УДК 656.073

Исследование способов диспетчеризации видео на множестве конечных устройств в контексте языка программирования Rust

Ермаков Сергей Геннадьевич — докт. техн. наук, профессор, заведующий кафедрой «Информационные и вычислительные системы». Область научных интересов: цифровая трансформация, цифровизация, обработка и анализ больших данных, эксплуатация автоматизированных систем управления, систем мониторинга, информационно-справочных и аналитических систем. E-mail: ermakov@pgups.ru

Ляпунов Владислав Евгеньевич — магистр, аспирант кафедры «Информационные и вычислительные системы». Область научных интересов: информационные системы. виртуальная реальность, дополненная реальность, иммерсивные технологии. E-mail: Bladl1967@yandex.ru

Шефнер Альберт — студент бакалавриата 3-го курса направления 09.03.01 «Информатика и вычислительная техника». Область научных интересов: информационные системы. E-mail: dev_n0emo@tuta.io

Ахмедов Хаджимурад Арсланович — студент бакалавриата 3-го курса направления 09.03.01 «Информатика и вычислительная техника». Область научных интересов: информационные системы. E-mail: tearus2002@mail.ru

Кот Никита Дмитриевич — студент бакалавриата 3-го курса направления 09.03.01 «Информатика и вычислительная техника». Область научных интересов: информационные системы, обработка больших данных. E-mail: nikitakot072@gmail.com

Петербургский государственный университет путей сообщения Императора Александра I, Россия, 190031, Санкт-Петербург, Московский пр., 9

Для цитирования: Ермаков С. Г., Ляпунов В. Е., Шефнер А., Ахмедов Х. А., Кот Н. Д. Исследование способов диспетчеризации видео на множестве конечных устройств в контексте языка программирования Rust // Интеллектуальные технологии на транспорте. 2024. № 4 (40). С. 59–66. DOI: 10.20295/2413-2527-2024-440-59-66

Аннотация. *Современный мир активно движется в сторону цифровизации, что требует эффективного распределения видеоконтента на множестве конечных устройств. Проблема заключается в том, что различные сценарии использования предъявляют разные требования к безопасности, качеству и задержке при передаче видео. Целью исследования является выявление наиболее подходящих протоколов для диспетчеризации видео в образовательных и реальных приложениях с использованием языка программирования Rust, известного своей безопасностью и производительностью. Исследованы и проанализированы такие протоколы, как HTTPS (TCP), BitTorrent, HLS, WebRTC, SRT и DASH. Результаты показывают, что HLS и DASH обладают преимуществами при адаптивном потоковом вещании в условиях меняющейся сети, в то время как SRT и WebRTC обеспечивают низкую задержку и высокую надежность для приложений, требующих работы в реальном времени. Практическая значимость этих выводов подтверждается успешной интеграцией в образовательные системы, что обеспечивает стабильную работу даже при изменении нагрузки. Обсуждение: в будущем рекомендуется изучить комбинированное использование нескольких протоколов, таких как HLS и SRT, для повышения общей эффективности и надежности передачи видеоданных, а также интеграцию с CDN для улучшения качества и снижения нагрузки.*

Ключевые слова: *Rust, потоковое вещание, HLS, DASH, SRT, WebRTC, BitTorrent, безопасность, CDN, мультимедийные системы*

2.3.5 *— математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей (технические науки),* ***2.3.6*** *— методы и системы защиты информации, информационная безопасность (технические науки)*

Благодарности. *Настоящая статья выполнена при поддержке Федерального государственного бюджетного образовательного учреждения высшего образования «Петербургский государственный университет путей сообщения Императора Александра I» инициативных научных работ, выполняемых студенческими научными коллективами*

Введение

В эпоху цифровизации образовательные учреждения активно используют технологии для улучшения качества обучения и взаимодействия между учащимися, педагогами и родителями. Одним из наиболее перспективных инструментов в этом направлении являются кросс-платформенные приложения, способные работать на различных устройствах и платформах [1]. Один из инструментов, позволяющий создавать такие приложения, — язык программирования высокого уровня Rust, отличающийся своими функциями безопасности памяти. Данная статья призвана исследовать актуальные способы отображения видеоконтента на цифровых фоторамках и прочих устройствах, используя Rust [2].

