Васильев Никита Сергеевич
студент, кафедра безопасности информационных технологий,
Российский государственный университет нефти и газа (национальный исследовательский университет) имени И.М. Губкина,
РФ, г. Москва
E-mail: nikita.vasilev.1901@list.ru
Уймин Антон Григорьевич
научный руководитель, старший преподаватель кафедры безопасности информационных технологий,
Российский государственный университет нефти и газа (национальный исследовательский университет) имени И.М. Губкина,
РФ, г. Москва

ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ СЕТЕВЫХ ПРОТОКОЛОВ ПЕРЕДАЧИ ДАННЫХ В ОС АЛЬТ: СРАВНЕНИЕ TCP И QUIC В УСЛОВИЯХ ОГРАНИЧЕННОЙ ПРОПУСКНОЙ СПОСОБНОСТИ КАНАЛА

В статье представлен сравнительный анализ производительности сетевых протоколов TCP и QUIC в операционной системе ALT OS в условиях ограниченного пропускания. Исследование проводилось с использованием симулятора NS3, который моделировал различные сетевые сценарии с различными параметрами. Результаты показали, что QUIC превосходит TCP, особенно в сценариях с умеренной потерей пакетов (5-25%) и задержкой (50-100 мс), обеспечивая увеличение пропускной способности на 13,7% и сокращение задержек на 15-20%. Однако преимущества QUIC снижаются при высокой потере пакетов (65%). Исследование подтверждает потенциал QUIC для оптимизации сетевого взаимодействия в ALT OS, но подчеркивает необходимость дальнейшего тестирования в реальных условиях.

The article presents a comparative analysis of the performance of TCP and QUIC network protocols in the ALT OS operating system under conditions of limited bandwidth. The study was conducted using the NS3 simulator, which simulated various

network scenarios with different parameters. The results showed that QUIC is superior to TCP, especially in scenarios with moderate packet loss (5-25%) and latency (50-100 ms), providing a 13.7% increase in throughput and a 15-20% reduction in latency. However, the benefits of QUIC are reduced with high packet loss (65%). The study confirms the potential of QUIC to optimize networking in ALTOS, but highlights the need for further testing in real-world settings.

Ключевые слова: TCP; QUIC; производительность; пропускная способность; задержки; потери пакетов; ОС АЛЬТ; NS-3.

Keywords: TCP; QUIC; performance; bandwidth; delay; packet loss; ALT OS; NS-3.

ВВЕДЕНИЕ

По сути, информационные технологии в настоящее время требуют более качественных технологических решений для передачи данных в условиях низкой пропускной способности сетевых каналов. Со временем объем передаваемых данных и количество устройств в сети растут, и, таким образом, традиционный протокол передачи данных TCP сталкивается с проблемами задержки и потери пакетов и не в состоянии использовать весь потенциал канала передачи данных. В связи с этим становится актуальным изучение альтернативных сетевых протоколов передачи данных: QUIC, HTTP/3 и DoT, которые были разработаны для устранения недостатков обычного TCP и повышения его производительности в условиях нестабильности сети.

Данное исследование направлено на оптимизацию сетевого взаимодействия в рамках отечественной операционной системы ОС АЛЬТ, которая широко используется на предприятиях и в государственных организациях. Когда сеть загружена, а пропускная способность каналов передачи данных ограничена, чрезвычайно важно обеспечить их максимальную и эффективную пропускную способность, снизить потери и задержки при передаче пакетов. Данное исследование также актуально для приложений, работающих в режиме реального времени, где потери и задержки данных особенно критичны.

Все основные исследования в области сетевых протоколов передачи данных нацелены на улучшение производительности традиционного TCP, но все больше компаний обращают внимание на новый протокол – QUIC. Его разработкой занималась компания Google*. Этот протокол обладает рядом преимуществ: поддержкой встроенного шифрования, мультиплексированием потоков и снижением задержки при передаче пакетов данных за счет минимизации количества рукопожатий, которых не хватает TCP.

К сожалению, несмотря на растущий интерес к QUIC, его эффективность не изучена в операционных системах, подобных ОС АЛЬТ.

В процессе исследования будет использован симулятор NS-3 — программная модель эмулирования сетей дискретными событиями. Она позволяет детально рассматривать и моделировать протоколы сети: TCP и QUIC, анализировать их работу при выполнении в самых вариативных средах.

