



ML против хакеров

Процессинг сотен тысяч событий в секунду

SmartData 2026

Николай Лыфенко

руководитель ML-разработки · Positive Technologies

@nlyfen

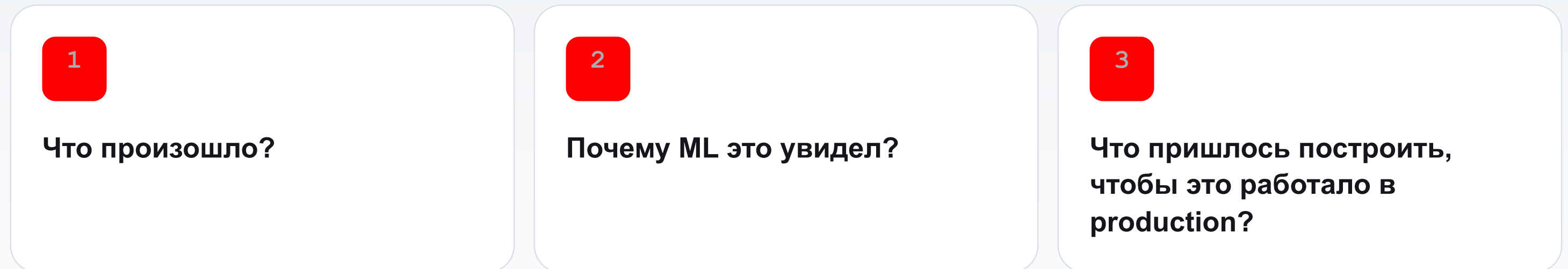
Николай Лыфенко



- Внедряю ML в продукты Positive Technologies
- Опыт в ИБ + МЛ домене более 10 лет
- Люблю MLOps



Проследим путь одной атаки



Сначала покажу атаку. Потом возьмём один подозрительный процесс и проведём его через инфраструктуру: на каждом этапе — не только решение, но и проблема, из-за которой оно появилось.

Кто такие хакеры



Атаки бывают разные

01

Показательный взлом

Дефейс. Популярно в 90-е и 00-е, сейчас встречается всё реже. Фокус сместился в сторону обогащения. Обычно ограничиваются взломом веб-сайта.

02

Финансовая выгода

Взломы финансовых организаций или шифрование инфраструктуры с требованием выкупа. Для достижения результата достаточно попасть на компьютеры сотрудников.

03

Нанесение ущерба

Ранее встречалось редко. Цель — максимальный урон компании. Играть роль специфические мотивы. Нужен привилегированный доступ во внутренние системы.

Киберполигон Standoff



Эта атака происходила на кибериспытаниях Standoff — исследователи легально проводят атаки, чтобы без риска для организаций проверить киберустойчивость.

110+

компаний уже вышли на кибериспытания

2500+

исследователей проверяют киберустойчивость организаций

60М ₺

максимум за недопустимое событие — плата за результат, а не за усилия

standoff365.com

Кибериспытания

Проверка киберустойчивости...

Проверить

Bug Bounty

Поиск уязвимостей в ИТ-системах

Найти

Кибербитва

Крупнейшее соревнование для специалистов по ИБ

Участвовать

Hackbase

Практика для белых хакеров

Практиковаться

Defend

Полигон для отработки тактик защиты бизнеса

Практиковаться

Киберучения

Аудит уровня зрелости команды ИБ

Проверить

17 июня, 11:07



powershell → SharpWsus.exe

запуск из powershell на сервере обновлений

uupd — сервер WSUS

Windows Server Update Services: качает патчи у Microsoft и раздаёт их на все компьютеры компании

фейковое обновление «KB5099999»

доставляется рабочим станциям штатно

раньше не наблюдался

ни на одном хосте инфраструктуры

SharpWsus.exe

инструмент, которого защитные системы не знали

ЧТО ТАКОЕ SHARPWSUS

Опенсорс-тулза (Nettitude, порт WSUSPendy) для атаки через WSUS: скомпрометировав сервер обновлений, атакующий создаёт фейковое «обновление» и «апрувит» его — рабочие станции сами скачивают и запускают payload при следующем чек-ине.

Хакер уже внутри. Он запускает на WSUS-сервере инструмент, которого защитные системы до этого не знали.

За следующие 25 часов атака развивается дальше



Время	Хост	Событие
17.06 11:07	uupd	SharpWsus.exe — фейк-обновление KB5099999
17.06 11:13	bjones	доставка m.exe («обновление» пришло)
17.06 13:31	bjones	mimikatz — выгрузка паролей и учётных данных из памяти
17.06 14:08	wsus	SharpWsus.exe — повторный запуск
18.06 11:53	ws11	CV_07.doc → rm.exe (фишинг)
18.06 12:40	ws11	туннель chisel к атакующим — утилита скрытых сетевых туннелей

Где защита впервые поняла, что происходит атака?

SharpWsus — это уже развитие атаки внутри инфраструктуры



Проникновение

хакер оказался внутри сети

Развитие атаки

разведка, привилегии, перемещение — здесь наш кейс

Ущерб

кража данных, разрушение систем

разведка

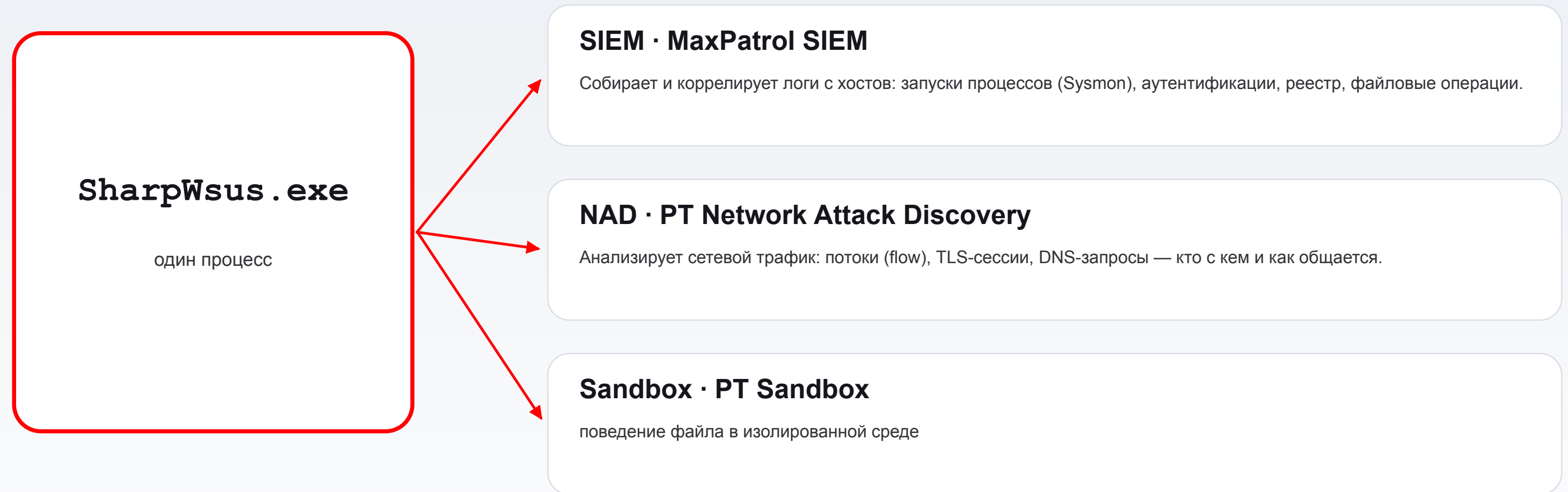
повышение привилегий

lateral movement

ключевые системы

Атакующий уже внутри — одного анализа периметра недостаточно. Нужно искать отклонения в поведении процессов внутри инфраструктуры.

