Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

n8n self-hosted для отдела продаж: контур автоматизаций

Как мы разворачиваем n8n 2.31.x в queue-режиме и связываем CRM, 1С, почту и мессенджеры

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Отдел продаж живёт в четырёх окнах: CRM, 1С, почта и мессенджер. Данные между ними менеджер переносит руками — заявку в карточку, остатки в письмо, продажу в сводную таблицу. Мы ставим заказчику n8n self-hosted, который делает эти переходы сам. Пока его нет, компания оплачивает копипаст по ставке менеджера, теряет заявки на стыках систем и не видит реальной воронки.

Почему это важно бизнесу

- Час менеджера уходит на перенос данных, а не на сделки — прямой расход фонда оплаты труда без выручки
- Заявки теряются на стыках почты, CRM и 1С: клиент не получает ответ и уходит к конкуренту
- Руководитель видит цифры воронки постфактум и с ошибками ручного ввода
- Облачные платформы автоматизации тарифицируют каждый запуск, self-hosted — только стоимость VPS
- Данные клиентов и цены не покидают контур компании, что снимает вопросы к обработке ПДн

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

2.31.4

Ветка 2.x под тегом stable, которую ставим в прод; DB_TYPE принимает лишь sqlite и postgresdb

тег stable образа n8nio/n8n

336 ч

Дефолт EXECUTIONS_DATA_MAX_AGE (14 суток). Режем до 168 ч: иначе данные исполнений съедают диск

EXECUTIONS_DATA_MAX_AGE, наш стандарт 168

10 000

Дефолт EXECUTIONS_DATA_PRUNE_MAX_COUNT — записей в базе; режем до 5000 плюс прунинг по возрасту

EXECUTIONS_DATA_PRUNE_MAX_COUNT

10 → 5

Флаг --concurrency воркера: дефолт 10, ниже 5 нельзя — воркеры выбирают пул соединений БД

n8n worker --concurrency

2 → 10

DB_POSTGRESDB_POOL_SIZE: дефолт 2 соединения мал для queue-режима, поднимаем под число воркеров

DB_POSTGRESDB_POOL_SIZE, наш стандарт

16 MiB

N8N_PAYLOAD_SIZE_MAX по умолчанию: выгрузка прайса или пачка PDF больше лимита упадёт на входе

N8N_PAYLOAD_SIZE_MAX

РЕАЛИЗАЦИЯ 1 ИЗ 3

Базовый контур: Compose, PostgreSQL, queue-режим

Что настраиваем

Одна VM в РФ-ЦОД: main-инстанс n8n, воркеры, PostgreSQL, Redis и reverse-proxy в Docker Compose

Как мы это делаем

1. Compose: n8n main, воркеры, PostgreSQL, Redis, reverse-proxy с Let's Encrypt; DB_TYPE=postgresdb, DB_POSTGRESDB_POOL_SIZE=10, EXECUTIONS_MODE=queue на всех ролях
2. N8N_ENCRYPTION_KEY генерируем один раз и одинаковым передаём в main и воркеры: иначе воркер не расшифрует credentials из базы и задачи падают
3. GENERIC_TIMEZONE=Europe/Moscow плюс TZ контейнера: дефолт America/New_York, при нём Schedule Trigger «в 9:00» отработает не тогда
4. Внешний контур: N8N_HOST и WEBHOOK_URL на боевой FQDN, N8N_PROXY_HOPS=1 за прокси, N8N_SECURE_COOKIE не выключаем; наружу открыт только путь /webhook/
5. Воркеры: n8n worker --concurrency=5, QUEUE_BULL_REDIS_HOST на общий Redis, QUEUE_HEALTH_CHECK_ACTIVE=true и опрос /healthz/readiness

РЕЗУЛЬТАТ

Панель не смотрит в интернет, исполнение отделено от интерфейса: перезапуск main не рвёт очередь, а пиковая нагрузка (утренний дайджест плюс поток вебхуков) размазывается по воркерам. Обновление или падение одного воркера не останавливает приём заявок.