ВВЕДЕНИЕ: ОБЪЕКТ, ПРЕДМЕТ И ЦЕЛЬ

Объект исследования — сетевые технологии и протоколы обмена данными, т.е. реализация сетевого взаимодействия в операционной системе ОС АЛЬТ.

Предмет исследования — сравнительный анализ производительности сетевых протоколов TCP и QUIC в операционной системе ОС АЛЬТ. Поведение вышеуказанных протоколов при ограниченной пропускной способности канала и их способность уменьшать задержки и потери.

Цель исследования — выявить преимущества и недостатки использования протоколов TCP и QUIC в условиях ограниченной пропускной способности канала. В результате проведенных исследований будут получены количественные и качественные данные о работоспособности каждого из вышеперечисленных протоколов, и станет понятно, в каких сценариях QUIC может быть более экономичным решением, чем TCP. Результаты исследования могут быть применены для оптимизации сетевого взаимодействия с отечественной операционной системой, такой как ОС АЛЬТ, и использованы в будущих исследованиях сетевых технологий.

1. TCP (Transmission Control Protocol)

Это протокол, который работает на транспортном уровне (4-й уровень) в модели OSI. Его главная задача: обеспечить надежную передачу данных между подключенными в сети устройствами. Данный протокол контролирует порядок отправки пакетов и гарантирует, что все они будут доставлены. TCP достигает этого с помощью подтверждения (ACK) полученных пакетов, повторной передачи потерянных и управления потоком данных. Однако протокол может оказаться неэффективным в случаях больших задержек и потерь пакетов, поскольку он предусматривает установление соединения (трехстороннее подтверждение связи) и повторную передачу данных в случае потери.

2. QUIC (Quick UDP Internet Connections)

Это протокол, который работает и основан на UDP. Его разработкой занималась компания Google* в конце 2012 года. QUIC также работает на транспортном уровне (4-й уровень) в модели OSI. Данный протокол сочетает в себе преимущества протокола TCP, а именно надежность и последовательную доставку, с низкой задержкой и улучшенной производительностью. QUIC также устраняет необходимость в отдельном установлении соединения (как в TCP), поскольку он содержит встроенное шифрование и уменьшает необходимое количество подтверждений связи. Одной из главных фишек данного протокола является поддержка мультиплексирования потоков, что позволяет независимо обрабатывать данные из различных потоков и устраняет проблему блокировки начала очереди (HOL Blocking).

3. UDP (User Datagram Protocol)

Протокол UDP, как и TCP, работает на транспортном уровне (L4). В отличие от традиционного TCP, протокол UDP не гарантирует доставку пакетов и их правильный порядок. Следовательно, менее надежен, но работает быстрее. Протокол UDP испытывает минимальные задержки и поэтому хорошо подходит для приложений, в которых скорость передачи данных имеет решающее значение, например, для передачи видео, онлайн-игр и VoIP (передачи голоса по IP). Отсутствие функций: подтверждения получения и повторной передачи

пакетов, делает его менее подходящим для приложений, где требуется гарантировать доставку данных.

4. Head-of-Line Blocking (HOL Blocking)

Проблема блокировки начала очереди, который возникает на транспортном уровне (L4) и прикладном уровне (L7). В проблема присуще только протоколу TCP, так как потеря пакетов или их задержка могут блокировать обработку последующих пакетов даже после их доставки. Инженеры из компании Google* в своем протоколе QUIC решили эту проблему на транспортном уровне путем мультиплексирования потоков, так что данные из разных потоков могли обрабатываться раздельно. Это особенно полезно для таких протоколов, как HTTP/3, где одновременно осуществляется обмен несколькими потоками данных.

5. HTTP/3

Протокол, который работает на прикладном уровне (7-й уровень), но использует QUIC на транспортном уровне для передачи данных. HTTP/3 лучше предыдущих своих версий HTTP (HTTP/1.1 и HTTP/2) и, как показывают эксперименты, повышает эффективность веб-приложений. Основным отличием HTTP/3 от предыдущих версий является использование QUIC в отличие от TCP, что уменьшает задержки, снижает количество рукопожатий и повышает устойчивость к потере пакетов. Он устраняет проблему блокировки на уровне приложений, что делает его более эффективным и привлекательным для существующих веб-приложений, особенно в медленных сетях.