Одна атака оставляет следы в нескольких источниках



Ни один источник сам по себе не знает всю историю.

Та же атака глазами телеметрии



Время	Где	Что произошло	Источник
17.06 11:07	uuprd	SharpWsus.exe — фейк-обновление KB5099999	SIEM
17.06 11:13	bjones	доставка m.exe («обновление» пришло)	SIEM + NAD
17.06 13:31	bjones	mimikatz — выгрузка паролей и учётных данных из памяти	SIEM
17.06 14:08	wsus	SharpWsus.exe — повторный запуск	SIEM
18.06 11:53	ws11	CV_07.doc → rm.exe (фишинг)	SIEM
18.06 12:40	ws11	туннель chisel к атакующим — утилита скрытых сетевых туннелей	SIEM + NAD

Увидеть событие ≠ понять, что это атака.

Известные инструменты правила увидели. SharpWsus — нет



Событие	Правила корреляции	ML
SharpWsus.exe (оба запуска, всего 55)	—	0.845
mimikatz — известный хакерский инструмент	сработали	сработала
chisel — известный туннель	сработали	сработала

Правила хорошо ловят то, что мы уже умеем описать. ML добавляет покрытие для неизвестного поведения.

Откуда взялся 0.845?

Что система узнала про SharpWsus.exe, чего не было в исходном событии?

Дальше — техническая часть: как несколько строк сырой телеметрии превратились в ML-признаки и затем в score.

А теперь таких событий — сотни тысяч каждую секунду



184 268

EPS

суммарный средний поток событий

23 166

ХОСТОВ

уникальных хостов за неделю

650+

ГБ/сутки

SIEM-телеметрии в сжатом виде

Окно 10 дней: от 104 млн до 5,1 млрд строк в день на клиента — разброс объёмов в десятки раз. Одним pandas-скриптом здесь не обойтись.

Один pipeline должен работать сразу для десятков инфраструктур



Сегодня

27

клиентов

хотим больше

100

- данные клиентов изолированы

- конфигурация клиента $\neq$ новый код

- тяжёлая аналитика не должна ломать ingestion

- новый клиент подключается автоматически

Значит, нам нужен не просто ML-классификатор. Нужна data/ML-платформа.

Классический стек



Airflow
оркестрация DAG-задач



Trino
SQL по Iceberg



Iceberg
табличный формат



Spark
feature matrix



MLflow
модели и метрики



Docker
упаковка пайплайнов



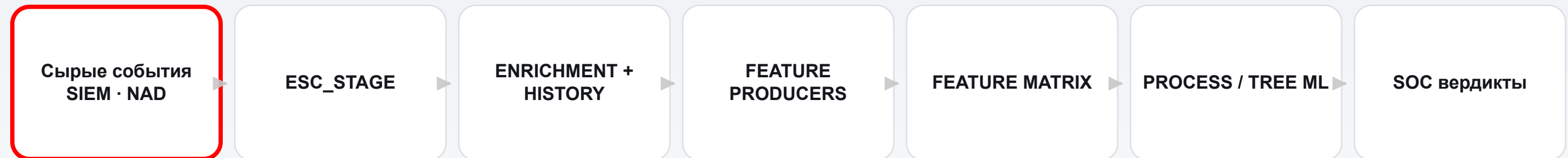
Python
язык пайплайнов

Здесь нет волшебной специализированной ML-платформы. Важнее то, как компоненты собраны вместе.

Путь SharpWsus внутри системы

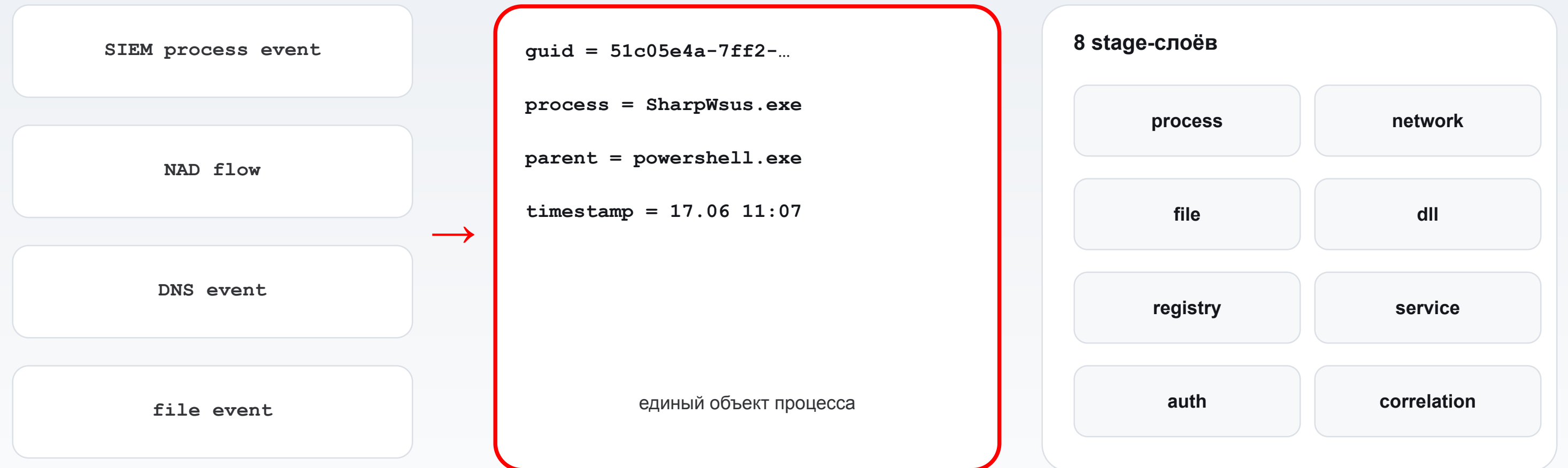


● SharpWsus.exe входит здесь



Вот весь путь — сейчас пройдем его слева направо. На каждом техническом слайде дальше в углу будет полоска этапов.

