DOI 10.37882/2223-2966.2025.05.16

АНАЛИЗ ВЛИЯНИЯ ПОТЕРИ ПАКЕТОВ НА СКОРОСТЬ ПЕРЕДАЧИ ДАННЫХ В СЕТИ

ANALYSIS OF THE IMPACT OF PACKET LOSS ON DATA TRANSMISSION SPEED IN THE NETWORK

K. Kamaletdinova
M. Myltseva
D. Yakupov

Summary. This paper is devoted to the study of the effect of packet loss on the effectiveness of the TCP and UDP network protocols. During the experiment, a Python script was used to simulate data transmission over these protocols, taking into account packet loss, as well as network settings that allow simulating different loss levels. Based on the collected data, latency, bandwidth, and response time were measured when the percentage of packet loss changed. The results of the experiment showed that TCP is more sensitive to packet loss, which leads to increased latency and reduced throughput. In contrast, UDP exhibits lower latency, but is less reliable at high levels of data loss. The work highlights the importance of taking packet loss into account when designing network applications and infrastructure, and provides recommendations on choosing protocols for various types of network services.

Keywords: network protocols, TCP, UDP, packet loss, latency, bandwidth, response time, Python script, loss modeling, data transfer efficiency.

Камалетдинова Камиля Рамилевна
ФГБОУВО «Поволжский государственный университет телекоммуникаций и информатики»
camaletdinova.kamilya@yandex.ru
Мыльцева Мария Николаевна
ФГБОУВО «Поволжский государственный университет телекоммуникаций и информатики»
kaktycmeya106@gmail.com
Якупов Денис Олегович
Старший преподаватель,
ФГБОУВО «Поволжский государственный университет телекоммуникаций и информатики»
d.yakupov@psuti.ru

Аннотация. Данная работа посвящена исследованию влияния потерь пакетов на эффективность сетевых протоколов TCP и UDP. В процессе эксперимента использовался Python-скрипт для имитации передачи данных через эти протоколы с учетом потерь пакетов, а также настроек сети, позволяющих моделировать различные уровни потерь. На основе собранных данных были проведены измерения задержки, пропускной способности и времени отклика при изменении процента потерь пакетов. Результаты эксперимента показали, что TCP более чувствителен к потере пакетов, что приводит к увеличению задержки и снижению пропускной способности. В отличие от этого, UDP демонстрирует меньшую задержку, но менее надежен при высоких уровнях потерь данных. Работа подчеркивает важность учета потерь пакетов при проектировании сетевых приложений и инфраструктуры, а также дает рекомендации по выбору протоколов для различных типов сетевых сервисов.

Ключевые слова: сетевые протоколы, TCP, UDP, потеря пакетов, задержка, пропускная способность, время отклика, Python-скрипт, моделирование потерь, эффективность передачи данных.

Введение

Сетевые технологии стали неотъемлемой частью нашей жизни, обеспечивая обмен данными между миллиардами устройств. Неизбежными остаются проблемы потерь пакетов, которые происходят, когда данные не достигают получателя из-за перегрузки сети, ошибок на канале или сбоев в маршрутизации. Потери пакетов влияют на качество сетевых соединений и производительность приложений, особенно при передаче мультимедийных данных, видеоконференциях и онлайн-играх. Объектом исследования являются протоколы TCP и UDP, которые отличаются по способу обработки потерь пакетов. TCP (Transmission Control Protocol) — это протокол с установлением соединения, гарантирующий доставку данных через механизмы подтверждения получения, повторную передачу и управление перегрузками. Эти механизмы повышают надежность, но увеличивают задержку при потере пакетов. Алгоритмы управления перегрузкой, такие как медленный старт и экспоненциальное отступление, снижают скорость передачи данных при обнаружении потерь. UDP (User Datagram Protocol) — более простой протокол без гарантии доставки, не требующий установления соединения и не использующий повторную передачу. UDP обладает меньшими накладными расходами и обеспечивает более высокую скорость, но при потерях пакетов данные теряются безвозвратно.

Исследование направлено на понимание влияния потерь пакетов на скорость передачи данных через эти протоколы. Будет проведен ряд экспериментов с Python-скриптами для измерения скорости при разных уровнях потерь. На основе полученных данных будут построены

графики зависимости скорости от процента потерь. Результаты помогут разработчикам выбрать подходящий протокол в зависимости от условий сети и требований приложения.

Задачи исследования: исследование влияния потерь пакетов на производительность протоколов TCP и UDP; разработка Python-скрипта для измерения скорости передачи данных; создание искусственных потерь пакетов с помощью утилиты tc qdisc; построение графиков зависимости скорости от процента потерь для TCP и UDP. Актуальность заключается в важности понимания влияния потерь пакетов на различные протоколы для проектирования сетевых приложений.

