Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

iptables и nftables: firewall для Linux-сервера и шлюза офиса

Наш базовый ruleset, deny-all, NAT-шлюз, тюнинг conntrack и переход на nft-бэкенд

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Две типовые ситуации. Первая: боевой Linux-сервер стоит в интернете с политикой ACCEPT на INPUT, наружу торчат 5432, 9100, 873 и панель управления. Вторая: правила есть, но набиты руками в консоли, после перезагрузки исчезают, а на шлюзе офиса раз в неделю «пропадает интернет» из-за переполненного conntrack. Мы приводим фильтрацию к воспроизводимому файлу правил.

Почему это важно бизнесу

- Открытый наружу порт СУБД или бэкапа — это утечка всей базы клиентов и остановка работы бухгалтерии на сутки
- Правила, не сохранённые в rules.v4, теряются при первой же перезагрузке — сервер молча остаётся голым
- Переполнение nf_conntrack на шлюзе выглядит как «интернет тормозит у всех» и съедает часы диагностики
- Ручные правила без комментариев и версий невозможно передать другому инженеру — растёт зависимость от одного человека
- Дропы без логирования превращают любой инцидент в гадание: почему 1С не видит сервер приложений

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

1.8.11

версия iptables в Debian 13: это фронтенд iptables-nft, legacy-бэкенд остался лишь опцией

iptables 1.8.11-2, Debian trixie

1.1.3

версия nft в Debian 13: атомарный nft -f, named sets, flowtable; upstream уже 1.1.7

netfilter.org, nftables downloads

432000 с

дефолт nf_conntrack_tcp_timeout_established (5 суток); на офисных шлюзах режем до 86400 с

docs.kernel.org, nf_conntrack-sysctl

x4

наше соотношение nf_conntrack_max к buckets; в ядре по умолчанию max = buckets

наш стандарт

5/min

лимит -m limit в цепочке LOGDROP: картина дропов видна, journald и диск не забиваются

наш стандарт

100

дефолт ip_list_tot модуля xt_recent — потолок адресов в таблице антибрута, поднимаем осознанно

man iptables-extensions

РЕАЛИЗАЦИЯ 1 ИЗ 3

Базовый периметр одиночного сервера: deny-all и conntrack

Что настраиваем

VPS или физический сервер с одной ролью (веб, 1С-публикация, почта, БД): Debian 12/13, Ubuntu 24.04 LTS, AlmaLinux 9

Как мы это делаем

1. Пишем не команды в консоли, а файл /etc/iptables/rules.v4 в формате iptables-save и применяем iptables-restore: порядок правил воспроизводим
2. Политики -P INPUT DROP, -P FORWARD DROP; первыми правилами -i lo -j ACCEPT и -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT, затем INVALID -j DROP
3. Управляющие порты (22/tcp, панели, 5432, 9100, 873) открываем только на источники из ipset/nft set office_nets; публичными оставляем строго 80 и 443
4. Антибрут на SSH: -m recent --set --name SSH --rsource плюс --update --seconds 60 --hitcount 4 -j LOGDROP; на нагруженных узлах дублируем fail2ban
5. Все DROP заворачиваем в свою цепочку LOGDROP: -m limit --limit 5/min -j LOG --log-prefix IPT-DROP --log-level 4, следом -j DROP

РЕЗУЛЬТАТ

Сервер отвечает только по тем портам, которые описаны в файле правил, а не по тем, что случайно слушает софт. Правила переживают перезагрузку через netfilter-persistent или iptables-services, а любое падение доступа диагностируется по префиксу IPT-DROP в journald за минуты.