Из разных событий делаем единый объект процесса



Почему ingestion пришлось отделить от аналитики



Ingestion

100k EPS → быстрые записи

писать нужно всегда и сразу



Analytics

JOIN · history · feature generation

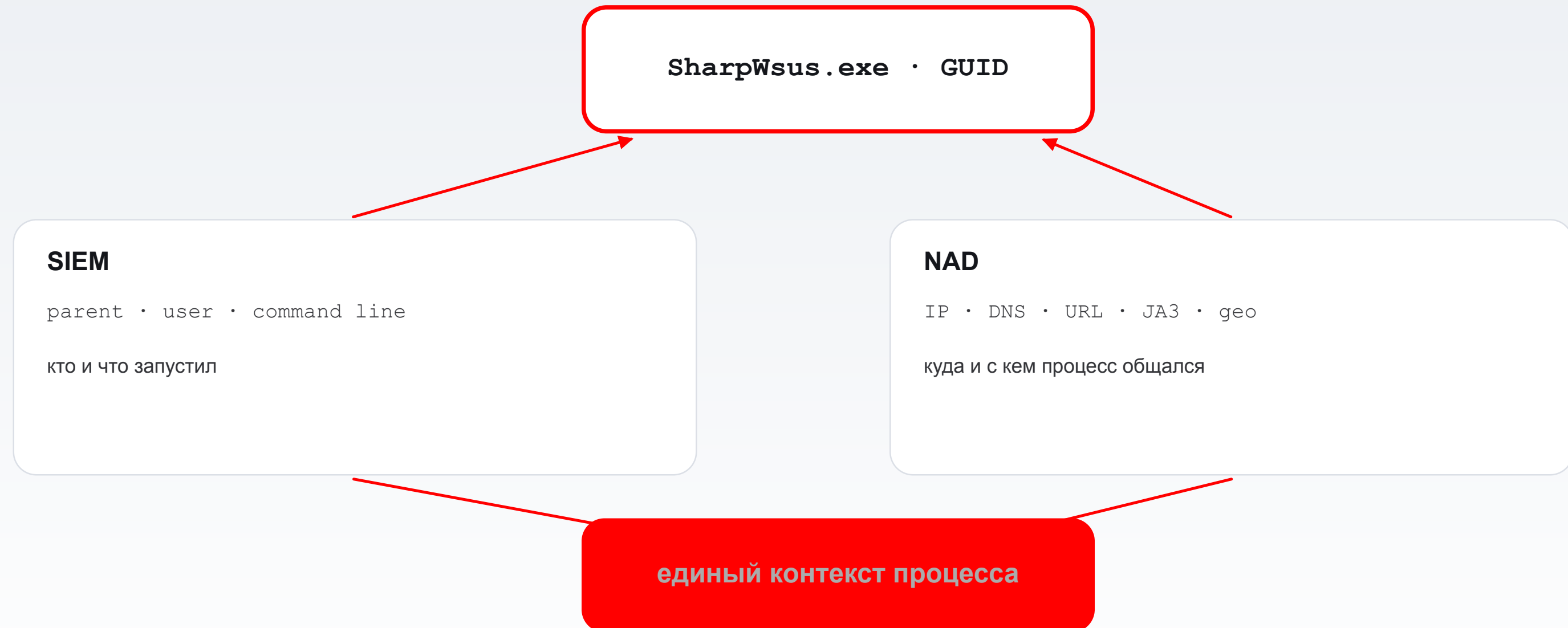
тяжёлые вычисления над историей

Ingestion ≠ Analytics

и event time ≠ ingestion time — опоздавшие события дописываются в прошлые партиции

Тяжёлый feature engineering не должен тормозить приём новых событий.

SIEM знает процесс. NAD знает, куда он ходил

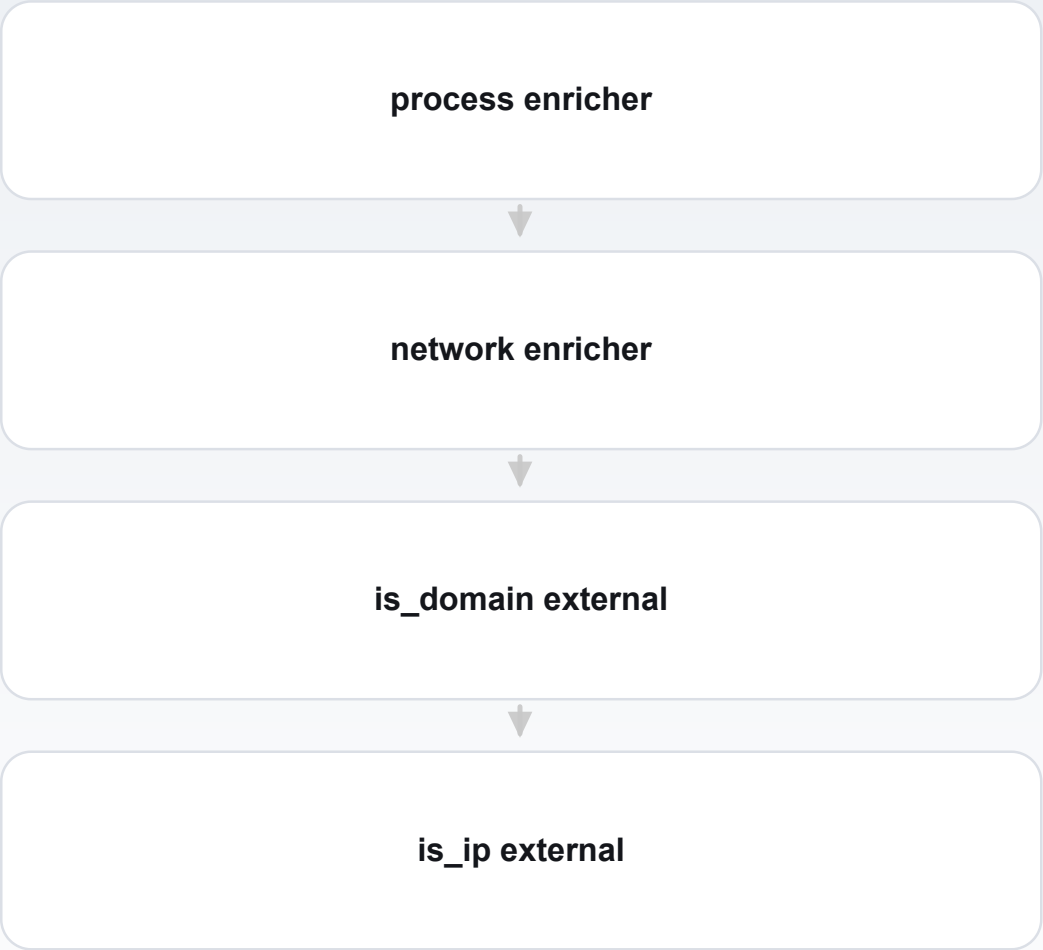


SharpWsus теперь — не просто имя процесса



SharpWsus.exe

parent	powershell.exe
command line	«C:\SharpWSUS.exe» create /payload:...
external IP in cmdline	да — 130.193.56.236
direct Internet connection	да — с сервера обновлений
DNS / IP context	внешний хост атакующих
встречался раньше?	нет — hosts: 0



Мы постепенно превращаем атомарное событие в описание поведения.

Не все признаки одинаково сложные



Если простой исторический счётчик уже говорит «такого никогда не было» — нет смысла сразу строить сложную модель.

Встречался ли этот объект раньше?



Видели процесс раньше?

count

На скольких хостах?

host_count

В скольких инфраструктурах?

infra_count

Сколько дней известен?

days_seen ·
days_known

SharpWsus.exe

```
count = 0 host_count = 0  
infra_count = 0 days_seen = 0
```

истории нет вообще

Ежедневный DAG сжимает историю в счётчики. История сама становится признаком.

А нормально ли именно такое сочетание?



`process → IP`

`process → DNS`

`process → JA3`

`process → пользователь`

`parent → child`

`SharpWsus → внешний IP`

пара никогда не встречалась

rare → high score

~16 KV-моделей · история `train_per_key` — отдельный DAG
каждые 8 часов

IP сам по себе может быть легитимным. Процесс сам по себе тоже. А вот конкретная пара может никогда раньше не встречаться.

Простые сигналы тоже становятся признаками



Флаг	Сигнал
UNKNOWN_PROCESS	редкое имя процесса в истории
EXTERNAL_IP_IN_CMDLINE	внешний IP прямо в командной строке
SUSPICIOUS_DIRECTORY	запуск из подозрительной директории
PARENT_CHILD_KV	новый child у известного parent
RARE_DNS	редкий registered domain

Всего 17 флагов — здесь наиболее интересные.

