Установка

- Обзор установки
- Системные требования
- Ограничения выпуска
- Reverse proxy
- Лицензия
- Диагностика установки

Обзор установки

Установка wiSLA

Для **первого выпуска** поддерживается один сценарий серверной части: **Ubuntu 24.04**, нативный установщик `.run` (**без контейнеров**). Агенты на мониторимых хостах ставятся отдельно — Агенты (флот) · Установка агента.

Номер продукта в этой документации **не фиксируется**: используйте выданный артефакт `wisla6-standalone-*.run` (+ `.sha256`).

Что **не** входит в выпуск: Ограничения выпуска.

Канон (shipping)

ОС	**Ubuntu 24.04** (x86_64), systemd, sudo/root
Установка	Native `.run` — инструкция оператора
Стек	PostgreSQL + ClickHouse + backend + nginx/UI на **одной** ВМ
Контейнеры	**Не используются** в поддерживаемом сценарии
Железо	Системные требования · калькулятор сайзинга

```
Установка (первый выпуск)
├── Обзор ← эта страница
├── Native Ubuntu 24 (.run)    ← единственный shipping
├── Системные требования
├── Ограничения выпуска
├── Reverse proxy (опционально)
└── Лицензия
```

После установки

1. Войти seed-учёткой ADMIN (только стенд) — Быстрый старт.
2. Settings → **Общие**: **Статус серверов** online.
3. Сменить пароль / MFA — Безопасность.
4. При необходимости — за reverse proxy, лицензия.
5. Настроить URL пакетов агента — Агенты (флот).
6. Установить агент на первый хост — **Инфраструктура → Агенты мониторинга → Скачать агент** (`/probes/install`).

Seed-учётки (только стенд / lab)

Login	Role	Password
admin	ADMIN	admin@123
operator	OPERATOR	operator@123
viewer	VIEWER	viewer@123
infosec	INFOSEC	infosec@123

На проде seed-пароли недопустимы.

Другие варианты (не в этом выпуске)

Материалы в репозитории сохранены для лаборатории и будущих волн. **Не** используйте их как инструкцию поставки первого выпуска.

Вариант	Документ	Статус для выпуска
Docker-пилот (простой инсталлер)	install-installer	**Не shipping** — lab / внутренний пилот
Compose только БД	install-compose	**Не shipping** — lab
Dev без `.run` (JDK/Node вручную)	install-standalone	**Не shipping** — разработка
Debian / Astra native	скрипты умеют распознать ОС	**Не заявляем** для выпуска (эталон — Ubuntu 24)
Kubernetes	install-kubernetes	**TBD / не поддерживается**
Кластер (HA)	install-cluster	**TBD / не поддерживается**
Postgres Pro	—	**Не поддерживается**

Компоненты (shipping / native)

Компонент	Назначение	Порт (native)
PostgreSQL	Конфиг, пользователи, аудит	5432
ClickHouse	Метрики time-series	8123, 9000
Backend (Spring Boot)	API, pollers, Flyway; UDP **162** trap / **514** syslog	8088
Frontend + nginx	Портал + `/docs/`	80 / 443
Агент wisla-wiprobe	Сбор на хосте	10050 (passive) и т.д.

Production (общее)

- За reverse proxy: reverse-proxy

- Диагностика: troubleshooting
- В prod-контуре native: `wisla.agent.auth-required=true` (standalone включает), `wisla.http.insecure-tls=false` где применимо

Системные требования

Системные требования wiSLA

Оценка железа и ОС для **серверной части** и **агента**. Базовый сценарий: **100 КЕ** (≈ 100 узлов/сервисов).

Цифры под текущий стек (ADR-001): PostgreSQL + ClickHouse + Spring Boot + Angular + wisla-wiprobe-agent. База — **100 КЕ**; планирование роста — до **~10 000 КЕ** (калькулятор сайзинга). **Kafka на пилоте / отделе не нужен** и в текущий стек не входит.

