DeepDive

Установка и администрирование

Версия 0.1.0 · 3 ноября 2026 г.

О DeepDive

DeepDive — ИИ-ассистент для анализа данных компании. Сотрудник задаёт вопрос на обычном языке — например, «почему упали продажи в Москве?» — и получает ответ по данным своей компании: выводы, таблицы и графики. Ассистент работает только с подключёнными данными и файлами пользователя, интернет для ответов не используется.

Для кого

- **Бизнес-пользователи** — руководители, аналитики, сотрудники подразделений.
- **Администратор организации** — источники данных, участники, бюджет, правила ассистента.
- **ИТ-служба и служба безопасности заказчика** — развёртывание, сопровождение, аудит.

Как устроено

Данные из источников — баз данных, витрин, учётных систем, файлов — приводятся к единой модели с одинаковыми показателями и справочниками.

Ассистент отвечает на вопросы по этой модели, графики из ответов собираются в дашборды, а автопилоты по расписанию проверяют данные и присылают сигналы о рисках и возможностях.

Платформа разворачивается как SaaS, on-premise в контуре заказчика или в частном облаке. Продукт везде один и тот же, различаются место хранения данных и то, кто обновляет платформу.

Состав документации

- Руководство пользователя — для пользователей и администраторов организации.
- **Установка и администрирование (этот документ)** — для ИТ-службы заказчика.
- Архитектура и безопасность — для архитекторов и службы ИБ.

Поддержка

Платформу предоставляет DeepDive. Вопросы о работе платформы — support@godeepdive.ru.

Содержание

Установка и администрирование

Варианты развёртывания 4
Состав платформы 9
Комплект поставки 12
Требования к площадке 16
Требования к вашему кластеру 24
Корпоративный Keycloak 30
Выбор компонентов 35
Установка 43
Обновление 53
Конфигурирование 57
Эксплуатация и диагностика 60
Данные, бэкапы и retention 64
Пользователи, роли и доступ 68
Панель администратора установки 72

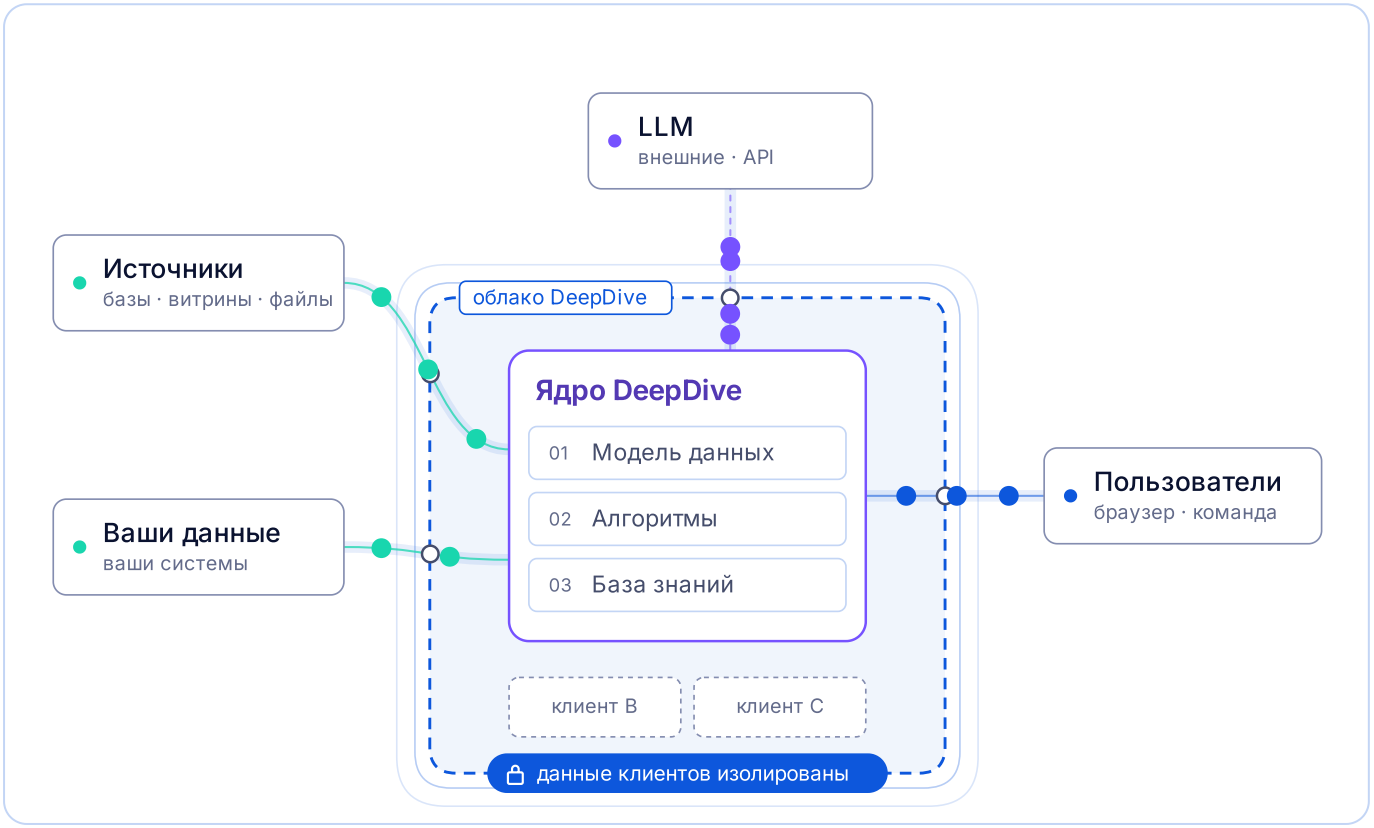
Варианты развёртывания

DeepDive запускается тремя способами. Продукт во всех трёх один и тот же — интерфейс, ассистент, модель прав. Различаются они тем, где работает платформа, где хранятся ваши данные и кто её обновляет. Выберите вариант, чтобы увидеть схему потока данных и подробности.

ВАРИАНТ 01

SaaS

Быстрый старт в нашем облаке, обновления и поддержка на нашей стороне

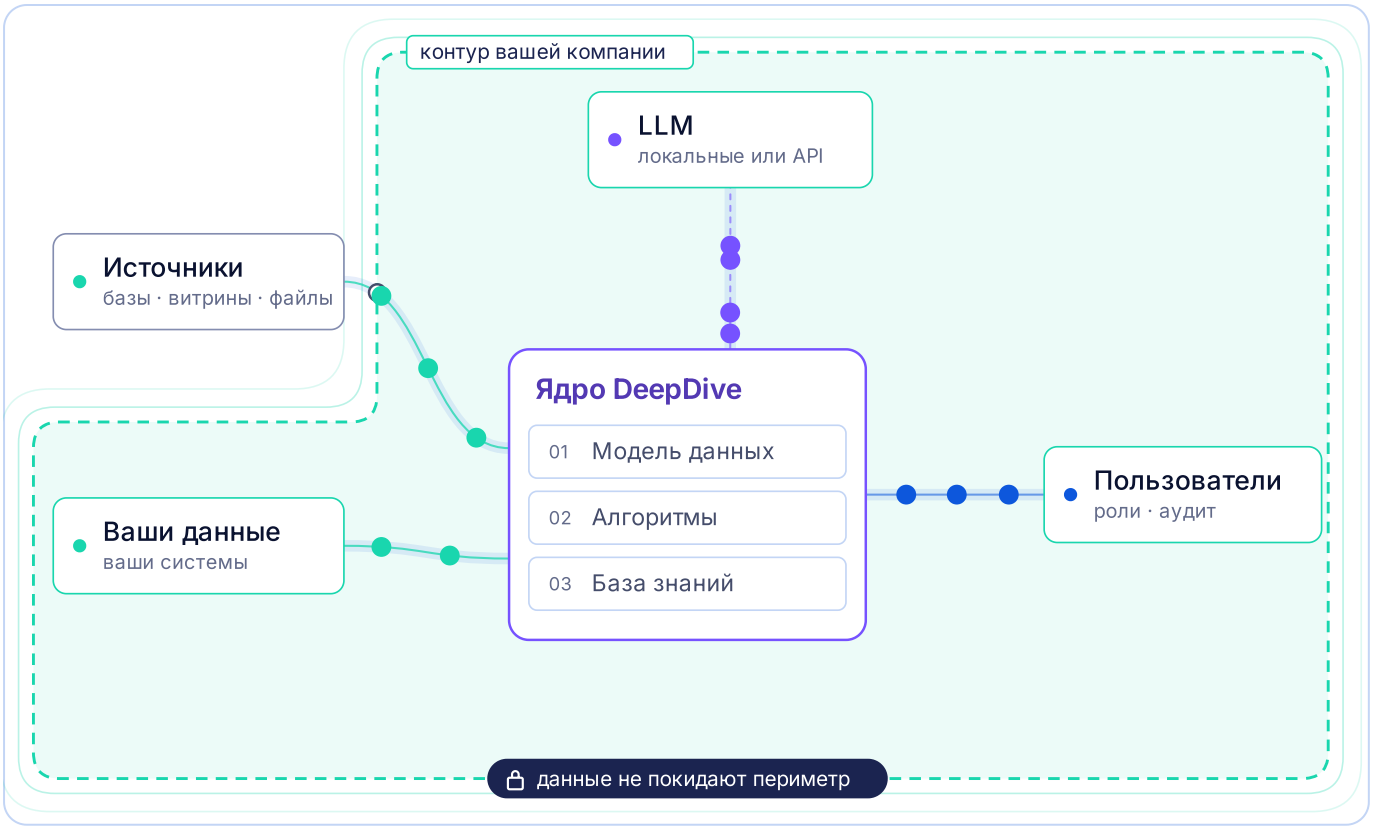


ГДЕ РАБОТАЕТ	В облаке DeepDive
ДАННЫЕ	Поступают через коннекторы к вашим системам и файлам; данные вашей компании изолированы от данных других клиентов **Опция: подключение к данным в вашем контуре**
МОДЕЛИ	Внешние LLM через API
ОБНОВЛЕНИЯ	Выпускаем мы, без участия вашего ИТ
ПОДХОДИТ	Для быстрого старта и пилота, когда нет требования держать данные в своём контуре

ВАРИАНТ 02

On-premise

Полностью в вашем контуре, данные не покидают периметр

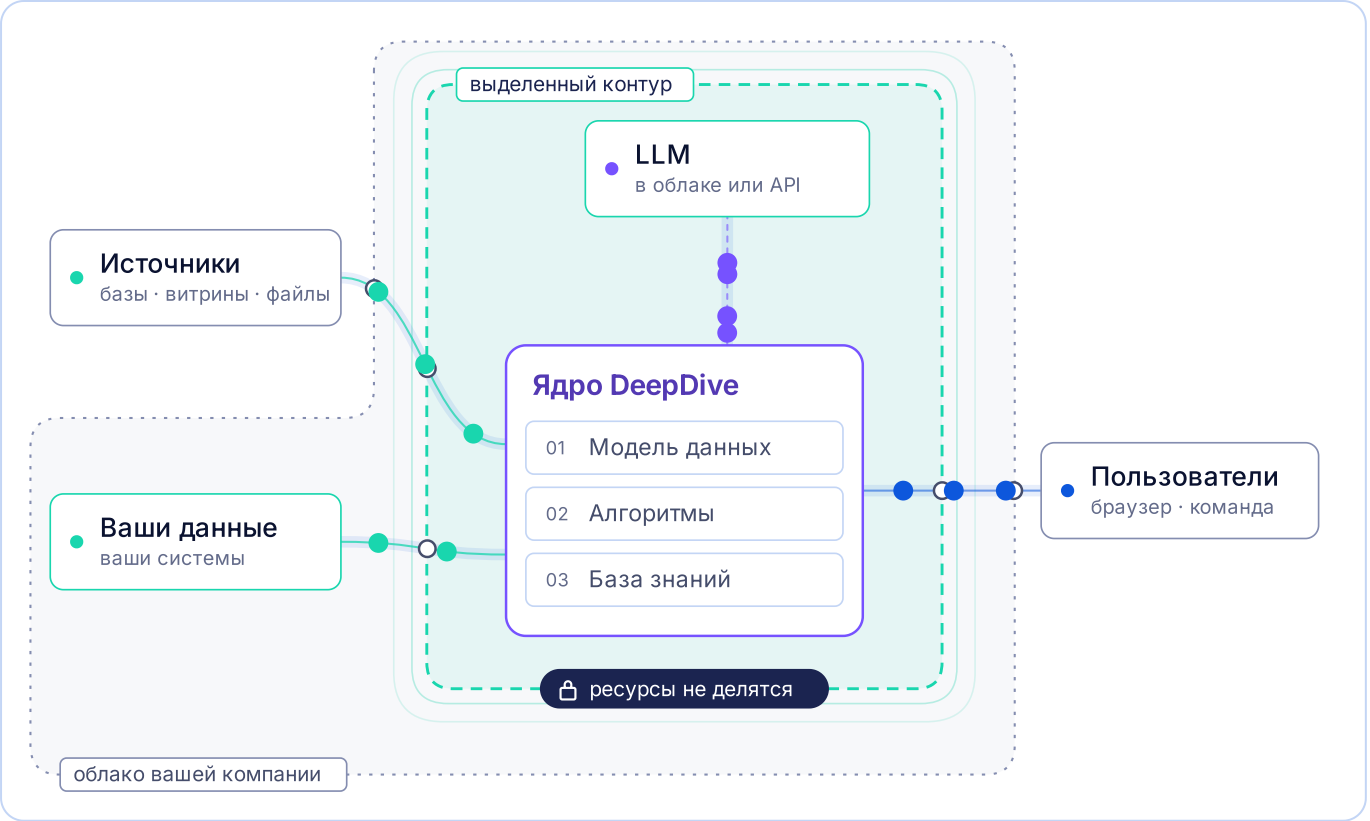


ГДЕ РАБОТАЕТ	В инфраструктуре вашей компании
ДАННЫЕ	Остаются внутри вашего периметра
МОДЕЛИ	Локальные LLM в вашем контуре или внешние через API
ДОСТУП	Корпоративная аутентификация, роли, аудит действий
ОБНОВЛЕНИЯ	Поставляем релизами, устанавливаются по согласованию с вашим ИТ
ПОДХОДИТ	При строгих требованиях информационной безопасности

ВАРИАНТ 03

Частное облако

Выделенный контур в облаке клиента



ГДЕ РАБОТАЕТ	В выделенном контуре в облаке, которое выбрала ваша компания
ДАННЫЕ	Остаются в вашем облачном аккаунте, ресурсы не делятся с другими клиентами
МОДЕЛИ	Внешние через API или развёрнутые в вашем облаке
ОБНОВЛЕНИЯ	По согласованному регламенту
ПОДХОДИТ	Компаниям с облачной стратегией, которым нужен изолированный контур

КОРОТКО: ТРИ ВАРИАНТА РЯДОМ

	SaaS	On-premise	Частное облако
ГДЕ ЖИВУТ ДАННЫЕ	В облаке DeepDive, изолированно от других клиентов; опция подключения к данным в вашем контуре	Внутри вашего периметра	В вашем облачном аккаунте

	SaaS	On-premise	Частное облако
ГДЕ РАБОТАЮТ МОДЕЛИ	Внешние LLM через API	Локальные в вашем контуре или внешние через API	Внешние через API или в вашем облаке
КТО ОБНОВЛЯЕТ	Мы, без участия вашего ИТ	Мы поставляем релизы, установка — по согласованию с вашим ИТ	По согласованному регламенту

Что это значит для установки

SaaS. Ставить и обслуживать в этом варианте нечего: остальная часть раздела «Установка и администрирование» к нему не относится.

On-premise. Платформа поставляется **офлайн-комплектом**: один архив, в котором есть всё для установки без доступа в интернет — образы, чарты, Kubernetes, пакеты ОС и инсталлятор. Что в нём лежит, — на странице «Комплект поставки» (стр. 12), как его поставить — на странице «Установка» (стр. 43).