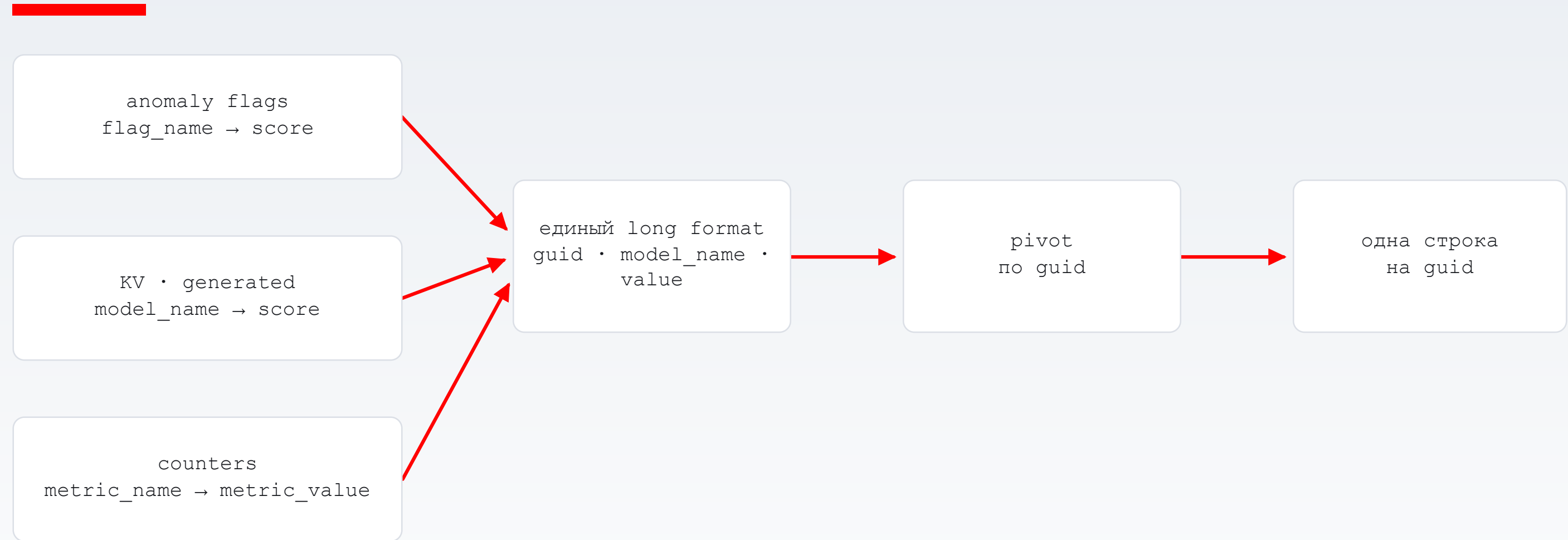
Что система знает о SharpWsus.exe перед инференсом



Фича	Значение	Что это значит
UNKNOWN_PROCESS	hosts: 0, runs: 0	процесс никогда не видели ни на одном хосте инфраструктуры
EXTERNAL_IP_IN_CMDLINE	130.193.56.236	внешний IP атакующих прямо в командной строке
INTERNET_CONNECTION	130.193.56.236	процесс ходил напрямую в интернет — с сервера обновлений
SUSPICIOUS_DIR	c:\	запуск из подозрительной директории (корень диска)
ATYPICAL_FILE_EXTENTION	exe	нетипичное размещение исполняемого файла

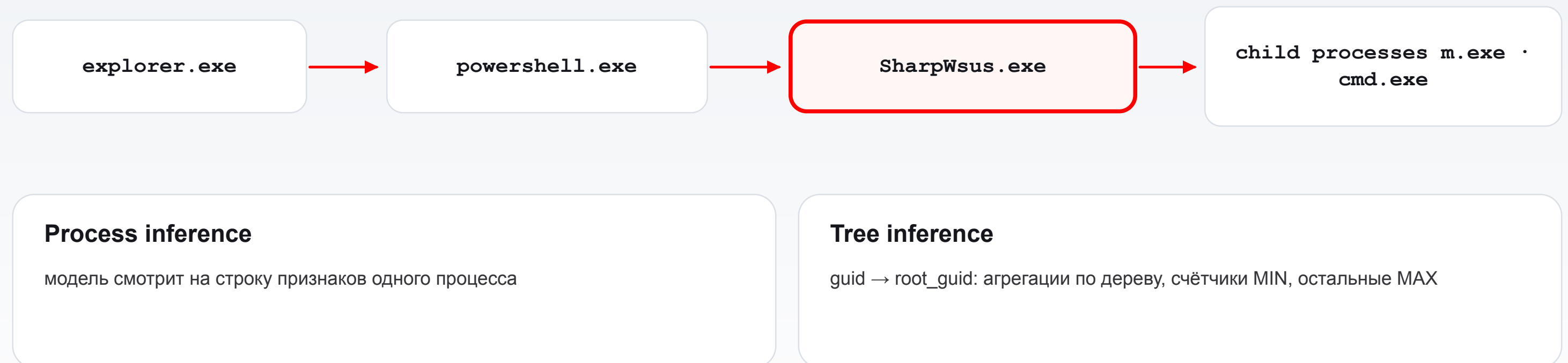
Ни один сигнал по отдельности не доказывает атаку. Совокупность — уже подозрительна.

Все producers должны договориться об одном формате



Единый ML-контракт. NULL ≠ 0: колонка может означать, что producer отсутствовал в этом run-е.

Один процесс может выглядеть нормально. Дерево — уже нет



Совокупность признаков → 0.845



0.845

X_LightGBM#42

порог срабатывания — выше 0.8

те самые признаки

UNKNOWN_PROCESS = 1

EXTERNAL_IP_IN_CMDLINE = 1

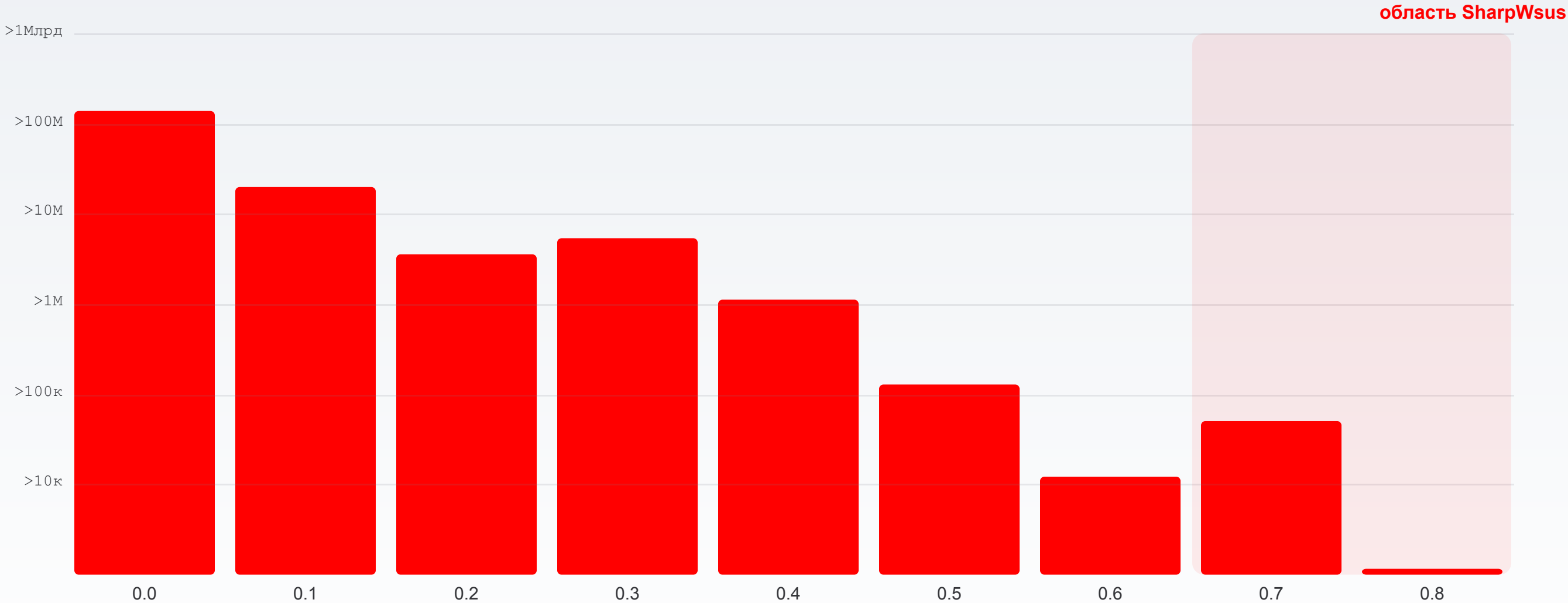
INTERNET_CONNECTION = 1

SUSPICIOUS_DIR = 1

process → IP: rare (high score)

