

Mask Gate — архитектура безопасности и поток данных

Mask Gate · trust pack · версия 1.1 · дата 25.10.2026 · проект документа, публикация — после подтверждения владельца

Версия: 1.1 · **Дата:** 25.10.2026 · **Статус:** проект документа, публикация — после подтверждения владельца.
Документ: trust pack, файл 1 из 4. Соседние: «Обработка ПДн и 152-ФЗ», «Поручение и субпроцессоры», «Качество детекции».

Что это. Описание того, что фактически происходит с данными при работе облачного сервиса Mask Gate (SaaS) и self-hosted поставки. Каждое утверждение опирается на код или артефакт репозитория; где опоры нет — это прямо указано. Документ не является юридическим заключением, сертификатом или аудиторским отчётом: сертификатов и независимого аудита у сервиса нет (см. раздел «Чего у нас нет»).

1. Что за система и где границы

Mask Gate — шлюз, который находит и маскирует персональные данные в тексте и файлах **до** того, как они покинут контур сервиса. Режимы работы:

Режим	Что делает	Кто оператор ПДн
SaaS (maskgate.ru)	Обработка на серверах сервиса в РФ; данные клиента приходят по HTTPS	ИП Цымбалюк А.В. (сведения оператора — в разделе 2)
Self-hosted (on-prem)	Тот же код развёрнут в контуре клиента; наружу данные не уходят	Клиент — оператор ПДн, он же настраивает политику и инфраструктуру

Принцип проектирования: **fail-closed** — если система не может определить, содержит ли текст ПДн, она не пропускает результат наружу (блокировка), а не «отправляет как есть».

2. Сведения об операторе (SaaS)

Поле	Значение	Источник
Оператор	ИП Цымбалюк Андрей Вадимович, ИНН 503123539105	публичные страницы сервиса (/security, /privacy, /offer), сведения оператора
Запись в реестре операторов	№ 77-25-184814	сведения оператора; в реестре РКН проверяются публично
Ответственный за организацию обработки	ИП лично	сведения оператора

Это **данные оператора о себе**, а не свойство кода: репозиторий их не подтверждает и не проверяет.

3. Поток данных в SaaS (по шагам)

```
Клиент (браузер / API / Excel-надстройка)
  | HTTPS, TLS 1.2/1.3
  ▼
nginx: HSTS, CSP, X-Frame-Options: DENY, Referrer-Policy
  | proxy_pass
  ▼
FastAPI-приложение (один процесс, uvicorn --workers 1)
```

- разбор текста/файла в оперативной памяти
- детекция ПДн по политике (48 категорий в политике на 25.10.2026)
- применение стратегии: block / redact / mask
- vault (только при обратимом режиме): AES-256-GCM, пара «токен-значение», TTL 30 суток
- аудит: HMAC-метки значений + счётчики (без содержимого), цепочка HMAC-SHA256
- ответ клиенту: результат + отчёт о находках
- сценарий «чат» (бэкенд реализован; публичный запуск — по решению владельца):
 - наш СЕРВЕРНЫЙ ключ `MG_CHAT_LLM_KEY``, провайдер DeepSeek (``api/chat_routes.py``, ``core/llm_client.py``); наружу уходит только замаскированный текст; ключ живёт в окружении/секрете, не в коде, не в БД, не в логах
 - (опционально, только по инициативе клиента) BYOK-шлюз ``/proxy/chat`` → внешняя LLM: наружу уходит ТОЛЬКО замаскированный текст; ключ провайдера вводит клиент, он не сохраняется и не логируется
 - (только при оплате) редирект в ЮKassa: данные карты обрабатывает платёжный провайдер

