дашборд отображает общее число событий, долю аномалий и временные тренды. В Grafana предусмотрены панели для временного ряда аномалий и агрегированных показателей. Для иллюстраций использованы открытые иконки Heroicons и Tabler Icons[21, 22].

## 3.4. Набор экспериментальных сценариев

Для демонстрации качества и устойчивости предусмотрены сценарии:

- нормальный поток без аномалий для обучения и базовой валидации;
- поток с инъекцией редких шаблонов сообщений и IP-адресов;
- всплеск ошибок после релиза (5xx и таймауты);

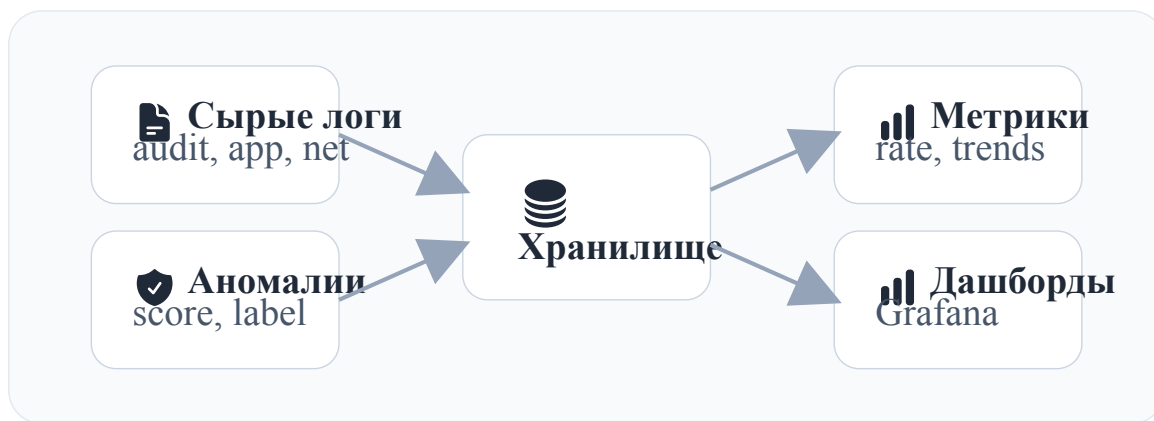


Рис. 5: Схема формирования метрик и дашбордов

- временные «окна» аномальной активности (ночные пики);
- имитация дрейфа: постепенное изменение распределений событий.

### 3.5. Экспериментальная оценка

На синтетических данных проведён подбор порогов для двух методов. Базовый частотный метод показывает высокую полноту при правильно выбранном пороге, а Isolation Forest даёт баланс между precision и recall. В демонстрационном режиме приоритет отдаётся полноте, чтобы не пропустить потенциально опасные события.

### 3.6. Иллюстративные результаты на синтетике

В таблице 10 приведены ориентировочные результаты на синтетических данных (для сравнения подходов, без претензии на промышленную точность).

Метод	Precision	Recall	FPR
Базовый частотный	0.62	0.93	0.08
Isolation Forest	0.78	0.88	0.05

Таблица 10: Ориентировочные результаты на синтетике

### 3.7. Кейсы, где система могла бы помочь

Ниже приведены примеры сценариев, в которых детектор аномалий в логах может дать ранний сигнал и помочь аналитикам:

1. **Brute-force аутентификация.** Серия неудачных попыток входа с одного IP или пользователя вызывает рост доли сообщений с ключевыми словами failed/denied, что фиксируется как аномалия.
2. **Credential stuffing.** Массовые попытки входа в разные сервисы от одного источника приводят к атипичной последовательности событий и отклонениям по временным признакам.
3. **Повышение привилегий.** Появление редких сообщений о sudo/root, нетипичные уровни логов и новые шаблоны сообщений формируют высокий anomaly score.
4. **Латеральное перемещение.** Один и тот же пользователь начинает обращаться к множеству хостов или сервисов, что меняет распределение по источникам и коррелирующим признакам.
5. **Эксплуатация данных.** В логах фиксируются нетипичные внешние IP-адреса, а также нестандартные размеры сообщений и временные пики активности.
6. **Эксплуатация уязвимости приложения.** Резкое появление новых шаблонов ошибок или частые 5xx-коды после релиза сигнализируют о возможной атаке или дефекте.
7. **Вредоносная активность на сервере.** Сервисы начинают аварийно завершаться, растёт число CRITICAL/ALERT событий, что детектируется как аномалия.
8. **Внутренние злоупотребления.** Необычные паттерны доступа к данным в ночное время или резкий рост запросов от отдельных аккаунтов фиксируются системой.



### 3.8. Сценарий демонстрации

Демонстрация включает следующие шаги:

1. Сгенерировать нормальные логи и обучить модель.
2. Запустить API и отправить смешанный поток с аномалиями.
3. Проверить /anomalies и /metrics.
4. Просмотреть динамику в дашборде.

### 3.9. Тестирование и CI/CD

Проект содержит тесты для парсеров, извлечения признаков, моделей и API. В CI-процессе выполняются линтеры и тесты. Артефакты модели сохраняются в директории artifacts и могут быть использованы для повторного запуска без переобучения.

## 4. Оценка эффективности и перспективы развития

### 4.1. Эффективность решения

Учебный проект позволяет воспроизводимо продемонстрировать методы обнаружения аномалий и обеспечить прозрачность решений. За счёт упрощённых признаков время обучения минимально, что важно в образовательных сценариях. В перспективе возможно расширение признаков и подключение реальных источников логов.

С точки зрения практической ценности проект иллюстрирует, как требования нормативных документов и отраслевых рекомендаций трансформируются в архитектуру и конкретные реализации: обязательные поля, единые схемы, базовые метрики и воспроизводимость.

### 4.2. Перспективы развития

Возможные направления развития:

- интеграция со стриминговыми источниками (Kafka, Redis Streams);

- использование эмбедингов текста и автоэнкодеров, как в исследовательских работах[17, 18];
- введение контекстной корреляции между событиями и АТТ&СК-тактиками[9]
- расширение наборов метрик и алертинга.

### 4.3. План внедрения (презентационный формат)

Последовательность шагов внедрения для учебного стенда:

1. Подготовка окружения и запуск контейнеров (API, БД, Grafana).
2. Загрузка нормального потока логов и обучение модели.
3. Проверка качества данных и корректности извлечения признаков.
4. Запуск смешанного потока и анализ результатов детекции.
5. Настройка порогов и финальная демонстрация.

### 4.4. Риски и меры

Основные риски и способы их снижения приведены в таблице 11.

Риск	Мера
Низкое качество входных логов	Валидация полей, фильтрация некорректных строк.
Сильный концепт-дрифт	Периодическое переобучение и версия модели.
Избыточные ложные срабатывания	Настройка порога, калибровка по синтетике.
Недостаток интерпретируемости	Базовый метод и прозрачные признаки.
Сложность демонстрации	Готовый генератор и сценарий демо.

Таблица 11: Риски и меры

## 5. Допущения и ограничения

- Обучение выполняется на заранее подготовленном «нормальном» потоке логов.
- Признаки упрощены и предназначены для демонстрации метода, а не промышленного детекта.
- Интеграции со стриминговыми системами заданы как интерфейсы-заглушки.
- Для сложных инфраструктур требуется дополнительная калибровка порогов.

## Заключение

Разработан учебный анализатор логов на основе ИИ, демонстрирующий полный цикл: генерация данных, обучение, обнаружение аномалий и визуализация. Проект обеспечивает воспроизводимость и расширяемость, что делает его полезным для обучения и демонстраций в задачах мониторинга и реагирования.