КЛЮЧЕВОЙ НЮАНС

Порядок правил важнее их количества: ESTABLISHED,RELATED всегда первым, иначе deny-all рвёт вашу же SSH-сессию. Перед применением всегда держим отложенный откат через at или systemd-run.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Шлюз офиса: NAT, форвардинг и ёмкость conntrack

Что настраиваем

Linux-шлюз на границе LAN до 50 рабочих мест: WAN-интерфейс, LAN 10.x/24, туннели VPN, проброс сервисов внутрь

Как мы это делаем

1. Маршрутизацию включаем декларативно: net.ipv4.ip_forward=1 в /etc/sysctl.d/, а не sysctl -w на живую; rp_filter=2 на мультиканальном шлюзе, send_redirects=0
2. SNAT для трафика LAN: MASQUERADE на динамическом WAN, но при статическом белом IP ставим -j SNAT --to-source — дешевле по CPU и стабильнее при флапе
3. FORWARD держим DROP: разрешаем LAN->WAN по интерфейсам, обратно только ESTABLISHED,RELATED; DNAT всегда дополняем парным правилом FORWARD на NEW
4. Тюним conntrack: hashsize в /etc/modprobe.d/nf_conntrack.conf, nf_conntrack_max = hashsize x4, свои udp_timeout и tcp_timeout_established
5. На шлюзах с большим PPS переводим ruleset на nft: базовый набор через nft -f /etc/nftables.conf, наборы адресов — named sets, форвард — flowtable

РЕЗУЛЬТАТ

Офис перестаёт ловить плавающие обрывы на пике нагрузки: таблица соединений имеет запас, а её заполнение видно на графике заранее. Проброшенные сервисы работают одинаково снаружи и изнутри, а не «только из дома». Ruleset шлюза лежит в git и разворачивается на резервный узел за минуты.

КЛЮЧЕВОЙ НЮАНС

Классическая ловушка — DNAT есть, а FORWARD забыт: пакет транслируется и тут же дропается политикой. Второй нюанс — nf_conntrack_buckets пишется только в initial netns, hashsize надёжнее задать модулю.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Эксплуатация: сохранение, миграция на nftables и диагностика

Что настраиваем

Парк Linux-узлов на обслуживании: смешанные Debian, Ubuntu LTS и RHEL-совместимые, разные поколения бэкендов netfilter

Как мы это делаем

1. Фиксируем бэкенд: update-alternatives для iptables на Debian, alternatives на RHEL; legacy и nft на узле не смешиваем — наборы не видят друг друга
2. Миграцию делаем через iptables-save и iptables-restore-translate, результат вычитываем руками: часть модулей xtables переводится не один в один
3. Сохранение: netfilter-persistent и rules.v4/rules.v6 на Debian, iptables-services на RHEL 9, на RHEL 10 — только nftables.service; файл правил в git
4. Диагностика по чек-листу: iptables -L -v -n --line-numbers, обнуление счётчиков -Z, TRACE в таблице raw или nft monitor trace, conntrack -L, ss -tlnp
5. Мониторинг: счётчики правил и nf_conntrack_count в Zabbix, отдельные триггеры на рост дропов и заполнение таблицы соединений выше 80%

РЕЗУЛЬТАТ

Правила становятся управляемым артефактом, а не устной легендой: изменения проходят через git, откат занимает одну команду. Инциденты вида «сервис не отвечает» закрываются по счётчикам и трассировке, а не перебором гипотез, и на этом экономится основная часть времени поддержки.

КЛЮЧЕВОЙ НЮАНС

IPv6 забывают почти всегда: пустой ip6tables при поднятом IPv6 сводит на нет весь ruleset v4. Правила v6 пишем и сохраняем синхронно с v4, отдельным файлом rules.v6.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- **Правила живут только в оперативной памяти**
  iptables не сохраняет ruleset сам. Держим правила в rules.v4 и применяем через netfilter-persistent или nftables.service при старте.
- **Смешаны бэкенды legacy и nft**
  Часть правил создана iptables-legacy, часть iptables-nft; наборы не видят друг друга. Фиксируем один бэкенд через update-alternatives.
- **Ставка на iptables там, где его уже нет**
  В RHEL 9 iptables-nft объявлен устаревшим, в RHEL 10 периметр строим на nftables и firewalld. Новые узлы сразу описываем в nft-синтаксисе.
- **Пустой ip6tables при живом IPv6**
  Сервис слушает на ::, а фильтруется только v4. Пишем зеркальный ruleset v6 и сохраняем в rules.v6, либо гасим IPv6 осознанно.
- **DNAT без парного правила в FORWARD**
  Трансляция срабатывает, а политика FORWARD DROP убивает пакет. Каждый проброс дополняем разрешением на --ctstate NEW по интерфейсам.
- **Дефолтная ёмкость conntrack на шлюзе**
  При переполнении ядро пишет table full и рвёт новые сессии. Считаем nf_conntrack_max от hashsize и держим таблицу под мониторингом.
- **DROP без логирования**
  Пакет исчезает без следа, и диагностика превращается в перебор. Все дропы уводим в цепочку LOGDROP с -m limit и понятным префиксом.
- **Применение правил без плана отката**
  Опечатка в правиле для SSH отрезает доступ к удалённому узлу. Перед применением ставим отложенный iptables-restore на прежний файл через at.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Политика DROP на INPUT и FORWARD, ACCEPT только на явно нужные порты
- Первое правило INPUT — ESTABLISHED,RELATED, второе — DROP для INVALID
- Правила сохранены в rules.v4/rules.v6 и поднимаются при загрузке
- Управляющие порты (SSH, панели, СУБД) закрыты на источники из белого списка

