Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

# Kubernetes для малого бизнеса: наш регламент внедрения

Когда кластер оправдан, а когда хватит Docker Compose: версии, пороги, наши шаблоны

Июль 2026

**itfresh.ru · ИТ-аутсорсинг для юридических лиц**

О ЧЁМ ЭТОТ РАЗБОР

# Суть проблемы

К нам приходят компании на 30-80 человек: 8-12 сервисов (сайт, API приложения, биллинг, обмен с 1С) живут на 3-5 виртуалках и деплоятся вручную по SSH. Релиз — 30-40 минут с простоем, откат делается копированием каталога, пик нагрузки кладёт сайт целиком, знания держатся в голове одного админа. Мы решаем не «внедрить Kubernetes», а убрать ручной деплой и единую точку отказа.

## Почему это важно бизнесу

- Простой на релизе бьёт по выручке: заказы не оформляются ровно в те 30-40 минут, когда обновляется сайт
- Откат сломанного релиза вручную занимает часы, а не минуты, и часто теряет часть данных
- Сезонный пик обслуживается «на глаз»: либо переплата за простаивающие ВМ, либо отказ в приём заказов
- Вся эксплуатация завязана на одного человека: отпуск или увольнение останавливает выпуск изменений
- Без изоляции сервисов один утёкший по памяти процесс роняет соседние приложения на той же машине

ЧТО МЫ ЗАКЛАДЫВАЕМ

# Ключевые параметры реализации

**1.37**

актуальная ветка ядра; поддерживаются три минорные версии, ветка живёт ~14 месяцев

*kubernetes.io, Releases*

**март 2026**

ingress-nginx снят с поддержки; новые кластеры собираем сразу на Gateway API v1.6

*kubernetes.io/blog, заявление проекта*

**110**

предел подов на узел в kubelet по умолчанию; наш потолок — 60 на узел 4 vCPU / 8 ГБ

*kubernetes.io, kubelet config*

**100Mi / 10%**

жёсткие пороги вытеснения kubelet по умолчанию: memory.available и nodefs.available

*kubernetes.io, Node-pressure Eviction*

**300 с**

окно стабилизации HPA на снижение по умолчанию; на рост по умолчанию 0 с, мы ставим 60 с

*kubernetes.io, HPA behavior*

**12 ч**

интервал снапшота etcd в k3s по умолчанию (0 */12 * * *), retention 5 снапшотов

*docs.k3s.io, etcd snapshots*

РЕАЛИЗАЦИЯ 1 ИЗ 3

# Managed-кластер под интернет-магазин с сезонным пиком

## Что настраиваем

*Торговая компания, 60 сотрудников: витрина, API приложения, админка, биллинг, обмен с 1С — было 5 ВМ и ручной деплой по SSH*

## Как мы это делаем

1. Неделя 1: инвентаризация сервисов, выносим PostgreSQL в managed-СУБД, а не в StatefulSet; поднимаем dev-кластер на поддерживаемой ветке 1.36
2. Неделя 2: Dockerfile с non-root USER и multi-stage, Helm-чарт из нашего шаблона itf-app: requests/limits, probes, PDB minAvailable: 1
3. Неделя 3: прод-кластер из 3 воркеров 4 vCPU / 8 ГБ, Gateway API v1.6 + cert-manager 1.21 (ACME HTTP-01), kube-prometheus-stack
4. Неделя 4: HPA по CPU 70% (min 2 / max 8), scaleUp stabilizationWindowSeconds: 60, scaleDown: 300; Velero 1.18 в S3, расписание 03:00
5. Стабилизация 4-6 недель: разбор алертов, правка requests по факту, обучение двух инженеров kubectl describe/logs/events

**РЕЗУЛЬТАТ**

Релиз с автодеплоем по тегу — 3-4 минуты без простоя, откат — одна команда отката релиза Helm. Пик закрывается автомасштабированием до 8 реплик и возвратом к 2 через 5 минут. Ночью кластер стоит дешевле пяти постоянно включённых ВМ, а падение любого узла переносит поды за 1-2 минуты.

**КЛЮЧЕВОЙ НЮАНС**

Мы никогда не тащим боевую СУБД в кластер первым шагом. Сначала переезжают stateless-сервисы, а база остаётся в managed-СУБД или уходит под оператор CloudNativePG отдельным этапом.

