OFFZONE 2026

ZEROFALSE ДЛЯ SAST: ЛОКАЛЬНЫЕ LLM, EVIDENCE-GATE И ТРИАЖ БЕЗ ГАЛЛЮЦИНАЦИЙ

сработки бывают двух типов — и доказываются по-разному

Юрий Туманов (Эльария) · AppSec · полгода исследования: 02–08.2026

OFFZONE 2026 · AppSec.Zone · 20–21 августа, Москва

ОБ АВТОРЕ

OFF ONE 2026

КТО

Юра Туманов
ака Эльария / Psycho Drake
«Ростелеком» · AppSec

ЭКСПЕРТИЗА

кибербез с 2001 года
кандидат технических наук @ НИЯУ «МИФИ»
кибербез + нейронки · пою

КОНТАКТЫ

@psychodrake
github.com/Eljees
3.14hell@gmail.com

ПОЧЕМУ ЭТА ТЕМА

Покупалась RTX 4080,
чтобы играть в киберпанк.

Служит — чтобы создавать
его в реальной жизни.

ДЕСЯТКИ ТЫСЯЧ ФОЛЗОВ В НЕДЕЛЮ

масштаб ручного разбора сработок SAST у нас в компании — оценки сверху

ДЕСЯТКИ ТЫСЯЧ

сработок SAST приходит в неделю в пиковые недели — это входящая очередь на разбор

95%

из них — фолзы: сканер кричит, а уязвимости нет. Но узнаёшь это, только разобрав сработку

< 10

человек в AppSec-команде на весь этот поток

70+ приложений · ~3 000 репозиториев · 150+ языков · в агрегаторе накоплено >1 млн сработок

Тысячи сработок на человека в неделю — меньше минуты на решение. Дальше в докладе — кому и по каким правилам отдать эту рутину, не теряя реальные уязвимости.

тезаурус: SAST — робот, читающий исходный код без запуска и кричащий на подозрительное · фолз — ложная тревога сканера · триаж — разбор: реальна ли проблема · агрегатор — система, где копятся сработки всех сканеров

ПЛАН ДОКЛАДА

семь остановок — от наивной гипотезы до цифр эксплуатации

01	Исходная гипотеза: «отдай сработку LLM — пусть решит», или как убедительный текст не оказался доказательством
02	Шесть поколений промптов (скилов) за полгода: от эссе к перечислению фактов
03	Архитектура ZeroFalse: модель предполагает — улики подтверждают — правила решают
04	Матмодель: классификация с правом сказать «Unknown — пусть смотрит человек»
05	Два типа сработок: property-based и trace-based — их нельзя обрабатывать одинаково
06	Доказательства вместо мнений: evidence-gate, алгоритмические проверки
07	Цифры: 2 546 решений людей · экзамен property-классов · экзамен изящных моделек · где споткнулись

СКЛАДНО ≠ ДОКАЗАНО

исходная гипотеза: один универсальный промпт, внутренняя большая LLM сама разберёт, что фолз, а что нет. Проверка: 100 сработок SQL-инъекций (PHP) через LLM-портал

РЕАЛЬНАЯ ВЫДАЧА, ДОСЛОВНО

```
вердикт: TP · confidence 1.00
ссылка: строка 32358
в выданном коде: 307 строк

«htmlspecialchars() нейтрализует
SQL-инъекцию» · conf 0.9
```

(htmlspecialchars экранирует HTML-символы — от SQL-инъекции она не защищает)

10 / 100 вердиктов ссылаются на строку кода, которой НЕТ в выданном фрагменте — модель «увидела» несуществующий код

9 / 10 из этих десяти закрывают сработку как реальную уязвимость с уверенностью 1.00

50% случаев модель противоречит сама себе, когда тот же фрагмент показывают повторно (17 из 34)

Текст безупречен — вердикт выдуман. Вопрос всего доклада: как не дать убедительности сойти за доказательство.

LLM-портал — Qwen2.5-72B: формат ответа не зафиксировать (ни JSON-схемы, ни guided decoding). Дальше в докладе — только локальные модели под полным контролем

ШЕСТЬ ПОКОЛЕНИЙ ПРОМПТОВ (СКИЛОВ) ЗА ПОЛГОДА

веса модели не трогали — меняли правила игры с ней — каждое поколение закрывало конкретный класс ошибок

P0 спросили текстом — получили эссе: складно, но непроверяемо

P1 жёсткая форма ответа (JSON): вердикты вместо сочинений

P2 правила под каждый класс дефектов — внутри зовём их скилами

P3 улики вместо целых файлов: модели передаётся только код, относящийся к делу

P4 система вправе отменить вердикт модели: алгоритмические проверки и пороги поверх ответа

P5 своя инструкция каждой модели парка — под её сильные и слабые места

P6 «перечисли факты — вердикт выводится сам»: evidence-gate (слайд 14)