Критерии

Каждый способ передачи видеоконтента оценивается по следующим критериям:

1. Безопасность — уровень безопасности использования этой технологии на сервере и надежность шифрования передаваемых данных.

2. Надежность — вероятность возникновения ошибки при передаче контента.

3. Качество — уровень качества воспроизводимого видео.

4. Эффективность — нагрузка на сеть при передаче данных.

5. Задержка — время, по истечении которого передаваемый контент становится доступным для воспроизведения на устройстве.

6. Наличие решений на Rust — возможность использования безопасной реализации протокола в рамках исследуемого языка программирования.

Предварительная загрузка по HTTPS (TCP)

HTTPS — зашифрованный вариант протокола HTTP, работающий на базе TCP протокола с TLS-шифрованием.

TCP (Transmission Control Protocol) — это протокол транспортного уровня, который обеспечивает надежную, упорядоченную и безошибочную передачу данных между устройствами в сети. Он является одним из основных протоколов, используемых в интернете, и работает в паре с протоколом **IP** (Internet Protocol), образуя стек **TCP/IP**.

Это решение можно использовать для предварительной загрузки видеофайлов, которые после загрузки могут воспроизводиться неограниченное количество раз без необходимости подключения к сети [3].

1. Высокая безопасность. Протокол **HTTPS** является полностью безопасным благодаря **TLS**-шифрованию [4].

2. Высокая надежность. Загрузка видеофайлов заранее гарантирует их доступность для воспроизведения в любое время, в том числе при отсутствии сети.

3. Высокое качество. Возможно использование более высоких битрейтов и разрешений, так как видео загружается заранее. Улучшение качества изображения и звука, что критично для образовательных и информационных приложений.

4. Средняя эффективность. На каждое устройство необходимо передавать видеоконтент отдельно. Также **TCP** несет в себе дополнительные издержки, но возможности интеграции с **CDN** позволяют распределить нагрузку на сеть и увеличить эффективность.

5. Задержка. Рассматривать задержку для передачи заранее некорректно, так как это не прямое потоковое вещание.

6. Наличие решений на **Rust**. Для языка программирования **Rust** существует множество безопасных реализаций протокола **HTTPS**, например hyper.

Предварительная загрузка по протоколу BitTorrent

BitTorrent — Peer-To-Peer **(P2P)** протокол, призванный увеличить эффективность загрузки контента по сети [5]. Он может быть актуальным в случае необходимости загрузки одинакового контента на множество устройств [6].

1. Низкая безопасность. Протокол **BitTorrent** не задумывался как способ безопасной передачи контента, и в его рамках не существует адекватных механизмов безопасности.

2. Высокая надежность. Загрузка видеофайлов заранее гарантирует их доступность для воспроизведения в любое время, в том числе при отсутствии сети.

3. Высокое качество. Возможно использование более высоких битрейтов и разрешений, так как видео загружается заранее.

4. Высокая эффективность. Контент передается по **P2P**-модели, что позволяет значительно снизить нагрузку на сеть.

5. Задержка. Рассматривать задержку для передачи заранее некорректно, так как это не прямое потоковое вещание.

6. Наличие решений на **Rust**. Существует безопасная реализация **BitTorrent**-клиента для языка **Rust**.

HLS

HLS (HTTP Live Streaming) — это протокол потокового вещания, разработанный компанией Apple, который обеспечивает эффективную передачу видео и аудио через интернет. Одной из ключевых особенностей **HLS** является возможность шифрования контента, что позволяет достичь хорошего баланса между безопасностью и адаптивностью к различным сетевым условиям. **HLS** предлагает хороший баланс между безопасностью и адаптивностью к различным сетевым условиям [7].

