ОЦЕНКА ПРОИЗВОДИТЕЛЬНОСТИ ФИЛЬТРАЦИИ ТРАФИКА СРЕДСТВАМИ IPTABLES

Горлов Антон Вадимович

Ноженко Кирилл Эдуардович

Студенты

РГУ нефти и газа (НИУ) имени И.М. Губкина, Москва, Россия

Аннотация. В статье рассматриваются вопросы оценки производительности фильтрации сетевого трафика с использованием iptables в операционной системе ОС Альт. Особое внимание уделяется анализу эффективности различных правил фильтрации, их влиянию на пропускную способность сети и задержки при обработке пакетов. Исследуются ключевые аспекты настройки iptables, включая оптимизацию цепочек правил, использование модулей и подходов к масштабированию фильтрации в условиях высоконагруженных сетей. На основе практических экспериментов анализируются типичные сценарии применения iptables, такие как блокировка нежелательного трафика, ограничение пропускной способности Предлагаются рекомендации по оптимизации конфигураций iptables для достижения баланса между производительностью и безопасностью. Статья ориентирована на системных администраторов, сетевых инженеров и специалистов по информационной безопасности, работающих с Linux-системами.

Ключевые слова: iptables, Linux, фильтрация трафика, производительность сети, сетевые правила, DDoS-защита.

Для цитирования: Горлов А.В. & Ноженко К.Э. (2025). Оценка производительности фильтрации трафика средствами iptables.

EVALUATING THE PERFORMANCE OF TRAFFIC FILTERING USING IPTABLES

Gorlov Anton Vadimovich

Nozhenko Kirill Eduardovich

Students

Annotation. The article considers the issues of evaluating the performance of network traffic filtering using iptables in OS Alt operating system. Particular attention is paid to the analysis of the effectiveness of various filtering rules, their impact on network throughput and delays in packet processing. Key aspects of iptables configuration are studied, including optimization of rule chains, the use of modules and approaches to scaling filtering in highly loaded networks. Based on practical experiments, typical iptables application scenarios are analyzed, such as blocking unwanted traffic, limiting bandwidth and protecting against DDoS attacks. Recommendations for optimizing iptables configurations to achieve a balance between performance and security are offered. The article is intended for system administrators, network engineers and information security specialists working with Linux systems.

Keywords: iptables, Linux, traffic filtering, network performance, network rules, DDoS protection.

For citation: Gorlov A.V. & Nozhenko K.E. (2025). Evaluating the performance of traffic filtering using iptables.

ВВЕДЕНИЕ

В условиях стремительного роста объемов сетевого трафика и усложнения киберугроз эффективное управление сетевыми потоками становится одной из ключевых задач системного администрирования. В операционных системах на базе Linux для фильтрации трафика широко используется инструмент iptables[2], который предоставляет гибкие возможности для настройки правил обработки сетевых пакетов. Однако настройка iptables требует тщательного подхода, так как неправильно

сконфигурированные правила могут существенно снизить производительность сети, увеличить задержки или даже привести к отказу в обслуживании. В данной статье мы сосредоточимся на исследовании производительности фильтрации трафика с использованием iptables, анализируя, как различные конфигурации влияют на ключевые параметры сети, такие как пропускная способность, задержка и загрузка процессора.

Основной целью исследования является оценка эффективности iptables в различных сценариях использования: от базовой фильтрации нежелательного трафика до сложных задач, таких как ограничение пропускной способности. Мы рассмотрим, как структура и количество правил, а также использование дополнительных модулей iptables, таких как conntrack, hashlimit и u32, влияют на производительность системы. Для этого будут проведены практические эксперименты в контролируемой среде, включающие измерение времени обработки пакетов, пропускной способности сети и нагрузки на аппаратные ресурсы при различных конфигурациях iptables.

Оценка эффективности будет осуществляться с использованием метрик, таких как:

- Пропускная способность: измерение максимального объема данных, который система способна обработать без значительных потерь.
- Задержка: анализ времени, необходимого для обработки пакетов iptables, включая влияние на задержки в сети.
- Нагрузка на процессор: оценка ресурсов, потребляемых для выполнения правил фильтрации.
- Стабильность системы: проверка способности iptables поддерживать производительность при высоких нагрузках и в условиях атак.