КЛЮЧЕВОЙ НЮАНС

Queue-режим включается на всех ролях сразу и требует общего ключа шифрования и общего Redis. Дефолт QUEUE_BULL_REDIS_HOST — localhost, поэтому воркер без явного адреса молча слушает себя.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Интеграции: почта, CRM, 1С по OData, мессенджер

Что настраиваем

Слой сценариев продаж: приём заявок, обогащение из 1С, уведомления и ночные сверки на стороне main-инстанса

Как мы это делаем

1. Триггеры: Webhook на формы сайта и платёжный шлюз, почтовый триггер на ящик продаж, Schedule Trigger на ночные сверки; у каждого вебхука свой path и проверка токена
2. 1С: HTTP Request к OData /odata/standard.odata/ с $format=json и фильтрами $filter/$select — XML не разбираем; сервисная учётка 1С только на чтение остатков и цен
3. CRM: постраничная выборка через Loop Over Items с нодой Wait — плотный цикл ловит лимит API и 429; на нодах включаем Retry On Fail, Max Tries=3 и Wait Between Tries
4. Общие шаги выносим в Execute Sub-workflow: нормализация телефона, поиск дубля контакта, журнал операций — один блок на весь набор сценариев вместо копий
5. Уведомления в мессенджер: в текст кладём ссылку на карточку, а не персональные данные — переписка бота не становится вторым хранилищем ПДн

РЕЗУЛЬТАТ

Заявка из письма или формы попадает в CRM за секунды с полным набором полей, остатки и цены приходят менеджеру по запросу без входа в 1С, а руководитель получает сводку по расписанию. Ручной перенос данных между окнами исчезает как класс работы.

КЛЮЧЕВОЙ НЮАНС

Узкое место всегда во внешнем API, а не в n8n: до проектирования сценария выясняем лимит запросов, формат авторизации и поведение при 429, и закладываем батчи с паузой между ними.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Эксплуатация: наблюдаемость, прунинг, бэкап, апгрейд

Что настраиваем

Регламент сопровождения: контроль ошибок, рост базы исполнений, резервные копии и обновление версии

Как мы это делаем

1. Каждому продовому воркфлоу назначаем Error Workflow: Error Trigger собирает имя сценария, id исполнения, сбойную ноду и текст ошибки и шлёт алерт админу в мессенджер
2. Прунинг: EXECUTIONS_DATA_SAVE_ON_SUCCESS=none, SAVE_ON_ERROR=all, MAX_AGE=168, EXECUTIONS_DATA_PRUNE_MAX_COUNT=5000 — иначе таблицы растут быстрее диска
3. Метрики: N8N_METRICS=true и N8N_METRICS_INCLUDE_QUEUE_METRICS=true, скрейп Prometheus с /metrics; алерты на длину очереди и на всплеск неуспешных исполнений
4. Бэкап: ночной pg_dump плюс экспорт командой n8n export:workflow --backup --output=/backup/ (флаг сам ставит --all --pretty --separate), ретеншн 30 дней
5. Обновление: образ пинуем точным тегом вместо stable, прогон на копии базы, окно вне рабочего дня, откат — прежний тег и дамп; перед мажором читаем breaking changes

РЕЗУЛЬТАТ

Сбой сценария виден инженеру раньше, чем менеджеру: алерт приходит с точным местом отказа. База не разрастается бесконтрольно, а обновление n8n перестаёт быть событием с риском — есть проверенная процедура отката на предыдущий тег.

КЛЮЧЕВОЙ НЮАНС

n8n по умолчанию сохраняет данные всех успешных исполнений — на потоке в десятки тысяч запусков это главный источник роста БД. Настройки прунинга задаём в первый же день, а не когда кончится диск.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

✕ SQLite в продакшене

Воркеры поверх SQLite не поддерживаются, MySQL и MariaDB в 2.0 удалены. Ставим PostgreSQL и поднимаем пул выше дефолтных 2 соединений.

✕ Потерянный ключ шифрования

Ключ генерируется сам при первом старте в каталоге данных: пересоздали том — credentials мертвы. Задаём N8N_ENCRYPTION_KEY явно и храним копию.

✕ Часовой пояс контейнера

Дефолт GENERIC_TIMEZONE — America/New_York, расписания уезжают на часы. Ставим Europe/Moscow в переменной и в TZ контейнера, проверяем Schedule-ноды.

✕ Разрастание данных исполнений

Дефолты держат 336 часов и 10 000 записей с полными payload. Ставим SAVE_ON_SUCCESS=none, MAX_AGE=168 и следим за размером таблиц исполнений.

