Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

ИИ-агенты в SOC: архитектура, при которой прод не падает

Как мы строим ИИ-триаж на Wazuh 4.14.7 + локальной LLM с HITL на Tier-0

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

У офиса на 30-50 РМ поток алертов Wazuh, EDR и Zabbix идёт тысячами в сутки, и почти всё — шум. Аналитик выгорает и перестаёт открывать события, а реальный инцидент проходит мимо. Мы ставим ИИ-агента на первичный триаж, корреляцию и классификацию по ATT&CK, но так, чтобы он физически не мог выполнить деструктивное действие без человека.

Почему это важно бизнесу

- Простой офиса от ошибочной автоизоляции DC или блокировки учётки директора стоит дороже всего проекта внедрения
- Пропущенный из-за усталости аналитика инцидент — это шифровальщик и остановка 1С и почты на несколько суток
- Отправка сырых логов в облачный LLM-API уводит ПДн за периметр и создаёт прямое нарушение 152-ФЗ
- Один аналитик ИБ на полставки перестаёт быть точкой отказа: агент держит первую линию, человек — решения
- Разбор инцидента с готовым резюме и доказательствами занимает минуты, а не часы оплачиваемого времени

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

4.14.7

Ветка Wazuh, на которой фиксируем стенд: с неё берём Server API и active response

documentation.wazuh.com, release notes

v18.1

Версия ATT&CK STIX-бандла в RAG: Detection Strategies и Analytics вместо прежних Detections

attack.mitre.org, ATT&CK v18.1

0.92

Дефолт --gpu-memory-utilization в vLLM: на карте, где рядом живёт Qdrant, опускаем до 0.90

docs.vllm.ai, Engine arguments

16384

max-model-len для Qwen3-32B AWQ на карте 24 ГБ: остаток памяти забирает KV-cache

карточка модели Qwen3-32B

900 с

Таймаут HITL-тикета на действия Tier-0: не подтвердили — действие не выполняется

наш регламент HITL

abstain

Пятое значение поля УВЕРЕННОСТЬ: агент обязан отказаться от вывода, а не угадывать

наш шаблон промпта триажа

РЕАЛИЗАЦИЯ 1 ИЗ 3

Слой инструментов: MCP-сервер с проверкой Tier перед вызовом

Что настраиваем

MCP-сервер на отдельной VM контура мониторинга; инструменты к Wazuh API, AD и firewall, модель не имеет прямых кредов ни к одному

Как мы это делаем

1. Базовая ревизия спецификации — 2025-06-18 и выше: structuredContent с outputSchema, сервер как OAuth Resource Server с метаданными по RFC 9728 и проверкой audience…
2. Каждый инструмент отдаём в двух вариантах: query_* (read-only) и act_* (меняет состояние); по умолчанию модель видит только read-only набор
3. В act_isolate_host первой строкой lookup_asset_tier(ip) по инвентарю в Qdrant; tier 0 → возврат pending_hitl с тикетом, а не выполнение
4. Креды Wazuh API и сервисной учётки AD живут в окружении MCP-сервера: LLM передаёт только имя инструмента и аргументы
5. Все вызовы act_* пишем в отдельный audit-лог с thread_id, чтобы постфактум разбирать цепочку рассуждение → действие

РЕЗУЛЬТАТ

Модель перестаёт быть субъектом действий: она формирует запрос, а решение о выполнении принимает код с явной политикой. Ошибка рассуждения или промпт-инъекция в худшем случае дают бессмысленный тикет, а не Disable-ADAccount на боевой учётке.

КЛЮЧЕВОЙ НЮАНС

Это класс LLM06 Excessive Agency: лечится не промптом, а сужением набора инструментов и прав. Права режем на стороне инструмента, а не в системном промпте — промпт переписывается входными данными, ACL инструме…

РЕАЛИЗАЦИЯ 2 ИЗ 3

HITL-контур на LangGraph: interrupt, checkpointer, Telegram

Что настраиваем

Оркестратор на LangGraph: граф триажа с точкой прерывания перед любым act_*-вызовом, состояние в Postgres-чекпойнтере