1. Высокая безопасность. Поддержка шифрования AES-128 обеспечивает защиту контента за счет индивидуального шифрования сегментов видео и хранения ключей на защищенном сервере, что предотвращает несанкционированный доступ.

2. Высокая надежность. **HLS** обеспечивает высокую надежность за счет сегментации контента и поддержки восстановления передачи при сбоях. Поток делится на небольшие сегменты, что позволяет клиенту продолжить воспроизведение с места последнего успешно загруженного сегмента, минимизируя прерывания.

3. Высокое качество. Возможности интеграции с сетями доставки контента **CDN** обеспечивает стабильное и быстрое воспроизведение видео даже при высоких нагрузках.

4. Средняя эффективность. Возможности интеграции с **CDN** позволяют снизить нагрузку на серверы, распределяя трафик между несколькими узлами.

5. Высокая задержка. Задержка **HLS** может достигать значений 45 секунд и выше, что может сделать выбор этого протокола непривлекательным, если требуется минимизировать задержку.

6. Наличие решений на **Rust**. Для **Rust** существует безопасная реализация чтения и создания m3u8-файлов, что позволит интегрировать данный протокол в существующий сервер или клиент. Также существует реализация сервера — **Xiu**.

WebRTC

WebRTC — технология, позволяющая обеспечить коммуникацию между устройствами в реальном времени. Она поддерживает передачу видео, голоса и данных между участниками, что позволяет создавать мощные решения для голосовой и видеосвязи.

1. Высокая безопасность. **WebRTC** включает в себя такие механизмы безопасности, как аутентификация и контроль доступа и использует использует протоколы **DTLS** (Datagram Transport Layer Security) и **SRTP** (Secure Real-time Transport Protocol) для обеспечения безопасности передаваемых данных. Это делает его подходящим

для использования в приложениях, требующих защиты данных.

2. Высокая надежность. **WebRTC** использует механизмы автоматического восстановления соединения и адаптивного контроля качества для обеспечения стабильности передачи данных даже в условиях сетевых сбоев и потерь пакетов. Это делает его подходящим для использования в реальных условиях с переменными сетевыми характеристиками [8].

3. Среднее качество. При передаче потокового видео уровень качества может ухудшаться в зависимости от условий среды [9].

4. Высокая эффективность. **WebRTC** позволяет устройствам обмениваться данными напрямую, что снижает нагрузку на серверы и уменьшает затраты на инфраструктуру.

5. Низкая задержка. **WebRTC** обеспечивает задержку в диапазоне от 0,2 до 2 с, что делает его привлекательным выбором для систем оповещения.

6. Наличие решений на **Rust**. **WebRTC.rs** обеспечивает первоклассную поддержку протокола в **Rust**.

SRT

Протокол потоковой передачи **SRT** (Secure Reliable Transport), разработанный компанией Hangzhou Hikvision Digital Technology Co., Ltd., представляет собой открытый протокол для передачи видео по интернет-сетям, включая короткие видеопотоки. Он создан для решения проблем, связанных с непредсказуемостью и ненадежностью интернета, акцентируя внимание на безопасности и надежности передачи данных на любые расстояния [10].

1. Высокая безопасность. Протокол **SRT** обеспечивает безопасность передачи данных с помощью шифрования **AES-256**, что защищает видеопотоки от несанкционированного доступа. Без правильного ключа доступ к содержимому становится невозможным [11].

2. Высокая надежность. **SRT** использует механизмы защиты от потери пакетов и метод **ARQ** (Automatic Repeat reQuest) для восстановления данных. Это позволяет минимизировать задержки и потери пакетов, обеспечивая надежную передачу даже в нестабильных сетях.

3. Среднее качество. Протокол адаптируется к изменяющимся условиям сети, включая колебания пропускной способности, что позволяет поддерживать стабильное качество видео, но качество может варьироваться в зависимости от состояния сети.

4. Высокая эффективность. **SRT** может увеличить скорость доставки видео в 3–5 раз по сравнению с традиционными протоколами, что делает его эффективным решением для потоковой передачи.

