Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Caddy vs Nginx в 2026: автоматический HTTPS без certbot

Как мы строим TLS-фронт на Caddy 2.11 и по каким критериям оставляем nginx 1.30

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

У типового клиента 10–50 доменов на одном nginx: сертификаты Let's Encrypt продлевает certbot, и каждое продление — точка отказа: занят порт 80, протух DNS-токен, новый сайт забыли внести в cron. Просроченный сертификат — это браузерная заглушка вместо сайта: встают ЛК, веб-1С и API-интеграции. Мы перестраиваем периметр так, чтобы выпуск и продление сертификатов не требовали р...

Почему это важно бизнесу

- Просроченный сертификат останавливает продажи: браузер показывает «не защищено», конверсия падает до нуля за минуты
- Интеграции (платежи, API, обмены 1С) рвутся молча — TLS-ошибку видно только в логах у партнёра
- Каждый ручной выпуск/починка сертификата — 20–30 минут инженера; на парке из 50 доменов это часы ежемесячно
- Ночной инцидент с продлением = простой до утра, если процесс не автоматизирован и не мониторится

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

2.11.x

Текущая ветка Caddy: ставим из официального deb-репозитория, версию фиксируем apt-пином

по докам caddyserver.com

1.30/1.31

nginx stable/mainline: в прод берём только stable, mainline — на стендах

по докам nginx.org

30 дней

Caddy продлевает сертификат при остатке 1/3 срока: для 90-дневных Let's Encrypt — за ~30 дней

Caddy: Automatic HTTPS

:2019

Локальный admin API Caddy (localhost:2019): конфиг накатываем POST /load без рестарта процесса

Caddy: API

10 с

health_interval активных проверок upstream в reverse_proxy; passive-отсечка fail_duration 30 с

наш стандарт

2/сутки

certbot renew по systemd-timer там, где остаётся nginx; deploy-hook обязан делать reload nginx

по докам certbot

РЕАЛИЗАЦИЯ 1 ИЗ 3

TLS-фронт на Caddy: от установки до on-demand TLS

Что настраиваем

Периметр клиента: один VPS-реверс-прокси перед сайтами, ЛК и веб-публикациями 1С (до 50 доменов)

Как мы это делаем

1. Ставим caddy из официального deb-репозитория (сервис caddy.service); глобальные опции: email для ACME-аккаунта, admin localhost:2019
2. Caddyfile: reverse_proxy на бэкенды с health_uri /health и health_interval 10s, encode zstd gzip, php_fastcgi unix//run/php/php8.3-fpm.sock для PHP-сайтов
3. Wildcard и контуры без открытого 80-го — DNS-01: собираем xcaddy build --with github.com/caddy-dns/<провайдер>, токен API кладём в env через systemd-override
4. Для мультиарендных доменов включаем on_demand_tls строго с ask-эндпоинтом — сертификат выдаётся только доменам из нашего белого списка
5. Перед боем прогоняем выдачу на staging-CA Let's Encrypt (директива acme_ca на staging-каталог), конфиг проверяем caddy validate

РЕЗУЛЬТАТ

Один бинарь без зависимостей закрывает выпуск и продление сертификатов (за ~30 дней до истечения), редирект 80→443, HTTP/2 и HTTP/3 из коробки. Добавление домена — 3–5 строк и caddy reload без обрыва соединений; cron-задач и certbot-хуков на хосте нет вообще.

КЛЮЧЕВОЙ НЮАНС

В разделе документации Automatic HTTPS ключевое: у Caddy издатель Let's Encrypt плюс резервный ZeroSSL (активен при заданном email), с повторами и бэкоффом — но выдача сработает, только если DNS домена уже ука…