✕ Лимиты API внешних систем

Цикл по сотням сделок ловит 429 и молча теряет операции. Батчи, нода Wait между итерациями, Retry On Fail с ростом паузы и обязательная ветка ошибки.

✕ Админка n8n наружу

Публичный UI — это доступ к сохранённым credentials CRM и 1С. Панель прячем за VPN и allow-list, наружу оставляем только вебхуки с проверкой токена.

✕ Бинарные данные в базе

В 2.0 режим in-memory убран: в queue-режиме вложения едут в PostgreSQL и раздувают дамп. Ставим N8N_DEFAULT_BINARY_DATA_MODE=s3 или filesystem.

✕ Вся логика в Code-ноде

Скрипт на 200 строк не диагностируется по шагам, а в 2.0 Code исполняется в task runner без $evaluateExpression. Логику держим в нодах и Sub-workflow.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- n8n и PostgreSQL в Docker Compose, HTTPS через reverse-proxy с Let's Encrypt
- N8N_ENCRYPTION_KEY задан явно, копия — в парольном хранилище
- GENERIC_TIMEZONE=Europe/Moscow и TZ, ночной pg_dump с ретеншном 30 дней
- UI закрыт VPN и allow-list, наружу только вебхуки с проверкой токена

НОРМАЛЬНО

- Queue-режим: Redis и два воркера с --concurrency=5, QUEUE_HEALTH_CHECK_ACTIVE=true
- Error Workflow на каждом боевом сценарии с алертом инженеру
- Прунинг: MAX_AGE=168, PRUNE_MAX_COUNT=5000, SAVE_ON_SUCCESS=none
- Версия образа зафиксирована тегом, есть staging-копия и план отката

ХОРОШО

- N8N_METRICS с queue-метриками в Prometheus, алерт на длину очереди
- N8N_DEFAULT_BINARY_DATA_MODE=s3 для вложений вместо хранения в базе
- Task runners в режиме external: сайдкар n8nio/runners с общим auth-токеном
- Воркфлоу в Git и ежемесячный тест восстановления из резервной копии

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Задан ли N8N_ENCRYPTION_KEY переменной и лежит ли копия в парольном хранилище?
- [ ] База — PostgreSQL, а не SQLite, и снимается ли с неё ночной pg_dump?
- [ ] Выставлены ли GENERIC_TIMEZONE и TZ контейнера в Europe/Moscow?
- [ ] Доступна ли админка n8n из интернета без VPN или IP allow-list?
- [ ] Назначен ли каждому боевому воркфлоу Error Workflow с алертом?
- [ ] Заданы ли EXECUTIONS_DATA_MAX_AGE и PRUNE_MAX_COUNT, знаете ли прирост таблиц?
- [ ] Проверяется ли подпись или токен вебхука до запуска бизнес-логики?
- [ ] Зафиксирована ли версия образа n8n тегом и есть ли проверенный откат?
- [ ] Учётки в CRM и 1С выданы сервисные, с минимально нужными правами?
- [ ] Известны ли лимиты внешних API и заложены ли ретраи с паузой?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Разворачиваем n8n self-hosted под ключ: Compose, PostgreSQL, Redis, HTTPS, бэкапы, мониторинг
- Составляем карту автоматизаций отдела продаж с оценкой экономии часов до начала работ
- Собираем интеграции с CRM, 1С по OData, почтой, мессенджером и платёжным шлюзом
- Ведём воркфлоу на абонентке: правки логики, обновления версий, разбор алертов об ошибках
- Настраиваем доступы, хранение секретов и журналирование под требования по обработке ПДн

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 Executions: переменные окружения исполнений (docs.n8n.io — 2.31)

02 Scaling: Enable queue mode и Control concurrency (docs.n8n.io — 2.31)

03 Database: переменные окружения PostgreSQL (docs.n8n.io — 2.31)

04 Endpoints и Security: переменные окружения (docs.n8n.io — 2.31)

05 Set up task runners: внешний режим (docs.n8n.io — 2.31)

06 Changelog: v2.0 Breaking changes и Release notes 2.x (docs.n8n.io — 2.31.4)

07 Шаблон ITfresh: docker-compose и .env для queue-режима (itfresh.ru — 2026)

08 Стандарт ITfresh: чек-лист приёмки воркфлоу в работу (itfresh.ru — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.