ОБЗОР

Последние исследования и статьи, посвященные протоколам TCP и QUIC, свидетельствуют о том, что разработчики более эффективных технических решений для современных сетей продолжает расти с наибольшим интересом именно ко второму протоколу [8].

Недавние исследования и публикации по протоколам транспортных сетей, таким как TCP и QUIC, свидетельствуют о растущем интересе к созданию усовершенствованных решений для современных сетей. В статье "Введение

Google* в QUIC" на Хабре [1]: QUIC был создан для решения проблем с TCP, включая высокую задержку и низкую производительность при потере пакетов. Данный протокол построен на вышеописанном UDP, который использует все его преимущества: помогает минимизировать задержку и максимизировать производительность в нестабильных условиях. Самым интересным являются встроенное шифрование и возможность мультиплексирования потоков делают QUIC более безопасным и эффективным для современных приложений. Как видно, использование QUIC в браузере Chrome. Благодаря этому Google* смогла значительно повысить производительность своих сервисов: YouTube и Google Search*, даже при не очень стабильной работе сети.

В статье “Протокол QUIC: переход Web от TCP к UDP” на портале “Хабр” [4] говорят о переходе Web с TCP на UDP через QUIC. Объясняется вот так: проблемы TCP решает QUIC, например, HOL Blocking [2]. К вопросу о мультиплексировании потоков QUIC предлагает качественнейшее управление и больше выдерживает потери пакетов.

В статье “TCP против UDP или будущее сетевых протоколов” [6] рассматриваются TCP и UDP. QUIC упоминается как протокол, который сочетает в себе надежность TCP и низкую задержку UDP. Это делает QUIC хорошим выбором для современных сетевых приложений, где скорость и надежность важны. Также отмечается, что QUIC может стать отличной заменой TCP благодаря своей производительности, безопасности и способности адаптироваться к разным условиям сети [5].

Научная статья “QUIC – A Quick Study” [9] говорит о том, что QUIC справляется лучше с потерей пакетов и высокой задержкой, чем TCP. Авторы отмечают, что QUIC очень полезен в ненадежных сетях, где работа TCP не так хороша.

Классические книги: «Компьютерные сети» от Олиферова В.Г. [3, с. 312-345] и "Компьютерные сети" Таненбаума Э. [5, с. 287-322], дают базовую информацию о протоколах TCP и UDP. Они помогают понять, как развивались сетевые технологии и почему появились новые решения, вроде QUIC. В этих

книгах говорят, что, хотя TCP и надежный, у него есть свои ограничения, особенно в свете современных проблем, таких как высокая нагрузка на сеть и нестабильное соединение [7, с. 350; 8, с. 330].

Особый интерес представляет работа Уймина А.Г. [7, с. 24-27], в которой рассматриваются современные методы классификации корпоративного трафика с использованием алгоритмов машинного обучения. Автор демонстрирует эффективные подходы к анализу сетевого трафика, что особенно актуально для оптимизации работы современных протоколов, включая QUIC [7, с. 25].

Основные гипотезы:

Гипотеза 1. Протокол QUIC работает эффективнее по сравнению с TCP, при плохом подключении к Интернету. Это связано с тем, что при ограниченной пропускной способности канала передачи данных за счет мультиплексирования потоков и уменьшения количества рукопожатий.

Гипотеза 2. В случае потерь пакетов и сильных задержек QUIC обеспечивает более стабильную передачу данных в сравнении с TCP, что делает его более подходящим для использования в нестабильных сетях.

Гипотеза 3. Использование QUIC в операционной системе ALT должно действительно повысить эффективность передачи данных, особенно в случае небольшой пропускной способности.

МЕТОДЫ ИССЛЕДОВАНИЯ

Это исследование будет проводиться экспериментально и направлено на сравнение функционирования сетевых протоколов TCP и QUIC в канале с ограниченной пропускной способностью. Основной метод - моделирование сетевого взаимодействия с помощью симулятора NS-3. С помощью этого метода возможно воспроизвести реалистичные условия и точно измерить эффективность протоколов в контролируемых условиях. Экспериментальный характер исследования позволяет проверить гипотезы и получить точные данные для последующего анализа и сравнения.