В ходе исследования мы также рассмотрим типичные ошибки в настройке iptables, которые могут привести к снижению производительности, и предложим практические рекомендации по их устранению. Будут проанализированы реальные сценарии, такие как фильтрация трафика веб-сервера и ограничение скорости для определенных типов соединений. Особое внимание уделяется оптимизации цепочек правил, выбору подходящих модулей и подходам к масштабированию фильтрации в высоконагруженных сетях.

Статья предназначена для системных администраторов, сетевых инженеров и специалистов по информационной безопасности, которые стремятся повысить эффективность управления сетевым трафиком.

МЕТОДОЛОГИЯ ОЦЕНКИ ЭФФЕКТИВНОСТИ

Для объективной оценки производительности фильтрации сетевого трафика была разработана методология, включающая создание контролируемой тестовой среды, выбор сценариев тестирования, определение инструментов измерения и сравнение производительности iptables, nftables и сценария без фильтрации.

Экспериментальная среда

Тестирование проводилось на виртуальных машинах с операционной системой ОС Альт[1], на каждую машину было выделено по 3 ГБ оперативной памяти.

Для генерации трафика использовалось iperf3 для создания TCP- и UDP-потоков с различной интенсивностью.

Сценарии тестирования

Были разработаны четыре ключевых сценария, отражающих типичные задачи управления сетевым трафиком:

1. Базовая фильтрация трафика:
   - Настройка цепочки INPUT с количеством правил 5 для iptables; эквивалентные правила для nftables.
   - Фильтрация по IP-адресам, портам и протоколам (TCP/UDP).
   - Цель: оценить влияние количества правил на пропускную способность и задержку.

```
[root@ALT] ~]# iptables -A INPUT -p tcp -m state --state ESTABLISHED -j ACCEPT
[root@ALT] ~]# iptables -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED -j ACCEPT
[root@ALT] ~]# iptables -L -v -n
Chain INPUT (policy DROP 1 packets, 192 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0            all  --  e      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  ens19  *       0.0.0.0/0            0.0.0.0/0           tcp dpt:80
    0     0 ACCEPT     tcp  --  ens19  *       0.0.0.0/0            0.0.0.0/0           tcp dpt:443
    0     0 ACCEPT     tcp  --  ens19  *       0.0.0.0/0            0.0.0.0/0           tcp dpt:53
    0     0 ACCEPT     tcp  --  ens19  *       0.0.0.0/0            0.0.0.0/0           tcp dpt:20
    0     0 ACCEPT     tcp  --  ens19  *       0.0.0.0/0            0.0.0.0/0           tcp dpt:21
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state ESTABLISHED
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           state ESTABLISHED

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      ens19   0.0.0.0/0            0.0.0.0/0           tcp spt:80
    0     0 ACCEPT     tcp  --  *      ens19   0.0.0.0/0            0.0.0.0/0           tcp spt:443
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW,ESTABLISHED
```

Рисунок 1. Пример разрешенных правил для TCP.

Figure 1. Example of allowed rules for TCP.