Как мы это делаем

1. Граф компилируем с checkpointer=PostgresSaver (пакет langgraph-checkpoint-postgres, .setup() на первом запуске): без чекпойнтера interrupt() не восстановит состояние
2. Перед узлом действия вызываем interrupt() с payload: цель, обоснование, id исходного алерта Wazuh и уровень уверенности
3. thread_id тикета = ключ возобновления; бот шлёт в Telegram inline-кнопки Разрешить/Отклонить, ответ возвращается через Command(resume=...)
4. Нет ответа за 900 с — узел завершается отказом, событие уходит аналитику обычным тикетом, автодействие не выполняется
5. Для Tier-2 (тестовые VM, dev) разрешаем act_* без прерывания, но с обязательной записью в audit-лог и уведомлением в дежурный чат

РЕЗУЛЬТАТ

Пауза агента становится штатным состоянием, а не аварией: процесс переживает перезапуск сервиса и возобновляется с того же шага. Дежурный видит не сырой лог, а готовое решение с обоснованием и жмёт одну кнопку — время до вердикта уходит в минуты.

КЛЮЧЕВОЙ НЮАНС

После resume узел стартует не со строки interrupt(), а с начала — весь код до прерывания выполняется повторно. Поэтому создание тикета и любые записи наружу делаем идемпотентными по thread_id, иначе тикет задв…

РЕАЛИЗАЦИЯ 3 ИЗ 3

Локальный инференс и RAG: vLLM + Qdrant внутри периметра

Что настраиваем

GPU-нода с одной картой 24 ГБ под vLLM, рядом Qdrant с политиками ИБ, инвентарём активов и ATT&CK; выхода в интернет у ноды нет

Как мы это делаем

1. Qwen3-32B в AWQ-4bit через vllm serve: --quantization awq, --max-model-len 16384, --gpu-memory-utilization 0.90 при дефолте 0.92
2. KV-cache считаем до запуска: 64 слоя × 8 KV-голов × 128 = 0.25 МБ на токен в fp16, --kv-cache-dtype fp8 режет вдвое; нативные 32768 берём только на карте от 32 ГБ
3. YaRN до 131072 не включаем: реализация статическая и штрафует короткие промпты, а алерт редко длиннее пары тысяч токенов
4. В Qdrant кладём инвентарь с полем tier, CIDR офиса, внутренние политики и enterprise-attack-18.1.json; коллекция со scalar quantization int8, HNSW — дефолтные m=16…
5. На массовой заливке ставим indexing_threshold=0 и после загрузки возвращаем 20000, иначе HNSW перестраивается на каждом батче

РЕЗУЛЬТАТ

Ни один фрагмент лога не покидает периметр: 152-ФЗ закрыт архитектурно, а не регламентом. Скорости одиночного запроса хватает на поток в несколько сотен событий в сутки, а стоимость тысячи алертов сводится к электричеству и амортизации карты.

КЛЮЧЕВОЙ НЮАНС

Дефолтные 0.92 выглядят безобидно, пока на той же карте не появляется Qdrant, а рядом с весами — KV-cache на 16 тысяч токенов. Память считаем до запуска: это дешевле, чем ловить OOM на боевом потоке алертов.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **Агент с правами Domain Admins**
  Сервисной учётке хватает чтения Wazuh API и security-логов. Запись в AD выносим в отдельный аккаунт под HITL, живущий вне Tier-0.

- ✕ **Сырые логи в облачный LLM-API**
  В события попадают ПДн и токены. Либо локальная модель, либо жёсткая маскировка полей до отправки; второй путь клиентам не рекомендуем.

- ✕ **Промпт-инъекция из тела письма**
  Текст лога — данные, а не инструкция. Оборачиваем в <user_data>, вычищаем конструкции вида «ignore previous», «system:», псевдо-теги и сверяем действ…

- ✕ **interrupt() без чекпойнтера**
  Граф не переживает рестарт и теряет незакрытые HITL-тикеты. Ставим PostgresSaver сразу, на пилоте тоже — иначе переделывать придётся под нагрузкой.

- ✕ **Инвентарь активов без поля tier**
  Без tier политика HITL не работает: агент считает DC обычным хостом. Заполненный инвентарь — условие до подключения первого act_*-инструмента.