НОРМАЛЬНО

- Ruleset лежит в git, применяется файлом, а не командами в консоли
- Своя цепочка LOGDROP с -m limit --limit 5/min и внятным префиксом
- Антибрут на SSH: xt_recent или fail2ban в отдельной цепочке
- Зеркальный ruleset для IPv6, а не молчаливо открытый ip6tables

ХОРОШО

- Новые шлюзы и все узлы RHEL 10 — на nftables: nft -f, named sets, flowtable
- Тюнинг conntrack: hashsize в modprobe.d, max = hashsize x4, свои таймауты
- Счётчики правил и nf_conntrack_count сняты в Zabbix, есть триггеры
- Отложенный откат ruleset и проверка правил на стенде перед боем

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Политика по умолчанию на INPUT и FORWARD стоит в DROP, а не в ACCEPT?
- [ ] Правила переживают перезагрузку — проверяли реальным ребутом, а не только на словах?
- [ ] Знаете, какой бэкенд активен на узле: iptables-legacy, iptables-nft или чистый nft?
- [ ] Для IPv6 есть свой набор правил, или ip6tables пустой при поднятом IPv6?
- [ ] Каждый проброс DNAT дополнен разрешающим правилом в цепочке FORWARD?
- [ ] СУБД, экспортеры метрик и rsync закрыты на конкретные источники, а не на 0.0.0.0/0?
- [ ] Дропы логируются с rate-limit и своим префиксом, или пакеты исчезают бесследно?
- [ ] Известна текущая заполненность nf_conntrack и порог, за которым шлюз начнёт рвать сессии?
- [ ] Файл правил хранится в системе контроля версий с комментариями к строкам?
- [ ] Есть отработанный сценарий отката, если новое правило отрежет удалённый доступ?

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 ruleset: находим открытые наружу сервисы, мёртвые и конфликтующие правила
- Разработка базового набора правил под роль сервера и его сохранение в rules.v4/rules.v6
- Сборка Linux-шлюза офиса: NAT, пробросы, VPN-сегменты, тюнинг conntrack и sysctl
- Миграция ruleset на nftables без простоя, с named sets и хранением конфигурации в git
- Подключение узла к нашему мониторингу Zabbix: дропы, счётчики правил, таблица соединений

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 iptables-extensions(8): conntrack, limit, recent, ipset (man7.org — 1.8.11)

02 xtables-nft(8): iptables поверх nftables API, выбор бэкенда (manpages.debian.org — trixie)

03 nftables wiki: ruleset, named sets, flowtable, nft monitor (wiki.nftables.org — 1.1.3)

04 nf_conntrack-sysctl: Netfilter Conntrack Sysfs variables (docs.kernel.org — 6.x)

05 Configuring firewalls and packet filters: nftables и firewalld (docs.redhat.com — RHEL 10)

06 Debian Wiki: nftables, iptables, netfilter-persistent (wiki.debian.org — 2026)

07 Netfilter flowtable infrastructure (nf_flowtable) (docs.kernel.org — 6.x)

Основано на официальной документации продуктов и нашей практике внедрения.