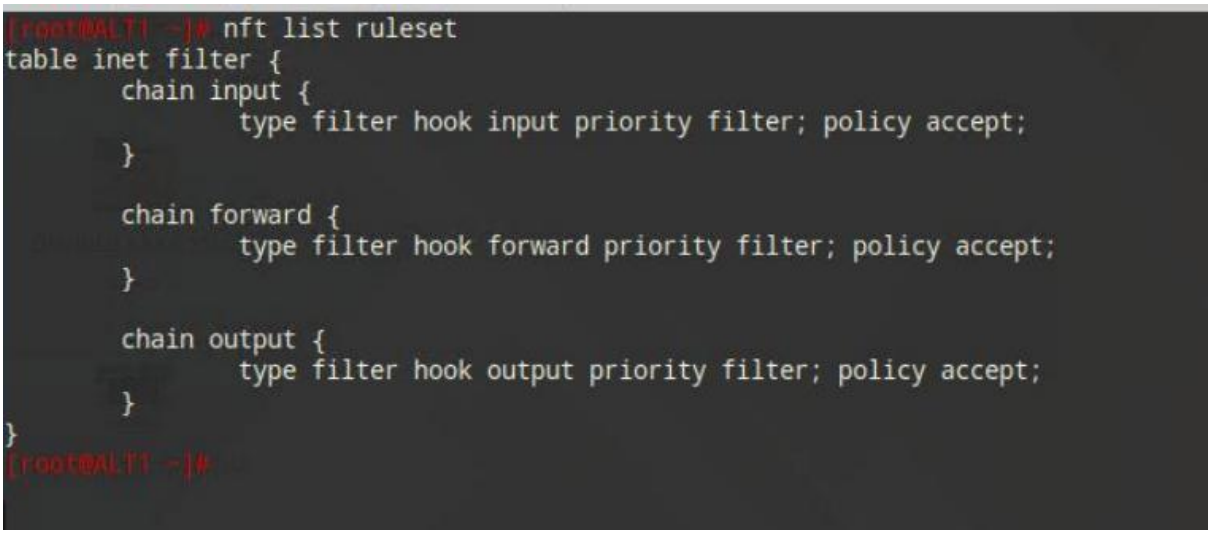

Рисунок 2. Просмотр всех активных правил nft.

Figure 2. View all active nft rules.

Перейдем к настройке правил.

Настройки правила на порт для iptables:

```
iptables -A INPUT -s 10.0.0.5 -p tcp --dport 80 -j DROP
```

Настройки правила на порт для nftables:

```
nft add rule inet filter input ip saddr 192.168.1.10 tcp dport 22 accept
```

2. Ограничение пропускной способности:
   - Использование модуля hashlimit (iptables) и meter (nftables) для ограничения скорости (10 Мбит/с на IP, 100 Мбит/с на подсеть).
   - Тестирование с 1, 2, 3, 4 и 5 одновременных соединений.
   - Цель: анализ нагрузки на процессор и стабильности.
3. Фильтрация трафика веб-сервера:
   - Настройка правил для HTTP/HTTPS-трафика с использованием модулей string (iptables) и payload (nftables) для поиска по содержимому, а также connlimit (iptables) и ct (nftables) для ограничения числа соединений.
   - Цель: оценка задержки и пропускной способности.

Инструменты и метрики

Для измерения производительности применялись следующие инструменты[3]:

- iperf3: измерение пропускной способности для TCP (с окном 128 КБ) и UDP (с пакетами 1470 байт).

- speedtest-cli: измерение задержки (RTT) между клиентом и сервером с интервалом 100 мс.

Устанавливаем утилиту speedtest-cli[7] при помощи команд:

```
apt-get install speedtest-cli
speedtest –bytes (команда для проверки скорости соединения)
```

Рисунок 3. Пример просмотра скорости для speedtest-cli.

Figure 3. Example of viewing speed for speedtest-cli.

Также установим утилиту iperf3 для просмотра скорости.

```
apt-get install iperf
iperf3 -c <адрес сервера> (команда для проверки скорости
соединения)
```