5. Низкая задержка. SRT обеспечивает передачу видео с минимальной задержкой (от 0,1 до 0,3 с), что критически важно для приложений, требующих реального времени, таких как видеоконференции и прямые трансляции.

6. Наличие решений на **Rust. srt-rs** обеспечивает безопасную поддержку протокола в Rust.

DASH

MPEG-DASH (Dynamic Adaptive Streaming over HTTP) — это технология адаптивного потокового вещания, которая оптимизирует качество аудио- и видеоконтента в зависимости от сетевых условий и возможностей устройства. Она минимизирует буферизацию и улучшает качество воспроизведения на разных устройствах [12].

1. Высокая безопасность. **MPEG-DASH** поддерживает шифрование данных с использованием **HTTPS**, обеспечивая защиту контента. Также возможны цифровые подписи и технологии **DRM**, предотвращающие несанкционированный доступ и копирование.

2. Высокая надежность. Система сегментации **MPEG-DASH** разбивает мультимедийный файл на небольшие сегменты, которые загружаются по мере воспроизведения, обеспечивая стабильную работу даже при нестабильном интернете и снижая риск потери данных.

3. Среднее качество. **MPEG-DASH** адаптирует качество видеопотока в реальном времени, выбирая сегмент с оптимальным битрейтом в зависимости от условий сети, минимизируя буферизацию и обеспечивая плавность воспроизведения.

4. Высокая эффективность. Адаптивное изменение качества потока позволяет эффективно использовать пропускную способность сети и уменьшает нагрузку на каналы связи, обеспечивая воспроизведение контента даже при изменяющихся условиях интернета.

5. Средняя задержка. **MPEG-DASH** минимизирует задержку при загрузке сегментов, что важно для динамичных сценариев, таких как онлайн-трансляции. Это улучшает пользовательский опыт, уменьшая время ожидания [13].

6. Наличие решений на **Rust**. **dash-mpd** позволяет эффективно обрабатывать потоки данных и обеспечивать адаптивное потоковое вещание.

Сравнительный анализ

Был произведен сравнительный анализ различных методов передачи видео на основе оцененных критериев (табл. 1).

Для сценариев, где важны *безопасность и качество*, например, для образовательного контента и трансляций, подойдут **HTTPS (TCP)** и **HLS**.

Для эффективного *массового распространения* одинакового контента, где безопасность не критична подойдет **BitTorrent**.

Для приложений *в реальном времени*, таких как видеозвонки и прямые трансляции, благодаря низкой задержке и высокой надежности идеальны **WebRTC** и **SRT**.

Для адаптивного *потокового вещания*, подходит **DASH**, обеспечивающий высокое качество при изменяющихся условиях сети.

Практический пример

В рамках проекта «Исследование оптимальных методов планирования и распределения вывода данных в мультимониторных системах в реальном времени с учетом динамической нагрузки» было реализовано отображение динамического расписания на день для определенной аудитории образовательного учреждения и системы оповещений. Это решение позволило эффективно интегрировать актуальные данные, получаемые в реальном времени, с использованием технологии **egui** для отображения информации на мультимониторных системах. Важным аспектом стало решение о выборе методов передачи медиаконтента: для отображения частых видео был использован протокол **HTTPS**, а для трансляций — технология **HLS**, что обеспечило стабильную работу системы и высокое качество передачи контента в условиях динамической нагрузки и изменяющихся данных.

Заключение

В ходе исследования были проанализированы различные способы диспетчеризации видеоконтента на конечные устройства с использованием языка программирования **Rust**. Протоколы **HLS** и **DASH** продемонстрировали хорошие возможности для адаптивного потокового вещания, обеспечивая стабильную передачу видео при переменной нагрузке и различных условиях сети. Протокол **SRT** с его низкой задержкой и высокой надежностью оказался подходящим для приложений, требующих реального времени, таких как трансляции и видеоконференции. **WebRTC** также проявил себя как эффективное решение для систем с минимальной задержкой, например для оповещений.