РЕАЛИЗАЦИЯ 2 ИЗ 3

Миграция парка сайтов с nginx+certbot на Caddy

Что настраиваем

Хост с 30–50 vhost: статика, PHP-FPM, reverse proxy на Node и публикации 1С

Как мы это делаем

1. Инвентаризация: выгружаем server_name/proxy_pass из /etc/nginx, сверяем с DNS; сайты с Lua/njs и спецмодулями помечаем «остаются на nginx»
2. Конвертируем в один Caddyfile: file_server, php_fastcgi, reverse_proxy; десятки vhost ужимаются с тысяч строк nginx-конфигов до пары сотен
3. Обкатываем Caddy параллельно на :8443 со staging-CA, каждый vhost проверяем curl --resolve до переключения
4. Переключаем одним окном: systemctl stop nginx && systemctl start caddy, выдачу всех сертификатов контролируем в journalctl -u caddy -f
5. Хранилище ключей /var/lib/caddy включаем в бэкап с сохранением владельца caddy:caddy; certbot и его таймеры с хоста убираем

РЕЗУЛЬТАТ

Инциденты с продлением уходят как класс: нет cron-задач certbot, нет ручных выпусков для новых доменов. Ввод нового сайта — минуты вместо десятков минут, конфигурация парка читается целиком в одном файле и живёт в Git.

КЛЮЧЕВОЙ НЮАНС

Главный риск миграции — боевые лимиты CA: у Let's Encrypt до 5 повторных выпусков на одинаковый набор имён в неделю. Все репетиции — только через staging-CA, боевой каталог включаем один раз при переключении.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Когда оставляем nginx — и приводим его к нашему стандарту

Что настраиваем

Фронты с нагрузкой 50k+ RPS, сборки с Lua/njs, Ingress-контроллеры Kubernetes

Как мы это делаем

1. Базис: nginx 1.30 stable, ssl_protocols TLSv1.2 TLSv1.3, ssl_session_cache shared:SSL:10m, ssl_session_timeout 1d, ssl_prefer_server_ciphers off
2. certbot по systemd-timer дважды в сутки; deploy-hook «nginx -t && systemctl reload nginx» — без reload воркеры держат старый сертификат в памяти
3. Отдельный default_server с ssl_reject_handshake on — по голому IP сервер не отдаёт сертификат первого попавшегося vhost
4. HTTP/3: пакеты nginx.org уже собраны с ngx_http_v3_module (проверяем nginx -V), listen 443 quic reuseport + заголовок Alt-Svc, на фаерволе открываем UDP/443
5. HSTS max-age=63072000 и security-заголовки выносим в общий include conf.d/security.conf, чтобы профиль был одинаков на всех площадках

РЕЗУЛЬТАТ

nginx остаётся там, где нужны fine-tuning и модули, но ведёт себя предсказуемо: сертификаты продлеваются таймером и подхватываются reload-ом автоматически, TLS-профиль единый на всех площадках клиента, по IP чужой vhost не светится.

КЛЮЧЕВОЙ НЮАНС

Let's Encrypt отключил свои OCSP-ответчики — ssl_stapling для их сертификатов больше не работает, из конфигов убираем; актуальность отзыва теперь закрывается коротким сроком жизни сертификата и продлением по A…

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

✕ **apt-обновление затирает xcaddy-сборку**
Пакет caddy ставит стоковый бинарь без DNS-модулей. Закрепляем свою сборку через dpkg-divert и пересобираем xcaddy той же версией, что в репозитории.

✕ **on_demand TLS без ask-эндпоинта**
Любой чужой CNAME на наш IP заставит сервер выпускать сертификаты и сожжёт лимиты CA. Всегда on_demand_tls { ask } с проверкой домена по белому списк…

✕ **Сожгли rate-limit Let's Encrypt**
Повторные тестовые выпуски упираются в лимит дубликатов (5/нед на набор имён). Все тесты — на staging-CA, боевой каталог включаем один раз.

✕ **ssl_stapling с Let's Encrypt**
LE выключил OCSP-ответчики: ssl_stapling on даёт только ошибки в error.log. Для LE-сертификатов директивы stapling из nginx-конфигов убираем.

✕ **Порт 80 занят при параллельной обкатке**
Пока nginx держит :80, Caddy не пройдёт HTTP-01. Обкатываем через DNS-01 и staging-CA, а HTTP-01 включается только в момент переключения.