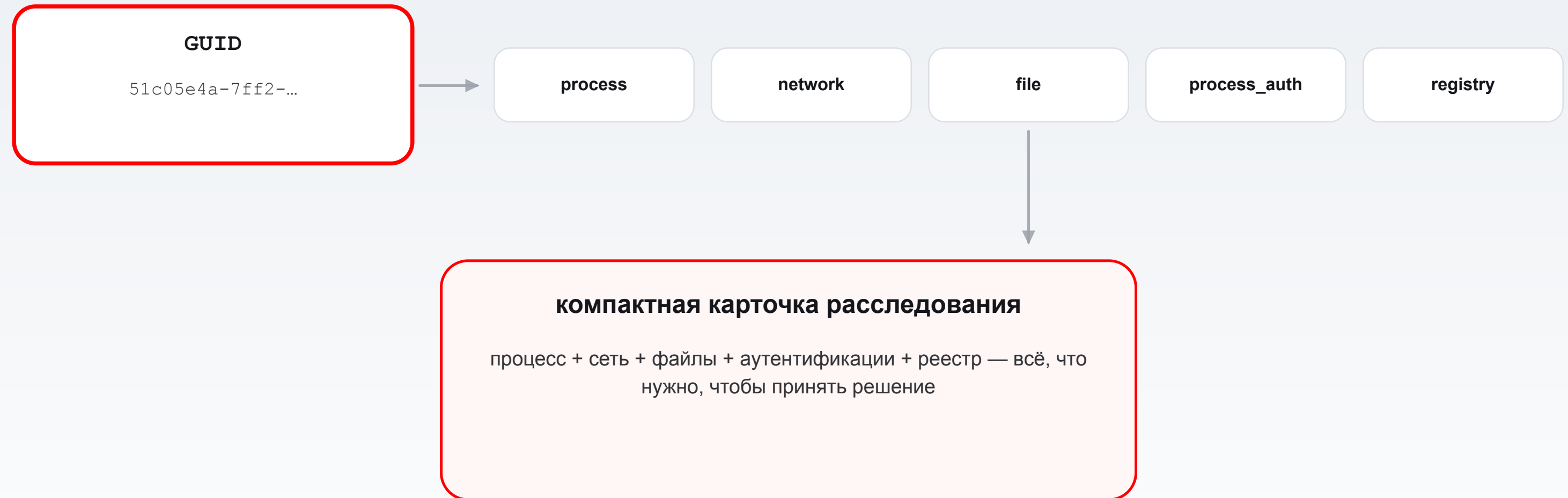
Вот откуда появился score, который мы увидели в начале.

Высокие score — редкость



Основная масса предсказаний — внизу распределения (~82% в бине 0.0–0.1). High-score объекты — небольшая выборка для дальнейшего разбора.

Score недостаточно — аналитику нужен контекст



Задача последнего этапа — не просто сказать «0.845», а вернуть человеку контекст для решения.

Для верхушки предсказаний — автоматический разбор контекста



Кандидаты

предсказания ML-скора ≥ 0.85 , топ-100 за сутки, один на GUID

Контекст

по каждому GUID: process · network · file · process_auth · registry

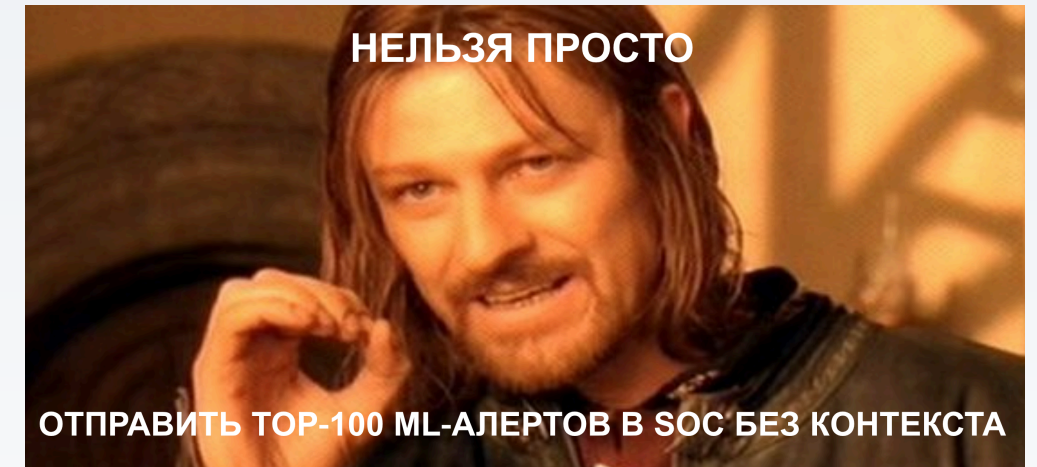
Разбор

самодеплойный LLM: вероятность вредоносности + объяснение (до 2 000 символов, temperature 0.1)

Вердикт

guid · malicious_probability · explanation; malicious при ≥ 0.5

Отдельный post-processing слой для high-score кандидатов — не обязательный этап пайплайна детекта.



Теперь мы знаем, что происходило внутри системы



Время	Событие	Реакция защиты
17.06 11:07	SharpWsus.exe на uupd	правила молчали · ML 0.845 — в ближайшем прогоне
17.06 11:13	вирус доставлен на bjones	правила частично · ML ~0.88
17.06 13:31	mimikatz	правила сработали · ML 0.88
17.06 14:08	SharpWsus.exe на wsus	правила молчали · ML 0.846 — в ближайшем прогоне
18.06 11:53	CV_07.doc → rm.exe	правила сработали · ML 0.84
18.06 12:40	туннель chisel	правила сработали · ML 0.89

Тот же таймлайн, что в начале — но теперь каждая строка заполнена: кто и когда поднял тревогу.

106 050

ML-срабатываний

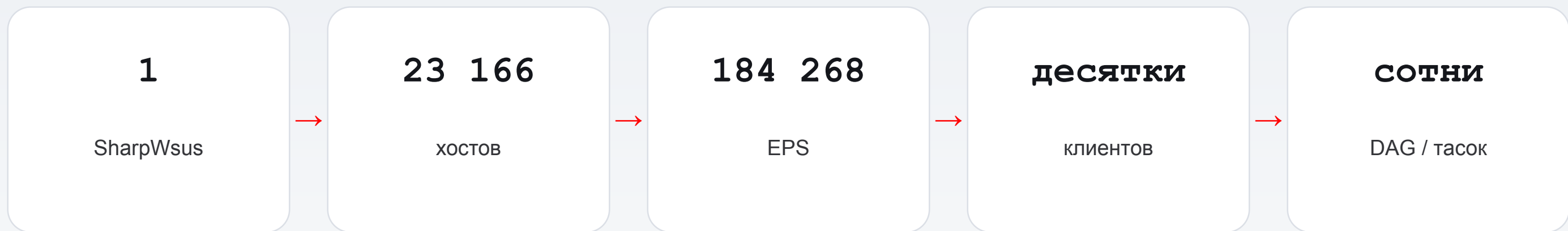
16 959 (16%)

на то, на что правила корреляции молчали

Окно 16–19 июня 2026 · ML-only: ML-сработка без сработавшего правила на тот же guid в окне ± 1 день (консервативно в пользу правил) — нижняя оценка прироста.