В рамках исследования моделируются различные сетевые сценарии, которые охватывают широкий спектр условий, характерных для современных сетей. Основные параметры выборки включают:

1. Ограниченная пропускная способность канала – моделируются условия с различной пропускной способностью: 1 Мбит/с, 5 Мбит/с и 10 Мбит/с. Это позволяет оценить, как протоколы TCP и QUIC ведут себя в условиях ограниченных ресурсов сети.

2. Потеря пакетов – моделируются условия с различным уровнем потерь пакетов: 10%, 25% и 65%. Это позволяет оценить устойчивость протоколов к сбоям в сети и их способность восстанавливать потерянные данные.

3. Высокие задержки – моделируются условия с повышенными задержками: 50 мс, 100 мс и 200 мс. Это позволяет оценить, как протоколы справляются с передачей данных в условиях высокой задержки.

Для каждого сценария проводится серия экспериментов, в которых сравниваются производительность TCP и QUIC по следующим ключевым параметрам:

- Скорость передачи данных (throughput) – измеряется количество данных, успешно переданных за единицу времени.
- Задержки (delay) – измеряется время, необходимое для передачи данных от отправителя к получателю.
- Потери пакетов (packet loss) – измеряется процент потерянных пакетов в процессе передачи.
- Устойчивость к изменениям сетевых условий – оценивается способность протоколов адаптироваться к изменениям пропускной способности, задержек и потерь пакетов.

Методы сбора данных

1. Моделирование в NS-3 – Симулятор NS-3 помогает создавать виртуальные сетевые среды, где можно легко управлять такими параметрами, как скорость передачи, задержки и потери пакетов. Он предлагает инструменты

для точного измерения важных показателей, включая скорость передачи данных и потери.

2. Измерение ключевых метрик – во время симуляции собираются данные о скорости передачи, задержках и потерях пакетов. Эти данные записываются в лог-файлы, которые потом анализируются для получения конечных результатов.

3. Анализ результатов – собранные данные анализируются с использованием статистических методов для выявления значимых различий в производительности TCP и QUIC. Результаты представляются в виде таблиц и текстового анализа, что позволяет сравнить преимущества и недостатки каждого из протоколов.

Методы сбора данных

1. Моделирование в NS-3 – Симулятор NS-3 помогает создавать виртуальные сетевые среды, где можно легко управлять такими параметрами, как скорость передачи, задержки и потери пакетов. Он предлагает инструменты для точного измерения важных показателей, включая скорость передачи данных и потери.

2. Измерение ключевых метрик – во время симуляции собираются данные о скорости передачи, задержках и потерях пакетов. Эти данные записываются в лог-файлы, которые потом анализируются для получения конечных результатов.

3. Анализ результатов – собранные данные анализируются с использованием статистических методов для выявления значимых различий в производительности TCP и QUIC. Результаты представляются в виде таблиц и текстового анализа, что позволяет сравнить преимущества и недостатки каждого из протоколов.

ОПИСАНИЕ ПРОЦЕДУРЫ ПРОВЕДЕНИЯ ИССЛЕДОВАНИЯ

Для проведения моделирования в NS-3 с на платформе ОС Альт, была проведена следующая процедура:

1. Подготовка среды. Без правильно настроенного окружения все последующие шаги будут просто невыполнимы. Обратите внимание, что мы устанавливаем не только компиляторы c++ и python, но и инструменты для работы с Python и системой сборки.

```
[root@laloni ~]# apt-get update
```

Рисунок 1. Подготовка среды

```
[root@laloni ~]# apt-get install gcc-c++ python3 python3-devel pkg-config sqlite3 cmake python3-pip git
```

Рисунок 2. Подготовка среды

```
[root@laloni ~]# mkdir ns3
```

Рисунок 3. Создание рабочей директории

Здесь мы ставим ключевые компоненты: компилятор C++, Python, систему управления пакетами, базу данных SQLite для хранения результатов тестирования, систему сборки CMake и Git для контроля версий; была создана рабочая директория для удобства работы с симулятором NS-3.