- ✕ **Слепая вера в один прогон модели**
  Один семпл при temperature > 0 даёт уверенный, но случайный вывод. Спорные алерты гоняем 3-5 раз с разными seed и уходим в abstain при разногласии.

- ✕ **Автоответ Wazuh поверх агента**
  firewall-drop срабатывает сам и путает разбор. Оставляем его на явный брутфорс (rules_id 5712, timeout 3600 при timeout_allowed yes), остальное ведём…

- ✕ **Нет метрик после запуска**
  Без precision на размеченной выборке агент незаметно деградирует. Размечаем 200 алертов в месяц, смотрим precision, recall, долю abstain и число HITL…

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Только чтение: агент читает Wazuh API и пишет резюме, никаких act_*-инструментов
- Инвентарь активов с заполненным полем tier готов до старта пилота
- Модель локальная, фрагменты логов не уходят за периметр компании
- Все ответы агента в один канал, решение по действию принимает человек

НОРМАЛЬНО

- MCP-сервер: read-only и изменяющие инструменты разделены, креды только у сервера
- LangGraph с PostgresSaver и interrupt() перед каждым act_*-вызовом
- RAG в Qdrant: политики, CIDR офиса, инвентарь, enterprise-attack-18.1.json
- Санитайзер входа и обёртка логов в <user_data> отдельно от system-промпта

ХОРОШО

- Автодействия только для Tier-2, Tier-0 и Tier-1 всегда через HITL
- Self-consistency 3-5 прогонов на спорных алертах, abstain при расхождении
- Audit-лог связки рассуждение → инструмент → результат по thread_id
- Ежемесячная разметка 200 алертов: precision, recall, доля abstain

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Есть ли инвентарь активов с полем tier, и попали ли туда все DC, PKI, гипервизоры и сервер бэкапов?
- [ ] Может ли агент выполнить действие, меняющее конфигурацию, без подтверждения человека?
- [ ] Разделены ли у MCP-сервера read-only и изменяющие инструменты по разным наборам прав?
- [ ] Видит ли LLM креды Wazuh API и сервисной учётки AD или они остаются на стороне инструмента?
- [ ] Пережил ли HITL-контур тестовый рестарт сервиса без потери незакрытых тикетов?
- [ ] Идемпотентен ли код до interrupt(): не задвоится ли тикет после возобновления графа?
- [ ] Уходят ли фрагменты логов во внешний API, и если да — маскируются ли ПДн до отправки?
- [ ] Обёрнуты ли данные логов в отдельный тег и отделены ли они от system-инструкций?
- [ ] Есть ли у агента режим abstain и что происходит с событием, когда он срабатывает?
- [ ] Снимаете ли вы precision и recall на размеченной выборке хотя бы раз в месяц?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Аудит потока алертов: считаем объём, долю шума и точки, где триаж реально экономит время
- Разворачиваем локальный инференс на вашем GPU: vLLM, квантизация, расчёт KV-cache, интеграция с Wazuh API
- Собираем MCP-сервер с разделением read-only и act-инструментов и политикой Tier-0
- Ставим HITL-контур на LangGraph с подтверждением в Telegram и аудит-логом решений
- Ведём три фазы внедрения: ассистент, полуавтомат, автомат только для Tier-2

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 Wazuh — Active response, ossec.conf (commands, active-response), Server API (documentation.wazuh.com — 4.14.7)

02 vLLM — Engine arguments, Quantization (AWQ), Optimization and tuning (docs.vllm.ai — 2026)

03 LangGraph — Interrupts, Persistence, PostgresSaver (docs.langchain.com — 2026)

04 MITRE ATT&CK Enterprise — Detection Strategies, Analytics, Log Sources (attack.mitre.org — v18.1)

05 Qdrant — Indexing (HNSW), Optimizer, Quantization (qdrant.tech — 2026)

06 Model Context Protocol — Specification, Authorization, Tools (modelcontextprotocol.io — 2026-07)

07 Qwen3-32B — карточка модели: контекст, YaRN, rope_scaling (huggingface.co — 2026)

08 OWASP Top 10 for LLM Applications — LLM01, LLM06 (genai.owasp.org — 2025)

Основано на официальной документации продуктов и нашей практике внедрения.