Минимум 4 vCPU / 8 ГБ — это пол после коротких SELECT inventory, **last() из in-memory value-cache** (скаляр last-value в Postgres для UI; кольцо не читается из PG на каждый триггер) и без cartesian JOIN шаблонов на каждый тик поллера. Иначе три агента уже насыщают CPU (см. heavy-poller-load). Память last-value кэша ≈ КЕ × метрик × N (`wisla.triggers.recent-keep`, default 60); cap `wisla.metrics.last-value.cache.maximum-weight` (64 МиБ) и `expire-after-access` 24 ч. Плюс heap-кэш эффективной конфигурации (`wisla.effective-config.cache.max-entries`, default 5000 сервисов) — снимок item/триггеров/макросов, не blob шаблона; объём строк в Postgres ≈ 40–70 на ОС-узел. На compact-боксе смотрите не только Java (`-Xmx1536m`): `docker-proxy` на опубликованных 5433/8123 тоже ест ядра, если backend на хосте ходит в localhost.

Shipping ОС для сервера: Ubuntu 24.04 (native `.run`, без Docker). Debian / Astra / контейнеры / Postgres Pro — не в этом выпуске.

Варианты установки: обзор. Интерактивная оценка: калькулятор сайзинга — есть пресеты **база / ОС+LLD / SNMP·сеть / HTTP·Proxmox**.

Что такое КЕ

КЕ = один объект мониторинга типа «узел / сервис» (`services`), не отдельная метрика.

Масштаб	КЕ	Комментарий
Пилот / отдел	**100**	этот документ
Отдел / площадка	**500–1000**	4–8 vCPU, 16 ГБ, 150–200 ГБ SSD
Целевой продукт (ADR-001)	**~10 000**	калькулятор + load test

Исходные допущения (100 КЕ)

Параметр	Значение
Число КЕ	100
Метрик на КЕ (в среднем)	**120** valued (≈ keys seed Linux by agent; без тяжёлого LLD)
Интервал	**60 с**
Retention сырых метрик (CH)	**30 дней**

Параметр	Значение
Операторов UI одновременно	**5–15**
Режим	одна ВМ, без HA
Снимки ИЗ (узлы)	строка в Postgres только при смене % / тона (событие), не каждые 5 мин

Типичные плотности для калькулятора (после LLD):

Сценарий	Метрик / КЕ (ориентир)
База / лёгкий микс	~120
ОС-агент + FS/IF discovery	~180
SNMP / сеть (if discovery)	~250
HTTP LLD (Proxmox и аналоги)	~400

Discovery SNMP/`images.discovery` в seed часто **1h**; прототипы трафика остаются частыми — в калькуляторе крутите «метрик на КЕ», а не только интервал discovery.

Нагрузка

≈ 100 × 120 / 60 ≈ 200 samples/s
≈ 17 млн строк/сутки

Для ClickHouse это **лёгкий** режим.

Слой	Оценка
Сырьё 30 дней	~10–20 ГБ
Rollups за год	+3–8 ГБ
CH на диске	**≈ 25–40 ГБ**
PostgreSQL	**< 1–2 ГБ**

Сервер на 100 КЕ (all-in-one)

Для 100 сервисов тяжёлый сервер не нужен. Канон: Ubuntu 24 native.

Ресурс	Минимум (пилот)	Рекомендуется
vCPU	**2**	**4**
RAM	**4 ГБ**	**8 ГБ**
Диск SSD	**40 ГБ**	**80 ГБ**
Сеть	100 Мбит/с	1 Гбит/с (по желанию)
ОС	**Ubuntu 24.04** x86_64	то же

RAM при 8 ГБ

Компонент	Доля
ClickHouse	1,5–2,5 ГБ (compact cap ~2 ГБ)
PostgreSQL	0,5–1 ГБ
Spring Boot	**0,75–1,5 ГБ** (`-Xmx1536m` в systemd / installer)
Nginx / UI + ОС	остальное

Диск при 80 ГБ

Назначение	Объём
ОС	15–20 ГБ
PostgreSQL	2–5 ГБ
ClickHouse	25–40 ГБ
Логи / бэкапы / запас	15–30 ГБ

Когда нужно больше

Сценарий	Ориентир
Retention 90+ дней, тяжёлая аналитика	диск ↑, RAM 8⟶12 ГБ
Рост до **500–1000 КЕ**	4–8 vCPU, 16 ГБ, 150–200 ГБ
HA / отдельные БД	не из-за 100 КЕ; **HA в выпуске не поддерживается** ⟶ ограничения

Разносить PG/CH/app на три машины для 100 КЕ **обычно не нужно**.