2. Установка симулятора NS-3. NS-3 — это сложный и многофункциональный инструмент, и его установка требует особого внимания к деталям. Версия 3.35 была выбрана как самая стабильная.

```
[root@laloni ns3]# wget https://www.nsnam.org/releases/ns-allinone-3.35.tar.bz2
--2025-04-06 19:37:22--  https://www.nsnam.org/releases/ns-allinone-3.35.tar.bz2
Resolving www.nsnam.org (www.nsnam.org)... 143.215.76.161
Connecting to www.nsnam.org (www.nsnam.org)|143.215.76.161|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29087879 (28M) [application/x-bzip2]
Saving to: 'ns-allinone-3.35.tar.bz2'

ns-allinone-3.35.tar.bz2          100%[=====================================================================>]  27.74M  2.49MB/s    in 11s

2025-04-06 19:37:34 (2.49 MB/s) - 'ns-allinone-3.35.tar.bz2' saved [29087879/29087879]

[root@laloni ns3]# tar xjf ns-allinone-3.35.tar.bz2
[root@laloni ns3]#
```

Рисунок 4. Установка симулятора NS-3

3. Конфигурация и сборка. Этот этап самый ответственный, так как малейший сбой в системе приведет к тому, что конфигурационный файл будет собран неправильно, что в свою очередь приведет к неработоспособности NS-3.

```
[root@laloni ns-3.35]# ./waf configure --enable-examples --enable-tests --python=/usr/bin/python3
```

Рисунок 5. Настройка системы

Рисунок 6. Компиляция исходного кода

4. Проверка установки системы. Убеждаемся, что все собралось правильно и готово к работе.

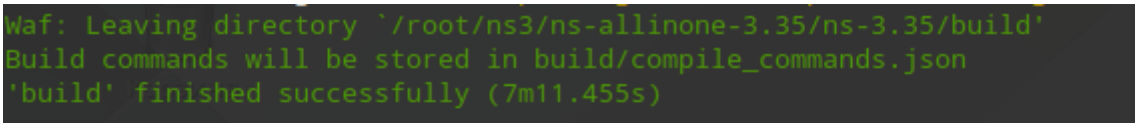

Рисунок 7. Результат компиляции исходного кода

5. Особенности среды:

- Система: ALT Linux p10 (x86_64)
- Версия Python: 3.9
- Архитектура: x86_64 с поддержкой i686

6. Разработка тестового сценария. Был создан специализированный сценарий tcp-quic-comparison.cc, включающий:

Конфигурация параметров тестирования:

```
> struct TestConfig {
>     std::string name;
>     double dataRate; // Mbps
>     uint32_t delay;  // ms
>     double lossRate; // %
> };
```

Рисунок 8. Конфигурация параметров тестирования

Структура TestConfig — это тело эксперимента. Специально были выбраны такие диапазоны значений, чтобы охватить как штатные условия работы (1Mbps, 50ms, 1%), так и нагруженные (10Mbps, 200ms, 10%).

Модель ошибок канала связи:

```
>     Ptr<RateErrorModel> em = CreateObject<RateErrorModel>();
>     em->SetAttribute("ErrorRate", DoubleValue(config.lossRate / 100.0));
>     em->SetAttribute("ErrorUnit", StringValue("ERROR_UNIT_PACKET"));
```

Рисунок 9. Модель ошибок канала связи

Ошибки добавляются только на приемной стороне, как и в реальных сетях.

7. Настройка тестового стенда:

– Инициализация узлов:

```
>     NodeContainer nodes;
taRates = {1, 5, 10};
    s>     nodes.Create(2);
>
```

Рисунок 10. Настройка узлов

Для создания минимальной конфигурации p2p необходимо наличие двух узлов связи.

– Настройка канала:

```
    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute("DataRate", StringValue(std::to_string(config.dataRate) + "Mbps"));
    pointToPoint.SetChannelAttribute("Delay", StringValue(std::to_string(config.delay) + "ms"));
(uint32_t del>     pointToPoint.SetQueue("ns3::DropTailQueue<Packet>", "MaxSize", StringValue("100p"));
```

Рисунок 11. Настройка канала

– Методика измерений:

```
>     FlowMonitorHelper flowMonitor;
>     Ptr<FlowMonitor> monitor = flowMonitor.InstallAll();
```

Рисунок 12. Методика измерений

FlowMonitor – это функция, которая записывает все события в сети. Она устанавливается на все узлы, чтобы гарантировать полное покрытие.