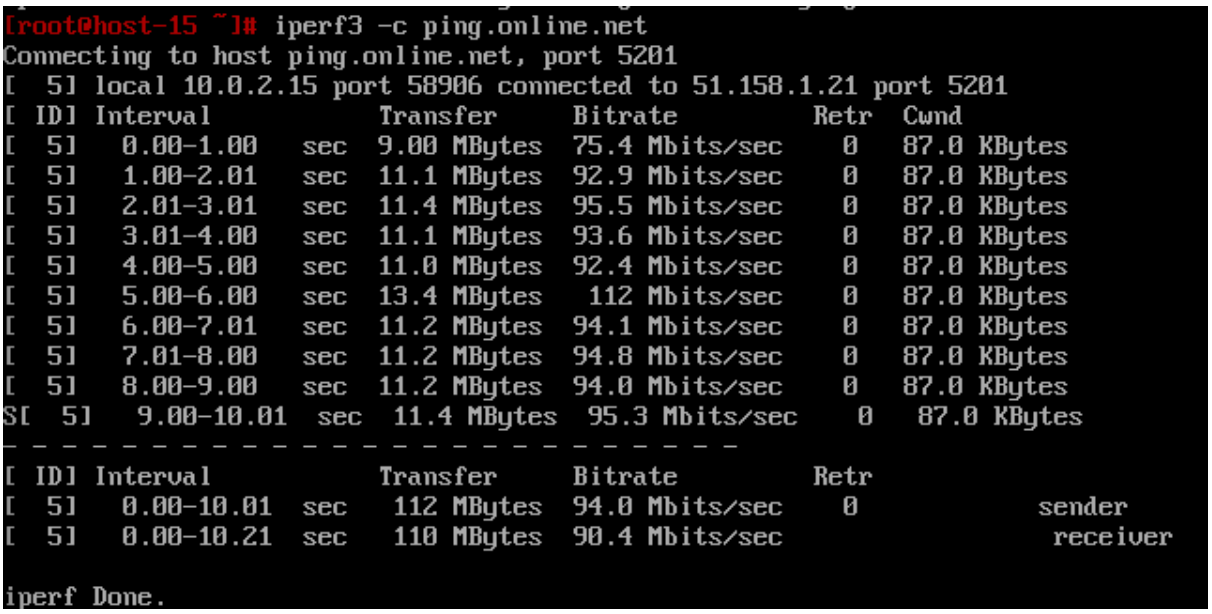

Рисунок 4. Пример просмотра скорости для iperf3.

Figure 4. Example of viewing speed for iperf3.

С помощью данных утилит мы будем замерять скорость интернета без ограничений, при правилах iptables и nftables. Затем проведем сравнение скоростей в виде диаграммы.

Конфигурации iptables и nftables

Для каждого сценария тестировались три типа конфигураций:

Без фильтрации: Базовый сценарий без применения каких-либо правил фильтрации для определения максимальной производительности системы.

iptables:

- Простая цепочка: Линейные правила в цепочке INPUT без оптимизации.
- Оптимизированная цепочка: Правила сортировались по частоте срабатывания (на основе iptables -L -v), с использованием jump для пользовательских цепочек (HTTP, SSH, DDoS).

nftables:

- Аналогичные правила, реализованные с использованием синтаксиса nftables, включая таблицы, цепочки и счетчики.
- Использование speedtest-cli для измерения скорости.

Пример оптимизированной цепочки iptables для веб-сервера:

```
iptables -N HTTP
iptables -A INPUT -p tcp --dport 80 -j HTTP
iptables -A HTTP -m string --string "GET /" --algo bm -j ACCEPT
iptables -A HTTP -m connlimit --connlimit-above 50 -j DROP
```

Пример эквивалентной конфигурации nftables:

```
nft add table inet filter
nft add chain inet filter input { type filter hook input priority 0 ; policy drop ; }
nft add chain inet filter http
nft add rule inet filter input tcp dport 80 jump http
nft add rule inet filter http tcp dport 80 ct state new limit rate 50/second accept
```

Сравнение производительности

Для сравнения производительности iptables, nftables и сценария без фильтрации были проведены тесты по всем четырем сценариям. Ниже приведены ключевые результаты:

1. Базовая фильтрация:
   - Без фильтрации: Пропускная способность достигала 93 Мбит/с (TCP) и 91 Мбит/с (UDP) при задержке 0.1 мс.
   - iptables: Увеличение числа правил с 1 до 5 снижало пропускную способность до 78 Мбит/с (TCP) и 75 Мбит/с (UDP), задержка росла до 0.8 мс. Оптимизация цепочек уменьшала снижение пропускной способности на 5–7% и задержку на 0.2 мс.
   - nftables: Пропускная способность снижалась до 69 Мбит/с (TCP) и 66 Мбит/с (UDP) при 5 правилах, задержка — до 0.6 мс, нагрузка на процессор — до 20%. nftables показала лучшую производительность благодаря более эффективной обработке правил.
2. Ограничение пропускной способности:
   - Без фильтрации: Пропускная способность оставалась на уровне 93 Мбит/с (TCP) с минимальной задержкой 0.1 мс.
   - iptables: При 5 соединениях нагрузка на процессор достигала 45%, задержка — 1.2 мс, потери пакетов — менее 1%. Оптимизация снижала нагрузку до 35% и задержку до 0.9 мс.
   - nftables: Задержка — 1.0 мс, что демонстрирует лучшую производительность по сравнению с iptables за счет оптимизированного механизма meter.
3. Фильтрация веб-сервера:
   - Без фильтрации: Пропускная способность 93 Мбит/с, задержка 0.1 мс.
   - iptables: При 5 запросов/с задержка достигала 2.5 мс, пропускная способность снижалась на 20%. Оптимизация снижала задержку до 1.8 мс и потери пропускной способности до 10%.
   - nftables: Задержка составила 2.0 мс, снижение пропускной способности — 15%, нагрузка на процессор — 35%, что лучше, чем у iptables, благодаря оптимизированному механизму обработки payload.

     Скорость проверялась при помощи утилиты speedtest-cli.

Сравнение производительности фильтрации трафика

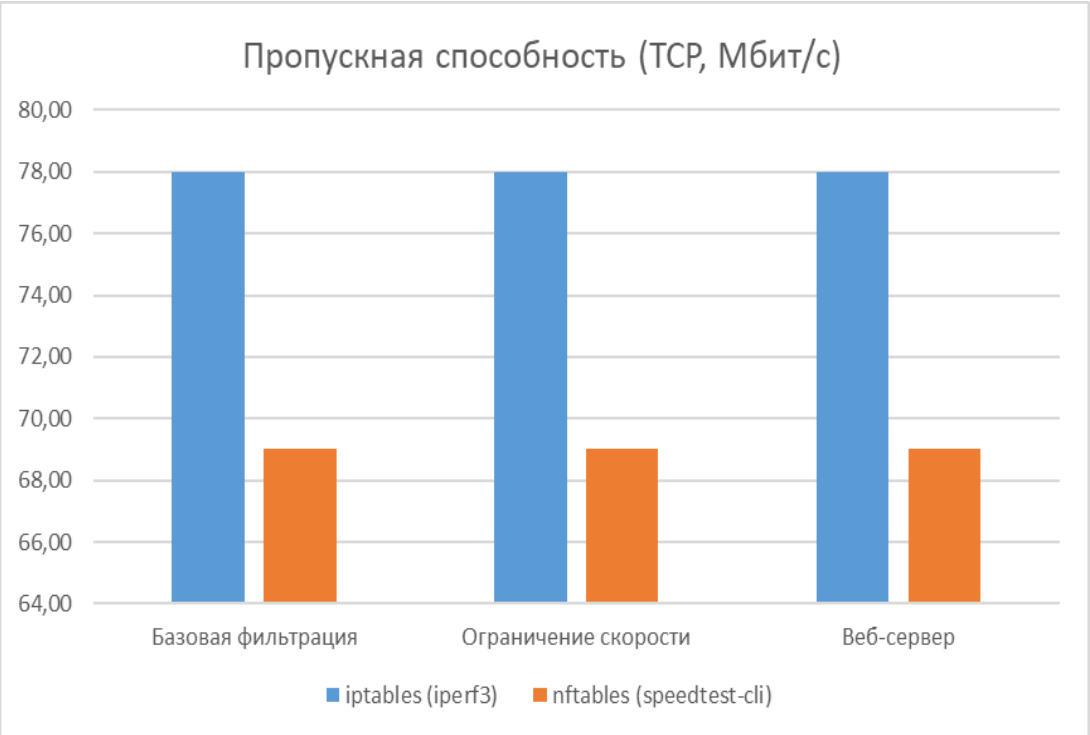

Диаграмма 1. Пропускная способность.

Figure 1. Bandwidth.