СТЕНД: RTX 4080 16 ГБ (vLLM + llama.cpp, модели 1,5–24B) + два ноутбука; большие модели пробовали через портал · триаж запускался живьём на смартфоне · код не покидает периметр · 143 ночных цикла настройки промптов

тезаурус: скил — регулярно используемый промпт, заточенный под класс дефекта

ВЕРСИИ ПРОМПТА НА ОДНОМ КЛАССЕ — 798, ПАРОЛЬ В КОДЕ

```
v1  словари: ghp_/AKIA → Confirmed;
placeholder/example → фолз
v2  + секции Confirmed / фолз / Unknown
v3  + среда: боевой конфиг или тестовый
```

инженерная процедура спасает реальные — тот же корпус, 2 модели:

АРХИТЕКТУРА ZEROFALSE

принцип: модель предполагает — улики подтверждают — правила проверяют — последнее слово всегда за системой правил · название — оммаж ZeroFalse (arXiv:2510.02534, 2025)

РОУТЕР
по классу CWE: тип сработки известен заранее

→ **property-based** › лестница алгоритмических проверок (слайд 11)

→ **trace-based** › пакет улик + evidence-gate (слайды 13–14)

МНЕНИЕ МОДЕЛИ

вердикт · обоснование · confidence — само по себе не решает ничего

→

УЛИКИ

трасса · код вокруг sink · строки происхождения данных

→

ПРАВИЛА

fastpath (алгоритмическое решение без LLM) · 120+ правил · алгоритмические проверки · пороги по классам

→

РЕШЕНИЕ СИСТЕМЫ

закрыть как фолз · завести дефект · отдать человеку

СКВОЗНОЙ ЖУРНАЛ: каждое решение воспроизводимо из лога прогона — все цифры этого доклада посчитаны из него

как читать цвета: серое — мнение (не решает) · синее — проверяемые факты · жёлтое — алгоритмика · яркое — единственное, что уходит наружу

МАТМОДЕЛЬ: КЛАССИФИКАЦИЯ С ПРАВОМ НА UNKNOWN

стоим на плечах титанов — классическое распознавание образов конца 1950-х

РАСПОЗНАВАНИЕ ОБРАЗОВ

сработка с уликами = объект с признаками; раскладываем на классы «фолз» и «реальная».

цены ошибок несимметричны:
лишний фолз человеку — минуты;
похороненная реальная — дыра в проде

ПРАВО НА ОТКАЗ — ПРАВИЛО ЧОУ

классика 1957/1970 (reject option): если ошибки дорогие — разреши третий ответ, Unknown: «пусть смотрит человек». Суммарно это дешевле, чем заставлять угадывать.

покрытие = доля сработок с ответом;
риск = доля ошибок среди закрытых

ФАКТОРИЗАЦИЯ ЗАДАЧИ

вместо одного «разбери всё» — много маленьких задач:

(тип сработки — их два, следующий слайд) × (класс CWE)

у каждой — свой инструмент, свой замер и свой порог

ДВА ТИПА СРАБОТОК: PROPERTY- И TRACE-BASED

тип известен заранее — его определяет класс CWE, который присвоил сканер; обрабатывать их надо по-разному

PROPERTY-BASED · «УЯЗВИМОСТЬ-СВОЙСТВО»

опасен сам ФАКТ в коде:
секрет или пароль в коде (798 · 257 · 259)
передача без шифрования (319)
слабый хеш или случайность (328 · 330)
слишком открытые права (276 · 732)
пустой catch — проглоченная ошибка (396 · 397)

ключевой вопрос: настоящий пароль или заглушка?

TRACE-BASED · «УЯЗВИМОСТЬ-ПУТЬ»

опасен МАРШРУТ данных: source › sink
SQL-инъекция (89) · XSS (79)
инъекция кода (94)
обход каталогов (22)
SSRF (918) · десериализация (502)

ключевой вопрос: дошли ли данные атакующего до опасного вызова — и была ли по дороге защита?

словарик: source — место, куда атакующий может подать свои данные; sink — вызов, где эти данные становятся опасными (запрос к БД, вывод в HTML, команда ОС)

~74%

потока — property-based (замер: 100 сработок, 56 категорий; в главном замере на 2 546 картина та же). Большинство сработок вообще не про «путь атаки».

СПРОСИ НЕ ТАК — ПОТЕРЯЕШЬ 9 ИЗ 10

OFF ONE 2026

взяли 38 классов дефектов — по 10 реальных и 10 фолзов на каждый — и задали всем сработкам один и тот же вопрос: «покажи путь атаки». Вот что вышло

TRACE-КЛАССЫ — ПУТЬ СУЩЕСТВУЕТ

XSS · инъекции · редиректы · SSRF · десериализация...

похоронено реальных: **0**

вопрос подошёл: у пути есть трасса, её можно проверить; все фолзы отсеяны верно

PROPERTY-КЛАССЫ — ПУТИ НЕТ

пароли · шифрование · случайность · права...