Будущие исследования могут быть направлены на использование нескольких протоколов в комбинации для улучшения эффективности и надежности видеопередачи. Например, сочетание **HLS** и **SRT**

Таблица 1

Сравнение различных методов передачи видеоконтента

Метод передачи	Безопасность	Надежность	Качество	Эффективность	Задержка
HTTPS (TCP)	Высокая	Высокая	Высокое	Средняя	Высокая
BitTorrent	Низкая	Высокая	Высокое	Высокая	Высокая
HLS	Высокая	Высокая	Высокое	Средняя	Высокая
WebRTC	Высокая	Высокая	Среднее	Высокая	Низкая
SRT	Высокая	Высокая	Среднее	Высокая	Низкая
DASH	Высокая	Высокая	Среднее	Высокая	Средняя

или **WebRTC** может сбалансировать адаптивность, минимальную задержку и надежность. Также стоит исследовать интеграцию адаптивных потоковых протоколов с **CDN** для оптимизации нагрузки и качества видео. Разработка таких гибридных решений улучшит пользовательский опыт в приложениях с высокими требованиями к качеству и стабильности, особенно в условиях динамичной сети.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Сидорова В. И. Применение универсальных протоколов для передачи изображений и видео // Молодой ученый. 2022. № 4 (399). С. 41–44.

2. Клабник С., Николс К. Программирование на Rust = The Rust Programming Language. СПб.: Питер, 2021. 592 с.

3. Enabling Secure and Efficient Video Delivery Through Encrypted In-Network Caching / X. Yuan, X. Wang, J. Wang [et al.] // IEEE Journal on Selected Areas in Communications. 2016. Vol. 34, iss. 8. P. 2077–2090. DOI: 10.1109/jsac.2016.2577301

4. Krawczyk H., Paterson K. G., Wee H. On the Security of the TLS Protocol: A Systematic Analysis // Canetti R., Garay J. A. (eds) Advances in Cryptology — CRYPTO 2013: Proceedings of the 33rd Annual Cryptology Conference (Santa Barbara, CA, USA, 18–22 August 2013). Part 1. Lecture Notes in Computer Science. Vol. 8042. Heidelberg: Springer-Verlag, 2013. P. 429–448. DOI: 10.1007/978-3-642-40041-4_24

5. BEP 3: The BitTorrent Protocol Specification. URL: http://www.bittorrent.org/beps/bep_0003.html (дата обращения: 24.11.2024).

6. Xia R. L., Muppala J. K. A Survey of BitTorrent Performance // IEEE Communications Surveys & Tutorials. 2010. Vol. 12, iss. 2. P. 140–158. DOI: 10.1109/SURV.2010.021110.00036

7. The QoS Improvement Using CDN for Live Video Streaming with HLS / W. E. Shabrina [et al.] // Proceedings of the 2020 International Conference on Smart Technology and Applications (ICoSTA), (Surabaya, Indonesia, 20 February 2020). Institute of Electrical and Electronics Engineers, 2020. 5 p. DOI: 10.1109/ICoSTA48221.2020.1570613984

8. WebRTC Security Measures and Weaknesses / B. Feher [et al.] // International Journal of Internet Technology and Secured Transactions. 2018. Vol. 8, no. 1. P. 78–102. DOI: 10.1504/IJITST.2018.092138

9. Fosser E., Nedberg L. Quality of Experience of WebRTC Based Video Communication. Norwegian University of Science and Technology, 2016. 131 p. URL: http://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2409900/15147_FULLTEXT.pdf

10. Adaptive Rate Control for Live Streaming Using SRT Protocol / R. Viola [et al.] // Proceedings of the 2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB) (Paris, France, 27–29 October 2020). Institute of Electrical and Electronics Engineers, 2020. 6 p. DOI: 10.1109/BMSB49480.2020.9379708