Стек (native shipping)

Сервис	Порты (native)
PostgreSQL 16+ (дистрибутив Ubuntu; **не** Postgres Pro)	`5432`
ClickHouse 24.8+	`8123`, `9000`
Spring Boot	`8088`
UI (nginx)	80/443

См. ADR-001. Compose-порты (`5433` и т.п.) — только lab.

Трафик агентов на 100 КЕ — **единицы Мбит/с** суммарно.

Агент на хосте

Ресурс	Типично	Пик
CPU	**< 1–2 %** ядра	до ~5 % (LLD/WMI)
RAM	**30–80 МБ**	100–150 МБ
Бинарь + конфиг	15–40 МБ	—
Сеть к серверу	обычно **< 0,5 Мбит/с**	—

С ролью **channel**: +10–40 МБ RAM, +1–5 % CPU.

Минимум на хосте: свободно ≥ 256 МБ RAM и ≥ 500 МБ диска; исходящий доступ к API. Runtime Go на хосте не нужен.

Сводка: 100 КЕ

Что	Рекомендация
Сервер	**4 vCPU, 8 ГБ RAM, 80 ГБ SSD**
Минимум	2 vCPU, 4 ГБ RAM, 40 ГБ SSD
Агент	~50–100 МБ RAM, <2 % CPU
Данные CH (30 дн.)	~25–40 ГБ

Как устроено

Документ	Содержание
ADR-001	PG + ClickHouse
Агент	host / channel
Установка	варианты развёртывания

Дальше

- Нагрузочный прогон и правка цифр по факту
- Рост через калькулятор сайзинга (пилот → отдел → до ~10 000 КЕ)
- HA / Kubernetes — не в этом выпуске

Ограничения выпуска

Ограничения первого выпуска

Честный список того, что **не** входит в поддерживаемую поставку. Канон установки: обзор · Ubuntu 24 native.

Номер будущего релиза продукта здесь **не** указывается.

Установка и платформа

Тема	Статус
Ubuntu 24.04 + native `.run` (без Docker)	**Поддерживается**
Debian (native)	Инсталлятор может распознать ОС; **для выпуска не заявляем**
Astra Linux 1.8 (native)	То же; часто нужен extended/offline mirror — **не заявляем**
Docker-пилот / полный стек в контейнерах	**Не shipping** (install-installer — lab)
Docker Compose как прод-установка	**Не shipping** (install-compose — lab/БД)
Kubernetes	**Не поддерживается** (каркас TBD)
Кластер / HA / несколько manager	**Не поддерживается** (каркас TBD)
Postgres Pro	**Не поддерживается** (стек — PostgreSQL из дистрибутива Ubuntu)
Установка из одного `.run` полностью без сети (airgap)	**Не поддерживается** — нужен apt/CH mirror или golden image (план)

Данные и масштаб

Тема	Статус
Одна BM all-in-one	Поддерживаемый режим
Автоматический failover БД / app	Нет
Разнос PG/CH/app на несколько узлов «из коробки»	Нет (ручная архитектура вне выпуска)

Агенты и пакеты

Тема	Статус
Установка wiSLA-агента с визарда `/probes/install`	Поддерживается — агенты

Тема	Статус
Подпись пакетов агента (GPG/secure boot policy)	Пакеты пока **unsigned** (lab OK) — packaging
Browser / Chromium web-scenario addon как прод	Не заявлять как готовый (stub / следующая волна)

UI и функции продукта

Отложенные или скрытые элементы UI (маршрут может существовать, пункта в меню нет) — см. beta-ui-deferred, в т.ч. конструктор отчётов в сайдбаре.