похоронено реальных: **7–9 из 10**

вопрос чужой: модель ищет путь, не находит — и решает «значит, не уязвимость». А путь здесь и не должен существовать

140 / 363

реальных уязвимостей похоронил один неправильный вопрос. Хуже всего пришлось самым простым и массовым классам — паролям и секретам в коде

«похоронить» = закрыть реальную уязвимость как фолз: сработка исчезает из очереди навсегда, дыра остаётся в проде

ФОЛБЕКИ ДЛЯ PROPERTY-BASED: ЛЕСТНИЦА ПРОВЕРОК

у «факта в коде» всегда есть ступень проверки, не зависящая от настроения нейронки; сработка спускается по лестнице, пока какая-то ступень не даст надёжный ответ

0 FASTPATH — алгоритмическое решение, LLM не зовём: ключ формата AKIA (боевой AWS) › Confirmed; файл generated/SBOM › фолз. Снимает ~68% шума класса, экономит ~6 000 токенов и ~4 с на сработку

1 LLM С PROPERTY-ПРОМПТОМ — вопрос по сути факта: «настоящий секрет или заглушка?»

2 120+ ДЕТЕРМИНИРОВАННЫХ ПРАВИЛ ПОСЛЕ LLM — структурные проверки ответа, могут поправить вердикт

3 ПОРОГИ ПО КЛАССАМ — слабый ответ класса превращается в Unknown

4 ЧЕЛОВЕК — Unknown уходит ревьюеру вместе с уликами; в боевую систему автомат в этом случае не пишет ничего

ПРАВИЛО

для property-сработки «не вижу пути атаки» — запрещённая причина закрытия;
такое закрытие отменяется правилами (поколение P4)

«с property хорошо», потому что под нейронкой — лестница алгоритмических проверок

МОДЕЛИ: ОТ 1,5 ДО 24 МИЛЛИАРДОВ ПАРАМЕТРОВ

весь парк — локальный, в 16 ГБ видеопамяти одной RTX 4080. B — размер модели в миллиардах параметров; q4 и AWQ — сжатие модели (квантование), чтобы она влезала в память

Qwen2.5-Coder-1.5B (q4)	единственная запускается на смартфоне, не превращая его в кирпич: живой триаж на Galaxy S25+ — вердикты совпали с видеокартой 37/39 · медиана 26,6 с против 3,5 с
Qwen2.5-Coder-3B (q4)	эксперименты и дешёвые классы; после автонастройки промптов точность на классе 732 (открытые права) выросла с 0,667 до 0,857 · главный экзамен изящных моделек — впереди
Qwen2.5-Coder-7B (q4)	массовые прогоны; самооценке не верим: за неделю 130 ответов с уверенностью 0.0 и 154 с 1.0 › в прод только с проверками поверх; подтверждать trace-уязвимости ей запрещено
Qwen2.5-Coder-14B (AWQ)	основная модель главного замера · 4,5 секунды на решение
Foundation-Sec-8B	дообучена под безопасность; самоуверенная фифа: просишь JSON — а она тебе поэму. Битый JSON 26%; подняли лимит длины ответа › 4,8%
Devstral-Small-2 (24B)	самая точная из локальных: на SQL-инъекциях 93,8% верных вердиктов против 76,2% у 14B и 72,2% у FSec-8B
Qwen2.5-72B-Instruct	антипример, в парк не входит: доступ через LLM-портал — формат ответа не зафиксировать › уверенные фантазии (слайд 5)

обе сильные модели хоронят ОДИН И ТОТ ЖЕ набор находок (совпадение 6/6 и 5/6) — смена модели не лечит системную ошибку; нарезка данных и процедура важнее размера модели

тезаурус: точность — доля верных вердиктов на контрольном наборе класса

TRACE-BASED: КАК ПОДАТЬ МОДЕЛИ КОНТЕКСТ

живой замер: 100 сработок (50 реальных + 50 фолзов), одна модель 14B — меняется только способ подать код. Линейка одна для всех строк: сравнение с боевыми вердиктами ревьюеров · цена — сколько контекста и вызовов съедает способ

способ подать код	покрытие	верно из решённых	похоронено	цена
full file — файл целиком	36%	92%	3	~7 600 токенов
distilled — дистиллят: только строки трассы	39%	92%	3	~3 400 (−55%)
no-Unknown — «решай всё», без права отказа	95%	56%	40	минимум контекста
staged — пошаговый разбор	79%	73%	17	3+ вызова модели
cascade — каскад: distilled, не уверен › staged	84%	77%	16	~3 900 · баланс

экстремум: сработка из запутанного (обфусцированного) кода тянула 418 000 токенов (токен ≈ 3–4 символа кода) › после дистилляции 1 200

Полный вход ≠ лучший вход. Решительность покупается опасными ошибками — 40 похороненных у «решай всё». Каскад — рабочий баланс; его 16 опасных дальше добивает evidence-gate (следующий слайд).