3.1. Что попадает на диск

Данные	Где лежит	Срок	Защита
Исходные тексты запросов	не пишутся: обработка в оперативной памяти процесса	—	—
Загруженные файлы	временный файл на время разбора, удаляется сразу после разбора	минуты	удаление в коде после парсинга
Зашифрованные пары «токен – значение» (vault)	SQLite-файл в томе <code>data/</code>	по умолчанию 30 суток (<code>MG_VAULT_TTL_DAYS</code>), затем удаляются janitor-процессом	AES-256-GCM; ключ из <code>VAULT_KEY</code> (вне БД); в базе только nonce + ciphertext
Журнал аудита	отдельный том <code>audit/</code>	по умолчанию 180 суток (<code>MG_AUDIT_RETENTION_DAYS</code>), ротация по возрасту	HMAC-метки значений; HMAC-цепочка; проверка <code>python -m core.audit_verify</code>
Учётные записи, платежи, API-ключи, коды входа	SQLite-файл в томе <code>data/</code>	пока жив аккаунт; журнал списаний — 365 суток (<code>USAGE_LEDGER_RETENTION_DAYS</code>)	пароли — PBKDF2-HMAC-SHA256; коды входа — <code>code_hash</code> + соль; API-ключи — только <code>key_hash</code> ; реквизиты карт не хранятся
Резервные копии БД	том <code>./backups</code> на хосте	30 суток , интервал 24 ч (<code>BACKUP_RETENTION_DAYS</code> , <code>BACKUP_INTERVAL_HOURS</code>)	онлайн-бэкап SQLite; содержит зашифрованный vault и хеши, не открытые тексты
Access-логи nginx	долговечная копия на хосте	не менее 30 суток (ротация <code>nginx/logrotate.conf</code> , проверка <code>scripts/check_log_retention.py --live</code>)	содержат IP-адреса — это персональные данные, см. раздел 5

Важная честная деталь: **шифруется только vault**. Остальные поля базы (email, суммы, статусы, счётчики) лежат в SQLite открытыми, но доступ к файлу БД требует доступа к серверу. Пароли, коды входа и API-ключи хранятся хешами — восстановить их из базы нельзя.

3.2. Что НЕ хранится

- тексты запросов и результаты проверок — не накапливаются, обрабатываются в памяти;
- исходные загруженные документы — не сохраняются после разбора (временный файл удаляется);
- данные банковских карт — не проходят через сервис, их обрабатывает платёжный провайдер;
- ключи LLM клиента (BYOK) — не сохраняются и не логируются (проверяется тестом

`tests/test_claims_vs_code_879.py` : ради этого в коде убрано логирование трассировки);

- серверный ключ чата (`MG_CHAT_LLM_KEY`) — только в окружении/секрете, не в коде, не в БД и не в логах (`core/llm_client.py`); адрес провайдера — фиксированный `allowlist`, а не значение из запроса;
- ключи шифрования и HMAC-ключи — только в окружении/секрете, не в БД и не в репозитории.

4. Шифрование и целостность

Механизм	Где применяется	Источник
TLS 1.2/1.3, ECDHE-шифры	соединения клиент ↔ сервис	<code>nginx/default.conf</code> (<code>ssl_protocols</code> , <code>ssl_ciphers</code>)
AES-256-GCM, AAD = (<code>session_id</code> , <code>token_id</code>)	значения vault перед записью в БД	<code>core/vault.py</code>
HMAC-SHA256, ключ <code>AUDIT_HMAC_KEY</code>	метки значений и цепочка аудита	<code>core/audit.py</code>
Ed25519 (публичный ключ в сборке)	лицензия on-prem; приватный ключ у вендора	<code>core/license.py</code>
JWT HS256	аутентификация пользователей	<code>core/jwt.py</code>
PBKDF2-HMAC-SHA256	пароли пользователей	<code>core/database.py</code>

Аудит устроен как **цепочка**: каждая запись содержит `prev` (значение цепочки предыдущей записи) и `chain` = HMAC(ключ, каноническая запись). Правка, удаление или перестановка строки ломает проверку с этой строки; при ротации первая запись новой генерации содержит `anchor` — значение цепочки последней записи предыдущей генерации, поэтому удаление или подмена **средней** ротированной генерации детектируется.

Границы целостности (честно):

- удаление файла **текущей** генерации или обрезка его хвоста — **не детектируется** (нужен внешний high-water mark, которого нет);
- удаление **самой старой** оставшейся генерации неотличимо от легальной ротации по сроку хранения — проверка сообщает об этом как о примечании, а не как о подделке;
- проверка осмысленна только при стабильном `AUDIT_HMAC_KEY` : с эфемерным ключом результат — «проверить нельзя», а не «всё цело».

5. Что раскрывается при инциденте

Сценарий: злоумышленник получил доступ к серверу или к резервной копии. Реалистичный объём:

Что может утечь	Что это даёт атакующему	Можно ли предотвратить
База <code>users.db</code>	email, хеши паролей (PBKDF2), платежные метаданные, хеши API-ключей, шифрованный vault	хеши необратимы; vault — только с ключом <code>VAULT_KEY</code>
Файл базы + <code>VAULT_KEY</code>	расшифровка пар «токен–значение» за последние 30 суток	ключ хранится отдельно от БД; ротация ключа — процедура эксплуатации
Журнал аудита	HMAC-метки значений, категории находок, время, origin; не сами значения	метки необратимы без <code>AUDIT_HMAC_KEY</code> ; ключ отдельно
Access-логи nginx	IP-адреса, URL и время запросов	ротация 30 суток, ограничение доступа к хосту
Резервные копии	копия БД на момент бэкапа (шифрованный vault, хеши)	30 суток, доступ к каталогу <code>./backups</code>

Что при этом не утекает: исходные тексты и документы (их нет на диске), данные карт (у платёжного провайдера), ключи LLM клиентов (не сохраняются).