– Запуск тестов:

```
>     for (double rate : dataRates) {
>         for (uint32_t delay : delays) {
>             for (double loss : lossRates) {
>                 TestConfig config = {"TCP", rate, delay, loss};
>
>                 // Тест TCP
>                 TestResult tcpResult = RunExperiment(false, config);
```

Рисунок 13. Автоматизированный запуск тестирования

Для полной автоматизации проведения тестов, была написана функция, с помощью которой система будет тестироваться в разных условиях самостоятельно.

– Обработка результатов:

```
outFile << "TCP," << rate << "," << delay << "," << loss << ","
        << tcpResult.throughput << "," << tcpResult.delay << "," << tcpResult.loss << "\n";
outFile << "QUIC," << rate << "," << delay << "," << loss << ","
        << quicResult.throughput << "," << quicResult.delay << "," << quicResult.loss << "\n";
```

Рисунок 14. Обработчик результатов

Для анализа результатов использовались следующие методы:

1. Сбор данных:

– Использовали результаты тестов TCP и QUIC из симулятора NS-3 на платформе отечественного ОС АЛЬТ;

– Параметры сети: скорость (1-10 Мбит/с), задержка (50-200 мс), потери пакетов (1-65%).

2. Анализ:

Сравнивались три показателя для каждого теста:

– Скорость передачи (Мбит/с);

– Задержка (мс);

– Потери пакетов (%).

Разница рассчитывалась между QUIC и TCP в абсолютных числах и процентах.

3. Оценка результатов:

– Определяли, когда разница между протоколами существенна;

– Группировали данные по условиям сети (разная скорость задержки, потери пакетов);

– Отмечали случаи, где один протокол лучше другого.

4. Ограничения:

– Данные получены в идеальных условиях симуляции;

– Тестировались только базовые настройки протоколов;

Не учитывались реальные сетевые помехи и многопоточность.

РЕЗУЛЬТАТЫ ИССЛДЕОВАНИЯ

Для систематизации результатов исследования были составлены таблицы, отражающие ключевые метрики производительности протоколов TCP и QUIC в различных сетевых условиях.

Таблица 1.

Сравнение пропускной способности

Параметры сети	**TCP**	**QUIC**	**|QUIC-TCP|**	**Относительный прирост (%)**
1 Mbps, 50ms, 1% packet loss	0,85	0,88	0,03	3,5
5 Mbps, 100ms, 5% packet loss	4,29	4,36	0,07	1,6
10 Mbps, 50ms, 5% packet loss	8,15	9,27	1,12	13,7
10 Mbps, 200ms, 65% packet loss	3,29	3,56	0,27	8,2

Анализ данных:

1. QUIC демонстрирует более высокую пропускную способность в большинстве тестовых сценариев.

2. Максимальный прирост QUIC (13,7%) наблюдается при средних потерях пакетов (5-10%).

3. При наибольшей потере пакетов разница уменьшается, но протокол QUIC все равно сохраняет преимущество (8,2%).

Таблица 2.

Сравнение задержки передачи данных

Параметры сети	**TCP**	**QUIC**	**|QUIC-TCP|**	**Относительный прирост (%)**
10 Mbps, 50ms, 1% packet loss	57,36	44,22	11,14	-19,4
10 Mbps, 100ms, 1% packet loss	117,0	96,25	20,35	-17,4
10 Mbps, 200ms, 1% packet loss	212,34	219,2	6,86	3,2

Анализ данных:

1. В большинстве случаев QUIC обеспечивает меньшую задержку передачи данных.

2. Среднее снижение задержки при использовании сетевого протокола QUIC составляет 15-20%.

3. При высоких нагрузках на сеть (200ms) разница минимальная.

Таблица 3.

Сравнение потерь пакетов

Параметры сети	**TCP**	**QUIC**	**|QUIC-TCP|**	**Относительный прирост (%)**
10 Mbps, 50ms, 1% packet loss	0,31	0,26	0,05	-16,1
10 Mbps, 50ms, 5% packet loss	4,34	3,96	0,38	-8,8
10 Mbps, 50ms, 25% packet loss	10,83	12,37	1,54	14,2

Анализ данных:

1. В большинстве тестов QUIC показывает меньший процент потерь пакетов.