EVIDENCE-GATE: «ПЕРЕЧИСЛИ ФАКТЫ ДО ВЕРДИКТА»

поколение P6: модель не имеет права сразу на вердикт — сначала обязана перечислить факты; вердикт вычисляется из таблицы фактов механически (пример — SQL-инъекции)

ШАГ 1 найди настоящий sink — строку, где реально выполняется опасный вызов. Сканер её часто теряет: в его описании строка пустая или комментарий (у нас 0 из 200 сработок доводили sink сами)

ШАГ 2 перечисли КАЖДУЮ переменную запроса: откуда пришла · какой строкой · была ли по дороге защита · в каком месте запроса стоит

ШАГ 3 выведи список незащищённых «грязных» переменных

ШАГ 4 вердикт МЕХАНИЧЕСКИ из списка: не пуст › Confirmed · пуст и происхождение всех известно › фолз · улики повреждены › всегда Unknown. «Не переопределяй интуицией»

63% → 100%

точность подтверждений на контрольном наборе, проверенном вручную: 19 сработок (12 действительно инъецируемых + 7 безопасных)

что дало прирост: обязательная таблица фактов + доставка строк происхождения кода — модель перестала «вспоминать» код, которого нет

guided decoding: поля ответа обязательны на уровне формата — пропустить шаг физически нельзя

дальше знания класса пополняются из ошибок прогонов: v4 — «переменная стоит в запросе» само по себе ещё не уязвимость; v5 — каждый разобранный промах превращается в новое правило

ЭКЗАМЕН: СИСТЕМА ПРОТИВ 2 546 РЕШЕНИЙ ЛЮДЕЙ

как проверяли: взяли 2 546 сработок, которые люди уже разобрали руками; система разобрала их же вслепую — человеческих ответов она не видела; потом ответы сверили. Из 2 902 исходных исключены 88 сработок, закрытых служебной автоматикой (не людьми), и 268 решений соавтора системы

СОВПАЛИ — 1 910 СРАБОТОК

система «фолз» и люди «фолз»: 1 856 · система «реальная» и люди «реальная»: 54

85,5%

сработок получили определённый ответ — «фолз» или «реальная» (это и есть покрытие)

РАЗОШЛИСЬ — 266

68 похороненных — реальные, закрытые как фолз: расплата за доверие всем вердиктам модели подряд

198 перестраховок — фолзы, отданные человеку как реальные: стоят минут ревьюера, дыр не создают

87,8%

ответов совпали с решением людей (из 2 176 отвеченных)

96,5%

вердиктов «фолз» верны (1 856 из 1 924) — главный показатель: именно фолзы уходят в автозакрытие

UNKNOWN — 370

система не решила и отдала человеку с уликами; реальных уязвимостей среди этих 370 — ноль

0,0%

технических сбоев на 2 902 прогона

68 похороненных = 3,53 потери на 100 автозакрытий; из-за них появились режимы строже — про них следующий слайд

ГДЕ АВТОМАТ ТОЧЕН, А ГДЕ ОБЯЗАН МОЛЧАТЬ

средней температуры нет — качество живёт по классам CWE; режим эксплуатации собирается из классов поимённо

ТОЧНЫЕ › В АВТОЗАКРЫТИЕ

класс 1027 (n=104): 98,9% закрыто, 1 потеря

классы 89 · 506 · 280 · 912 · 691: отсев 100% — люди там тоже сказали «всё фолзы»

МОЛЧАЩИЕ › ЛЮДЯМ

классы 250 · 311 · 362 · 244: Unknown 45–96%

в сработке мало улик — сомнение уходит человеку; так и задумано

ОШИБАЮЩИЕСЯ › В БЛОК-ЛИСТ

класс 257 (пароли, n=91): 46 похоронено — история дальше
класс 200: 13 · класс 325: 7

их автозакрытие выключено

дальше — экзамен property-классов на масштабе всего агрегатора › следующий слайд

ЭКЗАМЕН PROPERTY-КЛАССОВ

эксперимент: выгрузка метаданных всех 1 004 440 сработок агрегатора · классификация по типу локально, без прогона через боевой контур › 934 572 property (93%) · дальше — только property

1 корпус: 10 825 property-сработок со всех 180 классов, у каждой — вердикт человека · скил на каждый класс (87 боевых + 93 синтезированных) · 2 модели × 2 сборки промпта = 43 300 вердиктов

2 итог: первая очередь — 17 классов = 381 366 сработок = 40,8% property-потока (порог: ноль похороненных у всех четырёх конфигураций, точность закрытия не ниже 0,90)

тезаурус: первая очередь (Tier-1) — классы, допущенные к автозакрытию первыми

ЭКЗАМЕН ИЗЯЩНЫХ МОДЕЛЕК: КОГДА РАЗМЕР НЕ ИМЕЕТ ЗНАЧЕНИЯ

тот же срез, что у 14B, побайтово: 1 324 фолза + 101 канарейка · Qwen2.5-Coder 1.5B / 3B / 7B (q4)

1. гипотеза «возьмём модель побольше» опровергнута: 7B вдвое больше 3B — и хуже во всех трёх схемах ответа: через порог допуска 0 классов из 17 (у 3B — 8)

2. решает схема ответа — порядок полей, за который сервер физически не выпускает: «сначала дословные цитаты из кода, потом вердикт» › доля закрытых фолзов 0,801 против 0,345 у мягкой схемы. Фактам — да, поэзии — ни-ни!