Лицензии

Community (cloud) / Enterprise / Trial — см. лицензия. On-prem без ключа = **Trial** с жёстким лимитом создания сервисов; квота и срок enforce-ятся на create. Multi-node / Proxy licensing — вне этого выпуска.

Дальше

- Диагностика
- Системные требования
- Быстрый старт

Reverse proxy

Reverse proxy (HTTPS)

Как вынести **Ubuntu 24 native** стенд за reverse proxy (nginx / HAProxy / cloud LB). Это **не** кластер и **не** HA — один manager за TLS-терминацией. Ограничения выпуска.

Канон установки: Ubuntu 24 native.

Что проксировать

Путь / порт	Назначение
`/` (UI)	Портал Angular
`/api/`	Backend API
`/docs/`	Документация
`/swagger-ui/`, `/v3/api-docs`	API docs (если открываете снаружи)
`/agents/`	Каталог и пакеты агента (`index.json`, deb/rpm/msi)

Backend на стенде слушает **8088** (внутренний). UI/docs обычно отдаёт nginx на **80/443** самого стенда — либо proxy целится в этот nginx, либо (реже) разделяет UI и API на разные upstream.

Обязательные настройки

1. **Публичный URL менеджера** для агентов — в `wisla.properties` (или эквивалент на стенде):

   ```
   wisla.agent.manager-public-url=https://monitoring.example.com
   ```

   Без этого агенты и визард могут подставлять внутренний `http://IP:8088`.
2. **TLS** — сертификат на proxy (или Let's Encrypt на самом стенде через TUI **HTTPS**, если proxy нет).
3. **WebSocket / long poll** — если proxy буферизует ответы, отключите буферизацию для API при проблемах с live UI (зависит от proxy).
4. **Размер тела** для upload пакетов агента: `client_max_body_size` (nginx) ≥ **128m** на `location /api/` — иначе 413.
5. **CORS** — если UI и API на разных origin (нетипично для all-in-one), добавьте публичный origin в `WISLA_CORS_ORIGINS` / настройки CORS стенда.

Пример (схема)

```
Клиент / агент
        |
        ▼
https://monitoring.example.com  ← reverse proxy (TLS)
```

```
        |
        ├── /api/     → http://WISLA_HOST:8088
        ├── /agents/  → http://WISLA_HOST:80/agents/   (или тот же nginx стенда)
        ├── /docs/    → http://WISLA_HOST:80/docs/
        └── /         → http://WISLA_HOST:80/
```

Точные `proxy_pass` зависят от того, слушает ли внешний proxy nginx стенда (:80) или backend напрямую.

Чего нет в выпуске

- Несколько узлов manager с failover
- Автоматический баланс pollers / shared session store «из коробки»

См. диагностика · агенты (флот).

Лицензия

Лицензия (Community / Enterprise / Trial)

Активация ключа в UI. Документация описывает **фактическое** поведение продукта.

Где в UI

Настройки → раздел лицензии (ADMIN активирует / сбрасывает; INFOSEC — просмотр). Кнопка **Справка** у заголовка — модалка с видами лицензий и форматом ключа.

Редакции

Редакция	Поведение в продукте
Community	Облачная редакция wiSLA: лимиты задаются тарифом/тенантом в Cloud. **Не** локальный режим без ключа
Trial	On-premise по умолчанию без Enterprise-ключа: жёсткий лимит **50** сервисов, без календарного срока, пока не активирован ключ со сроком
Enterprise	Коммерческий on-prem по ключу: организация, срок (`yyyy-mm-dd` или `none`), max сервисов

Сброс ключа возвращает к **Trial** (не Community).