✕ **После renew nginx отдаёт старый серт**
certbot обновил файлы, но воркеры держат старые в памяти до reload. В deploy-hook обязательно nginx -t && systemctl reload nginx.

✕ **HTTP/3 «включён», но не работает**
Забытый UDP/443 на фаерволе или в облачном security-group. Открываем UDP и проверяем реальным curl --http3 с внешнего хоста.

✕ **Права на /var/lib/caddy после переноса**
Restore бэкапа теряет владельца caddy:caddy — сервис не читает ключи и перевыпускает всё заново. Проверяем права и владельца после любого переноса.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Единый TLS-фронт: Caddy 2.11 из официального репозитория, email ACME в глобальных оп…
- Если остаётся nginx — certbot по systemd-timer + deploy-hook с reload
- TLS только 1.2/1.3, HSTS и редирект 80→443 на каждом vhost

НОРМАЛЬНО

- DNS-01 для wildcard и закрытых контуров; DNS-токены в systemd env, не в конфиге
- Внешний мониторинг сроков сертификатов (Zabbix, алерт за 14 дней до истечения)
- Бэкап /var/lib/caddy и /etc/letsencrypt вместе с конфигами и правами
- Любые тесты выдачи — только через staging-CA Let's Encrypt

ХОРОШО

- on-demand TLS с ask-эндпоинтом для мультиарендных/клиентских доменов
- Конфиг в Git: caddy fmt + caddy validate в CI, деплой через admin API POST /load
- HTTP/3 на обоих: Caddy из коробки, nginx — listen 443 quic + открытый UDP/443
- Разделение ролей: Caddy — TLS-периметр, бэкенды за ним без публичных портов

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Продление сертификатов автоматическое и проверено вхолостую (certbot renew --dry-run или staging-CA)?
- [ ] Есть внешний мониторинг срока действия сертификатов с алертом минимум за 14 дней?
- [ ] Новый сайт вводится без ручной возни с сертификатом — одна секция конфига и reload?
- [ ] Wildcard-домены продлеваются по DNS-01, и токен DNS-API не лежит в открытом конфиге?
- [ ] Reload веб-сервера проходит без обрыва соединений и стоит в deploy-hook после каждого renew?
- [ ] Каталоги с ключами (/var/lib/caddy, /etc/letsencrypt) попадают в бэкап с сохранением прав?
- [ ] Тестовые выпуски идут на staging-CA и не сжигают боевые лимиты Let's Encrypt?
- [ ] По голому IP сервер не отдаёт сертификат первого попавшегося сайта (ssl_reject_handshake)?
- [ ] HTTP/3 реально работает: UDP/443 открыт и проверен curl --http3 снаружи?
- [ ] Из nginx-конфигов убран ssl_stapling для сертификатов Let's Encrypt?

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).

- Аудит текущего nginx+certbot: инвентаризация vhost, сроки сертификатов, точки отказа продления
- Миграция парка сайтов на Caddy: обкатка на staging-CA, переключение в одно технологическое окно
- Приведение остающегося nginx к нашему TLS-стандарту (нагрузка, Lua/njs, Ingress)
- Мониторинг сертификатов и доступности в Zabbix с алертами в Telegram
- Поддержка под ключ: обновления, бэкап ключей, разбор инцидентов с TLS

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 Automatic HTTPS (выпуск, продление, издатели) (caddyserver.com — v2.11)

02 Caddyfile: reverse_proxy, tls, on_demand_tls (caddyserver.com — v2.11)

03 Build from source / xcaddy и DNS-модули (caddyserver.com — 2026)

04 ngx_http_ssl_module (nginx.org — 1.30)

05 ngx_http_v3_module (QUIC/HTTP-3) (nginx.org — 1.30)

06 Certbot User Guide: renew, deploy-hooks (certbot.eff.org — 2026)

07 Let's Encrypt: Rate Limits, OCSP EoL, ARI (letsencrypt.org — 2026)

08 Шаблон ITfresh: Caddyfile TLS-фронта (itfresh.ru — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.