Частное облако. Технически это та же установка, что и on-premise: тот же офлайн-комплект, тот же инсталлятор, те же требования. Машины или кластер Kubernetes в этом случае — в вашем облачном аккаунте. Если у облака есть выход в интернет, становятся доступны два варианта, которых нет в закрытом контуре: внешние коммерческие модели по API и сертификаты Let’s Encrypt (см. «Выбор компонентов» (стр. 35)).

Состав платформы

Раздел «Установка и администрирование» написан для службы ИТ, которая ставит и обслуживает DeepDive в своём контуре — on-premise или в частном облаке (см. «Варианты развёртывания» (стр. 4)). Работа пользователей и администраторов организаций описана в разделах «Руководство пользователя» и «Руководство администратора».

Сервисы платформы

Платформа — шесть сервисов. Пять серверных собраны в один образ и различаются точкой входа, поэтому всегда обновляются одной версией; шестой — веб-интерфейс.

Сервис	Назначение	Порт
`config`	Централизованная конфигурация: раздаёт остальным сервисам их настройки	3001
`service-discovery`	Реестр сервисов и их адресов	3002
`identity`	Кто вошёл и что ему можно: пользователи, организации, роли, интеграция с Keycloak	3005
`instance`	Установка и её организации: идентификатор, оформление	3007
`api`	Основной сервис: GraphQL-API, ассистент, диалоги, база знаний, подключения к источникам данных	3000
`ui`	Веб-интерфейс, общается с `api` только по GraphQL	—

Перед платформой стоит **oauth2-proxy** — единственная точка входа снаружи. Он завершает TLS, проверяет сессию в Keycloak и передаёт платформе, кто пришёл. Интерфейс и API живут на одном домене за ним.

Порядок запуска

Сервисы стартуют по цепочке, и каждый ждёт готовности предыдущего по его `/healthz`:

```
config (3001)                  стартует первым, зависимостей нет
  └─ service-discovery (3002)
       ├─ identity (3005)
       └─ instance (3007)
            └─ api (3000)
                 └─ ui
```

Практическое следствие: если `config` не поднялся, не поднимется ни один сервис, — разбор всегда начинается с него.

У `api`, кроме `/healthz`, есть `/livez`: он выполняет `SELECT 1` через собственный пул соединений и отвечает `503`, если база не возвращает ответ. По нему Kubernetes перезапускает под, который потерял соединения с PostgreSQL (например, после пересоздания пода базы), — `/healthz` базу не проверяет.

Зависимости

Система	Для чего	Обязательна	Откуда берётся
PostgreSQL 16 с расширением `vector` (pgvector)	База платформы: настройки, организации, диалоги, состояние ассистента, база знаний; база сервиса `identity`; база Keycloak, если он из комплекта	Да	Из комплекта, на отдельной машине или ваш
Redis	Кэш платформы и сессии входа	Да	Из комплекта, на отдельной машине или ваш
NATS JetStream	Шина событий, поток ответа ассистента в интерфейс	Да	Из комплекта или ваш
S3-совместимое хранилище	Вложения диалогов и выгрузки	Да	Из комплекта, на отдельной машине или ваше
Keycloak	Вход в платформу (OIDC)	Да	Из комплекта или ваш
Языковая модель	Ответы ассистента; нужен вызов инструментов (tool calling) и потоковый ответ	Да	Ваш endpoint в контуре или внешний провайдер по API

Система	Для чего	Обязательна	Откуда берётся
Модель векторизации (embeddings)	Поиск по базе знаний	Да	Ваш endpoint в контуре или внешний провайдер по API
ClickHouse	Витрина бизнес-данных, таблицы из загруженных файлов	Нет	Из комплекта или ваш
Oracle, PostgreSQL	Источники бизнес-данных, если ассистент читает их напрямую	Нет	Ваши
Langfuse	Трассировка работы ассистента	Нет	Только ваш: из комплекта не ставится

Что из этого поднимает комплект, а что подключается ваше, решается при установке, — см. «Выбор компонентов» (стр. 35).

База знаний живёт в базе платформы. Отдельного сервера под неё нет: таблицу и индексы платформа заводит сама при старте, в той же PostgreSQL. Поэтому к серверу PostgreSQL одно требование сверху — расширение `vector`. Граф знаний на Neo4j, который был в платформе до версии 1.0, удалён: ни сервера, ни тома, ни пароля под него больше не нужно.

Источники бизнес-данных

Ассистент отвечает по данным, к которым платформа подключена как к **источникам**: ClickHouse, Oracle и PostgreSQL. Подключение — запись в базе платформы; витрину ClickHouse, названную при установке, заводит источником сама установка, остальные подключаются в интерфейсе. Платформа данные в источник **не загружает и не пишет** — наполнение витрины остаётся за вашей службой эксплуатации. Подробнее — в «Руководстве администратора» (см. «Руководство пользователя» → «Руководство администратора» → «Подключение источников данных»).

Комплект поставки

В свой контур DeepDive ставится **офлайн-комплектом** — одним архивом, в котором есть всё для установки без доступа в интернет и без внутреннего реестра образов. Комплект **универсальный**: он несёт полный набор компонентов, а что из них поднимать и что подключить своё, решается на площадке, в мастере установки (см. «Выбор компонентов» (стр. 35)).

Версия комплекта

Комплект — обвязка вокруг одного релиза платформы, и его версия называет обе половины:

```
aivo-1.9.0-dist.2
     ^^^^^      ^
     релиз      ревизия комплекта
     платформы
```

- **Релиз платформы** — версия сервисов: один номер на все её образы.
- **Ревизия комплекта** — всё вокруг релиза: чарты, пакеты ОС, инсталлятор, инструкция. Начинается с 1 заново на каждом релизе платформы. Поэтому разница между `…-dist.1` и `…-dist.2` — всегда про поставку, а не про платформу.

Что изменилось в ревизии, написано в `CHANGELOG.md` комплекта. Версию уже распакованного комплекта показывает `./aivo-installer info`.

Что внутри

```
aivo-<версия>/
├── configure, install,     # инсталлятор: четыре команды
│   verify, update
├── bootstrap.sh            # проверка окружения python
├── aivo-installer          # CLI: info, secrets, kb, …
├── installer/, vendor/     # код и библиотеки инсталлятора
├── components.yaml         # матрица компонентов для мастера
├── presets/                # готовые ответы под поставки
├── rke2/                   # Kubernetes (RKE2) для своих машин
├── tools/                  # kubectl и helm
├── images/                 # образы контейнеров (*.tar.zst)
│   └── import.sh           #   раскладка в реестр кластера
├── charts/                 # helm-чарты
├── postgres/, redis/       # офлайн-пакеты ОС для их машин
├── garage/                 # S3-хранилище, статический бинарь
├── keycloak/               # шаблон realm'а и клиентов
├── docs/                   # инструкция (PDF и markdown)
├── values/                 # настройки релизов
├── images.lock.json        # образы с digest'ами
├── MANIFEST.txt            # состав, размеры, база обновления
├── SHA256SUMS              # контрольные суммы всех файлов
├── sbom/                   # SBOM и отчёт сканирования
└── CHANGELOG.md
```

Комплект, собранный для установки в ваш собственный кластер Kubernetes, может не нести `rke2/`: тогда мастер не предлагает вариант «Kubernetes из комплекта», а `MANIFEST.txt` говорит об этом строкой `kubernetes_bundled=false`.

Инструкция в комплекте

В каталоге `docs/` лежит **полная инструкция по установке именно этой версии** — в PDF и markdown, с экранами мастера, отрендеренными из того же инсталлятора, и с решениями вашей поставки, если они согласованы заранее. Отдельными файлами рядом:

- **требования к корпоративному Keycloak** — для варианта, когда вход идёт через ваш realm (`docs/…-keycloak.pdf`); их отправляют службе эксплуатации Keycloak заранее (см. «Корпоративный Keycloak» (стр. 30));
- **требования к кластеру** — для установки в ваш кластер Kubernetes (`docs/…-cluster.pdf`; см. «Требования к вашему кластеру» (стр. 24)).

Эта документация описывает установку в целом и не привязана к ревизии. Если она расходится с инструкцией в комплекте — **верна инструкция в комплекте**: она собрана вместе с инсталлятором, который вы запускаете.

Целевые ОС

ОС	Что поддерживается
Ubuntu Server 24.04 LTS	машины комплекта и узлы Kubernetes
Enterprise Linux 9: RHEL 9.7+, Rocky Linux 9.7+, AlmaLinux 9.7+, CentOS Stream 9	то же; SELinux в режиме enforcing допускается — нужные политики едут в комплекте
РЕД ОС 8 (в том числе сертифицированная редакция)	то же, на наборе пакетов Enterprise Linux 9

Архитектура — x86-64. Один архив ставится на любую из перечисленных систем: пакеты для Debian и для Enterprise Linux лежат в нём рядом, и нужный набор выбирается на самой машине по её `/etc/os-release`. Машины одной установки могут быть на разных системах из списка. Совпадение ОС проверяет мастер установки и отклоняет систему, которой в списке нет.

От машины, с которой идёт установка, нужен только системный `python3` версии 3.9–3.12 (на RHEL 9 это 3.9, на Ubuntu 24.04 — 3.12). Ни `pip`, ни `venv`, ни доступа в сеть инсталлятору не нужно: его библиотеки лежат в комплекте уже распакованными.

Как передаётся

Комплект передаётся архивом `aivo-<версия>.tar` с файлом контрольной суммы `aivo-<версия>.tar.sha256` — по ссылке на загрузку или носителем, как согласовано в договоре. Большой архив может приходить частями (`….tar.part-000`, `….tar.part-001`, …) с отдельным файлом сумм частей.

Комплект весит несколько гигабайт; точный размер по каталогам записан в `MANIFEST.txt` и печатается при проверке целостности. Под распаковку и под образы на узле Kubernetes места нужно больше — см. «Требования к площадке» (стр. 16).

Проверка целостности

Перед установкой комплект проверяется на той машине, куда его доставили:

```
# если комплект пришёл частями — сначала собрать архив
sha256sum -c aivo-<версия>.tar.parts.sha256
cat aivo-<версия>.tar.part-* > aivo-<версия>.tar

sha256sum -c aivo-<версия>.tar.sha256    # архив цел
tar xpf aivo-<версия>.tar                 # -p сохраняет права на скриптах
cd aivo-<версия> && ./scripts/verify.sh   # файлы и состав комплекта
```

`verify.sh` должен закончиться без единого `FAIL` — иначе не продолжайте установку. Это POSIX sh без зависимостей: он работает раньше, чем на машине понадобится python.

Для службы информационной безопасности

Состав комплекта и результаты его проверки лежат в нём самом, в каталоге `sbom/`:

Файл	Что это
`sbom/images/*.cdx.json`	состав каждого образа в формате CycloneDX 1.6
`sbom/<комплект>.cdx.json`	состав комплекта целиком
`sbom/scan/*.json`	результаты сканирования уязвимостей
`sbom/SCAN-REPORT.md`	сводка: находки, какие из них закрываются обновлением, принятые исключения с обоснованием и сроком
`sbom/image-users.json`	от какого пользователя работает каждый образ

Комплект не выпускается, пока в нём есть уязвимость уровня Critical с доступным исправлением, не перечисленная в сводке с обоснованием.

Образы платформы и сторонних сервисов не работают от root, и это записано в самом образе: сторонний образ без инструкции `USER` комплект несёт производным — тот же образ, зафиксированный по digest, с объявленным непривилегированным пользователем. Исключение — системные образы Kubernetes (API-сервер, etcd, сетевой плагин): им root нужен по устройству, они перечислены в отчёте отдельным разделом с обоснованием.

Свой SCA-сканер лучше запускать по этим файлам, а не по распакованному дереву: разбирая образы эвристикой, сканеры находят уязвимости в файлах, которые в контейнере не исполняются.

Требования к площадке

Требования зависят от того, что поднимается из комплекта, а что подключается ваше (см. «Выбор компонентов» (стр. 35)): каждая база, которую вы даёте сами, — это машина, которую готовить не нужно. Ниже — что нужно в каждом случае.

Если вход в платформу пойдёт через **корпоративный Keycloak**, начните не с машин, а с требований к Keycloak (раздел «Корпоративный Keycloak», стр. 30): клиент и маппер групп на стороне Keycloak по опыту согласуются около недели. Остальную подготовку это не задерживает — её можно вести параллельно.

Машины

Калькулятор считает машины под офлайн-дистрибутив DeepDive по тем же вариантам, что предлагает мастер установки `./configure`: что ставится из комплекта — в кластер или на отдельную машину, — а что берётся у заказчика. Для каждого варианта видно, что установка проверит перед стартом и какие у него ограничения. Состояние калькулятора сохраняется в адресе страницы — ссылку на конфигурацию можно отправить.

Размеры машин взяты из эталонной раскладки дистрибутива. Встроенный ClickHouse — единственная часть, которая зависит от данных заказчика: его размер — оценка по объёму сырых данных, горизонту хранения и числу активных пользователей.

Готовые конфигурации.Первые три сборка комплекта устанавливает end-to-end на чистых машинах, четвёртая — первая из них на отказоустойчивом кластере; значения — раскладка дистрибутива `main (40b7e87)`.

Конфигурация	Машины	Итого	Предоставляет заказчик
Всё в одной машине Платформа, базы данных, вход и витрина — в кластере из комплекта на одной машине. Самый простой старт.	`aivo-app`: 12 vCPU, 64 ГиБ ОЗУ, 300 ГиБ диска	1 × ВМ: 12 vCPU, 64 ГиБ ОЗУ, 300 ГиБ диска	LLM — Своя модель в контуре; Embeddings — Своя модель в контуре; TLS-сертификаты доменов — Свои сертификаты

Конфигурация	Машины	Итого	Предоставляет заказчик
Внешние базы данных PostgreSQL и Redis — на отдельных машинах, вход через ваш Keycloak, витрина данных — ваша. Для площадок, где базы держат отдельно от кластера.	`aivo-app`: 12 vCPU, 64 ГиБ ОЗУ, 300 ГиБ диска `aivo-pg`: 4 vCPU, 16 ГиБ ОЗУ, 2 048 ГиБ диска `aivo-redis`: 4 vCPU, 8 ГиБ ОЗУ, 100 ГиБ диска	3 × ВМ: 20 vCPU, 88 ГиБ ОЗУ, 2 448 ГиБ диска	Keycloak (SSO) — Свой realm на вашем Keycloak; ClickHouse (витрина данных) — Внешняя витрина, у заказчика; LLM — Своя модель в контуре; Embeddings — Своя модель в контуре; TLS-сертификаты доменов — Свои сертификаты
Корпоративный контур Вход через ваш корпоративный Keycloak без прав администратора, витрина данных и трассировка — ваши.	`aivo-app`: 12 vCPU, 64 ГиБ ОЗУ, 300 ГиБ диска	1 × ВМ: 12 vCPU, 64 ГиБ ОЗУ, 300 ГиБ диска	Keycloak (SSO) — Ваш Keycloak, только вход; Langfuse — Внешний, у заказчика; ClickHouse (витрина данных) — Внешняя витрина, у заказчика; LLM — Своя модель в контуре; Embeddings — Своя модель в контуре; TLS-сертификаты доменов — Свои сертификаты
Отказоустойчивый кластер Всё в кластере из комплекта, но на трёх управляющих и нескольких рабочих узлах: отказ одного узла не останавливает кластер.	`aivo-cp-1`: 4 vCPU, 8 ГиБ ОЗУ, 250 ГиБ диска 2 × `aivo-cp-<n>`: 4 vCPU, 8 ГиБ ОЗУ, 150 ГиБ диска 2 × `aivo-worker-<n>`: 12 vCPU, 64 ГиБ ОЗУ, 150 ГиБ диска + 550 ГиБ под тома	5 × ВМ: 36 vCPU, 152 ГиБ ОЗУ, 1 950 ГиБ диска	LLM — Своя модель в контуре; Embeddings — Своя модель в контуре; TLS-сертификаты доменов — Свои сертификаты

Эталонная раскладка — для варианта «Kubernetes и базы из комплекта», когда PostgreSQL, Redis и S3-хранилище стоят на отдельных машинах:

Машина	Роль	vCPU	ОЗУ, ГиБ	Диск, ГиБ
`aivo-app`	Kubernetes (RKE2, один узел) и всё, что выбрано «в кластере»	12	64	300
`aivo-pg`	PostgreSQL 16 + pgvector — вариант «на отдельной машине»	4	16	2048
`aivo-redis`	Redis — вариант «на отдельной машине»	4	8	100
`aivo-s3`	S3-хранилище Garage — вариант «на отдельной машине»	4	8	100

Как меняется раскладка:

- **Всё в кластер.** PostgreSQL, Redis и S3-хранилище поднимаются в Kubernetes с томами на узле, и нужна одна машина `aivo-app`. Диск под тома тогда закладывается на неё: PostgreSQL — 200 ГиБ, S3-хранилище — 100 ГиБ, ClickHouse — 200 ГиБ, NATS — 20 ГиБ, Redis — 8 ГиБ (значения по умолчанию, размер томов PostgreSQL и ClickHouse задаётся в мастере).
- **Ваши базы.** PostgreSQL, Redis или S3-хранилище подключаются ваши — машины под них не нужны.
- **Ваш кластер Kubernetes.** Не нужна и машина `aivo-app`: платформа ставится в ваш кластер релизами Helm, требования к нему — на странице «Требования к вашему кластеру» (стр. 24).
- **Отказоустойчивый кластер из комплекта.** RKE2 из комплекта ставится и на несколько машин: три управляющих узла и рабочие узлы. Управляющие узлы держат etcd и нагрузку платформы не принимают; платформа, базы и тома живут на рабочих.