11. Biryukov A., Khovratovich D. Related-Key Cryptanalysis of the Full AES-192 and AES-256 // Matsui M. (ed.) Advances in Cryptology — ASIACRYPT 2009: Proceedings of the 15th International Conference on the Theory and Application of Cryptology and Information Security (Tokyo, Japan, 06–10 December 2009). Lecture Notes in Computer Science. Vol. 5912. Heidelberg: Springer-Verlag, 2009. P. 1–18. DOI: 10.1007/978-3-642-10366-7_1

12. Enhancing MPEG Dash Performance Via Server and Network Assistance / E. Thomas [et al.] // Proceedings of the International Broadcasting Convention Conference (IBC 2015) (Amsterdam, Netherlands, 11–15 September 2015). 8 p. DOI: 10.1049/ibc.2015.0014

13. Bouzakaria N., Concolato C., Le Feuvre J. Overhead and Performance of Low Latency Live Streaming Using MPEG-DASH // Proceedings of the Fifth International Conference on Information, Intelligence, Systems and Applications (IISA 2014) (Chania, Greece, 07–09 July 2014). Institute of Electrical and Electronics Engineers, 2014. P. 92–97. DOI: 10.1109/IISA.2014.6878732

Дата поступления: 25.11.2024
Решение о публикации: 27.11.2024

Investigation of Video Dispatching Methods on Multiple End Devices in the Context of the Rust Programming Language

Sergey G. Ermakov — Dr. Sci. in Engineering, Professor, Head of the Department "Information and Computing Systems". Research interests: digital transformation, digitalization, processing and analysis of big data, operation of automated control systems, monitoring systems, information-retrieval and analytical systems. E-mail: ermakov@pgups.ru

Vladislav E. Lyapunov — Master's Degree student, Postgraduate student department "Information and Computing Systems". Research interests: information systems, AR, VR immersive technologies. E-mail: Bladl1967@yandex.ru

Albert Shefner — 3rd year bachelor's degree program 09.03.01 "Computer Science and Engineering". Research interests: information systems. E-mail: dev_n0emo@tuta.io

Khadgimurad A. Akhmedov — 3rd year bachelor's degree program 09.03.01 "Computer Science and Engineering". Research interests: information systems. E-mail: tearus2002@mail.ru

Nikita D. Kot — 3rd year bachelor's degree program 09.03.01 "Computer Science and Engineering". Research interests: information systems, big data analysis. E-mail: nikitakot072@gmail.com

Emperor Alexander I St. Petersburg State Transport University, 9, Moskovsky pr., St. Petersburg, 190031, Russia

For citation: Ermakov S. G., Lyapunov V. E., Shefner A., Akhmedov K. A., Kot N. D. Investigation of Video Dispatching Methods on Multiple End Devices in the Context of the Rust Programming Language // Intellectual Technologies on Transport. 2024. № 4 (40). Pp. 59–66. DOI: 10.20295/2413-2527-2024-440-59-66. (In Russian)

Abstract. *The modern world is actively moving towards digitalization, which requires efficient distribution of video content on a variety of end devices. The problem is that different usage scenarios impose different requirements on security, quality, and latency when transmitting video. The aim of the study is to identify the most suitable protocols for video dispatching in educational and real-world applications using the Rust programming language, known for its security and performance. Protocols such as HTTPS (TCP), BitTorrent, HLS, WebRTC, SRT and DASH have been investigated and analyzed. The results show that HLS and DASH have advantages in adaptive streaming in a changing network environment, while SRT and WebRTC provide low latency and high reliability for applications requiring real-time operation. The practical significance of these findings is confirmed by successful integration into educational systems, which ensures stable work even with a change in workload. Discussion: in the future, it is recommended to explore the combined use of several protocols, such as HLS and SRT, to improve the overall efficiency and reliability of video data transmission, as well as integration with CDN to improve quality and reduce load.*

Keywords: *Rust, live streaming, HLS, DASH, SRT, WebRTC, BitTorrent, security, CDN, multimedia systems*