Методология

Для достижения цели исследования применяется экспериментальный подход с использованием Python-скриптов и утилиты tc qdisc для создания искусственных потерь пакетов. Эксперимент проводится в Linux, предоставляющей средства для управления сетью через команду tc и инструменты для контроля качества обслуживания. Выбор Linux обусловлен широкими возможностями настройки сетевого стека и распространенностью в серверных средах. Основным инструментом является Python, с помощью которого создается скрипт для измерения скорости передачи данных, задержек и времени отклика. Используются стандартные библиотеки Python, такие как time и socket. Для создания искусственных потерь применяется утилита tc qdisc, настраивающая уровень потерь пакетов в сети путем изменения параметров очереди. Моделируются различные уровни потерь от 0 % до 20 %. Выбор такого диапазона обусловлен тем, что потери более 20 % обычно свидетельствуют о серьезных проблемах в инфраструктуре.

Экспериментальная среда настроена в локальной сети для исключения влияния внешних факторов. На каждом этапе эксперимента изменяются параметры потерь, и для каждого уровня проводятся тесты на передачу данных через TCP и UDP. Для статистической значимости каждый тест повторяется несколько раз.

Измеряются такие параметры, как пропускная способность, задержка и время отклика. Пропускная способность — это скорость передачи данных (байт за единицу времени). Задержка — время прохождения пакета от отправителя к получателю. Время отклика — время от отправки пакета до получения подтверждения. Данные обрабатываются и визуализируются с помощью библиотек Matplotlib и Seaborn.

Ожидается, что с увеличением уровня потерь пакетов производительность сети будет снижаться, особенно для TCP, который использует механизм повторной передачи. UDP продемонстрирует меньшую задержку, но скорость передачи данных будет зависеть от потерь пакетов.

Таблица 1.

Измеряемые параметры для оценки производительности протоколов

Параметр	Описание	Единица измерения
Пропускная способность	Количество успешно переданных данных за единицу времени	Мбит/с
Задержка	Среднее время, необходимое для доставки одного пакета от отправителя к получателю	мс
Время отклика	Время между отправкой запроса и получением ответа	мс
Коэффициент потери пакетов	Отношение количества потерянных пакетов к общему количеству отправленных пакетов	%
Джиттер	Вариация задержки между последовательно принятыми пакетами	мс

Серверная часть

Для проведения эксперимента разработана серверная часть, принимающая и обрабатывающая данные от клиентов. Сервер реализован на Python с использованием библиотеки socket для работы с протоколами TCP и UDP. Цель серверной части — прием пакетов данных и анализ времени их получения, а также запись статистики о потерях и задержках. В серверной части создается сокет, который слушает входящие соединения или пакеты. Для TCP используется протокол SOCK_STREAM, требующий установления соединения. Для UDP используется SOCK_DGRAM, позволяющий передавать данные без постоянного соединения. Это фундаментальное различие влияет на поведение соединения при потере пакетов. Для TCP сервер создает сокет, связывает его с IP-адресом и портом, и начинает прослушивание порта. При подключении клиента сервер принимает соединение и создает новый сокет для обмена данными. После установления соединения сервер принимает данные и отправляет ответы методами recv() и send().

```
import socket

def start_tcp_server():
    server_socket = socket.socket(socket.AF_INET, socket.
SOCK_STREAM)
    server_socket.bind(('localhost', 12345))
    server_socket.listen(1)
    print(«TCP Server is listening on port 12345...»)
```

```
        while True:
         client_socket, client_address = server_socket.accept()
         print(f»Connection established with {client_address}»)
           data = client_socket.recv(1024)
           print(f»Received data: {data.decode()}»)
           client_socket.close()

    def start_udp_server():
       server_socket = socket.socket(socket.AF_INET, socket.
SOCK_DGRAM)
       server_socket.bind((‘localhost’, 12345))
       print(«UDP Server is listening on port 12345...»)

       while True:
         data, client_address = server_socket.recvfrom(1024)
           print(f»Received data from {client_address}: {data.
decode()}»)
```

Когда сервер получает данные, они обрабатываются для оценки потерь пакетов и задержек. Основное внимание уделяется анализу времени получения данных. Для каждого пакета сервер записывает время получения, что позволяет вычислить задержку. Это особенно важно при высоких уровнях потерь, когда задержки могут существенно возрасти.