Это заявка, рассчитанная на пилотный и небольшой промышленный контур. Для витрины ClickHouse из комплекта размер тома зависит от объёма ваших данных и горизонта хранения — его стоит считать отдельно.

Общие требования к машинам

- Чистая установка ОС из списка поддерживаемых (раздел «Комплект поставки», стр. 14), x86-64.
- Статические IP-адреса; все машины в одной сети или с маршрутизацией между ними.
- Синхронизированное время (chrony или systemd-timesyncd с внутренним NTP): расхождение часов ломает TLS, токены входа и проверку лицензии.
- Уникальные имена машин.
- На машине, с которой идёт установка, — `python3` 3.9–3.12. Больше инсталлятору ничего не нужно.

Kubernetes из комплекта

Если кластер ставится из комплекта (RKE2), на каждом его узле:

```
# swap выключен навсегда
swapoff -a
sed -i '/ swap /d' /etc/fstab

# хостовый firewall выключен либо открывает порты RKE2
systemctl disable --now firewalld 2>/dev/null || ufw disable 2>/dev/null || true
```

- Не меньше **120 ГиБ свободного места** под `/var/lib/rancher` — образы и тома. Мастер проверяет это до установки.
- На Enterprise Linux с SELinux в режиме enforcing политики `container-selinux` и `rke2-selinux` ставит сама установка — из комплекта, без репозиториев. На РЕД ОС 8 `container-selinux` из ОС требованию RKE2 не удовлетворяет, и установка заменяет его пакетом из комплекта (и пишет об этом в отчёте).
- Антивирус и агенты защиты не сканируют `/var/lib/rancher`, `/etc/rancher`, `/run/k3s`, `/var/log/pods`, `/var/log/containers` и каталог комплекта.

Вход в платформу публикуется через Gateway API на **Envoy Gateway** из комплекта: прокси занимает порты 80 и 443 узла, отдельного ingress-контроллера в кластере нет.

Кластер из нескольких узлов. Узел, на котором запускается установка, — первый управляющий; остальные установка присоединяет сама, по ssh. Для этого нужны учётная запись с `sudo` без пароля на всех узлах, ssh по ключу с узла установки и порты между узлами:

Порт	Протокол	Назначение	Между
9345	TCP	регистрация узлов RKE2	все узлы → управляющие
6443	TCP	API Kubernetes	все узлы → управляющие
2379, 2380, 2381	TCP	etcd	управляющие ↔ управляющие
10250	TCP	kubelet	все узлы ↔ все узлы
8472	UDP	сеть подов (VXLAN)	все узлы ↔ все узлы
9099	TCP	проверки сетевого плагина	все узлы ↔ все узлы
22	TCP	установка по ssh	узел установки → остальные узлы

Перед управляющими узлами нужен адрес регистрации (VIP, DNS-имя или балансировщик на 6443 и 9345), перед рабочими — балансировщик входящего трафика на 80 и 443. Подробный документ о подготовке узлов такого кластера кладётся в `docs/` комплекта.

> ⚠ **Осторожно**
>
> Обновление кластера из нескольких узлов `./update` не выполняет: платформу обновляет, а узлы Kubernetes оставляет на их версии.

Сеть

Доступы, которые нужно открыть заранее:

Откуда	Куда	Порт	Когда нужно
пользователи	узел Kubernetes (или балансировщик)	443/tcp	всегда: интерфейс и API
пользователи	Keycloak	443/tcp	Keycloak не из комплекта: вход идёт через его страницу
узел Kubernetes и поды	домены установки	443/tcp	всегда: на этих именах держится вход (см. ниже)
узел Kubernetes	PostgreSQL	5432/tcp	PostgreSQL на отдельной машине или ваш
узел Kubernetes	Redis	6379/tcp	Redis на отдельной машине или ваш
узел Kubernetes и поды	S3-хранилище	3900/tcp	S3-хранилище на отдельной машине
узел Kubernetes **и поды**	Keycloak	443/tcp	Keycloak не из комплекта: ключи подписи токенов
узел Kubernetes **и поды**	ClickHouse	8123, 9000/tcp	ваша витрина ClickHouse
узел Kubernetes	Oracle	1521/tcp	ассистент читает Oracle напрямую
узел Kubernetes	endpoint'ы языковой модели и векторизации	443/tcp	всегда

Откуда	Куда	Порт	Когда нужно
узел Kubernetes	Langfuse	443/tcp	трассировка в ваш Langfuse
узел Kubernetes	машины PostgreSQL, Redis, S3-хранилища	22/tcp	только на время установки, если машины ставит инсталлятор

К компонентам, поднятым в кластере, отдельных доступов не нужно: их трафик не выходит за пределы узла. Исходящий доступ в интернет платформе не нужен — кроме вариантов, которые сами в него ходят: внешний провайдер моделей по API и сертификаты Let’s Encrypt.

> ⚠ **Порт 3900 закрывает политика сети**
>
> За портом 3900 машины S3-хранилища — вложения и выгрузки платформы. Установка закрывает его хостовым firewall’ом, если тот включён, и прямо говорит, когда закрыть нечем. Если firewall на машинах выключен политикой организации, ограничить доступ к 3900, 5432 и 6379 одним узлом Kubernetes и его подами — часть подготовки площадки.

Доменные имена и TLS

Платформа занимает **одно имя** — например, `aivo.example.ru`: интерфейс и API, один вход и одна сессия. Если Keycloak поднимается из комплекта, ему нужно либо второе имя (`keycloak.example.ru`), либо путь на том же домене: `https://aivo.example.ru/auth` — тогда запись DNS, сертификат и TLS-секрет нужны одни. С корпоративным Keycloak второго имени нет — его realm живёт на своём домене.

Имя должно указывать на узел Kubernetes (или балансировщик перед ним) во внутреннем DNS.

> ⚠ **Имена должны работать и изнутри**
>
> Домены установки обязаны резолвиться и отвечать **с самого узла Kubernetes и из подов**, а не только с рабочих станций: через них идёт вход. Типовая ловушка — домен указывает на публичный адрес того же узла, а шлюз не разворачивает трафик обратно (NAT без hairpin). Снаружи всё работает, изнутри соединение висит до таймаута. Лечится hairpin-правилом на шлюзе либо записями в `/etc/hosts` узла и во внутреннем DNS кластера на внутренний адрес входа. Установка проверяет обе стороны и без этого не начинается.

Откуда берутся сертификаты — вопрос мастера, вариантов два:

- **Свои сертификаты** (по умолчанию). Сертификат и ключ вы передаёте мастеру — путём к файлу или вставив PEM, — и секреты в кластере создаёт установка. Сертификат нужен один на все домены установки (годится wildcard), промежуточные сертификаты дописываются

следом за ним, ключ — без пароля. Мастер сразу проверяет, что ключ подходит к сертификату, SAN покрывает все домены и срок не истёк. Продление — ручное: продлённый сертификат кладётся поверх того же файла, `./install` переносит его в секреты. В закрытом контуре это единственный вариант.

- **Let's Encrypt.** Сертификаты выпускает и продлевает cert-manager из комплекта. Нужны: доступ с узла к `acme-v02.api.letsencrypt.org`, домены в **публичном** DNS и порт 80 этих имён, открытый снаружи (проверка HTTP-01). Лимит Let's Encrypt — пять неудачных проверок на домен в час, поэтому проходимость удобно сначала проверить на staging-каталоге, который мастер предлагает на выбор.

Если Keycloak, Oracle или endpoint'ы моделей используют сертификаты корпоративного удостоверяющего центра, его корневой сертификат указывается в мастере. Проверка TLS не отключается ни в каком режиме: если цепочка не сходится, установка останавливается и говорит, на чём.

Языковая модель и векторизация

Ассистенту нужны две модели: **языковая** (ответы, вызовы инструментов) и **модель векторизации** (поиск по базе знаний). Для каждой мастер предлагает:

Вариант	Что это	Где доступен
Внешний провайдер моделей (нативный API)	Коммерческий провайдер по его собственному API; спрашивается только ключ и имя модели	только с доступом к API провайдера из контура
Внешний провайдер моделей (OpenAI-совместимый API)	Коммерческий провайдер по OpenAI-совместимому API; спрашивается только ключ и имя модели	только с доступом к API провайдера из контура
Своя модель в контуре	Ваш endpoint с OpenAI-совместимым API в контуре: адрес, ключ, имя модели	везде, в том числе в закрытом контуре
Настроить позже	Модель подключается после установки в интерфейсе платформы	только у языковой модели

Модель в комплект не входит: в закрытом контуре её разворачиваете вы, и подготовить endpoint нужно заранее. Мастер проверяет его теми же запросами, какими будет ходить платформа:

1. **Ответ на запрос:** `POST <адрес>/chat/completions` с выбранной моделью отвечает.
2. **Вызов инструментов:** запрос с блоком `tools` (в формате OpenAI-совместимого API) возвращает `tool_calls`. Без этого ассистент неработоспособен. Для моделей под vLLM на

сервинге должен быть включён парсер вызовов инструментов (например, `--enable-auto-tool-choice --tool-call-parser hermes`) — если `tool_calls` не приходят, это настройка сервинга, а не платформы.

3. **Потоковый ответ:** `"stream": true` возвращает SSE-поток.
4. **Векторизация:** endpoint возвращает вектор; зафиксируйте имя модели и размерность вектора.
5. **Лимиты:** размер контекста модели и ограничения на число запросов — зафиксируйте их.

> ⚠ **Размерность вектора меняется только на пустой базе знаний**
>
> По размерности модели векторизации платформа на первом старте строит колонку векторов базы знаний. После того как база знаний наполнена, смена модели означает переиндексацию всего загруженного. Поэтому модель векторизации отложить нельзя — у неё нет варианта «настроить позже».

Если языковая модель отложена, ассистент не отвечает, пока модель не подключат в панели администратора (раздел «Панель администратора установки», стр. 72).

Права и учётные записи

Настройка (`./configure`) идёт обычной учётной записью инженера и root не требует. Root нужен только шагам, которые меняют саму машину:

- **узел Kubernetes из комплекта** — импорт образов и установка RKE2 (бинарники, юнит systemd, конфигурация узла, политики SELinux);
- **машины PostgreSQL, Redis и S3-хранилища** — установка пакетов, юнита systemd и конфигурации. Инсталлятор заходит туда по ssh пользователем `root` либо пользователем с `sudo` без пароля, вход только по ключу;
- **ваш кластер Kubernetes** — root не нужен вовсе: установке хватает kubeconfig с ролью в namespace.

Заявку на права удобно подать одним заходом по плану: `./install --privileges` (то же — на последнем экране мастера) печатает «машина → шаг → что именно делается от root» формулировками, которые можно вставить в заявку.

Требования к вашему кластеру

Эта страница — для варианта, когда платформа ставится **в ваш кластер Kubernetes**: кластер разворачиваете и обслуживаете вы, комплект ставит в него релизы Helm и своего Kubernetes не везёт. Её удобно передать службе, которая эксплуатирует кластер; в комплекте то же самое лежит отдельным документом (`docs/…-cluster.pdf`) под вашу версию.

Всё, что здесь перечислено, установка проверяет сама до первого релиза и без этого не начинается.

Коротко

Что	Требование
Версия Kubernetes	1.31–1.37 (прогон поставки — на 1.35)
Namespace	один, выделенный, по умолчанию `aivo`
Права в нём	роль с полным списком глаголов, включая `update` и `patch`
Вход	ваш ingress-контроллер с объявленным `IngressClass` **или** ваш контроллер Gateway API с принятым `GatewayClass`
Pod Security Admission	совместим с уровнем `restricted`
Реестр образов	доступен узлам кластера; образы в него раскладываете вы
Тома	ваш `StorageClass` с `ReadWriteOnce` — если базы и хранилища поднимаются в кластере
Root на машине установки	не нужен: установка работает kubeconfig'ом
Кластерный уровень	только чтение: `ingressclasses`, `storageclasses`, `namespaces` — установке; метаданные namespace'а `kube-system` — платформе

Поставка живёт в одном namespace'е и за его пределы только смотрит: ни CRD, ни admission-webhook'ов, ни прав на чужие namespace'ы она не создаёт и не просит.

Версия

Поддерживается диапазон **1.31–1.37**: нижняя граница подтверждена действующими установками, верхняя — прогоном на стенде. Версию спрашивает сама установка и на выходе из диапазона останавливается: между минорными версиями меняются `apiVersion` объектов чартов и поведение Pod Security Admission.

`kubectl` комплекта — 1.37; против сервера 1.31 он формально выходит за поддерживаемое расхождение версий. На установку это не влияет, но работать своим `kubectl` под версию кластера никто не мешает.

Namespace и права

Роль установки в её namespace’е:

Объекты	Глаголы
`deployments`, `statefulsets`, `services`, `secrets`, `configmaps`, `ingresses`, `persistentvolumeclaims`, `serviceaccounts`, `roles`, `rolebindings`, `jobs`, `pods`, `poddisruptionbudgets`, `events`	`create`, `get`, `list`, `watch`, `update`, `patch`, `delete`
`replicasets`	`get`, `list`, `watch`
`pods/log`	`get`
`pods/exec`, `pods/portforward`	`create`
`gateways`, `httproutes` — только при входе через Gateway API	`create`, `get`, `list`, `watch`, `update`, `patch`, `delete`

И на уровне кластера — только чтение:

Объекты	Глаголы	Кому
`ingressclasses`, `storageclasses`	`get`, `list`	установке
`namespaces`	`get`	установке
`gatewayclasses` — только при входе через Gateway API	`get`, `list`	установке
`namespaces/kube-system`	`get`	сервису `api` платформы

Почему так:

- **`update` и `patch` — не запас.** `helm upgrade --install` обновляет секрет релиза уже на первой установке, а повторный запуск и обновление правят существующие объекты. Роль без них даёт установку, которая падает на середине.
- **`pods/exec` и `pods/portforward`** — так заводится realm Keycloak из комплекта и регистрируются модели в базе платформы, когда PostgreSQL в кластере.
- **`namespaces/kube-system`** читает не установка, а платформа своей учётной записью — и ровно метаданные одного объекта: UID этого namespace'а, к которому привязана лицензия. Содержимое `kube-system` ей недоступно.