6. Доступ и управление

- пользовательские пароли и коды входа — только в виде хешей (`code_hash` + соль);
- API-ключи B2B — только `key_hash`; лимиты RPM и суточные лимиты на ключ;
- аутентификация веб-клиента — JWT (HS256) + refresh-cookie; административные маршруты требуют административной роли;
- публичные страницы сервиса ничего не журналируют и не имеют собственного писателя в журнал аудита;
- при `MG_ENV=production` сервис **не стартует** без обязательных секретов: `JWT_SECRET`, `AUDIT_HMAC_KEY`, `VAULT_KEY`, `SMTP_HOST` (`.env.example`, `core/config.py`, `core/vault.py`);
- rate-limit: общий лимитер плюс прогрессивная блокировка повторных неудачных попыток входа.

7. Self-hosted: поток данных в контуре клиента

- код тот же, что в SaaS; отличие — конфигурация (`MG_EDITION=onprem`), собственная почта/домен, собственная политика маскирования (`.env` и `docker-compose.override.yml` клиента);
- клиент — оператор ПДн; данные не покидают его контур; вендор не получает содержимое и не имеет доступа к серверу клиента;
- лицензия — **офлайн-токен** Ed25519: сборка проверяет подпись публичным ключом, приватный ключ остаётся у вендора. Проверка лицензии **не передаёт данные вендору**;
- обновления — поставка образов (tar-архив), установка и откат — по инструкции `docs/ON_PREM.md`.

8. Границы и «чего у нас нет»

- **нет сертификатов** ISO 27001, SOC 2, ФСТЭК и любых других — ни у SaaS, ни у on-prem;
- **нет независимого аудита** безопасности и **нет отчёта пентеста** в репозитории;
- **нет SLA** и формальных гарантий доступности; сервис предоставляется «как есть» (раздел 8 оферты);

- **нет DLP-функций** уровня предприятия (агент на рабочих станциях, контроль каналов утечки);

сервис **не заменяет полную DLP-систему** — он снижает риск в конкретном сценарии;

- **шифрование БД не полное**: шифруется только vault (см. 3.1);
- **размещение серверов в РФ (Санкт-Петербург, Beget)** — утверждение оператора об инфраструктуре;

код его не подтверждает, независимого подтверждения в документе нет;

- **аудит-журнал не является неизменяемым** за пределами описанных гарантий (см. 4);
- **полнота детекции не 100 %**: на нашем тестовом корпусе recall 99,81 %, остаточный риск пропуска

есть (подробно — документ «Качество детекции»);

- **операционные лимиты и цены** (квоты, бесплатные операции) в этом документе намеренно не

зафиксированы числами: они меняются по решению владельца сервиса, а зафиксированное число устарело бы в день изменения. Актуальные значения публикуются на `/pricing` и в интерфейсе; для договора — на дату подписания.

9. Как проверить этот документ

Утверждение	Чем проверяется
Маскирование, стратегии, отчёт о находках	код <code>core/engine.py</code> , <code>api/processing.py</code> ; публичная страница <code>/quality</code>
AES-256-GCM, TTL 30 суток	<code>core/vault.py</code> , <code>.env.example</code> (<code>MG_VAULT_TTL_DAYS</code>)
Аудит без содержимого, HMAC-цепочка	<code>core/audit.py</code> , проверка <code>python -m core.audit_verify</code>
Временный файл и его удаление	<code>api/files_routes.py</code>
Резервные копии: 30 суток, 24 ч	<code>core/backup.py</code> , <code>.env.example</code>
Ротация access-логов ≥ 30 суток	<code>nginx/logrotate.conf</code> , <code>scripts/check_log_retention.py --live</code>
Отсутствие сырых данных в журналах/логах	тесты <code>tests/test_security_props.py</code> , <code>tests/test_claims_vs_code_879.py</code>
Требование секретов в production	<code>.env.example</code> , <code>core/config.py</code> , <code>core/vault.py</code>

Вопросы по безопасности и инцидентам: security@maskgate.ru.

Документ не является юридическим заключением. Сертификатов, независимого аудита и SLA у сервиса нет — это прямо указано в тексте.

Вопросы: security@maskgate.ru