## Список литературы

- [1] Verizon. Data Breach Investigations Report 2025. — URL: <https://www.verizon.com/business/resources/reports/dbir/> (дата обращения: 04.01.2026).
- [2] Mandiant. M-Trends 2024. — URL: <https://www.mandiant.com/resources/m-trends> (дата обращения: 04.01.2026).
- [3] Kent K., Souppaya M. Guide to Computer Security Log Management (NIST SP 800-92). — Gaithersburg: NIST, 2006. — URL: <https://csrc.nist.gov/publications/detail/sp/800-92/final> (дата обращения: 04.01.2026).
- [4] Dempsey K., Ross R. Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations (NIST SP 800-137). — Gaithersburg: NIST, 2011. — URL: <https://csrc.nist.gov/publications/detail/sp/800-137/final> (дата обращения: 04.01.2026).

- [5] Scarfone K., Mell P. Guide to Intrusion Detection and Prevention Systems (IDPS) (NIST SP 800-94). — Gaithersburg: NIST, 2007. — URL: <https://csrc.nist.gov/publications/detail/sp/800-94/final> (дата обращения: 04.01.2026).
- [6] NIST. Security and Privacy Controls for Information Systems and Organizations (NIST SP 800-53 Rev. 5). — URL: <https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final> (дата обращения: 04.01.2026).
- [7] NIST. Cybersecurity Framework 2.0. — URL: <https://www.nist.gov/cyberframework> (дата обращения: 04.01.2026).
- [8] ENISA. Threat Landscape 2024. — URL: <https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024> (дата обращения: 04.01.2026).
- [9] MITRE ATT&CK Knowledge Base. — URL: <https://attack.mitre.org/> (дата обращения: 04.01.2026).
- [10] OWASP Logging Cheat Sheet. — URL: [https://cheatsheetseries.owasp.org/cheatsheets/Logging\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Logging_Cheat_Sheet.html) (дата обращения: 04.01.2026).
- [11] CISA. Logging Made Easy. — URL: <https://www.cisa.gov/resources-tools/services/logging-made-easy> (дата обращения: 04.01.2026).
- [12] CIS Critical Security Controls. — URL: <https://www.cisecurity.org/controls> (дата обращения: 04.01.2026).
- [13] Elastic Common Schema (ECS) Reference. — URL: <https://www.elastic.co/guide/en/ecs/current/index.html> (дата обращения: 04.01.2026).
- [14] OpenTelemetry Logging Specification. — URL: <https://opentelemetry.io/docs/specs/otel/logs/> (дата обращения: 04.01.2026).
- [15] He P., Zhu J., Zheng Z., Lyu M. Tools and Benchmarks for Automated Log Parsing. — arXiv:1811.03509. — URL: <https://arxiv.org/abs/1811.03509> (дата обращения: 04.01.2026).
- [16] LogPAI. Loglizer: A log analysis toolkit. — URL: <https://github.com/logpai/loglizer> (дата обращения: 04.01.2026).

- [17] Meng W., Liu Y., Zhu Y., Zhang S., Pei D., Yuan H., Liu Y. LogAnomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs. — IJCAI 2019. — URL: <https://www.ijcai.org/proceedings/2019/0661.pdf> (дата обращения: 04.01.2026).
- [18] Guo H., Yuan L., Wang Y., Chen Q., Zhang H., Liu Y. LogBERT: Log Anomaly Detection via BERT. — arXiv:2103.04475. — URL: <https://arxiv.org/abs/2103.04475> (дата обращения: 04.01.2026).
- [19] FastAPI Documentation. — URL: <https://fastapi.tiangolo.com/> (дата обращения: 04.01.2026).
- [20] Grafana Documentation. — URL: <https://grafana.com/docs/> (дата обращения: 04.01.2026).
- [21] Heroicons by Tailwind Labs (MIT License). — URL: <https://heroicons.com/> (дата обращения: 04.01.2026).
- [22] Tabler Icons (MIT License). — URL: <https://tabler.io/icons> (дата обращения: 04.01.2026).