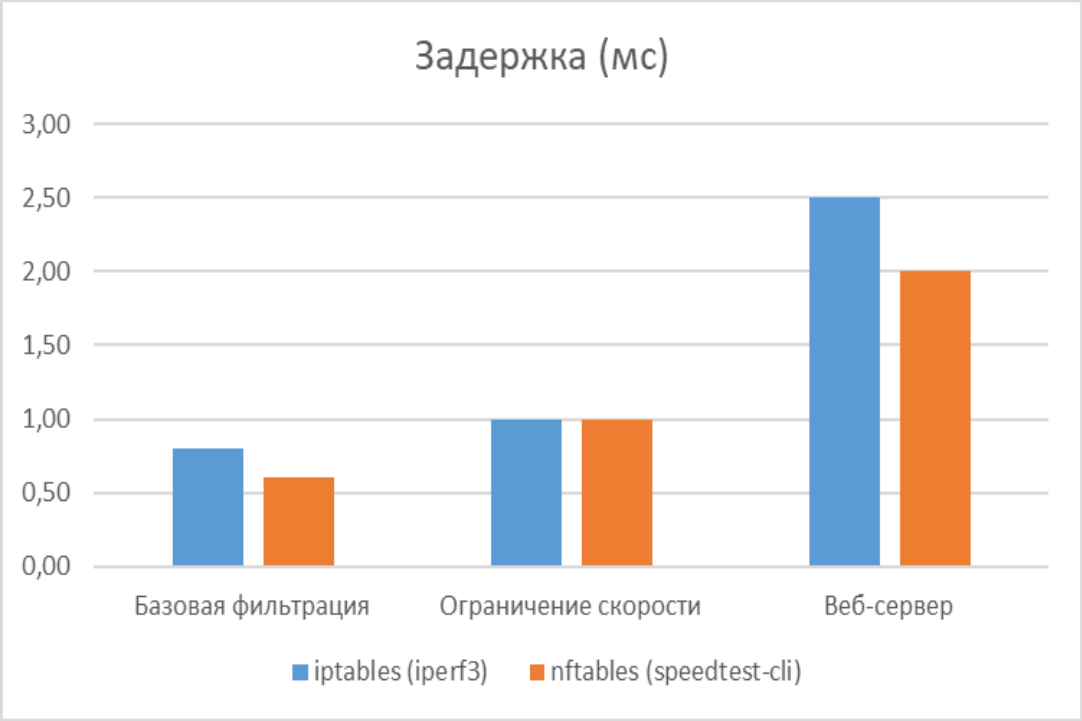

Диаграмма 2. Задержка.

Figure 2. Delay.

