Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Linux и Docker в малом офисе: наш стандарт эксплуатации

Как мы собираем платформу: Debian 13, Docker Compose, Traefik, Prometheus 3 и BorgBackup

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Внутренние сервисы компании — GitLab, Nextcloud, CRM, wiki — часто живут на одном сервере «как поставили»: Docker из репозитория дистрибутива, логи без ротации, порты открыты наружу, бэкапов нет. Сбой диска или кривое обновление кладёт всё сразу, а восстановление растягивается на дни, потому что конфигурация нигде не зафиксирована.

Почему это важно бизнесу

- Простой сервиса = простой отдела: без CRM, файлов и задач работа встаёт по всей компании
- Восстановление «по памяти» занимает дни; из git-репозитория со стеками — часы
- Диск, забитый логами контейнеров, роняет сервер внезапно и в худший момент
- Порты контейнеров, открытые в обход фаервола, — прямой путь к взлому и утечке
- Зависимость от одного админа: без зафиксированных конфигов его уход — риск для бизнеса

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

29.x

Docker Engine: ставим docker-ce из репо download.docker.com, а не docker.io из дистрибутива

Docker Engine 29 · docs.docker.com

Debian 13

Базовая ОС хоста: trixie, unattended-upgrades только security, отдельный том /var/lib/docker

наш стандарт

10m × 3

Ротация логов json-file в daemon.json (max-size/max-file) — иначе логи съедают диск за месяцы

docs.docker.com · daemon.json

15s / 30d

Prometheus 3: scrape_interval 15s, хранение метрик --storage.tsdb.retention.time=30d

Prometheus 3.13 LTS · prometheus.io

7/4/6

borg prune: keep-daily 7, keep-weekly 4, keep-monthly 6 + ежемесячный borg check --verify-data

BorgBackup 1.4 · наш стандарт

2 мин

Порог алерта up == 0 for 2m на падение хоста или экспортёра — ловим аварию без флаппинга

наш стандарт

РЕАЛИЗАЦИЯ 1 ИЗ 3

Хост-платформа: закалённый Linux + Docker Compose

Что настраиваем

Одиночный сервер клиента под внутренние сервисы: Debian 13, 4–8 vCPU, LVM, все стеки в /opt/stacks

Как мы это делаем

1. Закалка хоста: SSH key-only (PasswordAuthentication no), nftables, unattended-upgrades, отдельный LVM-том под /var/lib/docker
2. Docker Engine 29.x из apt-репо download.docker.com; daemon.json: json-file (max-size 10m, max-file 3), live-restore true, default-address-pools
3. Каждый сервис — свой Compose-стек в /opt/stacks/<имя>: изолированная сеть, пин версии образа, .env с правами 600, обязательный healthcheck
4. Traefik v3 — единая точка входа: entryPoints 80/443, ACME Let's Encrypt, маршруты через labels; порты приложений наружу не публикуем
5. /opt/stacks в git; деплой = git pull + docker compose up -d --pull always; откат = checkout предыдущего тега

РЕЗУЛЬТАТ

Сервер восстанавливается с нуля за 2–3 часа по репозиторию и бэкапу; обновления и откаты предсказуемы; новый сервис добавляется шаблоном стека за 30–60 минут без риска для соседних.

КЛЮЧЕВОЙ НЮАНС

live-restore: true даёт рестарт демона без падения контейнеров; default-address-pools задаём сразу — дефолтная 172.17.0.0/16 конфликтует с VPN и филиальными сетями клиентов.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Мониторинг: Prometheus 3 + Alertmanager + Grafana

Что настраиваем

Все Linux-хосты и контейнеры клиента; отдельная VM мониторинга вне продакшн-сервера

Как мы это делаем

1. node_exporter systemd-юнитом на каждом хосте, cAdvisor — по контейнерам; порт 9100 открыт только для сети мониторинга
2. Prometheus 3.x: scrape_interval 15s; правила в rules/*.yml — диск <15%, inode, load, память, up == 0 for 2m
3. Alertmanager → Telegram: маршруты по severity, group_by [alertname, instance], repeat_interval 4h, inhibit_rules против каскада
4. Grafana 13: Node Exporter Full + наш дашборд «Сервисы клиента» со статусами healthcheck и сроками сертификатов
5. blackbox_exporter: HTTPS-проверки сайта и публикаций (1С, Nextcloud) каждые 60 с, алерт на истечение TLS за 14 дней

РЕЗУЛЬТАТ

Об аварии узнаём мы, а не бухгалтер утром: падение хоста, диска или сервиса приходит в Telegram за 2–3 минуты; по дашбордам деградация (диск, RAM, латентность) видна за недели до отказа.

КЛЮЧЕВОЙ НЮАНС

Без inhibit_rules падение одного хоста рождает десятки алертов по контейнерам — глушим дочерние правилом по instance; условие for 2m убирает флаппинг при коротких сетевых морганиях.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Бэкап и проверка восстановления: BorgBackup

Что настраиваем

Тома контейнеров, дампы БД и /opt/stacks; borg-репозиторий на втором узле + оффсайт-копия

Как мы это делаем

1. Перед архивом — консистентные дампы: docker exec pg_dump / mysqldump в /backup/dumps; тома БД «на живую» не бэкапим
2. borg create с compression zstd и дедупликацией; расписание — systemd timer OnCalendar 02:30 + RandomizedDelaySec, OnFailure= юнит алерта в Telegram
3. borg prune --keep-daily 7 --keep-weekly 4 --keep-monthly 6; ежемесячно borg check --verify-data
4. Репозиторий в режиме borg serve --append-only — шифровальщик с хоста не удалит историю копий
5. Раз в квартал — тестовое восстановление: borg extract на резервный хост, docker compose up, проверка входа в приложение

РЕЗУЛЬТАТ

RPO — сутки (для БД можно чаще), проверенный RTO — 2–4 часа на полный стек; дедупликация со сжатием zstd держит 6 месяцев истории в 1,5–2 объёмах исходных данных.