РЕАЛИЗАЦИЯ 2 ИЗ 3

# Self-hosted k3s на своём железе под требования к локализации

## Что настраиваем

*Производственная компания, 45 мест: персональные данные обязаны лежать на собственных серверах, есть три простаивающих узла*

## Как мы это делаем

1. Проектирование: 3 узла с ролью server (кворум etcd 3 из 3) на k3s ветки 1.36, отдельный сегмент сети и статические адреса, без выхода API наружу
2. Установка: k3s с отключённым traefik, свой ingress-слой на Gateway API v1.6; хранилище — Longhorn с 3 репликами, для БД — оператор CloudNativePG 1.30
3. Резервирование: снапшоты etcd раз в 12 часов с retention 5 и выгрузкой на внешний узел, Velero 1.18 для namespace-объектов и PVC
4. Эксплуатация: ежеквартальное обновление на одну минорную версию, окно на выходных, drain узлов по очереди с cordon и проверкой PDB

**РЕЗУЛЬТАТ**

Данные не покидают периметр, кластер переживает потерю одного узла без вмешательства человека. Плановое обновление минорной версии занимает 40-60 минут и не приводит к простою приложений с двумя и более репликами.

**КЛЮЧЕВОЙ НЮАНС**

Self-hosted оправдан только при выполненных условиях: три узла на кворум, проверенное восстановление etcd из снапшота и двое обученных инженеров. Один узел — это не кластер, а лишний слой абстракции.

РЕАЛИЗАЦИЯ 3 ИЗ 3

# Перевод с ingress-nginx на Gateway API без простоя

## Что настраиваем

*Унаследованный кластер клиента: 14 Ingress-объектов на ingress-nginx, снятом с поддержки в марте 2026, TLS через cert-manager*

## Как мы это делаем

1. Аудит: выгружаем все Ingress, помечаем нестандартные аннотации (rewrite, таймауты, размер тела) — их эквиваленты в Gateway API ищем поимённо
2. Конвертация: ingress2gateway 1.2 генерирует черновые Gateway/HTTPRoute, мы правим руками; TLS переносим на listener шлюза, ClusterIssuer не трогаем
3. Параллельный запуск: новый Gateway на отдельном адресе, проверка по hosts-файлу и синтетическими запросами, затем перевод DNS с TTL 300
4. Снятие старого контроллера через 7 дней после переключения, удаление Ingress-объектов и его RBAC

**РЕЗУЛЬТАТ**

Периметр кластера снова получает обновления безопасности, TLS и маршрутизация описаны стандартными ресурсами, откат на любом шаге — возврат DNS-записи в течение 5 минут.

**КЛЮЧЕВОЙ НЮАНС**

Автоконвертер закрывает базовые правила, но не аннотации. Мы всегда закладываем неделю параллельной работы двух точек входа и переключаем трафик только через DNS с коротким TTL.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

# Подводные камни

- ✕ **Кластер там, где нужен Compose**
  До 8-10 независимо выкатываемых сервисов кластер добавляет работы больше, чем экономит. Мы начинаем с Docker Compose и CI.

- ✕ **Боевая СУБД в простом StatefulSet**
  Без оператора нет корректного failover, бэкапа WAL и point-in-time. Только managed-СУБД или CloudNativePG с проверенным восстановлением.

- ✕ **Нет requests и limits**
  Под без requests планируется куда угодно и вытесняется первым. Один утёкший процесс уводит узел под MemoryPressure и роняет соседей.

- ✕ **Одна реплика и нет PDB**
  При drain узла на обновлении единственная реплика уезжает вместе с ним. Минимум две реплики плюс PodDisruptionBudget.

- ✕ **Устаревшая версия ядра**
  Поддерживаются только три минорные ветки. Пропустив два цикла, клиент остаётся без исправлений безопасности и без пути обновления в один шаг.

- ✕ **Секреты в git**
  Объект Secret — это base64, а не шифрование. Мы храним их в Sealed Secrets или внешнем хранилище, в репозиторий попадает только шифротекст.

- ✕ **Отсутствие probes**
  Без readiness трафик идёт в неготовый под, без liveness зависший процесс живёт вечно. Проверки задаём отдельными эндпоинтами, а не одним.