Объекты кластерного уровня (роль установки и `ClusterRole` + `ClusterRoleBinding` для лицензии) создаёте вы — готовый манифест лежит в документе комплекта. Установка их не создаёт: прав на это у неё нет. Проверить объекты для лицензии до установки она тоже не может, это делает `./verify` строкой «Платформа читает UID кластера» — после первого входа в интерфейс.

Установке передаётся **kubeconfig с этой ролью**; если в нём несколько контекстов, мастер спросит нужный.

Вход: Ingress или Gateway API

Комплект своего контроллера в ваш кластер не ставит. Вариантов публикации два.

Объекты Ingress на вашем контроллере (по умолчанию). Подойдёт любой контроллер, который выполняет условия ниже; значения комплекта написаны без аннотаций какого-либо одного контроллера. Рекомендуемый — Traefik: на нём поставка проверена. ingress-nginx работает, но проект выведен из эксплуатации и обновлений безопасности больше не получает — установка назовёт это отдельной строкой в отчёте и продолжит; решение за вами и вашей службой безопасности.

Gateway API. Установка заводит в своём namespace'е `Gateway` на названном вами `GatewayClass` (с TLS на секретах установки) и маршруты `HTTPRoute`. Нужен контроллер Gateway API v1 стандартного канала **не ниже v1.2**: маршруту платформы нужно поле `timeouts` со значением `request: 0s`, иначе поток ответа ассистента рвётся на умолчании контроллера.

Что требуется от контроллера в обоих случаях:

1. **Тело запроса — не меньше 30 МБ.** Столько весит предельный файл, который пользователь загружает в диалог; собственный потолок JSON-тела у платформы — 5 МБ (`api.maxHttpBodyBytes`). Контроллер не должен резать тело раньше платформы, иначе вместо внятной ошибки пользователь получает `413`. У ingress-nginx по умолчанию — 1 МиБ (`proxy-body-size`).

2. **Ожидание ответа бэкенда — не меньше 300 секунд.** Долгие запросы `/graphql` отвечают минутами.
3. **Чтение запроса с телом — не меньше 300 секунд**: загрузка 30 МБ по узкому каналу в стандартные 60 секунд не укладывается.
4. **Только HTTPS** либо редирект с HTTP: cookie сессии помечена `Secure`.

Сертификаты доменов в кластере лежат секретами `kubernetes.io/tls` в namespace'е установки. Их создаёт установка из сертификата и ключа, переданных мастеру, — либо вы заранее, до запуска мастера.

Pod Security Admission

Поды поставки рассчитаны на уровень **`restricted`** и послаблений не требуют: непривилегированный пользователь, `allowPrivilegeEscalation: false`, `drop: ["ALL"]`, `seccompProfile: RuntimeDefault`. Ни одному контейнеру не нужны root, `hostPath`, `hostNetwork` или привилегированный режим.

Перед установкой в ваш namespace уходит пробный под с тем же `securityContext` — с `--dry-run=server`: ваши admission-политики (PSA, Kyverno, Gatekeeper) отрабатывают полностью, а объект не создаётся.

Образы

Образы лежат в комплекте архивами; в ваш реестр их кладёт скрипт комплекта:

```
./images/import.sh --registry registry.corp.local/aivo
```

Он сохраняет путь образа и меняет только адрес реестра (`ghcr.io/aivoagent/agent/backend:<версия>` → `registry.corp.local/aivo/aivoagent/agent/backend:<версия>`). Тот же адрес вы называете мастеру, и установка подставит его в файлы значений. Скрипту нужны любой клиент реестра (`skopeo`, `crane`, `docker` или `podman`) и `zstd`. Если реестр настроен зеркалом на уровне containerd узлов, адрес называть не нужно.

Ресурсы и тома

Сколько поставка просит у планировщика, зависит от выбранных компонентов; точная таблица заявок `requests` — в документе комплекта. Квоту namespace'а, если она есть, ставьте по ней.

Если PostgreSQL, Redis, NATS, S3-хранилище или ClickHouse поднимаются в кластере, установка заводит им тома (`PersistentVolumeClaim`) вашим классом хранения. Нужен класс с `ReadWriteOnce`; сетевой он или локальный — решаете вы. С локальными томами под с состоянием привязан к своему узлу.

PostgreSQL в кластере — одна реплика: отказ узла с её томом останавливает платформу до возврата тома или восстановления из копии. Дамп всех баз комплект снимает перед каждым обновлением; регулярные копии по расписанию — на вашей стороне (см. «Данные, бэкапы и retention» (стр. 64)).

Сеть

Из сети подов должны быть доступны все системы, которые вы даёте платформе: машины поставки (PostgreSQL 5432, S3-хранилище 3900, Redis 6379), ваш Keycloak, витрина данных и endpoint’ы моделей. Установка проверяет это изнутри кластера — короткоживущим подом в вашем namespace’е: иначе правило в сети выглядело бы как поломка платформы. Исходящий доступ в интернет кластеру не нужен.

Чего поставка не просит

- `StorageClass` и `PersistentVolume` — только `PersistentVolumeClaim` в своём namespace’е;
- `CustomResourceDefinition`, admission-webhook’и;
- ничего кластерного на запись;
- `hostPath`, `hostNetwork`, `NET_ADMIN`, привилегированные контейнеры;
- сервисы `NodePort` и `LoadBalancer` — весь вход через ваш контроллер;
- права на узлы и доступ к содержимому `kube-system`;
- исходящий доступ в интернет;
- root на машине, с которой идёт установка.

Чек-лист для передачи

- ☐ Версия кластера — в диапазоне 1.31–1.37.
- ☐ Создан namespace установки.
- ☐ Выдан kubeconfig с ролью из таблицы — со всеми глаголами, включая `update` и `patch`; при нескольких контекстах назван нужный.
- ☐ Применены объекты для лицензии (`ClusterRole` + `ClusterRoleBinding` на `namespaces/kube-system`).

- [ ] Назван `ingressClassName` либо `gatewayClassName`; подтверждены лимиты контроллера (30 МБ, 300 секунд, только HTTPS).
- [ ] Назван уровень Pod Security Admission namespace'а.
- [ ] Образы разложены в реестр и назван его адрес (или подтверждено зеркало containerd).
- [ ] Назван `StorageClass`, и на нём хватает места под тома — если что-то поднимается в кластере.
- [ ] Подтверждена доступность ваших систем из сети подов.
- [ ] Готовы сертификат и ключ доменов установки (или созданы TLS-секреты).

Корпоративный Keycloak

Эта страница — для варианта, когда вход в платформу идёт через **ваш существующий realm** (вариант Keycloak «Корпоративный» в мастере, см. «Выбор компонентов» (стр. 37)). Вы остаётесь его единственным владельцем, а платформа получает ровно два права. Страницу удобно отправить службе эксплуатации Keycloak примерно за неделю до работ — столько по опыту занимает согласование клиента и маппера. В комплекте то же самое лежит отдельным документом (`docs/…-keycloak.pdf`).

Если для платформы на вашем сервере заводится **отдельный realm** (вариант «Realm установки на корпоративном сервере»), требования другие: там нужны права Admin API и сервисная учётная запись, а realm и клиенты заводятся по шаблону из каталога `keycloak/` комплекта.

Коротко: два права и ни одного администраторского

Платформе нужно:

1. аутентифицировать пользователя — стандартный OIDC-вход;
2. видеть в его токене список групп, в которых он состоит.

Роли `realm-management` (в том числе `view-users`) выдавать не нужно, и установка их не запрашивает. Ни при установке, ни в работе платформа не выполняет ни одного вызова Admin API: не читает справочник пользователей, не создаёт учётные записи, не читает список групп realm’а. Сервисная учётная запись клиенту тоже не нужна.

Собрать нужно пять вещей — каждая из них ответ мастера установки:

#	Что
1	Confidential-клиент для oauth2-proxy: Client ID, Client Secret, redirect URI
2	Claim `groups` в ID-токене этого клиента — с фильтрацией
3	Публичный адрес realm’а и, если отличается, внутренний
4	Путь группы, чьи участники администрируют установку
5	Пути групп, которым соответствуют организации в платформе

Клиент для oauth2-proxy

Единственная точка входа в платформу — oauth2-proxy, который выполняет OIDC-вход в вашем realm. Для него нужен отдельный клиент:

Параметр	Значение
Client ID	по вашим правилам именования
Client authentication	**on** (confidential, с секретом)
Standard flow	включён
Direct access grants	не нужен
Service accounts	**не нужен**, оставьте выключенным
Valid redirect URIs	`https://<домен платформы>/oauth2/callback`
Web origins	`https://<домен платформы>`

Передать инженеру установки: **Client ID** и **Client Secret**. Адрес возврата указывается точный: * Keycloak понимает только в конце строки, и при несовпадении вход заканчивается ошибкой `Invalid parameter: redirect_uri`.

Группы в токене — обязательно с фильтрацией

В какую организацию попадает сотрудник и какие у него права, платформа определяет **по списку его групп в ID-токене**. Клиенту нужен маппер, который кладёт в токен claim `groups` с **полными путями** групп (`/Corp/Sales/Analysts`, а не `Analysts`).

Список групп должен быть отфильтрован — только группы, имеющие отношение к платформе. У сотрудника, который состоит в сотнях групп AD, нефильтрованный токен перестаёт проходить через вход, и войти он не может вовсе. Платформа в этом случае показывает ошибку и **не** впускает с урезанным списком: молча потерянная группа означала бы молча потерянные права.

Способы фильтрации, в порядке предпочтения:

1. **Через роли клиента.** Нужные группы связываются с ролями клиента oauth2-proxy, а в токен добавляется маппер *User Client Role* с *Token Claim Name* `groups`. Фильтрация штатными средствами, без расширений.
2. **Ограничить импорт из AD.** В `group-ldap-mapper` федерации указать *Groups DN* на OU с нужными группами или задать *Custom LDAP Filter*.

3. **Фильтрующий маппер** (script mapper или *Filtered Group Membership Mapper*) — только если первые два не подходят.

Штатный маппер *Group Membership* без фильтрации подходит, только если групп у сотрудников единицы.

Членство в группах для входа не требуется. Сотрудник вне всех групп входит и видит экран «вы не состоите ни в одной организации». Заводить служебную группу «все сотрудники» ради того, чтобы claim был в токене, не нужно.

Всё, что платформа читает, она читает из самого ID-токена — к `userinfo` не обращается. Обязателен только `sub`: по нему платформа узнаёт человека, поэтому вход работает и у учётной записи без e-mail. `groups` нужен настолько, насколько нужны организации. E-mail, логин и отображаемое имя — по вашему выбору.

Адреса realm’а

Адрес	Что это	Зачем платформе
публичный	по нему в realm ходит браузер сотрудника	по нему проверяется `iss` каждого токена
внутренний	по нему realm виден из кластера платформы	по нему забираются ключи подписи (JWKS)

Публичный адрес обязателен; внутренний — только если он отличается от публичного или публичный из кластера недоступен. Keycloak подписывает токены тем issuer’ом, который задан в его hostname-конфигурации, поэтому внутренний адрес вместо публичного ломает проверку каждого токена.

Если сертификат realm’а выписан корпоративным удостоверяющим центром, приложите **корневой сертификат УЦ в PEM**. Без него oauth2-proxy не проходит OIDC discovery и не стартует; проверка TLS при этом не отключается.

Группы, которые использует платформа

Группы подразделений. Для каждой организации, которая должна появиться в платформе, — путь группы, членство в которой даёт доступ в неё:

Путь группы	Организация в платформе	Роль
`/Corp/Sales`	Продажи	администратор организации

Путь группы	Организация в платформе	Роль
`/Corp/Support`	Поддержка	участник

Право работать в платформе = попадание хотя бы в одно такое соответствие; отдельного списка «кому разрешён вход» нет. Соответствие — по префиксу пути и **с учётом регистра**: правило для `/Corp/Sales` покрывает `/Corp/Sales/Analysts`, но не `/Corp/SalesOps` и не `/corp/sales`.

Группа администраторов установки. Одна группа инженеров, выполняющих первичную настройку. Её участники — администраторы установки с первого входа: это единственный способ войти в свежую установку, где ещё нет ни одной организации. После первичной настройки права переносятся в платформу.

Время жизни токена

Менять ничего не нужно, но значение нужно знать: **Access Token Lifespan** вашего realm'а (по умолчанию 5 минут). Интервал обновления сессии на стороне платформы ставится меньше него — иначе вход перестаёт работать через несколько минут после успешного входа.

Что платформа делает с данными сотрудников

- Сохраняет **только тех, кто хотя бы раз вошёл**: идентификатор (`sub`) и — по вашему выбору — логин, e-mail, отображаемое имя. Состав полей можно сократить до одного идентификатора.
- Сохраняет **пути групп**, приходившие в токенах: так администратор установки видит, из каких групп реально приходят люди.
- **Не создаёт и не изменяет** учётные записи в вашем Keycloak и не читает справочник пользователей.

Ограничения, которые из этого следуют

- **Справочник пользователей в платформе неполный**: в нём только те, кто входил. Пригласить из платформы сотрудника, ни разу не входившего, нельзя.
- **Отзыв доступа действует не мгновенно.** Вывод из группы вступает в силу при следующем входе или обновлении сессии — обычно минуты. Немедленная блокировка — отключение учётной записи у вас и блокировка пользователя в платформе.
- **Лицензионное место занимается при первом входе** и освобождается только явной блокировкой пользователя в платформе.

- **Фоновые задачи** (автопилоты, обновление дашбордов) выполняются от имени создавшего их сотрудника по последнему известному членству и прекращаются при его следующем входе после вывода из группы.

Чек-лист для передачи

- ☐ Создан confidential-клиент для oauth2-proxy; переданы Client ID и Client Secret.
- ☐ Зарегистрирован redirect URI `https://<домен платформы>/oauth2/callback`.
- ☐ В ID-токене клиента есть claim `groups` с полными путями, и он отфильтрован; назван способ.
- ☐ Передан публичный адрес realm'а и, если отличается, внутренний.
- ☐ Передан корневой сертификат УЦ в PEM, если УЦ не публичный.
- ☐ Передан список «группа → организация → роль».
- ☐ Передан путь группы администраторов установки.
- ☐ Сообщено значение Access Token Lifespan.
- ☐ Проверено на сотруднике с **наибольшим** числом групп: вход проходит.
- ☐ Проверено на сотруднике **вне всех** групп: вход заканчивается сообщением «нет доступной организации», а не ошибкой.

Выбор компонентов

Комплект несёт полный набор компонентов, а что из них поднимать, решается в мастере `./configure`. Мастер идёт по компонентам в том порядке, в каком они перечислены ниже, и для каждого спрашивает вариант и реквизиты этого варианта. Вопросы мастер строит из матрицы компонентов — файла `components.yaml` в корне комплекта, — по ней же `./install` строит план, поэтому мастер и установка разойтись не могут.

Варианты

Вариант	Что означает
`in-cluster`	Поднимается из комплекта в Kubernetes: чарт, образы, том
`vm`	Ставится из комплекта на отдельную машину systemd-сервисом. Поставить его может инсталлятор сам, по ssh, или инженер — по напечатанной инструкции
`external`	Подключается ваш, уже существующий: мастер спрашивает реквизиты и проверяет их
`disabled`	Не используется

Не у каждого компонента есть все варианты. Первым мастер предлагает то, что комплект умеет поставить сам: подключить свой PostgreSQL, Redis, Keycloak или ClickHouse — осознанное решение, а не умолчание.