Для каждого теста сервер записывает статистику о времени получения пакетов. Эти данные используются для вычисления пропускной способности и задержки. Потери пакетов влияют на TCP, поскольку он инициирует повторную отправку, увеличивая нагрузку на сеть. UDP не восстанавливает потерянные данные, что делает его более устойчивым к задержкам, но менее надежным.

Таблица 2.

Результаты измерения задержки (мс) для различных уровней потерь пакетов

Протокол	0 % потерь	1 % потерь	5 % потерь	10 % потерь	20 % потерь
TCP	12.5	45.2	118.7	203.5	312.4
UDP	10.2	19.8	34.6	48.3	65.1

Сервер также измеряет джиттер — вариацию задержки, важный показатель стабильности соединения для приложений реального времени. Высокий джиттер может негативно сказаться на качестве воспроизведения аудио или видео.

Клиентская часть

Клиентская часть активно отправляет данные и играет ключевую роль в эксперименте. Клиент с помощью сокетов отправляет пакеты на сервер и измеряет время их отправки и получения для вычисления задержки и пропускной способности. Клиент разработан на Python с использованием тех же библиотек, что и сервер.

Для взаимодействия с сервером клиент использует сокеты. Для TCP клиент устанавливает соединение перед отправкой данных, а для UDP отправляет пакеты без постоянного соединения. Сокеты создаются с использованием протоколов SOCK_STREAM (TCP) и SOCK_DGRAM (UDP).

```
    import socket
    import time

    def start_tcp_client():
    client_socket = socket.socket(socket.AF_INET, socket.
SOCK_STREAM)
     client_socket.connect((‘localhost’, 12345))

    message = «Test packet»
    start_time = time.time()
    client_socket.send(message.encode())
    data = client_socket.recv(1024)
    end_time = time.time()

    print(f»Received: {data.decode()}»)
    print(f»Time taken: {end_time — start_time} seconds»)
    client_socket.close()

    def start_udp_client():
    client_socket = socket.socket(socket.AF_INET, socket.
SOCK_DGRAM)

    message = «Test packet»
    start_time = time.time()
    client_socket.sendto(message.encode(), (‘localhost’,
12345))
    data, server_address = client_socket.recvfrom(1024)
    end_time = time.time()

    print(f»Received: {data.decode()} from {server_address}»)
    print(f»Time taken: {end_time — start_time} seconds»)
    client_socket.close()
```

Для точной оценки времени передачи данных клиент записывает время до и после отправки каждого пакета. Чем выше уровень потерь, тем дольше задержка, особенно для TCP. Для обеспечения точности клиент может использовать time.perf_counter() вместо time.time(). При моделировании потерь пакетов используется утилита tc qdisc. Например, команда «sudo tc qdisc add dev eth0 root netem loss 10 %» настраивает потерю 10 % пакетов. Утилита netem позволяет эмулировать различные свойства сети, такие как потери, задержки и дублирование пакетов. Клиент отправляет серию пакетов разного размера для оценки зависимости производительности от размера пакета при различных уровнях потерь. Для каждого размера пакета и уровня потерь проводится несколько тестов для статистической значимости результатов.

Таблица 3.

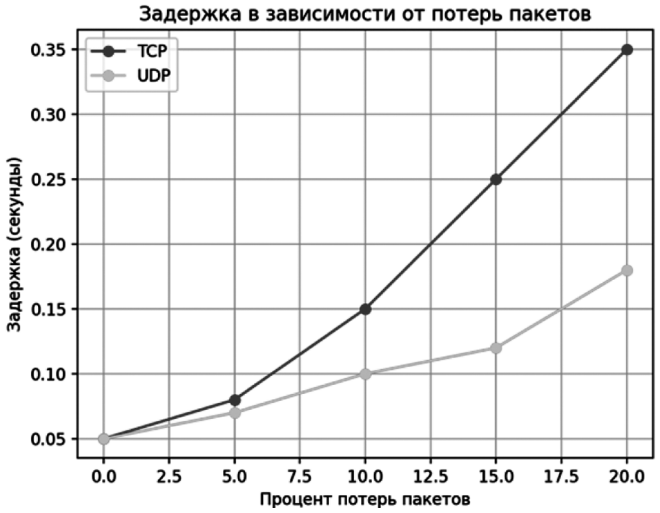
Результаты измерения пропускной способности (Мбит/с) для различных уровней потерь пакетов

Протокол	0 % потерь	1 % потерь	5 % потерь	10 % потерь	20 % потерь
TCP	95.3	72.5	43.8	28.6	14.2
UDP	98.1	96.3	92.8	87.4	77.9

Анализ результатов

После проведения эксперимента, где сервер и клиент обменивались пакетами данных через TCP и UDP с искусственно настроенной потерей пакетов, был получен ряд статистических данных. В данной главе проводится анализ результатов для выявления зависимостей между потерями пакетов и временем отклика, а также пропускной способностью для каждого протокола.