- ✕ **Бэкап только на уровне ВМ**
  Снапшот узла не восстановит объекты кластера. Нужны Velero для манифестов и PVC плюс снапшоты etcd на self-hosted.

УРОВНИ РЕАЛИЗАЦИИ

# Как правильно

## МИНИМУМ

- Поддерживаемая минорная версия ядра, обновление не реже раза в полгода
- Две реплики, requests/limits и readiness-проба у каждого сервиса
- Gateway API вместо снятого с поддержки ingress-nginx, TLS через cert-manager
- Velero в объектное хранилище с ежедневным расписанием и проверкой восстановления

## НОРМАЛЬНО

- HPA по CPU 70%, min 2 / max 8, окно scale-up 60 с, scale-down 300 с
- kube-prometheus-stack с алертами по CrashLoopBackOff, OOMKilled, PVC 85%
- Отдельные namespace с ResourceQuota и LimitRange под каждое окружение
- Секреты через Sealed Secrets, доступ по RBAC на уровне namespace

## ХОРОШО

- GitOps: состояние кластера только из репозитория, ручной kubectl apply запрещён
- PodDisruptionBudget и topologySpreadConstraints по узлам для всех сервисов
- CloudNativePG 1.30 с проверяемым восстановлением на точку во времени раз в квартал
- Политики Kyverno: запрет latest, privileged и подов без limits

ПРОВЕРЬТЕ У СЕБЯ

# Чек-лист самопроверки

- [ ] Версия ядра входит в три поддерживаемые ветки, дата окончания поддержки записана в паспорт стенда
- [ ] У каждого пода заданы requests и limits по CPU и памяти, ключевые сервисы в классе Guaranteed
- [ ] У каждого сервиса не менее двух реплик и PodDisruptionBudget с minAvailable
- [ ] Readiness и liveness пробы отвечают на разных эндпоинтах и не ходят во внешние зависимости
- [ ] Точка входа переведена на Gateway API, старые Ingress-объекты и их контроллер удалены
- [ ] Сертификаты выпускаются cert-manager автоматически, срок до продления контролируется алертом
- [ ] Velero делает ежедневную копию манифестов и PVC, восстановление проверено на тестовом namespace
- [ ] На self-hosted снапшоты etcd раз в 12 часов уезжают на внешний узел, восстановление отработано
- [ ] Секреты не хранятся в репозитории в открытом виде, доступ разграничен по RBAC
- [ ] Диагностику pod, events и logs умеют выполнять минимум два человека в команде клиента

*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).

- Считаем честный TCO и честно говорим, когда кластер не нужен и хватит Docker Compose с CI
- Собираем кластер под ключ: Gateway API, cert-manager, мониторинг, HPA, бэкапы, наши Helm-шаблоны
- Переводим унаследованные кластеры с ingress-nginx на Gateway API через параллельный контур
- Ведём эксплуатацию: обновление минорных версий по регламенту, дежурство по алертам, разбор инцидентов
- Обучаем ваших инженеров работе с kubectl и разбору отказов, чтобы вы не зависели от одного человека

**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 Kubernetes Releases и Patch Releases: срок поддержки ветки (kubernetes.io — 2026)

02 Заявление проекта о снятии ingress-nginx с поддержки (kubernetes.io/blog — 2026)

03 Node-pressure Eviction и Kubelet Configuration (kubernetes.io — 2026)

04 Horizontal Pod Autoscaling: поведение масштабирования (kubernetes.io — 2026)

05 Gateway API v1.6 и утилита ingress2gateway 1.2 (gateway-api.sigs.k8s.io — 2026)

06 cert-manager 1.21: Supported Releases и ACME Issuer (cert-manager.io — 2026)

07 Velero 1.18: расписания резервных копий и восстановление (velero.io — 2026)

08 CloudNativePG 1.30: Backup, Recovery, Supported Releases (cloudnative-pg.io — 2026)

09 k3s: снапшоты etcd и высокая доступность (docs.k3s.io — 2026)

10 Наши шаблоны: Helm-чарт itf-app и паспорт стенда (itfresh.ru — 2026)


*Основано на официальной документации продуктов и нашей практике внедрения.*