Формат ключа

Принимаются **оба** варианта:

1. Структурированный plaintext:

```
WSL1/<edition>/<org>/<yyyy-mm-dd|none>/<max>
```

- `edition`: `enterprise` / `trial` (локальная активация `community` отклоняется)
- Допускается префикс `WSL.1/` (нормализуется к `WSL1/`)
- Opaque-ключ ≥12 символов (не WSL1 и не расшифровываемый) → Trial на **30 дней** с лимитом 50

1. Зашифрованный ключ поставщика (IT-MANAGER AES / AES-GCM v2), если на сервере задан `wisla.license.vendor-encryption-key` (`WISLA_LICENSE_ENCRYPTION_KEY`):
   - COMMERCIAL → Enterprise
   - CE JSON → Trial с `maxActiveServices` и сроком из ключа

Ключ хранится зашифрованным; в UI — tip и квота `used/max`.

Ограничения

- Лимит сервисов и истекший срок **блокируют создание** новых сервисов (`403`, коды `LICENSE_QUOTA_EXCEEDED` / `LICENSE_EXPIRED`). Существующие сервисы продолжают работать.
- Конфиг Trial max: `wisla.license.trial-max-services` (по умолчанию 50).

Вне scope этого выпуска

Лицензирование **multi-node** и схем с **Proxy** — отдельная задача (не описано как поддерживаемое поведение здесь).

API: `GET/POST/DELETE /api/v1/settings/license` (см. Swagger).

См. также: Настройки системы, Ограничения выпуска.

Диагностика установки

Диагностика для администратора

Канон стенда: Ubuntu 24 native. Что не входит в выпуск: ограничения.

Shipping (Ubuntu 24 native)

Симптом	Куда смотреть	Действие
UI / API не отвечают	TUI **Состояние стенда** / `doctor`	`sudo /opt/wisla6/wisla-standalone.sh status` · `doctor`; journal служб
Backend unhealthy	`:8088/actuator/health`	`curl -fsS http://127.0.0.1:8088/actuator/health`; логи backend в TUI **Журналы**
Нет `/docs/`	nginx стенда	Проверить nginx и путь `/docs/`; не путать с dev `:3400`
Swagger 404	backend :8088	backend up, `/swagger-ui/`
Агент offline при `auth-required`	`/etc/wisla/agent.yaml`	Нужен Bearer-токен; на same-host: `wisla-standalone.sh local-agent` — агенты
`dpkg`: «не Debian archive»	HTML вместо `.deb`	`curl -I http(s)://HOST/agents/...deb` ⟶ 200 + octet-stream; версия из `/agents/index.json`
Визард даёт пустой/старый пакет	каталог пакетов	`wisla.agent.packages-dir` / publish в shared; флот
Upload пакетов — 413	nginx body size	`client_max_body_size 128m` на `/api/` — reverse-proxy
Нет метрик на узле	агент online, шаблон	heartbeat, access_point, poller logs
Нет писем/webhook	Settings ⟶ Уведомления	enabled, credentials, severity, maintenance
MFA не пускает	Аккаунт, время OTP	синхронизация часов, backup code
Сессии «чужие»	`/access/sessions`	revoke, max_sessions_per_user
Много LOGIN_FAILURE	Аудит, lockout	IP/актор, lockout_seconds

Агенты и wiProbe (дополнительно)

Симптом	Куда смотреть	Действие
wiProbe не появляется	handshake / URL	Перезапуск backend после смены кода; SLAMON: `http://HOST:8088`; SLAMon

Симптом	Куда смотреть	Действие
После wipe БД нет агентов	ожидаемо	Регистрация заново после install + heartbeat
Геокарта без маркеров	адрес на агенте	Заполнить **Расположение** на карточке

Lab / dev only (не путь выпуска)

Симптом	Примечание
`:4300/login` + CORS / Docker pilot	Dev UI + pilot compose — не shipping; см. archive `install-installer`
docs-site `:3400` локально	Только разработка документации
Prometheus / Hikari при волне агентов	runtime metrics — lab

Общий FAQ: guide/faq.

Стендовый прогон (внутр.): alpha-smoke.