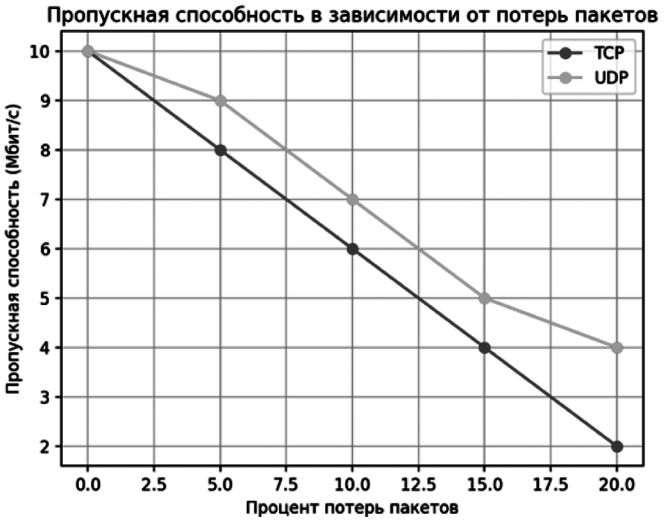
Задержка является важным параметром для оценки качества сети. Результаты показали, что для TCP с увеличением уровня потерь пакетов задержка значительно возрастала. Это объясняется механизмом повторной передачи и алгоритмами управления перегрузкой. На графике ниже показано, как увеличивается задержка с ростом потерь пакетов для обоих протоколов.



Результаты: Как видно из графика, **TCP** испытывает значительно большие задержки при росте потерь пакетов по сравнению с **UDP**. Это связано с тем, что **TCP** использует механизмы подтверждения и повторной отправки пакетов, что увеличивает время передачи данных при потере пакетов. В то время как **UDP**, не имея такого механизма, теряет пакеты, но быстрее передает оставшиеся данные.

При анализе данных стало очевидно, что **TCP** страдает от потерь пакетов гораздо сильнее, чем **UDP**. Для **TCP** пропускная способность уменьшается с ростом потерь пакетов, так как повторные отправки пакетов занимают время и увеличивают общий объем переданных данных. В случае с **UDP** пропускная способность также падает с увеличением потерь, но в меньшей степени, поскольку UDP не тратит время на повторную отправку данных.

На графике ниже показано, как пропускная способность изменяется с увеличением процента потерь пакетов.



Результаты: Из графика видно, что пропускная способность **TCP** значительно снижается с увеличением потерь пакетов, что свидетельствует о высокой чувствительности этого протокола к потере данных. В отличие от этого, **UDP** демонстрирует более стабильную пропускную способность, хотя она также падает с ростом потерь, но не так резко.

Основываясь на полученных данных, можно сделать вывод, что TCP имеет преимущество в стабильных сетях с минимальными потерями пакетов. Однако при повышении потерь его производительность ухудшается из-за необходимости повторной передачи данных. UDP, несмотря на отсутствие механизма повторной отправки, показал лучшую производительность в условиях потерь, но это приводит к потере данных.

Из эксперимента можно сделать несколько выводов. Во-первых, потери пакетов существенно влияют на производительность обоих протоколов. Во-вторых, для приложений, где высокая надежность имеет первостепенное значение, лучше использовать TCP. Для приложений, требующих высокой скорости с меньшими требованиями к надежности, UDP будет более подходящим выбором. Результаты могут быть полезны разработчикам сетевых приложений при выборе протокола в зависимости от условий работы и требований приложения.

Таблица 4.
Относительная производительность протоколов при различных уровнях потерь пакетов (% от максимальной)

Протокол	Метрика	0 % потерь	1 % потерь	5 % потерь	10 % потерь	20 % потерь
TCP	Пропускная способность	100 %	76 %	46 %	30 %	15 %
TCP	Задержка	100 %	362 %	950 %	1628 %	2499 %
UDP	Пропускная способность	100 %	98 %	95 %	89 %	79 %
UDP	Задержка	100 %	194 %	339 %	473 %	638 %