3. лучшие связки «класс › модель + схема»: 13 классов = 99,4% объёма первой очереди на моделях 1,5–3 млрд · строим на 3B: 0,7–1,7 с на сработку против ~7 с у 14B — и меньше похороненных, чем у 1.5B

4. проверка на прочность: в срез подмешана 101 канарейка — сработка, в которой дефект точно настоящий и подтверждён человеком. Малые модели похоронили до 48 из 101 — отличать настоящий дефект от ложного они сами не умеют

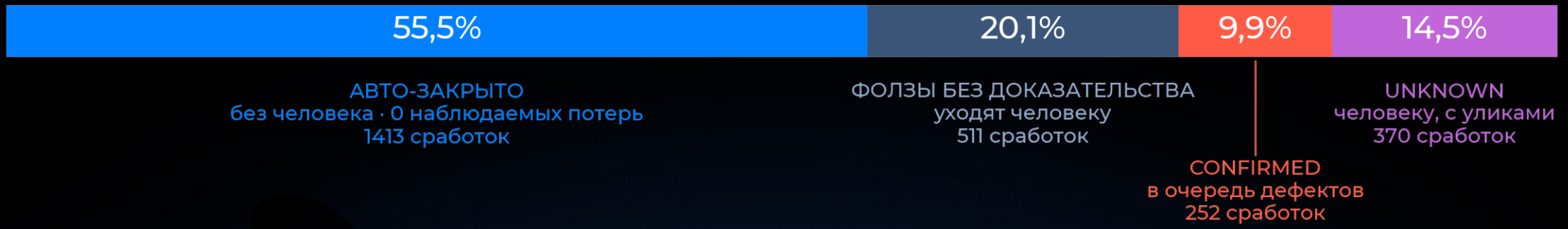
5. защита по обоснованию: модель нередко сама пишет в обосновании «критичный риск» — и тем же ответом закрывает сработку как фолз. Такое закрытие автоматически понижается до Unknown › заметив это, были введены дополнительные проверки на логику принятия решения моделью — результат: на треть больше корректных распознаваний

тезаурус: канарейка — подмешанная сработка с заведомо реальным дефектом · схема ответа — список и порядок полей, которые модель обязана заполнить

ПРАКТИЧЕСКИЙ ЭФФЕКТ: 55,5% ОЧЕРЕДИ — БЕЗ ЧЕЛОВЕКА

OFF ONE 2026

консервативный режим на главном замере: что происходит с очередью из 2 546 сработок



часы ручного разбора в неделю › останется с системой:

при 2 мин*	33 ч › 15 ч
при 5 мин	83 ч › 37 ч
при 10 мин	167 ч › 74 ч

* время на одну сработку — допущение сценария, не замер

ЧЕМ ЭТО ОБОРАЧИВАЕТСЯ В ДЕНЬГАХ

аппсек в среднем: 3 000 сработок в неделю — ~13 000 в месяц

при ЗП 200 тыс ₽/мес — ~15 ₽ за решение

одна RTX 4080: 1 000 сработок за ~3 часа — ~8 000 в сутки, ~240 000 в месяц

первая очередь (370 тыс.): месяц на 1 4В › неделя на 3В

= пропускная способность ~18 аппсеков

— ~3,7 млн ₽/мес (~44 млн ₽/год)

и это уже не стенд: почти 100 000 сработок прошло через модели (97 202 в срезе production-масштаба, июль, 99 классов CWE)

ГДЕ МЫ СПОТКНУЛИСЬ

три грабли за полгода — и все три поймали перепроверки

1 НАРЕЗКА ДАННЫХ · на смеси проектов 93,8% верных › внутри каждого проекта 41,7–60%, на Go/Ruby — 0%: скил подогнан под PHP. Вторая модель — та же яма. Лечение: мерить внутри каждого проекта

2 БОЕВОЙ ТОКЕН В ТЕСТОВЫХ ФАЙЛАХ · один настоящий GitLab-токен × 46 сработок; модель поверила пути: «placeholder в тестовом конфиге», уверенность 1.0 — все 46 похоронены. Лечение: алгоритмическая проверка «жив ли токен» на ступени 0

3 СБОРЩИК МОЛЧА РЕЗАЛ СКИЛ · лимит 16 800 символов, скил дорос до 22 800 — середину вырезал без предупреждения. Подняли лимит › класс 366 (гонка потоков): с 0,010 до 0,810. Лечение: порог на доставку скила