Пароли и ключи придумывать не нужно. Всё, что можно сгенерировать, — пароли ролей PostgreSQL, пароли Redis, ключи S3 и ClickHouse, ключ шифрования cookie — инсталлятор генерирует сам. Если значение задано корпоративной политикой, на экране компонента есть «Задать свои значения». Посмотреть получившееся можно в любой момент: `./aivo-installer secrets show` (с `--reveal` — значения целиком).

Проверки площадки. У каждого варианта, который подключает ваше, есть проверки: мастер действительно подключается к указанному и показывает результат построчно. Обязательные проверки останавливают установку, если не прошли; справочные — только предупреждают. Ниже они перечислены у каждого компонента.

Kubernetes

Вариант	Что это
`vm` — RKE2 из комплекта (по умолчанию)	Kubernetes ставится установкой на ваши машины: один узел или отказоустойчивый кластер из трёх управляющих узлов, при желании с рабочими узлами
`external` — кластер заказчика	Готовый кластер на площадке; из комплекта ставятся только релизы Helm

RKE2 из комплекта. Проверяется: остальные узлы доступны по ssh, ОС узла совпадает с поддерживаемой, на `/var/lib/rancher` не меньше 120 ГиБ, ресурсов узла хватает на выбранные компоненты (справочно).

Кластер заказчика. Мастер спрашивает kubeconfig, контекст, namespace, класс ingress или Gateway API и адрес реестра образов. Проверяется: версия кластера в поддерживаемом диапазоне, права роли, класс ingress или `GatewayClass`, совместимость подов с Pod Security Admission, наличие образов в реестре. Образы в реестр кладёте вы (`images/import.sh`), а ingress-контроллер, класс хранения и сам кластер остаются на вашем обслуживании. Подробно — «Требования к вашему кластеру» (стр. 24).

Класс хранения

Тома для компонентов, поднятых в кластере.

Вариант	Что это
`in-cluster` (по умолчанию)	local-path-provisioner из комплекта: тома — каталоги на узле. Под с таким томом на другой узел не переедет
`external`	Класс хранения вашего кластера. Проверяется, что он существует и умеет `ReadWriteOnce`
`disabled`	Томов в кластере нет вовсе: всё состояние — на отдельных машинах или в ваших сервисах

Вариант `disabled` возможен, только если ни один компонент не просит том: PostgreSQL, NATS с файловым хранилищем, S3-хранилище, ClickHouse, Redis и Keycloak в кластере с ним несовместимы. Зато кластер остаётся без состояния: потеря узла — это переустановка, а не потеря данных.

Платформа

Ставится всегда из комплекта. Мастер спрашивает идентификатор и название установки, домен и — если лицензия уже получена — путь к её файлу. Проверяется: домены установки резолвятся с узла и соединение с ними не виснет, файл лицензии разбирается как лицензия.

Вход в платформу (oauth2-proxy)

Ставится всегда из комплекта: единственная точка входа снаружи. Мастер спрашивает имя TLS-секрета домена и интервал обновления сессии; ключ шифрования cookie генерирует сам.

Keycloak

Вариант отвечает на один вопрос — **чей Keycloak**, — и вместе с ним решается, откуда установка берёт организации, роли и справочник людей.

Вариант в мастере	Чей realm	Что установка в нём делает	Откуда организации	Кто заводит пользователей
Внутренний: поднимается из комплекта (`in-cluster`, по умолчанию)	установки	всё: realm, клиенты, первая организация, первый пользователь	организации Keycloak	администратор установки в консоли Keycloak
Realm установки на корпоративном сервере (`external`)	установки, сервер ваш	ничего: realm и клиентов заводит администратор SSO по шаблону из комплекта	организации Keycloak	администратор realm'а
Корпоративный: только вход и группы, без Admin API (`corporate`)	ваш	ничего, ни одного вызова Admin API	правила «группа → организация → роль» в платформе	только ваша служба эксплуатации Keycloak

Продукт при этом один и тот же: интерфейс, API и модель прав от выбора не меняются.

Внутренний. Нужны домен Keycloak (или путь `/auth` на домене платформы) и ещё около 2 ГиБ памяти на узле; база `keycloak` заводится в PostgreSQL установки. Проверяется: realm, созданный установкой, отдаёт OIDC discovery.

Realm установки на корпоративном сервере. Заранее, до установки, администратор вашего SSO заводит realm по шаблону из каталога `keycloak/` комплекта — с включёнными Organizations, клиентом платформы (confidential, с сервисной учётной записью) и клиентом oauth2-proxy — и передаёт их реквизиты. Проверяется: сертификат issuer'а, OIDC discovery, ключи подписи, получение токена клиентом платформы, роли `realm-management` у его сервисной учётной записи (справочно).

Корпоративный. Мастер спрашивает адрес realm'а, клиент oauth2-proxy, группу администраторов установки, при необходимости внутренний адрес ключей подписи и корневой сертификат УЦ. Что подготовить на стороне Keycloak — «Корпоративный Keycloak» (стр. 30). Проверяется: сертификат issuer'а, OIDC discovery, ключи подписи (и по внутреннему адресу), что клиент oauth2-proxy существует и адрес возврата в нём зарегистрирован; справочно — состав claim'а `groups` и срок жизни токена против интервала обновления сессии. Ограничения варианта: пользователей и группы заводит только ваша служба эксплуатации Keycloak; организации и правила доступа пишутся в панели администратора после установки.

PostgreSQL

Основная база: настройки платформы, организации, диалоги, состояние ассистента, база знаний; база сервиса `identity`; база Keycloak, если он из комплекта.

Вариант	Что это
`in-cluster` (по умолчанию)	PostgreSQL 16 + pgvector из комплекта в кластере, одна реплика и один том (200 ГиБ по умолчанию)
`vm`	PostgreSQL 16 + pgvector systemd-сервисом на отдельной машине, из офлайн-пакетов комплекта
`external`	Ваш PostgreSQL

В кластере. Переключения при отказе узла нет. Резервное копирование не настраивается — см. «Данные, бэкапы и retention» (стр. 64).

На отдельной машине. Базы (`aivo`, `aivo_checkpointer`, `aivo_identity` и `keycloak`, если Keycloak из комплекта) и роли заводит скрипт установки. Проверяется: машина отвечает по ssh и `sudo` не спрашивает пароль, узел приложения достаёт до неё маршрутизируемым адресом, ОС совпадает с той, под которую собраны пакеты, роли входят своими паролями, база `identity` принимает роль платформы.

Ваш. Нужны: версия не ниже 14, база платформы и база сервиса `identity` (или право их создать), расширение `vector` и желательно `pg_trgm`. Проверяется: соединение, версия, базы, права

создавать и удалять таблицы, расширение `vector` (обязательно: без него поиск по базе знаний работает только по словам), `pg_trgm` (справочно: без него текстовый поиск идёт полным сканированием). Расширение `vector` не входит в доверенные, и роль платформы на чужом сервере часто не может его создать — тогда его ставит администратор СУБД: `CREATE EXTENSION vector;` в базе платформы.

Redis

Кэш платформы и сессии входа (базы 1 и 3).

Вариант	Что это
`in-cluster` (по умолчанию)	Valkey (сервер протокола Redis, форк Redis 7.2 под лицензией BSD) из комплекта в кластере, том 8 ГиБ
`vm`	Redis systemd-сервисом на отдельной машине, из пакетов ОС в комплекте (7.x на Ubuntu, 6.2 на Enterprise Linux 9)
`external`	Ваш Redis

Ваш. Пароль необязателен: пустое поле означает «подключаться без пароля». Проверяется: `PING` с этим паролем, доступность баз 1 и 3, политика вытеснения не мешает сессиям (справочно).

NATS JetStream

Шина событий платформы и поток ответа ассистента в интерфейс.

Вариант	Что это
`in-cluster` (по умолчанию)	NATS из комплекта с файловым хранилищем JetStream на отдельном томе (20 ГиБ)
`in-cluster-memory`	NATS из комплекта без тома: стримы только в памяти и не переживают перезапуск пода. Платформе этого достаточно, стороннему потребителю той же шины — нет
`external`	Ваш NATS; JetStream обязателен. Проверяется: брокер отвечает, JetStream включён

S3-совместимое хранилище

Вложения диалогов и выгрузки (бакет `aivo-files`).

Вариант	Что это
`in-cluster` (по умолчанию)	Garage из комплекта в кластере, том 100 ГиБ
`vm`	Garage systemd-сервисом на отдельной машине — статический бинарь, от ОС не зависит. Бакет и ключи заводит скрипт установки; административный порт слушает только локальный интерфейс
`external`	Ваше S3-хранилище: endpoint, регион, ключи, бакет. Проверяется: пробный объект записывается, читается и удаляется; адресация в стиле path работает (справочно)

Langfuse

Трассировка работы ассистента: промпты, вызовы инструментов, длительность шагов, стоимость.

Вариант	Что это
`disabled` (по умолчанию)	Без трассировки. Платформа работает полностью, недоступен только разбор поведения ассистента по шагам
`external`	Ваш Langfuse: адрес и пара ключей проекта. Проверяется: API отвечает с этими ключами

Из комплекта Langfuse не ставится.

ClickHouse

Витрина бизнес-данных. Не единственный возможный источник: платформа читает и Oracle, и PostgreSQL, подключённые в интерфейсе.

Вариант	Что это
`in-cluster` (по умолчанию)	ClickHouse из комплекта: один экземпляр со своим томом (200 ГиБ по умолчанию), пустая база `aivo` и пользователь `aivo`
`external`	Ваша витрина: адрес, пользователь, пароль, база. Проверяется: `/ping`, запрос с этими реквизитами, наличие таблиц в базе (справочно)

Вариант	Что это
`disabled`	Без витрины. Ассистент отвечает по базе знаний, загруженным файлам и источникам, подключённым в интерфейсе

Витрину, названную в мастере, установка сама заводит источником данных и привязывает к организациям установки. **Загрузку данных комплект не выполняет и не настраивает ни в одном варианте**: ClickHouse из комплекта после установки пуст. У него одна реплика без переезда на другой узел, резервное копирование не настраивается.

Языковая модель

Вариант в мастере	Что это
Внешний провайдер моделей (нативный API)	Коммерческий провайдер по его собственному API; адрес известен заранее — нужен ключ и имя модели
Внешний провайдер моделей (OpenAI-совместимый API)	Коммерческий провайдер по OpenAI-совместимому API; адрес известен заранее — нужен ключ и имя модели
Своя модель в контуре (по умолчанию)	Ваш endpoint с OpenAI-совместимым API: адрес, ключ, имя модели
Настроить позже	Модель подключается в интерфейсе после установки

Имя модели вводится любое: перечисленные рядом с полем — проверенные на момент выпуска комплекта, а не единственно допустимые. Проверяется: модель отвечает, возвращает вызов инструмента и отвечает после его результата, отдаёт ответ потоком (справочно). Выбранную модель установка сама заводит в реестре моделей платформы и делает моделью по умолчанию.

Внешним провайдерам нужен доступ к API провайдера из контура — в закрытом контуре эти варианты недоступны. «Настроить позже» оставляет ассистента без модели, пока её не подключат в панели администратора (раздел «Панель администратора установки», стр. 72).

Модель векторизации

Вариант в мастере	Что это
Внешний провайдер моделей (нативный API), **Внешний провайдер моделей (OpenAI-совместимый API)**	Модели векторизации провайдера по API; размерность известна по модели

Вариант в мастере	Что это
Своя модель в контуре (по умолчанию)	Ваш endpoint с OpenAI-совместимым API; размерность вектора задаётся вручную

Отложить этот выбор нельзя: экрана для модели векторизации в платформе нет, а по её размерности строится колонка векторов базы знаний. Проверяется: endpoint возвращает вектор заявленной размерности; один и тот же текст даёт один и тот же вектор (справочно).

TLS-сертификаты доменов

Вариант	Что это
`manual` (по умолчанию)	Ваши сертификаты: файлом или PEM в мастере, либо секреты, заранее созданные в кластере. Единственный вариант в закрытом контуре
`letsencrypt`	Сертификаты выпускает и продлевает cert-manager из комплекта

Требования к обоим вариантам и что проверяется — в разделе «Доменные имена и TLS» (раздел «Требования к площадке», стр. 21).

Публикация доменов

Вариант	Где	Что это
`envoy-gateway` (по умолчанию для RKE2 из комплекта)	только кластер из комплекта	Gateway API на Envoy Gateway из комплекта; прокси на портах 80/443 узла
`cluster` (по умолчанию для вашего кластера)	только ваш кластер	Объекты Ingress на вашем ingress-контроллере
`gateway`	только ваш кластер	Gateway на вашем `GatewayClass`; контроллер Gateway API v1 не ниже v1.2

Установка

Комплект ставит инсталлятор из этого же комплекта. Весь путь — три команды:

```
./configure    # мастер: что откуда берётся, реквизиты, проверки площадки, план
./install      # выполнение плана по шагам, с проверкой после каждого
./verify       # те же проверки отдельно, в любой момент после установки
```

Перед ними — подготовка площадки («Требования к площадке» (стр. 16)), после них — первый вход, лицензия и приёмка (ниже на этой странице). Пошаговая инструкция под вашу версию, с теми же шагами руками без инсталлятора, лежит в `docs/` комплекта.

1. Приёмка комплекта

Распакуйте комплект на машине, с которой пойдёт установка, — при Kubernetes из комплекта это узел кластера (первый управляющий узел), при вашем кластере — любая машина с доступом к нему. Принятое место — `/opt/aivo-dist`. Проверка целостности описана на странице «Комплект поставки» (стр. 14); `./scripts/verify.sh` должен закончиться без единого `FAIL`.

На машины PostgreSQL, Redis и S3-хранилища переносить ничего не нужно, если их ставит инсталлятор по ssh: свою часть комплекта он относит туда сам.

Необязательная проверка окружения — `./bootstrap.sh`: ничего не устанавливает, печатает найденный python и версии библиотек. `./aivo-installer info` показывает сведения о комплекте.

2. Мастер настройки `./configure`

```
./configure                      # мастер
./configure --preset <имя>       # стартовать с решениями поставки из presets/
./configure --fresh              # начать с чистого листа, без прошлого захода
```

Мастер идёт по компонентам (см. «Выбор компонентов» (стр. 35)): для каждого — выбор варианта и поля реквизитов этого варианта. Обязательные поля помечены *. Если в комплекте есть пресет вашей поставки, мастер стартует с его решениями, и спрашивать остаётся только реквизиты площадки — адреса, домены, ключи.

Экран выбора варианта — на примере Keycloak:

```
                  … — настройка установки
 шаг 5 из 17: Keycloak (SSO)

 Keycloak (SSO)

 Провайдер входа. Вариант отвечает на один вопрос — КОМУ
 ПРИНАДЛЕЖИТ Keycloak, и вместе с ним решается, откуда установка
 берёт организации, роли и справочник людей.

 ┌──────────────────────────────────────────────────────────────────────┐
 │ (•) Внутренний: поднимается из комплекта                             │
 │ ( ) Realm установки на корпоративном сервере                         │
 │ ( ) Корпоративный: только вход и группы, без Admin API               │
 └──────────────────────────────────────────────────────────────────────┘

 Домен Keycloak (может совпадать с доменом платформы,
 если задан путь) *
   например: keycloak.example.ru
 ┌──────────────────────────────────────────────────────────────────────┐
 │ keycloak.example.ru                                                  │
 └──────────────────────────────────────────────────────────────────────┘

 Путь Keycloak на домене (пусто — Keycloak на своём домене)
   например: /auth
 ┌──────────────────────────────────────────────────────────────────────┐
 │ /auth                                                                │
 └──────────────────────────────────────────────────────────────────────┘

       [ Назад ]   [ Дальше ]

^q выйти  ^n дальше  ^p назад  ^r перепроверить  ^s записать
```

Реквизиты вашей системы — на примере языковой модели в контуре:

```
шаг 12 из 17: LLM

Языковая модель, которой отвечает агент. Требуется поддержка
tool calling.

┌──────────────────────────────────────────────────────────────────────────┐
│ ( ) Внешний провайдер моделей (нативный API)                             │
│ ( ) Внешний провайдер моделей (OpenAI-совместимый API)                   │
│ (•) Своя модель в контуре                                                │
│ ( ) Настроить позже Модель подключается в интерфейсе платформы          │
└──────────────────────────────────────────────────────────────────────────┘

Базовый URL (…/v1) *
  например: https://llm.example.ru/v1
┌──────────────────────────────────────────────────────────────────────────┐
│ https://llm.example.ru/v1                                                │
└──────────────────────────────────────────────────────────────────────────┘
Ключ *
┌──────────────────────────────────────────────────────────────────────────┐
│                                                                          │
└──────────────────────────────────────────────────────────────────────────┘
Имя модели *
┌──────────────────────────────────────────────────────────────────────────┐
│                                                                          │
└──────────────────────────────────────────────────────────────────────────┘
```

Клавиши: `Ctrl+N` — дальше, `Ctrl+P` — назад, `Ctrl+R` — перепроверить площадку, `Ctrl+S` — записать и выйти, `Ctrl+Q` — выйти, не записывая. Экран длиннее терминала прокручивается колесом и `PageUp` / `PageDown`; `Tab` всегда приводит поле в видимую область.