ML не заменяет правила. Он добавляет ещё один слой покрытия.

А ПОТОМ МЫ ВКЛЮЧИЛИ ЭТО ДЛЯ ДЕСЯТКОВ КЛИЕНТОВ



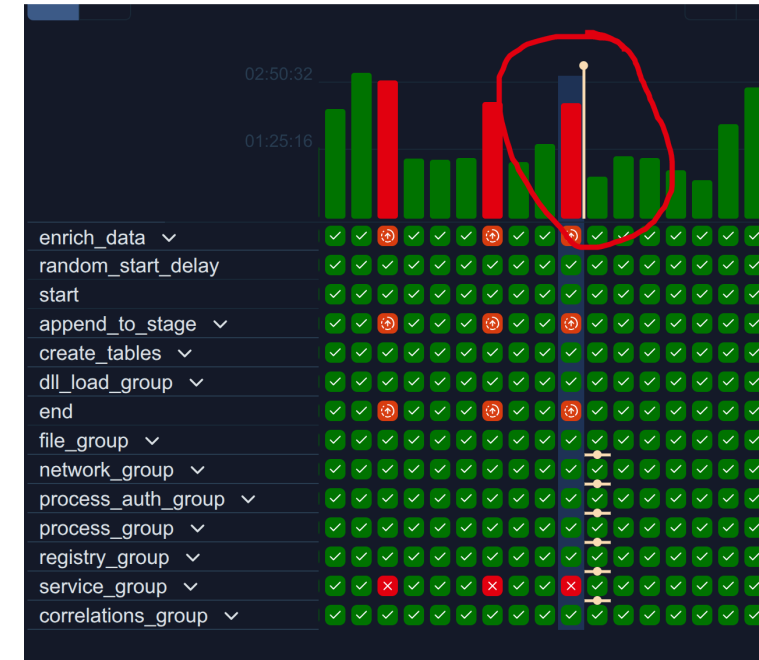
И здесь начались настоящие Data Engineering проблемы.

Урок 1: версия pipeline должна быть наблюдаема



Проблема

«Код изменился. Какая версия реально работает на стенде — непонятно.»



- в Ansible-плейбук деплоя DAG-а — версия пайплайна, которым он собран: видно, кто, из какой feature-ветки и когда опубликовал бандл

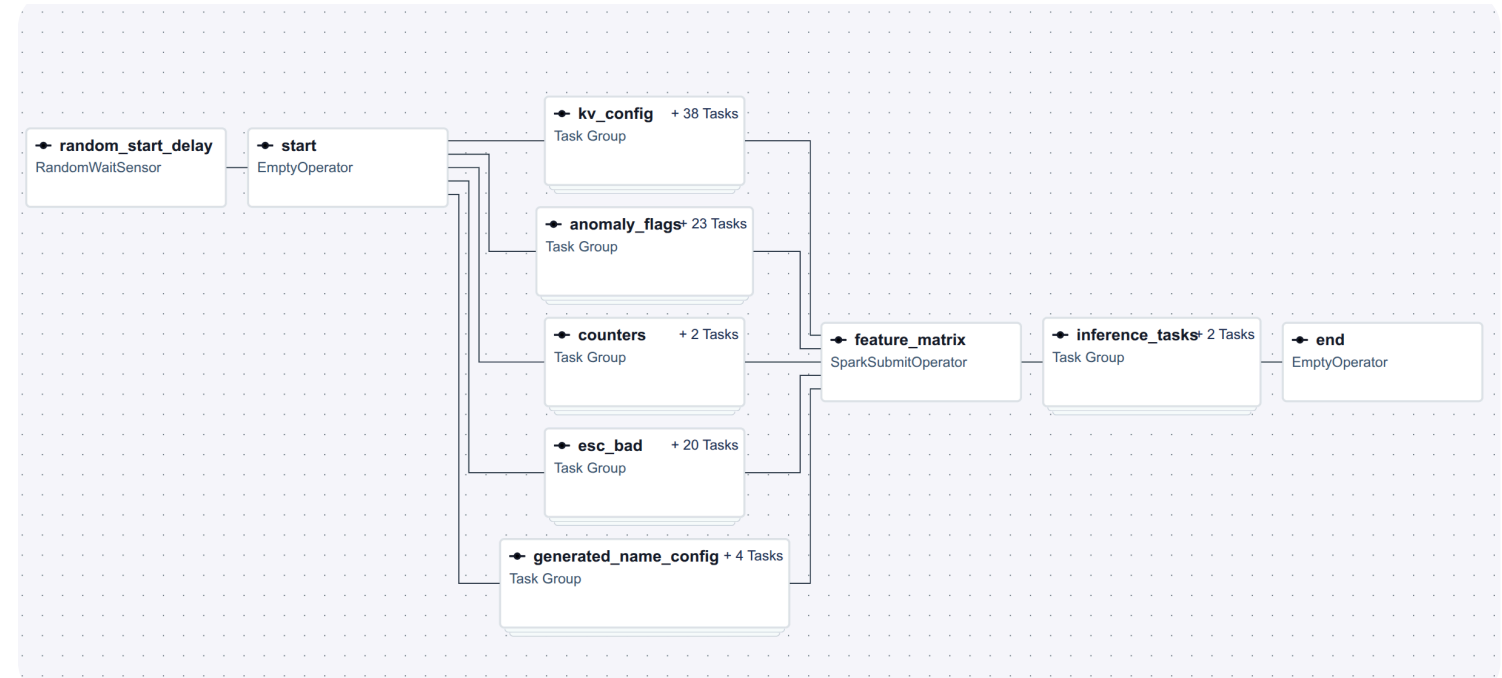
Артефакт, код и DAG должны иметь один идентификатор версии.

Урок 2: слишком много независимых DAG



Проблема

«DAG на каждую группу фичей: конфигурации расходятся между dev и prod.»