Все три грабли предотвратили перепроверки. Снаружи цифра, которая рассыпается при смене нарезки, и скил, который не доехал, выглядели правдоподобно

«Неважно, как часто ты падаешь. Важно, как часто ты поднимаешься» — Джейсон Стетхем

НАСТРОЙКА НА НОВЫЙ КЛАСС: 100 › 10 000

как класс дефектов автоматизируется: ~100 примеров от людей превращаются в 10 000+ разобранных машиной — штатная процедура эксплуатации

ШАГ 1 · ЛЮДИ	ревьюеры разбирают ~100 сработок класса (примерно день работы): ~40 идут на настройку промпта, ~15 — в замороженный контрольный набор
ШАГ 2 · НОЧЬ	автоцикл promptloop предлагает правку промпта; правка принимается ТОЛЬКО если качество на контрольном наборе выросло. Веса модели не трогаем; каждый шаг пишется в журнал
ШАГ 3 · КОНВЕЙЕР	разбирает 10 000+ таких же сработок за сутки с небольшим машинного времени

почему это работает: очереди однотипны — вспомните 46 потерь из ОДНОГО токена; в топ-классах тысячи повторов одного и того же паттерна

новая настройка (август): схема ответа подбирается под класс — одна смена схемы подняла долю закрытых фолзов с 0,345 до 0,801

за полгода — 143 цикла. подъёмы качества (F1 на контрольном наборе): слабый хеш (328): 0,615 › 1,0 · десериализация (502): 0,667 › 1,0 · инъекция кода (94): 0,667 › 1,0

СЕГОДНЯ МЫ УЗНАЛИ МНОГО НОВОГО

пять вещей, которые стоит унести с собой

1. Сработки бывают двух сортов. Property-based доказываются проверкой факта, trace-based — проверкой пути данных. Один вопрос «покажи путь» на всех — это потеря 9 из 10 реальных property-уязвимостей на ровном месте.

2. Property — алгоритмические проверки, trace — улики. Модель никогда не хоронит сработку единолично: под ней лестница проверок и правила.

3. Unknown — механизм безопасности. В главном замере ни одна реальная уязвимость не спряталась в Unknown (0 из 370): всё сомнительное уходит человеку.

4. Уверенность (confidence) модели — не пропуск в автозакрытие: её ответы с уверенностью 1.0 оказались хуже, чем с 0.8. Пропуск — только доказательство, проверенное алгоритмически.

5. Дисциплина важнее размера. Доставленный скил и жёсткая схема ответа двигают качество сильнее, чем лишние миллиарды параметров: 7B вдвое хуже 3B, а 87% похороненных — при недоставленном скиле.

НЕ ЗАСТАВЛЯЙТЕ МОДЕЛЬ УГАДЫВАТЬ. ЗАСТАВЬТЕ СИСТЕМУ ПРОВЕРЯТЬ.

ЧТО ДАЛЬШЕ

1. Защита по обоснованию: вердикт «фолз» при тревожном обосновании модели понижается до Unknown · на непросмотренном срезе — треть похороненных возвращена за 6 п.п. полноты
2. Оркестратор (уже в работе): центральная маршрутизация потока — доказанные лёгкие property-классы малым моделям, тяжёлые trace большим, под мощность модели и железа
3. Версионирование промптов-скилов с автоматическим уточнением под размер контекста и характеристики конкретной модели
4. Дообучение отдельных моделей (LoRA) на корпусах безопасности — по примеру Foundation-Sec-8B, дообученной на данных CVE и CWE
5. Внутренняя RAG-система: база знаний классов, проектов и прецедентов под рукой у каждой модели
6. Автогенерация и валидация патчей для подтверждённых сработок — отдельное большое направление
7. Структурная генерация скилов: сборка по контракту — атомарная доставка, бюджет места, схема ответа под класс

«Neo, sooner or later you're going to realize, just as I did, there's a difference between knowing the path and walking the path» — Морфеус

СПАСИБО ЗА ВНИМАНИЕ!

Вопросы?

Юрий Туманов (Эльария)

AppSec · локальные LLM · SSDLC

@psychodrake
github.com/Eljees
3.14hell@gmail.com

WAKE THE F**K UP, SAMURAI.
WE HAVE A CITY TO BURN

A1 · ФОРМУЛЫ И ПРАВИЛА UNKNOWN

Матрица: a=real→Conf · b=real→FP (ПОХОРОНЕНА) · c=real→Unk · d=fp→Conf · e=fp→FP · f=fp→Unk

P = a/(a+d) — точность вердиктов «реальная» (precision Confirmed)

Rб = (a+c)/Np — доля НЕ потерянных реальных (safety-recall; Unknown = сохранено) Rд = a/Np (Unknown = промах)

Ротс = e/(e+b) — точность авто-отсева (доля правды среди закрытых как фолз) Ротс = e/Nл Q = e/N

coverage = решённые/N · abstention = (c+f)/N · риск = 100·b/(e+b) на 100 авто-отсевов

Unknown: НИКОГДА не «правильный ответ»; при сверке с людьми — несовпадение; в боевую систему не пишется.

parse/timeout/llm-error — отдельные статусы, не вердикты; сломанный JSON → FP запрещён.

confidence ниже порога класса → авто-skip к человеку. Пороги по классам CWE, запасной 0.6.