Заключение

Исследование показало значительное влияние потерь пакетов на работу протоколов TCP и UDP. В ходе эксперимента использовался Python-скрипт для отправки данных и имитации потерь пакетов. Были проанализированы изменения задержки, пропускной способности и времени отклика в зависимости от уровня потерь. Результаты подтвердили теоретические предположения о поведении протоколов и предоставили количественные оценки их производительности. Для TCP повышение потерь пакетов приводит к увеличению задержки из-за механизма повторной отправки, что замедляет передачу данных. UDP сохраняет стабильную задержку, однако потеря пакетов влияет на надежность передачи данных. Эти особенности делают TCP подходящим для приложений, требующих надежной доставки, а UDP — для приложений, где важна низкая задержка. Пропускная способность TCP существенно снижается при увеличении потерь пакетов из-за необходимости повторной передачи. Пропускная способность UDP остается более стабильной, но потери пакетов ведут к потере данных. Выбор между TCP и UDP должен основываться на компромиссе между надежностью и скоростью, с учетом требований приложения и условий сети. Выбор протокола зависит от характеристик приложения и условий сети. Для приложений, где важна целостность данных, предпочтительнее TCP. Для приложений, где важна высокая скорость и допустимы потери, например в потоковых сервисах, предпочтительнее UDP. В современных приложениях часто используются оба протокола для разных типов данных.

Результаты исследования подчеркивают важность учета потерь пакетов при проектировании сетевых приложений и инфраструктуры. Эти данные помогают выбрать наиболее подходящий протокол для конкретных условий. Понимание влияния потерь пакетов на протоколы позволяет разработчикам создавать более эффективные и адаптивные приложения, способные поддерживать высокую производительность даже в неблагоприятных условиях сети.

ЛИТЕРАТУРА

1. Таненбаум А.С., & Ветеролл, Д.Дж. Компьютерные сети (5-е изд.). Питер. — Режим доступа: [URL: https://edu.library.moscow.ru/bb17194]
2. Куросе Дж.Ф., & Росс К.У. Компьютерные сети: принципы, технологии и протоколы (7-е изд.). Питер. — Режим доступа: [URL: https://books.google.ru/books?id=gECtAgAAQBAJ]
3. Постель Дж. Транспортный протокол управления (RFC 793). — Режим доступа: [URL: https://tools.ietf.org/html/rfc793]
4. Стюарт Р. TCP/IP: Иллюстрации, том 1: Протоколы (2-е изд.). Addison-Wesley Professional. — Режим доступа: [URL: https://www.pearson.com/store/p/tcp-ip-illustrated-volume-1-the-protocols/P100000156607]
5. Кешав С. Математика коммуникационных сетей. Cambridge University Press. — Режим доступа: [URL: https://www.cambridge.org/core/books/mathematics-of-communication-networks/AB32AFC04A5C87C2885AB4E88B35F9A]
6. Форузан Б.А. Коммуникации данных и сети (5-е изд.). McGraw-Hill. — Режим доступа: [URL: https://www.mheducation.com/highered/product/data-communications-networking-forouzan/M9780073376225.html]
7. RFC 5681. Управление перегрузками в TCP. — Режим доступа: [URL:https://tools.ietf.org/html/rfc5681]
8. Хеди Л., & Халил С. (2016). «Оценка производительности TCP и UDP при различных уровнях потерь пакетов». Международный журнал управления сетями, 26(3), 220–233. — Режим доступа: [URL:https://doi.org/10.1002/nem.2006]
9. «Управление трафиком в Linux» (2018). Linux Foundation. — Режим доступа: [URL:https://www.kernel.org/doc/Documentation/networking/tc.txt]
10. Комер Д.Э. Взаимодействие с TCP/IP: принципы, протоколы и архитектура (6-е изд.). Питер. — Режим доступа: [URL:https://www.ozon.ru/context/detail/id/30189593/]
11. Дромс Р. Протокол динамической конфигурации хоста (DHCP). RFC 2131. — Режим доступа: [URL: https://tools.ietf.org/html/rfc2131]
12. Стюарт Р. Производительность TCP и выбор протоколов для приложений реального времени (2008). — Режим доступа: [URL:https://ieeexplore.ieee.org/document/4533907]
13. «Понимание UDP и TCP в сетевых технологиях». NetworkWorld. — Режим доступа: [URL:https://www.networkworld.com/article/2883597/understanding-tcp-and-udp.html]
14. «Введение в сетевые технологии: TCP/IP и модель OSI». Cisco Networking Academy. — Режим доступа: [URL: https://www.netacad.com/courses/packet-tracer]

© Камалетдинова Камиля Рамилевна (camaletdinova.kamilya@yandex.ru); Мыльцева Мария Николаевна (kaktycmeya106@gmail.com);
Якупов Денис Олегович (d.yakupov@psuti.ru)
Журнал «Современная наука: актуальные проблемы теории и практики»