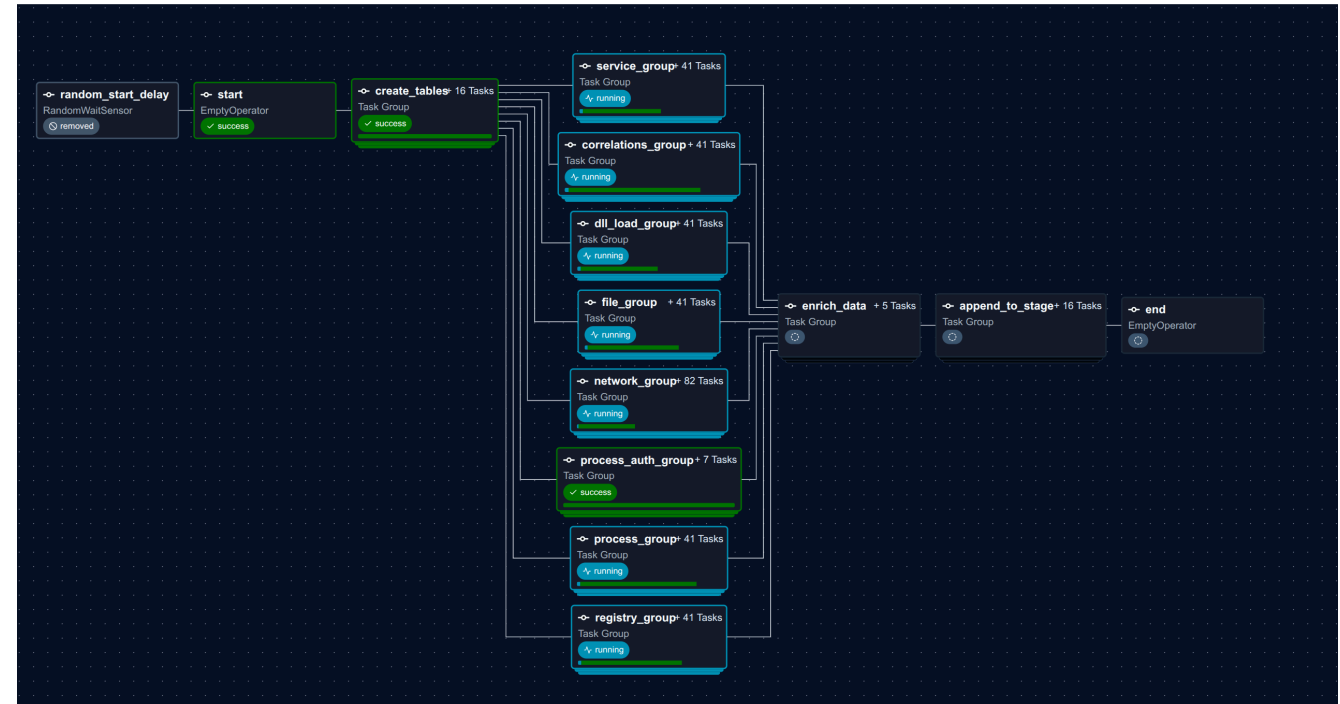
Для нашей структуры producers один DAG оказался проще сопровождать.

Урок 3: аккуратнее с concurrency



Проблема

```
ICEBERG_COMMIT_ERROR, message="Failed to commit  
the transaction during insert: Commit failed:  
Requirement failed: branch main has changed:  
expected id 1606132530018329058 !=  
2535512302574202991"
```



Taskгруппы параллелим, таски внутри — чейним; pool используем разумно.

Урок 4: DAU вырос



Проблема

«Пользователей всё больше, запросы тяжелеют, а одна учётка на все команды не позволяет понять, кто создаёт нагрузку.»



Нагрузку нужно не только масштабировать, но и атрибутировать.

Урок 5: distributed cron создаёт пики



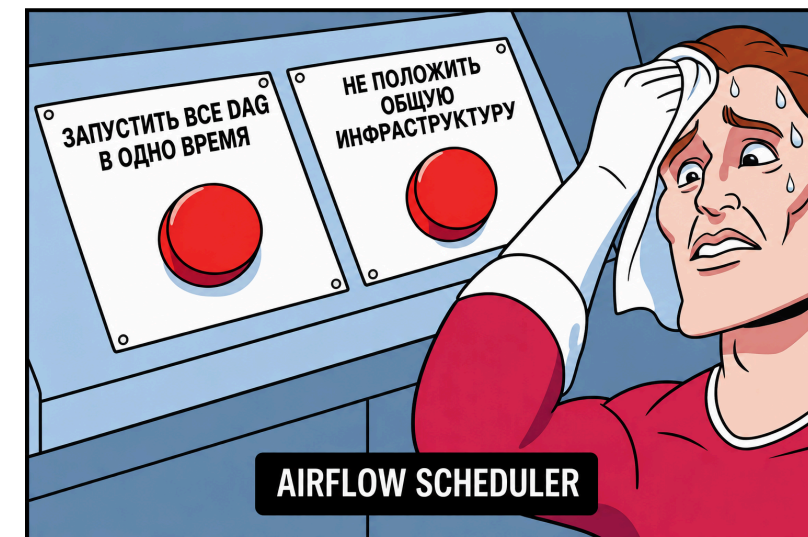
Проблема

«Если все DAG стартуют в 00 минут — это почти DDoS самого себя.»



Решение

RandomWaitSensor



деферируемый сенсор в начале каждого DAG: случайная задержка старта → джиттер запусков → сглаживание пиковой нагрузки на общую инфраструктуру

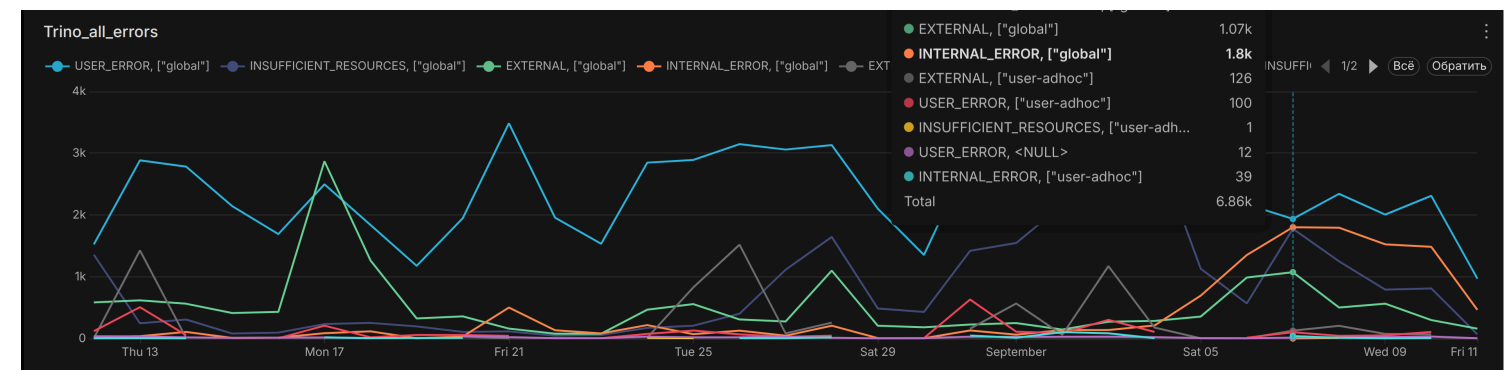
Расписания — тоже часть нагрузки: пики запусков нужно разносить.

Урок 6: на больших объёмах стоимость плохого JOIN заметна



Проблема

«Тяжёлые JOIN-ы и широкие окна тормозили отчётность: запросы висели часами.»



- балансир перед тринами — распределяет нагрузку между кластерами
- MCP для Trino: читает DDL, пишет и исполняет хорошие рабочие запросы

Плохой SQL масштабируется лучше хорошего — и дороже.