Доверительные интервалы: ±9 п.п. при n=40 (p≈0.9); больше 20 п.п. при n=14 —

различия меньше интервала не интерпретируются.

A2 · ПОЛНЫЕ МАТРИЦЫ ГЛАВНОГО ЗАМЕРА

ВСЕ РЕШЕНИЯ ЛЮДЕЙ (n=2 814):

	люди:FP	люди:Conf
система FP	1 969	166
система Conf	221	84
система Unk	374	0

совпадение 73.0% · без Unknown 84.1% · точность «фолз» 92.2% · Unknown 13.3%

ПОСЛЕ ОЧИСТКИ — только ревьюеры, не видевшие ответов системы (n=2 546):

система FP	1 856	68
система Conf	198	54
система Unk	370	0

совпадение 75.0% · без Unknown 87.8% · точность «фолз» 96.5% · Unknown 14.5%

срез соавтора системы (n=268): совпадение 53.4%; 98 из 166 «Conf→FP» полного набора — его решения

пересчёт recompute_megasrez.py — матрицы сошлись бит-в-бит с замороженными

A3 · СРЕЗ ПО ПРОЕКТАМ: SQL-ИНЪЕКЦИИ (CWE-89)

Devstral-24B, процедура v3, полные корпуса приложений (порог приёмки P≥85%, Rб≥95% — задан ДО прогонов).
Обозначения — в A1; baseline = доля фолзов в корпусе (столько отсеял бы вердикт «всё фолз»).

Проект A (PHP)	n=40	P 52.9	Rб 100.0	похоронено 0	Ротс 100.0	baseline 50.0
Проект B (PHP)	n=40	P 57.1	Rб 100.0	похоронено 0	Ротс 100.0	baseline 52.5
Проект C (PHP)	n=40	P 60.0	Rб 75.0	похоронено 5	Ротс 64.3	baseline 55.0
Проект D (Go/Ruby)	n=30	P 41.7	Rб 60.0	похоронено 6	Ротс 0.0	baseline 76.7

Qwen-14B на тех же корпусах: P 51.9–61.5 · похоронено 12/75 · те же два проекта ниже порога

Пересечение похороненных: Проект D — 6 из 6 совпали; Проект C — 5 из 6 (+1 только Qwen)

разброс по 4 проектам — иллюстрация; факт — минимум и максимум (P4). CCTV-подмножество n=14:
доверительный интервал больше 20 п.п. — не интерпретируется. Имена приложений анонимизированы.

A4 · PROMPTLOOP: АВТОЦИКЛ НАСТРОЙКИ ПРОМПТОВ

Механика (10 шагов): выбор класса CWE (по решениям людей) → живой прогон → замороженный срез
→ базовый замер (train/val) → диагноз ошибок → промпт-кандидат → проверка, что промпт собрался без потерь
→ замер кандидата → принятие правки (+SHA до/после) → автотесты → повторный живой прогон → журнал

Значимые циклы (val, по решениям людей): CWE-328 f1 0.615→1.0 · CWE-502 0.667→1.0 · CWE-94 0.667→1.0
CWE-89 0.333→0.6 · CWE-732(3b) 0.667→0.857 · дальше NO-OP на потолке контрольного набора
120 строк журнала · 3 модели (14b/7b/3b) · 17 CWE · train≈40 / val≈15 — публикуется только с N

Порог в CI: падение качества (F1/точность/полнота) больше 0.005 или рост Unknown больше 0.02
автоматически блокирует изменение (triage-eval.yml)

A5 · КАЧЕСТВО ПО КЛАССАМ · ВОСПРОИЗВОДИМОСТЬ

Главный замер (после очистки), классы с n≥20: «годно» (точность фолз-вердикта F1≥85%) — 33 класса; примеры:

1027 (n=104, риск 1) · 89/506/280/912/691/778/20/472 (100% отсева) · 798 (83%) · 79 (93%)

«слабо»: 257 (46 потерь!) · 200 (13) · 325 (7) · 250/311/362/244/190 (Unknown 45–96%)