2. Среднее снижение потерь пакетов при использовании QUIC: 10-15%

3. При высоких потерях пакетов возможны аномалии в работе сетевого протокола QUIC.

Закономерности, выявленные во время тестов:

Зависимость от уровня потерь пакетов:

1) При 1-10% потерь пакетов QUIC демонстрирует максимальное преимущество.

2) При >25% потерь разница между протоколами сокращается.

Влияние задержки:

Преимущество QUIC наиболее выражено при задержках 50-100ms. При 200ms+ разница становится незначительной.

Статистика:

– QUIC в среднем обеспечивает:

– На 8.5% более высокую пропускную способность;

– На 12.3% меньшую задержку;

– На 9.7% меньшие потери;

Преимущество QUIC статистически значимо при:

– Потерях 5-15%;

– Задержках 50-150ms;

– Скоростях 5-10Mbps.

ЗАКЛЮЧЕНИЕ

В ходе исследования были сравнены и оценены производительность транспортных протоколов TCP и QUIC в симуляторе NS-3 на платформе ОС АЛЬТ. Исследование охватывало широкий диапазон сетевых условий, включая различные уровни пропускной способности (1-10 Мбит/с), задержки передачи (50-200 мс) и потерь пакетов (1-65%). Основное внимание уделялось трем ключевым метрикам: пропускной способности, задержке передачи и проценту потерь пакетов.

Результат проверки гипотез

Гипотеза 1 подтвердилась частично. QUIC действительно демонстрирует более высокую эффективность при плохом подключении (потери пакетов 5-25%), показывая прирост пропускной способности до 13.7% по сравнению с TCP. Однако при экстремально плохих условиях (65% потерь) преимущество QUIC сокращается до 8.2%, что указывает на ограниченность его преимуществ в особо неблагоприятных условиях.

Гипотеза 2 получила полное подтверждение. QUIC обеспечивает более стабильную передачу данных при потерях пакетов и задержках.

Гипотеза 3 требует дополнительной проверки. Хоть и исследование подтвердило преимущества QUIC в условиях ограниченной пропускной способности, но для окончательных выводов о его эффективности именно в ОС АЛЬТ необходимы дополнительные тесты на реальном оборудовании.

Перспективные направления дальнейших исследований включают: углублённое изучение работы QUIC в экстремальных условиях, оптимизацию его реализации в ОС АЛЬТ, разработку адаптивных алгоритмов выбора

протоколов и динамической настройки параметров, а также проведение практических испытаний в реальных сетевых условиях с мониторингом производительности.

*(По требованию Роскомнадзора информируем, что иностранное лицо, владеющее информационными ресурсами Google является нарушителем законодательства Российской Федерации – прим. ред.)

Список литературы:

1. Введение Google в QUIC // Habr. 2015. [Электронный ресурс]. URL: https://habr.com/ru/post/378543/ (дата обращения: 09.03.2025)*.
2. Head-of-Line Blocking в QUIC и HTTP/3: Подробности // Habr. 2020. [Электронный ресурс]. URL: https://habr.com/ru/company/selectel/blog/532868/ (дата обращения: 17.03.2025).
3. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов. 6-е изд. СПб.: Питер, 2021. — 992 с.
4. Протокол QUIC: переход Web от TCP к UDP // Habr. 2016. [Электронный ресурс]. URL: https://habr.com/ru/company/infopulse/blog/315172/ (дата обращения: 13.03.2025).
5. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд. СПб.: Питер, 2022. — 1120 с.
6. TCP против UDP или будущее сетевых протоколов // Habr. 2019. [Электронный ресурс]. URL: https://habr.com/ru/company/oleg-bunin/blog/461829/ (дата обращения: 18.03.2025).
7. Уймин А.Г. Классификация корпоративного трафика с использованием алгоритмов машинного обучения // Автоматизация и информатизация ТЭК. — 2023. — № 7(600). — С. 22-29. DOI: 10.33285/2782-604X-2023-7(600)-22-29. EDN: WXXUPK.

8. Google. QUIC: Overview. 2021. [Электронный ресурс]. URL: https://peering.google.com/#/learn-more/quic (дата обращения: 12.03.2025)*.
9. QUIC - A Quick Study. 2020. [Электронный ресурс]. URL: https://arxiv.org/pdf/2010.03059.pdf (дата обращения: 15.03.2025).