Урок 7: любой DAG должен безопасно перезапускаться



Проблема



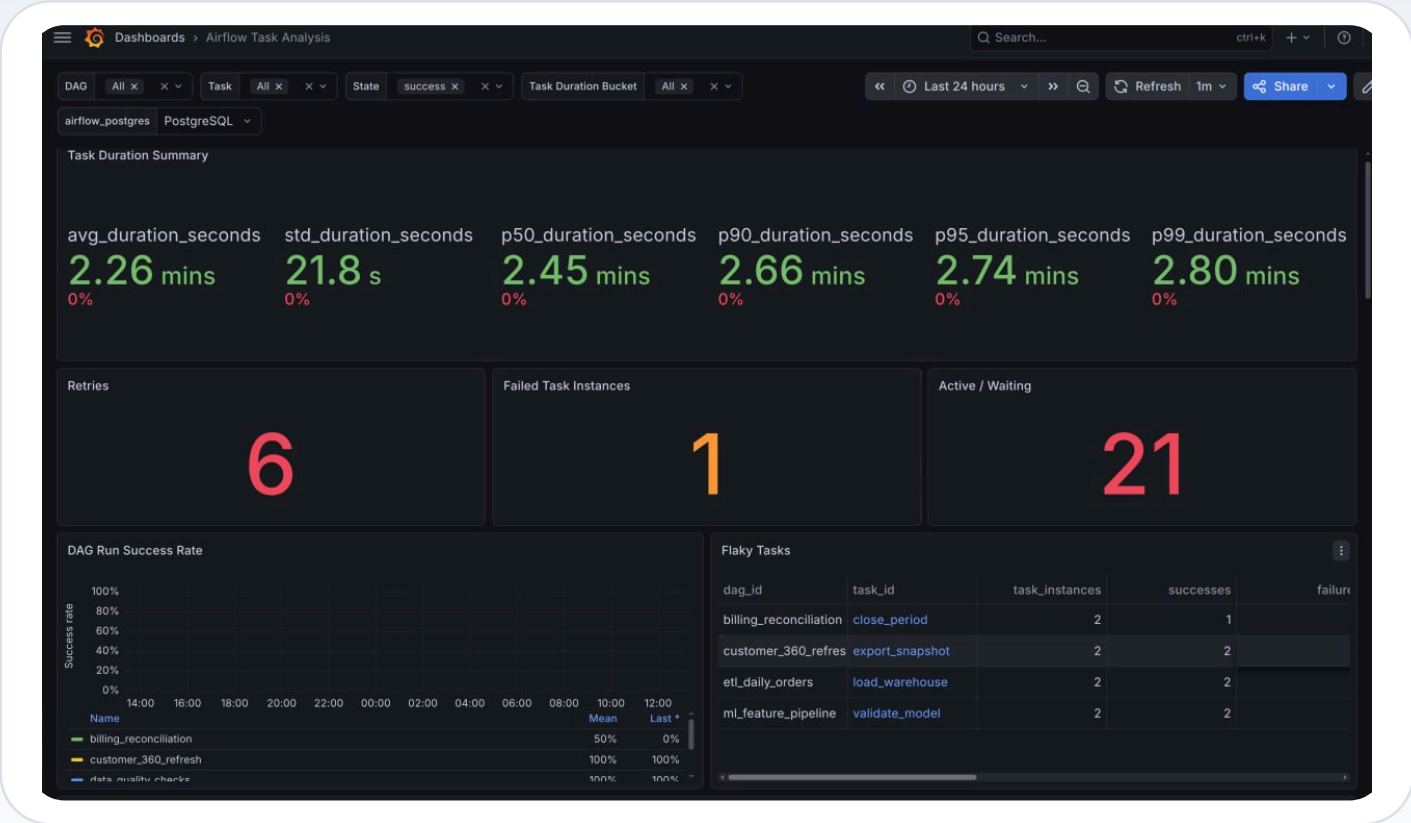
Решение

Code review rule: DAG must be idempotent

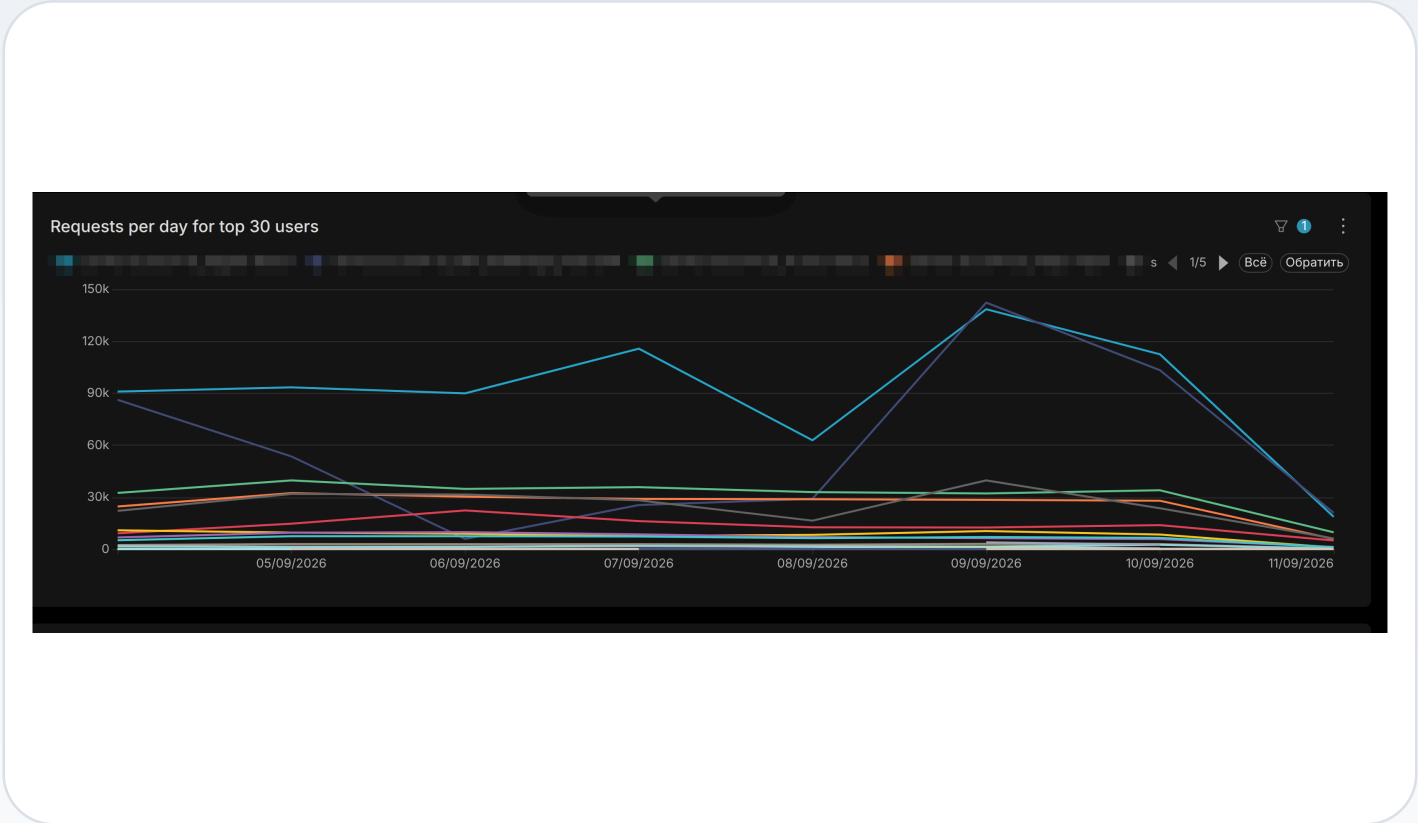
чек-лист идемпотентности обязателен на ревью каждого DAG:
повторный запуск не должен дублировать данные

Неидемпотентный pipeline однажды обязательно перезапустят.

Урок 8: если pipeline не наблюдаем — его нет



Airflow · Grafana



Trino · Superset

что падает?

что медленно?

кто создаёт нагрузку?

где растут ошибки?

Monitoring — часть ML-системы, а не инфраструктурный бонус.

Что на самом деле оказалось важным



- Правила и ML решают разные части задачи: правила сильны на известном, ML добавляет поиск необычного поведения
- Хороший ML начинается с контекста: process + network + history ценнее отдельного события
- Feature engineering — это distributed data system: основная сложность далеко не только в обучении модели
- Production-система определяется эксплуатацией: versioning, idempotency, resource isolation, observability — часть качества детекта

Спасибо за внимание!

Вопросы?



Николай Лыфенко • @nlyfen • SmartData 2026



Наш Telegram-канал

t.me/falseposi