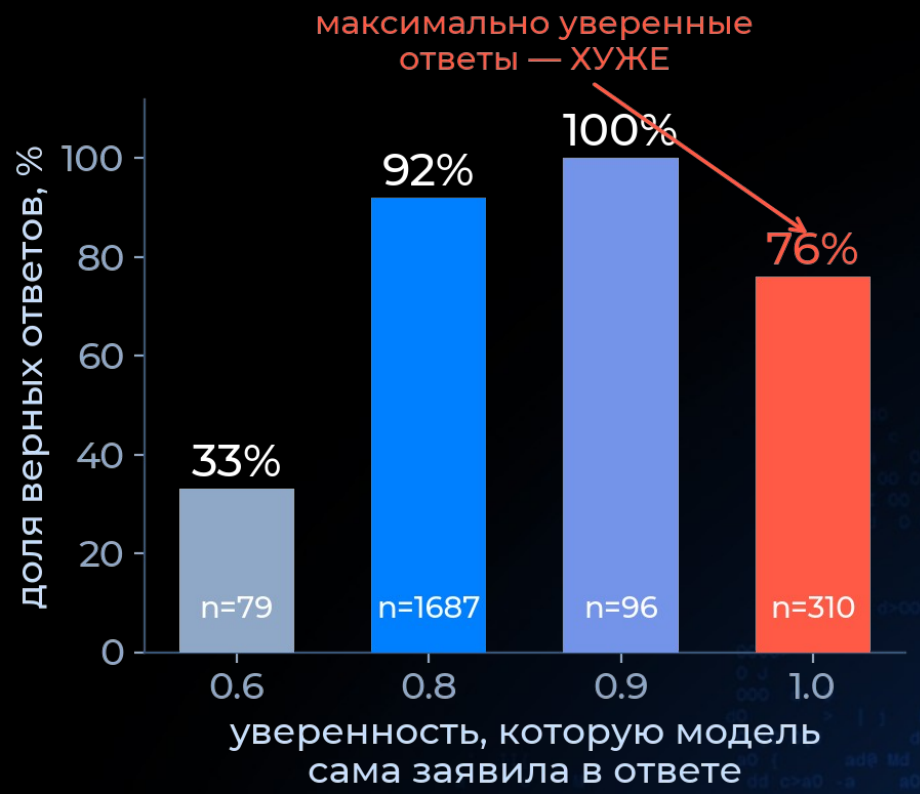
Обзорный замер (38 классов × 10+10, голый промпт): безопасные 79/94/601/918/502/1033/1035; худшие 259/319/330…

Воспроизводимость: каждая цифра деки — в 07_claims_ledger.csv (источник, формула, N, допущения); эксперименты 11–15.08 — в манифестах и журналах property_cwe_*;

пересчёт — scripts/*.py; методика и правила Unknown — 04_metrics_methodology.md;

чего пока не хватает — 08_missing_evidence.md. Полный пакет: v7_offzone_metrics/

A6 · УВЕРЕННОСТЬ МОДЕЛИ — НЕ ПОВОД ДЛЯ АВТОЗАКРЫТИЯ



точность ответов по заявленной самой моделью уверенности (главный замер, после очистки):

`confidence 0.8 › 92% верных (n=1 687)`

`confidence 1.0 › 76% верных (n=310) — максимально уверенные ответы ХУЖЕ`

порог «пускать в автозакрытие только confidence ≥ 0.9» дал бы 13,5 потерь на 100 автозакрытий — вчетверо хуже простого приёма всех фолз-вердиктов

причина кластера самоуверенных ошибок: 46 копий одного и того же боевого токена, закрытых как «placeholder» с уверенностью 1.0

A7 · ЭКЗАМЕН ИЗЯЩНЫХ: ЛУЧШАЯ СВЯЗКА ПО КЛАССАМ

замер 14–15.08: срез 1 324 фолза + 101 канарейка, побайтово тот же корпус, что у 14B. Уилсон — нижняя граница доверия с поправкой на размер выборки

477 устаревшие API — 209 781 в агрегаторе — 1.5B v2 — 1,000 (Уилсон 0,951)

248 неперехваченное исключение — 146 306 — 1.5B v2 — 0,953 (0,871)

778 недостаточное логирование — 10 486 — 3B v2 24k — 0,970 (0,915)

338 слабый генератор случайности — 2 530 — 1.5B v2 — 0,990 (0,946)

346 проверка origin — 2 200 — 3B v2 — 0,987 (0,930)

1022 target=_blank без noopener — 1 978 — 3B v2 — 1,000 (0,955)

474 двусмысленная функция — 1 975 — 3B v2 — 1,000 (0,946)

829 недоверенный источник — 1 616 — 3B v2 — 0,941 (0,870)

595 сравнение объектов по ссылке — 1 416 — 3B v2 — 1,000 (0,963)

284 разграничение доступа — 628 — 1.5B v2 — 0,940 (0,875)

426 путь поиска библиотек — 148 — 3B v2 24k — 1,000 (0,963)

676 опасная функция — 72 — 3B v2 — 1,000 (0,930)

203 различие в поведении — 62 — 3B v2 24k — 1,000 (0,908)

итог: 13 классов = 379 198 сработок = 99,4% объёма первой очереди = 40,6% property-потока

не прошли порог: 366 · 1035 · 693 · 1028 — вместе 2 168 сработок (0,6% объёма)

v2 — схема «сначала цитаты, потом вердикт» · 24k — поднятый бюджет сборщика · изящные = 1.5B и 3B