REFERENCES

1. Sidorova V. I. Primenenie universal'nyh protokolov dlya peredachi izobrazhenij i video // Molodoj uchenyj. 2022. No. 4 (399). S. 41–44. (In Russian)

2. Klabnik S., Nikols K. Programmirovanie na Rust = The Rust Programming Language. SPb.: Piter, 2021. 592 s. (In Russian)

3. Enabling Secure and Efficient Video Delivery Through Encrypted In-Network Caching / X. Yuan, X. Wang, J. Wang [et al.] // IEEE Journal on Selected Areas in Communications. 2016. Vol. 34, iss. 8. P. 2077–2090. DOI: 10.1109/jsac.2016.2577301

4. Krawczyk H., Paterson K. G., Wee H. On the Security of the TLS Protocol: A Systematic Analysis // Canetti R.,

Garay J. A. (eds) Advances in Cryptology — CRYPTO 2013: Proceedings of the 33rd Annual Cryptology Conference (Santa Barbara, CA, USA, 18–22 August 2013). Part 1. Lecture Notes in Computer Science. Vol. 8042. Heidelberg: Springer-Verlag, 2013. P. 429–448. DOI: 10.1007/978-3-642-40041-4_24

5. BEP 3: The BitTorrent Protocol Specification. URL: http://www.bittorrent.org/beps/bep_0003.html (дата обращения: 24.11.2024)

6. Xia R. L., Muppala J. K. A Survey of BitTorrent Performance // IEEE Communications Surveys & Tutorials. 2010. Vol. 12, iss. 2. P. 140–158. DOI: 10.1109/SURV.2010.021110.00036

7. The QoS Improvement Using CDN for Live Video Streaming with HLS / W. E. Shabrina [et al.] // Proceedings of the 2020 International Conference on Smart Technology and Applications (ICoSTA), (Surabaya, Indonesia, 20 February 2020). Institute of Electrical and Electronics Engineers, 2020. 5 p. DOI: 10.1109/ICoSTA48221.2020.1570613984

8. WebRTC Security Measures and Weaknesses / B. Feher [et al.] // International Journal of Internet Technology and Secured Transactions. 2018. Vol. 8, no. 1. P. 78–102. DOI: 10.1504/IJITST.2018.092138

9. Fosser E., Nedberg L. Quality of Experience of WebRTC Based Video Communication. Norwegian University of Science and Technology, 2016. 131 p. URL: http://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2409900/15147_FULLTEXT.pdf

10. Adaptive Rate Control for Live Streaming Using SRT Protocol / R. Viola [et al.] // Proceedings of the 2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB) (Paris, France, 27–29 October 2020). Institute of Electrical and Electronics Engineers, 2020. 6 p. DOI: 10.1109/BMSB49480.2020.9379708

11. Biryukov A., Khovratovich D. Related-Key Cryptanalysis of the Full AES-192 and AES-256 // Matsui M. (ed.) Advances in Cryptology — ASIACRYPT 2009: Proceedings of the 15th International Conference on the Theory and Application of Cryptology and Information Security (Tokyo, Japan, 06–10 December 2009). Lecture Notes in Computer Science. Vol. 5912. Heidelberg: Springer-Verlag, 2009. P. 1–18. DOI: 10.1007/978-3-642-10366-7_1

12. Enhancing MPEG Dash Performance Via Server and Network Assistance / E. Thomas [et al.] // Proceedings of the International Broadcasting Convention Conference (IBC 2015) (Amsterdam, Netherlands, 11–15 September 2015). 8 p. DOI: 10.1049/ibc.2015.0014

13. Bouzakaria N., Concolato C., Le Feuvre J. Overhead and Performance of Low Latency Live Streaming Using MPEG-DASH // Proceedings of the Fifth International Conference on Information, Intelligence, Systems and Applications (IISA 2014) (Chania, Greece, 07–09 July 2014). Institute of Electrical and Electronics Engineers, 2014. P. 92–97. DOI: 10.1109/IISA.2014.6878732

Received: 25.11.2024
Accepted: 27.11.2024