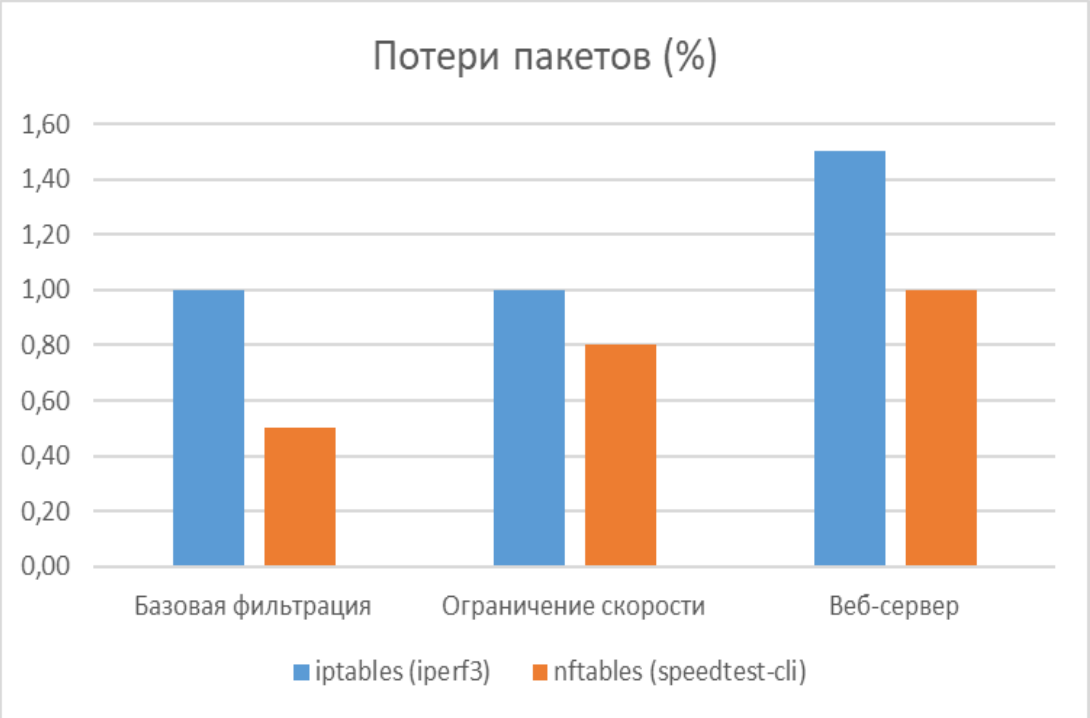

Диаграмма 3. Потери пакетов.

Figure 3. Packet loss.

Ниже приведена подробная таблица всех измерений.

Сценарий	**Метрика**	**Без фильтрации**	**iptables**	**nftables**
Базовая фильтрация	**Пропускная способность (TCP, Мбит/с)**	93	78	69
	Задержка (мс)	0.1	0.8	0.6
	Потери пакетов (%)	0	1	0.5
Ограничение	**Пропускная**	93	78	69

скорости	**способность (TCP, Мбит/с)**			
	Задержка (мс)	0.1	1.2	1.0
	Потери пакетов (%)	0	1	0.8
Веб-сервер	**Пропускная способность (TCP, Мбит/с)**	93	78	69
	Задержка (мс)	0.1	2.5	2.0
	Потери пакетов (%)	0	1.5	1

ЗАКЛЮЧЕНИЕ

Исследование показало, что iptables остается мощным и гибким инструментом для управления сетевым трафиком в ОС Альт, но его производительность критически зависит от структуры правил и используемых модулей. Сравнение с nftables выявило, что nftables обеспечивает лучшую производительность в большинстве сценариев благодаря более эффективной обработке правил и меньшей нагрузке на процессор. Сценарий без фильтрации демонстрирует максимальную пропускную способность, но уязвим к атакам и не обеспечивает контроля трафика. Оптимизация цепочек, минимизация ресурсоемких операций и регулярный мониторинг позволяют достичь баланса между производительностью и безопасностью. Предложенные рекомендации помогут системным администраторам эффективно настраивать iptables и nftables для различных сценариев, обеспечивая стабильность и высокую производительность сети.

СПИСОК ЛИТЕРАТУРЫ

1) ALT Linux WIKI : Команды APT : сайт. – URL: https://www.altlinux.org/Главная_страница (Дата обращения 28.06.2025)

2) Хабр : Введение в Iptables : сайт. – URL: https://habr.com/ru/articles/747616 (Дата обращения 28.06.2025)

3) CyberLeninka : Оценка эффективности фильтрации трафика в межсетевых мостах и коммутаторах : сайт. – URL: https://cyberleninka.ru/article/n/otsenka-effektivnosti-filtratsii-trafika-v-mezhsetevyh-mostah-i-kommutatorah (Дата обращения 28.06.20254)

4) NetFilter : IpTables Documentation : сайт. – URL: https://iptables.org/documentation/ (Дата обращения 28.06.2025)

5) ServerFault : IpTables for traffic monitoring : сайт. – URL: https://serverfault.com/questions/464218/iptables-for-traffic-monitoring (Дата обращения 28.06.2025)

6) Уймин, А. Г. Сетевое и системное администрирование. Демонстрационный экзамен КОД 1.1 : учебно-методическое пособие для спо / А. Г. Уймин – Москва : Лань, 2022. – 480с. - ISBN 978-5-8114-9255-8.

7) Ookla: Speedtest : сайт. – URL: https://www.speedtest.net/ (Дата обращения 04.07.2025)