КЛЮЧЕВОЙ НЮАНС

Бэкап без тестового восстановления — это гипотеза: раз в квартал реально поднимаем стек из архива; append-only и отдельные ключи доступа — репозиторий переживает компрометацию хоста.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **Docker из репозитория дистрибутива**
  Пакет docker.io отстаёт на мажорные версии и security-патчи; ставим docker-ce из официального apt-репо download.docker.com
- ✕ **Логи контейнеров без ротации**
  json-file по умолчанию не ограничен — гигабайты в /var/lib/docker; задаём max-size=10m, max-file=3 в daemon.json
- ✕ **Порты в обход фаервола**
  -p 8080:80 пишет правила в iptables мимо nftables/ufw INPUT; публикуем на 127.0.0.1 и выпускаем наружу только через Traefik
- ✕ **Прод на образах :latest**
  Обновление прилетает незаметно и ломает совместимость; пиним major.minor и обновляем через staging-прогон
- ✕ **Сети Docker поверх корпоративных**
  Дефолтная 172.17.0.0/16 пересекается с VPN и филиалами — трафик уходит не туда; задаём default-address-pools заранее
- ✕ **Секреты в environment**
  Переменные видны в docker inspect любому с доступом к сокету; .env с правами 600, секреты — файлами, сокет в контейнеры не пробрасываем
- ✕ **restart: always вместо healthcheck**
  Процесс жив, приложение висит — рестарта не будет; пишем healthcheck (interval 30s, retries 3) и алертим по его статусу
- ✕ **Бэкап тома с работающей БД**
  Файлы PostgreSQL/MySQL в момент копирования неконсистентны — архив не восстановится; только дамп или остановка контейнера

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- docker-ce из официального репо + ротация логов 10m×3 в daemon.json
- SSH только по ключам; порты контейнеров — на 127.0.0.1, наружу через прокси
- Ночной BorgBackup дампов БД и томов на отдельный узел
- Проверка доступности сервисов с алертом в Telegram (blackbox/Uptime Kuma)

НОРМАЛЬНО

- Все Compose-стеки и конфиги в git, деплой только из репозитория
- Prometheus 3 + Alertmanager + Grafana, экспортёры на всех хостах
- Traefik v3 с ACME-сертификатами как единая точка входа
- systemd-таймеры вместо cron: журнал в journald + OnFailure-алерт

ХОРОШО

- Ansible-плейбуки: закалка, Docker, экспортёры — идемпотентно на весь парк
- Append-only borg-репозиторий + квартальное тестовое восстановление
- Staging-прогон обновлений образов перед продом, пин версий
- Rootless Podman / userns-remap для сервисов с внешним трафиком

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Docker Engine установлен из официального репозитория и обновляется по плану, а не «как получится»?
- [ ] Ограничена ли ротация логов контейнеров (max-size/max-file в daemon.json)?
- [ ] Порты контейнеров закрыты фаерволом или привязаны к 127.0.0.1 за reverse proxy?
- [ ] Все Compose-файлы и конфиги лежат в git и разворачиваются одной командой?
- [ ] У каждого продакшн-контейнера есть healthcheck и алерт при его падении?
- [ ] Бэкапятся ли дампы БД, а не только файлы томов «на живую»?
- [ ] Проверялось ли восстановление из бэкапа за последние 3 месяца?
- [ ] Придёт ли алерт в Telegram при падении хоста, диске <15% и недоступном сервисе?
- [ ] Версии образов зафиксированы, или прод живёт на :latest?
- [ ] Docker-сети не пересекаются с VPN и сетями филиалов (default-address-pools)?

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 платформу под ключ: закалённый Linux-хост, Docker, Traefik, мониторинг, бэкап — за 3–5 дней
- Переносим существующие сервисы (GitLab, Nextcloud, CRM, wiki) в Compose-стеки с конфигами в git
- Настраиваем Prometheus/Grafana/Alertmanager или Zabbix с алертами в Telegram под вашу инфраструктуру
- Строим бэкап BorgBackup по схеме 3-2-1 с проверочными восстановлениями и отчётом
- Сопровождаем: обновления Docker и образов, реакция на алерты, дежурство по SLA

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 Docker Engine — install, daemon.json, release notes (docs.docker.com — 29.x)

02 Compose Specification — healthcheck, networks, secrets (docs.docker.com — 2026)

03 Prometheus — configuration, storage/retention (prometheus.io — 3.13 LTS)

04 Alertmanager — routing, inhibition (prometheus.io — 0.33)

05 Grafana — dashboards & alerting (grafana.com — 13.x)

06 BorgBackup — create/prune/check, append-only (borgbackup.readthedocs.io — 1.4)

07 Traefik Proxy — ACME / Let's Encrypt (doc.traefik.io — v3.7)

08 systemd.timer — OnCalendar, RandomizedDelaySec (freedesktop.org — 258)

Основано на официальной документации продуктов и нашей практике внедрения.