Экран не отпускает, пока на нём не всё в порядке: незаполненное обязательное поле или значение не того формата подсвечивается с причиной. Сертификаты и ключи вводятся путём к файлу или вставкой PEM целиком — вписанный PEM мастер сразу кладёт в файл с правами `0600` и в ответы записывает только путь.

Проверки площадки — предпоследний экран. Мастер подключается ко всему, что вы указали, — PostgreSQL, Redis, S3, Keycloak, ClickHouse, языковой модели — и показывает результат построчно, с именем компонента в начале строки (`[llm]`, `[keycloak]`), чтобы было видно, куда возвращаться. Настройку эти проверки не задерживают — записи DNS и доступы часто заводят как раз по её результату. Обязательными они становятся на шаге `preflight` установки: `./install` без них не начнётся.

Последний экран — что будет установлено:

```
шаг 17 из 17: Готово к установке

Что будет установлено

  kubernetes   vm             Kubernetes
  storage      in-cluster      Класс хранения (StorageClass)
  platform     in-cluster      Платформа
  auth-proxy   in-cluster      Вход в платформу (oauth2-proxy)
  keycloak     in-cluster      Keycloak (SSO)
  postgres     in-cluster      PostgreSQL 16 + pgvector
  redis        in-cluster      Redis
  nats         in-cluster      NATS JetStream
  objectstore  in-cluster      S3-совместимое хранилище
  langfuse     disabled        Langfuse
  clickhouse   in-cluster      ClickHouse (витрина данных)
  llm          external        LLM
  embeddings   external        Embeddings
  tls          manual          TLS-сертификаты доменов
  ingress      envoy-gateway   Публикация доменов (ingress)

Незаполненное:
  - kubernetes/vm: не заполнено «nodeAddress»
    (Адрес узла, на котором поднимается RKE2)

      [ Назад ]   [ Записать ]
```

`Ctrl+S` записывает результат.

Настройку можно прерывать. Мастер сохраняет введённое при каждом переходе между экранами, и следующий `./configure` предложит продолжить. Но незаконченный заход настройкой не считается: пока лежит черновик, `./install` и `./update` отказываются работать — иначе они поставили бы по старым ответам. Выход — довести мастер до `Ctrl+S` или удалить черновик `answers-draft.yaml`.

Повторная настройка уже поднятой установки — обычное дело: так добавляют файл лицензии, меняют адрес внешней системы, включают компонент. Пароли и ключи, которые придумал инсталлятор, при этом не меняются никогда. Новые генерируются только явным `--fresh` — на работающей установке делать этого не нужно: пароль первого пользователя Keycloak после этого не восстановить.

Неинтерактивный режим

Та же установка без человека — для повторяемых стендов и автоматизации:

```
./configure --answers answers.yaml --non-interactive
./configure --answers answers.yaml --non-interactive \
  --set llm.model=<имя модели>
./install --yes
./verify
```

`--answers` принимает файл, записанный мастером, или пресет, дополненный вашими значениями; `--set КОМПОНЕНТ.ПОЛЕ=ЗНАЧЕНИЕ` переопределяет отдельные ответы. Ответы проверяются тем же валидатором, что и экраны мастера, и неполный файл отклоняется со списком **всех** проблем сразу. На одинаковых ответах мастер и неинтерактивный режим дают одинаковый план.

Что записал мастер

Всё — в каталоге установки `~/.local/share/aivo/installation/`, **вне каталога комплекта**: следующая ревизия распаковывается поверх комплекта, а состояние установки должно это пережить. Каталог создаётся с правами `0700`, файлы с секретами — `0600`; переопределяется `--out` у `./configure` или переменной `AIVO_INSTALL_DIR`.

Файл	Что это
`answers.yaml`	ваши ответы; ими повторяется эта же установка (`./configure --answers …`)
`install-plan.txt`	план словами, без паролей — прочитайте его до установки
`install-config.yaml`	тот же план для `./install`
`values/`	файлы значений релизов с вашими реквизитами
`preflight-report.txt`	результаты проверок площадки
`vm/<компонент>.env`	пароли для сервисов на отдельных машинах
`install-state.yaml`	что из плана выполнено и какая ревизия стоит

`install-config.yaml`, `values/` и `vm/` содержат пароли и ключи — наружу их не отдавайте.

3. Установка `./install`

```
sudo ./install
```

Либо root только шагам узла, остальное — обычной учётной записью:

```
sudo ./install --only images.import,kubernetes.rke2
./install
```

Права проверяются до первого шага: `./install` без root, которому root нужен, отказывает сразу и перечисляет все такие шаги. С вашим кластером Kubernetes root не нужен вовсе.

В шапке запуска напечатано, **чьими ответами** идёт установка: путь к `answers.yaml`, время записи и отпечаток. Первые шаги плана — целостность комплекта и **`preflight`**: проверки площадки по данным ответам. Не прошла обязательная — установка не начинается (код возврата 4), причина — в сообщении и в `preflight-report.txt`.

Дальше инсталлятор идёт по плану: **шаг → проверка → отметка в состоянии**. Релиз считается установленным, только когда его поды готовы. Установка занимает десятки минут. Пока идёт долгий шаг, в его строке видно, сколько он уже идёт, сколько идёт обычно и сколько осталось до предела, и что сейчас в кластере: сколько подов готово и что мешает остальным. Команда для второго терминала напечатана тут же:

```
kubectl -n aivo get pods -w
```

Если шаг не прошёл, инсталлятор называет причину словами (нехватка памяти, не найден образ, поду некуда встать) и печатает статусы подов, последние события и хвост лога контейнера. Состояние, из которого кластер сам не выйдет, заканчивает шаг сразу, не дожидаясь предела.

- **Прервали или упало** — повторный `./install` продолжит с первого незавершённого шага.
- **Изменили реквизиты** — затронутые шаги выполнятся заново; `./install --list` показывает план, его состояние и такие шаги заранее.
- **Отдельный шаг** — `./install --from-step <шаг>` или `./install --only <шаг>`.

Шаги, которые делает человек. Если сервис на отдельной машине вы решили ставить руками, установка останавливается на нём и печатает, что перенести, что выполнить и на какой машине. Шаги, которые сейчас ни на что не влияют (например, загрузка лицензии в корпоративном контуре — её делает администратор своей учётной записью), перечисляются в конце со статусом «ждёт вас». И то и другое — код возврата 5; после ручного действия запустите `./install` ещё раз.

4. Проверка `./verify`

```
./verify
```

Целостность комплекта, готовность ваших систем, состояние кластера, релизов, подов, доменов, базы знаний, лицензии и промптов ассистента. Запускать можно когда угодно — в том числе через месяц после установки, чтобы понять, что сломалось. Каждая строка помечена компонентом (`[postgres]`, `[platform]`), в конце — итог по компонентам; отчёт остаётся файлом рядом с планом.

Проверка меняет ровно три вещи и называет их в отчёте: создаёт недостающую базу в PostgreSQL, устанавливает расширение `vector`, записывает и удаляет один пробный объект в S3.

Коды возврата

Код	Что означает
0	успех
1	ошибка; подробности — в логе (`logs/` каталога комплекта)
2	ошибка в аргументах командной строки
4	проверки не прошли: площадка не готова или установка не считается выполненной
5	установка выполнена, остались шаги, которые делает человек

5. Первый вход

Адрес платформы — `https://<домен платформы>`: браузер уходит на страницу входа Keycloak и возвращается в интерфейс.

Keycloak из комплекта. Установка завела realm, клиентов платформы, первую организацию и первого пользователя — администратора установки, состоящего в этой организации. Его логин и пароль, придуманный инсталлятором:

```
./aivo-installer secrets show --reveal
```

Остальных пользователей заводят в консоли Keycloak — см. «Пользователи, роли и доступ» (стр. 68).

Realm установки на корпоративном сервере. Realm, клиентов и учётные записи завёл администратор SSO по шаблону из комплекта; входит тот, кому он выдал роль администратора.

Корпоративный Keycloak. Первым входит участник группы администраторов установки, названной в мастере. Если организаций ещё нет, платформа заводит одну — `Default` — и правило, которое даёт этой группе роль администратора в ней; переименовать её можно в разделе «Тенанты». Дальше — организации и правила доступа по «Пользователям, ролям и доступу» (раздел «Пользователи, роли и доступ», стр. 69).

6. Лицензия

Без лицензии платформа работает в ограниченном режиме: вход, интерфейс, панель администратора, настройки и подключения доступны, а функции ассистента — диалог, автопилоты — отвечают отказом `LICENSE_RESTRICTED`. Режим виден баннером в интерфейсе и строкой «Лицензия установки» в `./verify`.

Лицензия выдаётся на конкретный кластер, поэтому запрос формируется на уже поднятой установке:

1. **Запрос.** В интерфейсе: **Администрирование → Инсталляция → Лицензия → «Запросить лицензию»**. Платформа соберёт файл запроса: UID кластера, название установки, версия платформы и открытый ключ установки. Секретов в нём нет, передавать его можно любым каналом.
2. **Отправка.** Файл запроса уходит поставщику платформы по каналу, оговорённому договором; обратно приходит файл лицензии.
3. **Загрузка.** Там же, **«Загрузить лицензию»**. Подпись проверяется до записи; функции ассистента включаются сразу, без перезапуска.

Если лицензия получена заранее, путь к её файлу можно назвать в мастере, и `./install` загрузит её сам. В корпоративном контуре загрузка остаётся за человеком: у установки нет учётной записи сотрудника, чтобы войти в платформу.

Продление — такая же загрузка новой лицензии. За месяц до конца срока интерфейс показывает предупреждение, `./verify` — число оставшихся дней. После истечения действует отсрочка, записанная в самой лицензии.

Что сказала платформа при загрузке	Что это значит
«это лицензионный ЗАПРОС, а не лицензия»	перепутаны файлы запроса и лицензии
подпись не проверяется	файл повреждён при передаче либо выдан не для этой версии платформы
лицензия выдана другому кластеру	кластер пересобран или это резервная площадка: действует отсрочка, за которую нужно получить лицензию на новый кластер
лицензия ещё не действует, расхождение часов	часы узла разошлись с датой выпуска — проверьте NTP

7. Что установка сделала сама

После установки руками заводить не нужно:

- **Языковая модель** — заведена в реестре моделей платформы и выбрана по умолчанию (если не выбрано «Настроить позже»).
- **Витрина ClickHouse** — заведена источником данных и привязана к организациям установки. Если организации на момент установки ещё не было, привяжите витрину после её появления: `./install --only datasource.register`. Наполнение витрины данными — за вашей службой эксплуатации.
- **Промпты ассистента** — глобальные системные сообщения из релиза платформы. Без них ассистент формально исправен, но отвечает без рамки назначения и без правил оформления. Шаг повторяется на каждом `./install` и доставляет только изменившееся; правки этих промптов в панели администратора новый релиз перезаписывает.
- **База знаний** — таблицу и индексы платформа заводит сама при старте. Наполняется база знаний в интерфейсе; сразу после установки она пуста, и это норма. Проверить её состояние: `./aivo-installer kb status`.

Источники Oracle и PostgreSQL подключаются в интерфейсе — см. «Подключение источников данных» («Руководство пользователя» → «Руководство администратора» → «Подключение источников данных»). Учётной записи источника достаточно прав только на чтение, и лучше, чтобы других у неё не было: ассистент выполняет SQL, который пишет языковая модель.

8. Приёмка

#	Проверка	Ожидаемый результат
1	`./verify`	все проверки прошли
2	`https://<домен платформы>`	вход через Keycloak, интерфейс открывается
3	Корпоративный Keycloak: вход пользователем из отображённой группы	попадает в свою организацию
4	Корпоративный Keycloak: вход пользователем вне всех отображённых групп	сообщение «нет доступной организации», а не ошибка
5	Корпоративный Keycloak: вход сотрудником с наибольшим числом групп	вход проходит
6	Администрирование → Инсталляция → Лицензия	лицензия активна, назван срок
7	Вопрос ассистенту	ответ строится, текст появляется потоком
8	Вопрос по данным	цифры из подключённого источника
9	`./aivo-installer kb status`	код 0: таблица и колонка векторов на месте, размерность совпадает с моделью
10	Трассировка, если подключён ваш Langfuse	трейсы диалогов пп. 7–8 со спанами модели и инструментов
11	Резервное копирование PostgreSQL	в каталоге назначения появился первый дамп (см. «Данные, бэкапы и retention» (стр. 64))

Установка без инсталлятора

Если инсталлятор запустить негде или служба безопасности требует выполнять всё поимённо, те же шаги выполняются руками — по приложению «Установка руками» инструкции в `docs/` комплекта. Оно же помогает понять, что делала установка, если она остановилась.

Обновление

Обновления приходят офлайн-комплектами следующих ревизий: новые образы, чарты и `CHANGELOG.md`. Обновляет установку сам комплект — командой `./update`, которая запускается **из нового комплекта**:

```
sha256sum -c aivo-<новая версия>.tar.sha256 && tar xpf aivo-<новая версия>.tar
cd aivo-<новая версия>
sudo ./update     # найти установку, показать план, спросить подтверждение
```

Команда	Что делает
`./update --plan`	только показать план, ничего не меняя
`./update --from <каталог>`	указать установку явно, если найдено несколько
`./update --backup`	снять дамп PostgreSQL из комплекта и обновиться
`./update --backup-taken "<чем и куда>"`	бэкап снят вашим регламентом — записать факт
`./update --yes`	без подтверждения, для автоматизации
`./update --rollback`	вернуть релизы на ревизии до обновления

Что делает `./update`

1. **Находит текущую установку.** Состояние живёт вне комплекта, в `~/.local/share/aivo/installation`, поэтому смена комплекта его не трогает. Ответы прошлой установки используются заново: решать, что и где ставится, не нужно. Ключи, которые вписывались вручную (например, секрет клиента корпоративного Keycloak), переносятся тоже.
2. **Проверяет цепочку ревизий.** Комплект объявляет в `MANIFEST.txt`, с каких ревизий он обновляет (`supported_upgrade`; то же показывает `./aivo-installer info`). Ревизия попадает в этот список, только когда обновление с неё прогнано на стенде. Прыжок через ревизию отклоняется: нужно поставить промежуточную и повторить.
3. **Показывает план и `CHANGELOG.md`**: какие релизы переустанавливаются, какие образы новые, обратимы ли миграции и какой ожидается простой. Настройка новой ревизии до подтверждения собирается в стороне, поэтому `--plan` и отказ на подтверждении ничего не меняют.

4. **Проверяет готовность** — до первого изменения в кластере:
    - есть свежий бэкап PostgreSQL (не старше суток);
    - хватает места под новые образы;
    - текущая установка исправна — `./verify` зелёный. Обновлять сломанную установку нельзя: после обновления уже не отличить старую поломку от новой;
    - каждый релиз может назвать свою текущую ревизию — иначе откат вернул бы установку не целиком.
5. **Выполняет** тем же исполнителем, что и `./install`: релизы по порядку, миграции при старте сервисов, итоговый `./verify`. Прерванное обновление повторный запуск продолжает с первого незавершённого шага.

Компоненты, которых не было в прошлой установке, обновление само не включает: в плане они идут как «не выбран — пропускается», а включаются осознанно через `./configure` после обновления.

Флага «обойти проверку готовности» нет намеренно: обновление, которое её обходит, её и не проверяет.

Бэкап перед обновлением

Без свежего бэкапа обновление не начнётся.

- **PostgreSQL из комплекта, в кластере** — `./update` снимает дамп сам (`pg_dumpall` в поде) в `~/.local/share/aivo/backups/`; `--backup` или `--yes` это подтверждает.
- **PostgreSQL на отдельной машине или ваш** — дамп снимает его администратор; на машине из комплекта для этого есть `postgres/backup/enable-backup.sh`. Факт записывается `--backup-taken "<чем и куда>"`.

Миграции

Схема базы обновляется по правилу **expand/contract**: ревизия добавляет то, что нужно новому коду, и только следующая убирает лишнее. Поэтому схема ревизии N+1 читается кодом ревизии N, и откат кода не требует отката данных. Ревизия, которая это правило нарушает, объявляет себя необратимой (`migrations_reversible=false` в `MANIFEST.txt`): `./update` предупреждает об этом до начала, а `CHANGELOG.md` описывает, чем грозит откат.

Базы, которых не было в прошлой ревизии, `./update` не заводит сам: `CREATE DATABASE` на рабочей СУБД делает человек. Если такая база понадобится, проверка готовности не пустит обновление, пока её нет, а `CHANGELOG.md` назовёт команду. Порядок жёсткий: сначала база, потом обновление.

Откат

```
sudo ./update --rollback
./verify          # из комплекта ПРОШЛОЙ ревизии — он снова актуален
```

Откат возвращает релизы на ревизии, записанные перед первым `helm upgrade`, и настройку прежней ревизии в каталог установки. Данные не трогаются. Если ревизия объявлена необратимой, откат вернёт код, но не схему — тогда возврат возможен только восстановлением из бэкапа.

Старый комплект не удаляйте, пока обновление не подтверждено: `./verify` и `./install` для работающей ревизии — из него, и из него же берутся бинарники Kubernetes, если придётся возвращать узел.

Kubernetes из комплекта

Узел одноузлового RKE2 обновление поднимает до версии комплекта само — если разрыв не больше одной минорной версии Kubernetes. Это офлайн-процедура: снимок etcd, артефакты комплекта, перезапуск `rke2-server`. Узел двигается до первого обновления релизов, так что упавший подъём останавливает обновление, пока ни один релиз не тронут. Платформа в этом окне недоступна 10–20 минут.

Обновление **не трогает узел** и говорит об этом в плане, если:

- разрыв больше одной минорной версии — промежуточных версий комплект не везёт;
- узел новее комплекта;
- в кластере больше одного узла;
- версию узла выяснить не удалось.

Платформа при этом обновляется своим чередом. **Откат узла не касается**: `./update --rollback` возвращает только релизы и прямо говорит об этом, если обновление подняло Kubernetes, — понизить версию узла можно только восстановлением снимка etcd, то есть с потерей всего записанного после него. Порядок ручного подъёма и возврата узла — в приложении инструкции в `docs/` комплекта.

Ваш кластер Kubernetes

Образы новой ревизии раскладываете в реестр вы, **до** обновления:

```
./images/import.sh --registry <реестр>    # из НОВОГО комплекта
sudo ./update --yes
```

Ingress-контроллер и сам кластер комплект не трогает ни при установке, ни при обновлении.

Ограничения

> ⚠ **Перехода с версий 0.x нет**
>
> Платформа 1.0 изменила форму части данных (параметры промптов, элементы библиотеки, начальные подсказки) так, что первый старт версии 1.x на базе версии 0.x их повреждает. Комплекты 1.x такой переход не поддерживают: `./update` на установке 0.x отказывает сам.
>
> **Не запускайте образы 1.x на базе 0.x** и не ставьте 1.x «как новую установку» поверх работающей — на установке с данными это их потеря. Переход с 0.x согласуйте с поставщиком платформы.

- **Список ревизий, с которых комплект обновляет, может быть пустым.** Тогда `./update` отказывает на любой установке и говорит почему. Список ведётся по прогонам на стенде: переход, который никто не проходил, комплект не объявляет. Проверьте `supported_upgrade` в `MANIFEST.txt` до окна работ и, если вашей ревизии там нет, обратитесь к поставщику.
- **Смена состава поставки — решение заказчика, а не шаг обновления.** Если ревизия выводит компонент из комплекта вместе с его данными, переход с установки, где этот компонент работает, не объявляется — до решения владельца установки, описанного в `CHANGELOG.md`.
- **Кластер из нескольких узлов** `./update` не обновляет — только платформу в нём.
- **Смена входа кластера из комплекта** (в 1.5 — Traefik → Envoy Gateway) на действующем узле необратима: откат вернёт релизы, но не контроллер узла.

Конфигурирование

Где живут настройки

Настройки установки лежат в трёх местах, и у каждого своя роль:

Где	Что	Как меняется
Ответы мастера — `answers.yaml` в каталоге установки	Решения и реквизиты: что откуда берётся, адреса, домены, ключи, модели	`./configure`, затем `./install`
Файлы значений — `values/` в каталоге установки	Значения релизов Helm, отрендеренные из ответов; среди них конфигурация платформы	Пересобираются мастером из ответов
База платформы	Модели, источники данных, промпты, организации, правила доступа, оформление, лицензия	В панели администратора (раздел «Панель администратора установки», стр. 72), без перезапуска

Каталог установки — `~/.local/share/aivo/installation/` учётной записи, которой шла настройка (подробнее — «Установка» (стр. 47)).

Как поменять настройку

Всё, что спрашивает мастер — адрес внешней системы, сертификат, модель, вариант компонента, файл лицензии, — меняется повторной настройкой:

```
./configure        # мастер предложит продолжить с текущими ответами
./install          # выполнит заново только затронутые шаги
./verify
```

Пароли и ключи, которые придумал инсталлятор, при повторной настройке не меняются. `./install --list` до запуска покажет, какие шаги изменение затронет.

Сертификат продлевается так же: новый файл кладётся поверх прежнего, и `./install` переносит его в секреты кластера; перезапуск не нужен.

Параметр, которого в мастере нет. Файлы значений можно править руками: изменившийся файл меняет отпечаток шага, и следующий `./install` выполнит этот шаг заново. Но следующий `./configure` пересоберёт файлы из ответов и ручную правку уберёт. Поэтому ручная правка —

временная мера: параметры, которые нужны вашей установке постоянно, согласуйте с поставщиком, чтобы они появились в мастере.

> ⚠ **Осторожно**
>
> Не меняйте объекты в кластере командами `kubectl edit` / `kubectl apply` и не делайте `helm upgrade` в обход инсталлятора: план и его состояние перестанут описывать то, что стоит в кластере, и следующий `./install` или `./update` вернёт всё к плану. Команды чтения (`get`, `describe`, `logs`) безопасны.

Как платформа читает конфигурацию

Сервисы платформы не читают настройки из переменных окружения по отдельности:

1. Сервис `config` получает конфигурацию всех сервисов из файла значений платформы.
2. Каждый сервис при старте запрашивает у `config` свою секцию.
3. Секция проверяется схемой. Если параметр отсутствует или имеет неверный тип, сервис **не стартует** и пишет в лог, какое поле не прошло проверку.

Отсюда практическое следствие: сервис, который не поднимается сразу после изменения настройки, почти всегда означает ошибку в значении — смотрите первые строки лога его пода. Пароли и ключи в конфигурацию не пишутся открытым текстом: они подставляются из секретов Kubernetes.

Параметры `api`, о которых стоит знать

Основная секция — у сервиса `api`. Большинство её значений задаёт мастер; ниже — те, о которых чаще спрашивают при эксплуатации.

Параметр	Назначение
`db`	Подключение к базе платформы, лимиты пула соединений
`checkpointer`	Где хранится рабочее состояние ассистента между шагами (`postgres`) и политика его очистки, см. «Данные, бэкапы и retention» (стр. 67)
`nats`	Адрес брокера: через него ответ ассистента идёт в интерфейс потоком
`redis`	Кэш платформы
`fileStorage`	S3-хранилище вложений и выгрузок

Параметр	Назначение
`fileUpload`	Ограничения загружаемых файлов. Должны согласовываться с лимитом тела запроса на входе — см. «Требования к вашему кластеру» (стр. 26)
`maxHttpBodyBytes`	Потолок JSON-тела запроса к API, по умолчанию 5 МБ
`embeddings`	Модель векторизации: провайдер, модель, размерность. Размерность меняется только на пустой базе знаний
`langfuse`	Трассировка в ваш Langfuse: адрес и ключи проекта
`documents.allowAgentWrites`	Разрешено ли ассистенту записывать в базу знаний. По умолчанию — нет, только читает
`recursionLimit`	Максимальное число шагов ассистента в одном запросе
`subagent`	Лимиты подзадач ассистента: шаги, параллельность, число вызовов
`billing.enabled`	Учёт расходов на языковые модели и бюджеты организаций. По умолчанию выключен
`cors`	Разрешённые источники запросов к API

Модели, доступные пользователям, источники данных организаций, промпты и оформление задаются не здесь, а в панели администратора (раздел «Панель администратора установки», стр. 72) и в «Руководстве администратора» (см. «Руководство пользователя» → «Руководство администратора» → «Администрирование организации»).

Другие сервисы

- **`identity`** — провайдер входа. Режим задаётся вариантом Keycloak в мастере: Keycloak установки (организации и роли из Keycloak) или корпоративный (организации и роли — из правил доступа в базе платформы).
- **`instance`** — идентификатор и название установки.
- **`service-discovery`** — внутренние адреса сервисов.
- **`ui`** — адреса GraphQL-API (HTTP и WebSocket) для веб-интерфейса. Если после смены домена интерфейс не загружается, проверяйте в первую очередь их.

Эксплуатация и диагностика

Проверка работоспособности

Первое действие при любой проблеме — `./verify` из комплекта, которым стоит установка. Он проверяет всю цепочку — комплект, ваши системы, кластер, релизы, поды, домены, базу знаний, лицензию, промпты — и называет конкретное место; каждая строка помечена компонентом.

```
./verify
```

Дальше — сам кластер:

```
kubectl -n aivo get pods     # все поды Running, рестарты не растут
kubectl -n aivo get deploy   # имена сервисов платформы
```

При Kubernetes из комплекта kubeconfig администратора — `/etc/rancher/rke2/rke2.yaml` (читается от root); установка кладёт копию в `~/.kube/config` учётной записи, которая её запускала. Namespace по умолчанию — `aivo`.

Признаки здоровой установки:

- `./verify` проходит без блокирующих строк;
- все поды в `Running`, счётчик рестартов не растёт;
- страница входа открывается и после входа отдаёт интерфейс;
- пробный вопрос ассистенту возвращает ответ потоком — это проверяет сразу ассистента, модель и поток через NATS.

Где смотреть, что происходит

Что нужно	Куда смотреть
Почему не прошла установка или обновление	Лог инсталлятора — `logs/` в каталоге комплекта; путь печатается при каждом запуске
Логи сервиса платформы	`kubectl -n aivo logs deploy/<имя> --tail=200`
Почему под не стартует	`kubectl -n aivo describe pod <под>` — события и причина

Что нужно	Куда смотреть
Сервис на отдельной машине (PostgreSQL, Redis, S3-хранилище)	`journalctl -u <юнит>` на этой машине
Kubernetes из комплекта	`journalctl -u rke2-server -f` на узле
Что делал ассистент в конкретном диалоге	Ваш Langfuse, если трассировка подключена; иначе — история диалогов в панели администратора (раздел «Панель администратора установки», стр. 72)
Секреты, придуманные инсталлятором	`./aivo-installer secrets show`

Системы сбора логов и метрик комплект не ставит: логи подов и сервисов собирайте средствами вашей площадки.

Типовые сбои

Симптом	Куда смотреть
Инсталлятор не запускается	Версия python (нужна 3.9–3.12), целостность комплекта (`./scripts/verify.sh`); `./bootstrap.sh` назовёт причину
`./install` встал на шаге	Сообщение шага и разбор под ним; исправить и запустить `./install` снова — он продолжит с этого места
`rke2-server` не стартует	`journalctl -u rke2-server`. Частое: swap не выключен, мало места, SELinux без пакета `rke2-selinux`
Под в `ImagePullBackOff`	Kubernetes из комплекта: образы не импортированы — архивы лежат в `/var/lib/rancher/rke2/agent/images/`? Ваш кластер: образов нет в реестре или адрес назван неверно — `./images/import.sh --registry <реестр> --dry-run` покажет, под какими именами они должны лежать
Под в `Pending`	Не хватает ресурсов узла или квоты namespace'а, не выделен том, не пропускает Pod Security: `kubectl describe pod`
`config` не готов	Лог пода: ошибка проверки конфигурации, поле будет названо
Сервис висит в ожидании	Он ждёт `/healthz` предыдущего по цепочке — смотрите предыдущий сервис (см. «Состав платформы» (стр. 9))

Симптом	Куда смотреть
`api` не видит PostgreSQL или Redis	Firewall и маршрутизация между машинами, `pg_hba.conf`; `./verify` проверяет это отдельными строками
Вход не работает	Keycloak: клиент заведён, адрес возврата — `https://<домен>/oauth2/callback`, секрет клиента актуален, OIDC discovery доступен **с узла и из пода**; у Keycloak за корпоративным УЦ указан его корневой сертификат. У Keycloak из комплекта всё это делает шаг установки — повторите его: `./install --only keycloak.realm`
Вход прошёл, но каждый запрос отвечает отказом	Корпоративный Keycloak: в отказе назван код. `AUTH_CLAIMS_UNTRUSTED` — не совпадает issuer (нужен публичный адрес realm'а), audience или истёк токен; если вход работает несколько минут и перестаёт — интервал обновления сессии длиннее времени жизни токена. `AUTH_CLAIMS_KEYS_UNAVAILABLE` — ключи подписи не забираются из пода: проверьте внутренний адрес ключей
Вход работает у всех, кроме некоторых	`AUTH_CLAIMS_TOO_LARGE`: список групп в токене не отфильтрован. Чинится в Keycloak, а не поднятием лимита (см. «Корпоративный Keycloak» (стр. 31))
«Нет доступной организации» у того, у кого она должна быть	Корпоративный Keycloak: ни одно правило доступа не совпало. Сверьте путь группы из токена с правилом — обычно расходится регистр
Ассистент отвечает, но без инструментов	Вызов инструментов на сервинге модели: включён ли парсер tool-вызовов
Ассистент отвечает «модель не настроена»	Модели нет в реестре платформы: `./install --only llm.register` или подключить в панели администратора
Вопрос по данным без ответа, источников нет	Витрина не заведена источником или организация к ней не привязана: `./install --only datasource.register`
Ответы без цифр	Доступность витрины с узла и из пода, наполненность её данными
Ассистент отвечает не по назначению, таблицы без оформления	Нет глобальных промптов или они старее релиза: `./install --only prompts.apply`
Поиск по базе знаний находит по словам, а не по смыслу	Нет расширения `vector`: `CREATE EXTENSION vector;` в базе платформы и перезапуск подов `api`. Состояние базы знаний — `./aivo-installer kb status`
Ассистент отвечает `LICENSE_RESTRICTED`	Лицензии нет, она истекла или выдана другому кластеру — см. «Установка» (стр. 50); строка `./verify` «Лицензия установки» назовёт причину

Симптом	Куда смотреть
Не загружаются файлы	Лимит тела запроса на входе (нужно не меньше 30 МБ) и доступность S3-хранилища
Ответ ассистента обрывается	Таймауты на входе: ожидание ответа и чтение запроса — не меньше 300 секунд, у Gateway API — `request: 0s`
В вашем Langfuse нет трейсов	Проверка `langfuse.api` при настройке: отказ 401 — проекта с этими ключами нет. Если адреса и ключей в настройке нет, трассировка выключена, и это штатное состояние

Плановые операции

- **Перезапуск сервиса:** `kubectl -n aivo rollout restart deploy/<имя>`. Данные не теряются: состояние диалогов хранится в базе.
- **Повтор одного шага установки:** `./install --only <шаг>`; список шагов с состоянием — `./install --list`.
- **Изменение настройки** — через мастер, см. «Конфигурирование» (стр. 57).
- **Обновление** — только новым комплектом, см. «Обновление» (стр. 53).
- **Продление сертификата** — новый файл поверх прежнего и `./install`.
- **Продление лицензии** — загрузка новой в панели администратора.

Данные, бэкапы и retention

Где что хранится

Хранилище	Что содержит	Пользовательские данные	Сколько хранится по умолчанию
PostgreSQL, база платформы	Настройки, организации, диалоги, библиотека, дашборды, промпты, база знаний (тексты и векторы)	Да	Бессрочно
PostgreSQL, база рабочего состояния ассистента	Состояние ассистента между шагами диалога	Частично	Очищается автоматически, см. ниже
PostgreSQL, база сервиса `identity`	Организации, роли, правила доступа, администраторы установки, копии вошедших пользователей	Учётные данные	Бессрочно
PostgreSQL, база Keycloak (только Keycloak из комплекта)	Пользователи, пароли, группы, клиенты	Учётные данные	Бессрочно
ClickHouse	Витрина бизнес-данных; таблицы из загруженных пользователями файлов	Да	Витрина — по вашему регламенту; таблицы из файлов, загруженных в диалог, — 30 дней

Хранилище	Что содержит	Пользовательские данные	Сколько хранится по умолчанию
S3-хранилище	Вложения диалогов и выгрузки	Да	Файлы, загруженные в диалог, — 30 дней, выгрузки — сутки
Redis	Кэш и сессии входа	Транзитно	До перезапуска
NATS JetStream	Поток ответа ассистента в интерфейс	Кратковременно	Минуты
Каталог установки — `~/.local/share/aivo/installation/` на машине установки	Ответы мастера, пароли и ключи, придуманные инсталлятором, план, состояние	Нет, но секреты	Пока существует установка

Базы PostgreSQL называются `aivo`, `aivo_checkpointer`, `aivo_identity` и `keycloak`.

Что бэкапить

Обязательно:

- **PostgreSQL** — база платформы, база `identity` и база Keycloak, если он из комплекта. Это то, что нельзя восстановить пересчётом.
- **S3-хранилище** — вложения пользователей.
- **Каталог установки** с машины, с которой шла установка. В нём пароли и ключи, которые придумал инсталлятор, и ответы, которыми установка повторяется и обновляется. Без него следующий комплект установку не найдёт, а пароли придётся восстанавливать из секретов кластера. Храните копию как секрет.

По ситуации:

- **ClickHouse** — витрину бэкапить не нужно, если она полностью восстанавливается повторной загрузкой из источника; таблицы из файлов, загруженных в диалог, живут 30 дней.

- **База рабочего состояния ассистента** — можно не бэкапить: это кэш, полная история диалогов хранится в базе платформы, и после восстановления без неё диалоги продолжаются.
- **Redis** и **NATS** — не бэкапятся: потеря Redis означает только повторный вход пользователей.

Чем бэкапить

Регулярного резервного копирования комплект за вас не настраивает — кроме машины PostgreSQL из комплекта, где для него есть готовый скрипт.

Где стоит PostgreSQL	Чем
Отдельная машина из комплекта	`postgres/backup/enable-backup.sh` из комплекта: ставит ежедневный дамп с ротацией таймером systemd. `--target` — каталог, куда машина может писать (локальный диск, NFS, любое смонтированное хранилище), `--keep` — сколько дней хранить (по умолчанию 14), `--at` — время запуска (по умолчанию 02:30). Выносить копии в объектное хранилище — вашими средствами
В кластере	Снимки тома средствами вашего хранилища или `pg_dump` по расписанию вашим регламентом. Одна реплика: отказ узла с томом останавливает платформу до возврата тома или восстановления из копии
Ваш PostgreSQL	Вашим регламентом

```
# на машине PostgreSQL, в её каталоге комплекта (/opt/aivo-dist/postgres)
sudo ./backup/enable-backup.sh --target /mnt/backup/aivo --keep 14 --at 02:30
```

Перед каждым обновлением `./update` требует свежий бэкап и, если PostgreSQL из комплекта в кластере, снимает его сам (`pg_dumpall` в поде) — см. «Обновление» (стр. 54). Это копия на случай отката, а не замена регулярной.

S3-хранилище и ClickHouse в кластере резервным копированием не покрываются — снимки томов либо ваш регламент.

Проверка восстановления

Резервная копия, из которой ни разу не восстанавливались, ничего не гарантирует. Периодически поднимайте установку из того же комплекта на тестовой площадке с копией каталога установки,

восстанавливайте в неё дамп PostgreSQL и проверяйте, что пользователи входят и видят свои диалоги.

Сроки хранения

- **Рабочее состояние ассистента** очищается автоматически, раз в час: у каждого диалога остаются два последних снимка, а диалог без активности 30 дней теряет их все. Полная история при этом остаётся в базе платформы, и диалог продолжается с того же места. Настройки — `checkpointer.retention` в конфигурации `api`.
- **Файлы, загруженные в диалог**, и таблицы, созданные из них в ClickHouse, хранятся 30 дней; **выгрузки** — сутки, ссылка на скачивание действует 15 минут.
- **Диалоги** удаляют пользователи в интерфейсе; массовая очистка — на стороне базы.
- **Трассировка** — в вашем Langfuse и по его настройкам хранения. Трейсы содержат промпты, ответы и фрагменты данных: обращайтесь с ними как с пользовательскими данными.

Защита данных

- Шифрование дисков машин и томов с базами — средствами площадки.
- Учётная запись только на чтение для источников данных ассистента: он выполняет SQL, который пишет языковая модель, и единственная гарантия, что оттуда не придёт запись, — права на вашей стороне.
- Доступ к истории диалогов в панели администратора и к Langfuse — только тем, кому он нужен по роли: эти экраны показывают содержимое запросов.
- Каталог установки и файлы значений содержат пароли: права `0700` / `0600` ставит инсталлятор, не ослабляйте их и не передавайте эти файлы по открытым каналам.

Пользователи, роли и доступ

Вход в платформу всегда идёт через Keycloak, а вот где живут организации и роли, зависит от варианта Keycloak, выбранного при установке (см. «Выбор компонентов» (стр. 37)):

	Keycloak установки — из комплекта или realm установки на вашем сервере	**Корпоративный Keycloak**
Кто заводит пользователей	администратор realm'а в консоли Keycloak	только ваша служба эксплуатации Keycloak
Откуда организации	организации (Organizations) Keycloak	разделы «Тенанты» и «Правила доступа» панели администратора
Откуда роли в организации	группы `/org/<id организации>/admins` и `/org/<id организации>/members`	правила «путь группы → организация → роль»
Кто администратор установки	держатели роли `admin` клиента платформы в Keycloak	группа администраторов из мастера, затем — записи в базе платформы

Продукт в обоих случаях один: интерфейс, роли и права одинаковы.

Роли

- **Участник организации** — работает с ассистентом: диалоги, дашборды, библиотека, автопилоты, свои файлы.
- **Администратор организации** — всё, что участник, и администрирование своей организации: источники данных, модель данных, промпты, участники, бюджет. Описано в «Руководстве администратора» (см. «Руководство пользователя» → «Руководство администратора» → «Администрирование организации»).
- **Администратор установки** — отвечает за платформу целиком: модели, лицензия, организации, правила доступа, оформление по умолчанию. Это свойство человека, а не роль в организации: оно не требует членства ни в одной организации и действует во всех.

Роль в организации действует внутри неё: один человек может быть администратором одной организации и участником другой. Данные чужой организации недоступны ни через интерфейс, ни через ассистента.

Keycloak установки

Создание пользователя

1. Откройте консоль Keycloak (при Keycloak из комплекта — `https://<домен Keycloak>/admin` или `https://<домен платформы>/auth/admin`, если он под путём) и войдите администратором Keycloak. Его пароль — `./aivo-installer secrets show --reveal`.
2. В realm'е платформы: **Users → Add user** — логин и e-mail.
3. Вкладка **Credentials** — пароль. Оставьте `Temporary`, если пользователь должен сменить его при первом входе.
4. Добавьте пользователя в организацию: **Organizations → организация → Members → Add member**.
5. Выдайте роль в организации членством в группе: `/org/<id организации>/members` — участник, `/org/<id организации>/admins` — администратор организации. Группы создаются вместе с организацией.
6. Передайте пользователю адрес платформы.

Администратор установки — роль `admin` клиента платформы (**Clients → клиент платформы → Roles**; выдаётся на вкладке пользователя **Role mapping**). Realm-роли платформа не читает. Первый пользователь, заведённый установкой при Keycloak из комплекта, уже держит эту роль и состоит в первой организации.

Организации

Новая организация заводится в Keycloak (**Organizations → Create organization**). В разделе **Администрирование → Инсталляция → Тенанты** её можно только включить или выключить: название приходит из Keycloak и перезаписывается при синхронизации.

Корпоративный Keycloak

Учётные записи и группы ведёт ваша служба эксплуатации Keycloak, и только она: списка людей realm'а платформа не видит и завести никого не может. Требования к realm'у — «Корпоративный Keycloak» (стр. 30).

Первичная настройка

1. **Войдите участником группы администраторов установки** — той, что названа в мастере. Это единственный способ войти в свежую установку. Если организаций ещё нет, платформа заводит одну, `Default`, и правило, которое даёт этой группе роль администратора в ней.

2. **Дайте каждой нужной группе разок войти.** Список групп realm'а платформе недоступен, поэтому она запоминает каждый путь, который хоть раз пришёл в токене, и предлагает его во вкладке **Группы** раздела «Правила доступа». Попросите службу эксплуатации Keycloak о тестовом пользователе на подразделение; отказанные входы список тоже наполняют.
3. **Заведите организации** в разделе **Тенанты** — по одной на подразделение. `Default` можно переименовать.
4. **Напишите правила** в разделе **Правила доступа**: «участники такого-то пути группы имеют такую-то роль в такой-то организации». Сопоставление — по префиксу пути и с учётом регистра: `/Corp/Sales` покрывает `/Corp/Sales/Analysts`, но не `/Corp/SalesOps` и не `/corp/sales`. Пути копируйте из вкладки «Группы», а не набирайте руками. Каждой организации — хотя бы одно правило с ролью администратора, иначе настраивать её будет некому. Вкладка **Симуляция** показывает, что получит человек с заданным набором групп.
5. **Проверьте настоящим пользователем каждого подразделения** — пункты приёмки 3–5 на странице «Установка» (стр. 51).

Состав организаций и роли пересчитываются из правил при каждом входе, поэтому в карточке организации они не редактируются — меняется правило. Вкладка **Аудит** показывает, кто и когда менял организации, состав и правила; в корпоративной установке это единственная запись об этом.

Ограничения

- В справочнике пользователей платформы только те, кто хотя бы раз входил; пригласить того, кто не входил, нельзя.
- Вывод из группы действует со следующего входа или обновления сессии — обычно минуты.
- Лицензионное место занимается при первом входе.

Закрыть доступ

Задача	Keycloak установки	Корпоративный Keycloak
Временно закрыть вход	В Keycloak выключить у пользователя `Enabled`	Заблокировать в платформе (ниже) и/или отключить учётную запись в вашем Keycloak
Убрать из одной организации	Исключить из организации Keycloak и её групп `/org/<id>/*`	Вывести из группы в вашем Keycloak или изменить правило; немедленно — «Исключить из тенанта» в платформе
Закрыть доступ ко всей установке	Удалить или отключить пользователя в Keycloak	**Заблокировать в инсталляции** в платформе

Блокировка и исключение в платформе — в разделе **Участники** администрирования организации: «Исключить из тенанта» и «Заблокировать в инсталляции» (второе доступно администратору установки). Они переживают пересчёт по правилам и снимаются там же. Пользователь с заблокированной учётной записью видит при входе, что это решение администратора установки. Блокировка освобождает лицензионное место.

История диалогов удалённого или заблокированного пользователя остаётся в базе платформы.

Панель администратора установки

Часть настроек установки задаётся не мастером, а в интерфейсе — в разделе **Администрирование → Инсталляция**. Он доступен администраторам установки (см. «Пользователи, роли и доступ» (стр. 68)). Изменения в нём вступают в силу без перезапуска сервисов.

Администрирование отдельной организации — источники данных, модель данных, документы, участники, бюджет — описано в «Руководстве администратора» (см. «Руководство пользователя» → «Руководство администратора» → «Администрирование организации»). Администратор установки может открыть его в любой организации.

Модели и данные

Раздел	Что настраивается
Модели LLM	Модели и провайдеры, на которых работает ассистент: подключение, параметры, проверка связи. Модель, выбранная в мастере, уже заведена здесь установкой; если в мастере было «Настроить позже», модель подключается здесь
Хранилище таблиц файлов	База ClickHouse, в которую попадают таблицы из загруженных пользователями файлов, — отдельно для каждой организации. Работает, когда загрузка файлов в таблицы включена в конфигурации (`fileIngest.enabled`)

Кнопка «Проверить подключение» сразу показывает проблемы с сетью или ключом.

Поведение ассистента

Раздел	Что настраивается
Промты	Глобальные системные сообщения ассистента: назначение, правила ответа, оформление таблиц и графиков. Установка заводит и обновляет их из релиза платформы
Скиллы агента	Глобальные скиллы — инструкции для типовых задач, которые ассистент подгружает, когда вопрос подходит под их описание. Скиллы организаций задаются в их администрировании
События	Именованные триггеры, вызываемые извне — например, после обновления данных. На глобальное событие могут подписываться автопилоты и дашборды всех организаций

> ⚠ **Осторожно**
>
> Глобальные промпты влияют на ответы всех пользователей всех организаций. Новый релиз платформы перезаписывает их своими текстами (прежнее содержимое сохраняется снимком), а `./verify` сообщает о расхождении с релизом отдельной строкой. Собственные правила лучше задавать на уровне организации — их обновление не трогает.

Организации и доступ

Раздел	Что настраивается
Тенанты	Организации установки. С Keycloak установки — только включить или выключить (организации заводятся в Keycloak); с корпоративным Keycloak — завести, переименовать, включить или выключить
Правила доступа	Правила «путь группы → организация → роль» для корпоративного Keycloak, список групп, которые видела установка, симуляция и журнал изменений. С Keycloak установки правила хранятся, но не действуют
Биллинг	Тарифы, кредиты и лимиты расхода организаций. Раздел есть, только если учёт расходов включён в конфигурации (`billing.enabled`)
Лицензия	Состояние лицензии (серийный номер, владелец, срок, число пользователей), запрос и загрузка лицензии, история. Порядок — «Установка» (стр. 50)
Внешний вид	Оформление по умолчанию для всех организаций: название, логотип, тема, цвета, размер текста, и можно ли организациям менять его. Применяется без перезапуска

Наблюдение

Раздел	Что показывает
История диалогов	Диалоги всех организаций: кто, когда и о чём спрашивал ассистента. Для разбора инцидентов и жалоб на качество ответов
Аналитика диалогов	Активность по организациям и пользователям

История диалогов содержит пользовательские данные — выдавайте права администратора установки только тем, кому они нужны по роли.

Сборка 8817d38 от 3 ноября 2026 г.