**3GPP TSG SA WG4 #114e *S4-211108***

**E-meeting, 19-28 May, 2021**

|  |
| --- |
| *CR-Form-v12.0* |
| **Pseudo CHANGE REQUEST** |
|  |
|  | **26.804** | **CR** | **<CR#>** | **rev** | **1** | **Current version:** | **0.4.1** |  |
|  |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network |  | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | [FS\_5GMS-EXT] New Transport Protocols - Conclusions and Recommendations |
|  |  |
| ***Source to WG:*** | Tencent  |
| ***Source to TSG:*** | SA4 |
|  |  |
| ***Work item code:*** | FS\_5GMS-EXT |  | ***Date:*** | 2021-08-12 |
|  |  |  |  |  |
| ***Category:*** | **B** |  | ***Release:*** | Rel-17 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)Rel-12 (Release 12)**Rel-13 (Release 13)Rel-14 (Release 14)Rel-15 (Release 15)Rel-16 (Release 16)* |
|  |  |
| ***Reason for change:*** | Provide rationale, conclusions, and recommendations for Section 5.4 |
|  |  |
| ***Summary of change:*** | Provide rationale, conclusions, and recommendations for Section 5.4 |
|  |  |
| ***Consequences if not approved:*** |  |
|  |  |
| ***Clauses affected:*** |  |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** |  |
| ***56***  |  |
| ***This CR's revision history:*** |  |

**===== CHANGE =====**

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] Akamai Blog, "A QUICk Introduction to HTTP/3", April 2020, <https://developer.akamai.com/blog/2020/04/14/quick-introduction-http3>

[3] Fielding, R., Nottingham, M., and J. Reschke, "HTTP/1.1", Work in Progress, Internet-Draft, draft-ietf-httpbis-messaging-13, 14 December 2020, http://www.ietf.org/internet-drafts/draft-ietf-httpbis-messaging-13.txt

[4] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, May 2015, https://www.rfc-editor.org/info/rfc7540

[5] draft-ietf-quic-http-34, "Hypertext Transfer Protocol Version 3 (HTTP/3)", February 2021

[6] D. Bhat, A. Rizk, and M. Zink, "Not so QUIC: A Performance Study of DASH over QUIC," NOSSDAV'17: Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and VideoJune 2017 Pages 13–18 https://doi.org/10.1145/3083165.3083175

[7] AWS, "Achieving Great Video Quality Without Breaking the Bank", Streaming Media June 2019, [[https://pages.awscloud.com/rs/112-TZM-766/images/GEN elemental-wp-achieving-great-video-quality-without-breaking-the-bank.pdf](https://pages.awscloud.com/rs/112-TZM-766/images/GEN%20elemental-wp-achieving-great-video-quality-without-breaking-the-bank.pdf)](https://pages.awscloud.com/rs/112-TZM-766/images/GEN%20elemental-wp-achieving-great-video-quality-without-breaking-the-bank.pdf)

[8] Netflix, "Optimized shot-based encodes: Now Streaming!", Netflix Blog, May 2018, https://netflixtechblog.com/optimized-shot-based-encodes-now-streaming-4b9464204830

[9] DASH-IF/DVB Report on Low-Latency Live Service with DASH, July 2017, available here: <https://dash-industry-forum.github.io/docs/Report%20on%20Low%20Latency%20DASH.pdf>

[10] DASH-IF IOP Guidelines v5, Low-latency Modes for DASH, available here: <https://dash-industry-forum.github.io/docs/CR-Low-Latency-Live-r8.pdf>

[11] ISO/IEC 23009-1, "Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats".

[12] IETF RFC 8673, "HTTP Random Access and Live Content".

[13] 3GPP TR 26.939: "Guidelines on the Framework for Live Uplink Streaming (FLUS)".

[14] 3GPP TS 26.238: "Uplink Streaming".

[15] 3GPP TS 26.501: "5G Media Streaming (5GMS); General description and architecture".

[16] 3GPP TS 26.512: "5G Media Streaming (5GMS); Protocols".

[17] ISO/IEC 13818-1:2019: "Information technology — Generic coding of moving pictures and associated audio information — Part 1: Systems".

[18] SCTE 35 2020: "Digital Program Insertion Cueing Message", <https://www.scte.org/pdf-redirect/?url=https://scte-cms-resource-storage.s3.amazonaws.com/SCTE-35-2020_notice-1609861286512.pdf>

[19] ISO/IEC 23000-19:2020 Information technology — Multimedia application format (MPEG-A) —Part 19: Common media application format (CMAF) for segmented media

[20] ISO/IEC 23009-1:2019/DAMD1 Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats — Amendment 1: CMAF support, events processing model and other extensions [21] VSF TR-06-01, RIST Simple Profile, <https://www.videoservicesforum.org/download/technical_recommendations/VSF_TR-06-1_2018_10_17.pdf>

[22] VSF TR-06-02: "RIST Main Profile", <https://www.videoservicesforum.org/download/technical_recommendations/VSF_TR-06-2_2020_03_24.pdf>

[23] 3GPP TS 23.501: "System architecture for the 5G System (5GS)".

[24] 3GPP TS 23.502: "Procedures for the 5G System (5GS)".

[25] 3GPP TS 29.517: "5G System; Application Function Event Exposure Service; Stage 3".

[26] 3GPP TS 29.244: "Interface between the Control Plane and the User Plane nodes; Stage 3".

[27] IETF RFC 6733: "Diameter Base Protocol".

[28] 3GPP TS 29.514: "5G System; Policy and Charging Control over Rx reference point; Stage 3".

[29] IETF RFC 7657: "Differentiated Services (Diffserv) and Real-Time Communication", November 1995.

[30] IETF RFC 3168: "The Addition of Explicit Congestion Notification (ECN) to IP", September 2001.

[31] C. Krasic, M. Bishop, and A. Frindell, Ed., draft-ietf-quic-qpack-21, "QPACK: Header Compression for HTTP/3", Work in Progress, Internet-Draft, 2 February 2021

[32] J. Iyengar, Ed. and M. Thomson, Ed., draft-ietf-quic-transport-34, "QUIC: A UDP-Based Multiplexed and Secure Transport", Work in Progress, Internet-Draft, 15 January 2021

[33] M. Thomson, Ed. and S. Turner, Ed., draft-ietf-quic-tls-34, "Using TLS to Secure QUIC", Work in Progress, Internet-Draft, 15 January 2021

[34] J. Iyengar, Ed. and I. Swett, Ed., draft-ietf-quic-recovery-34, "QUIC Loss Detection and Congestion Control", Work in Progress, Internet-Draft, 15 January 2021

[35] IETF RFC 5681, "TCP Congestion Control".

[36] M. Kuehlewind and B. Trammell, draft-ietf-quic-manageability-11, “Manageability of the QUIC Transport Protocol”, Work in Progress, Internet-Draft, 21 April 2021

[37] N. Cardwell et. al. “BBR Updates: Internal Deployment, Code, Draft Plans”, 9 March 2021, https://datatracker.ietf.org/meeting/110/materials/slides-110-iccrg-bbr-updates-00.pdf

[38] ETSI TS 103 799: "Publicly Available Specification (PAS); DASH-IF Content Protection Information Exchange Format".

[39] ISO/IEC JTC1/SC29/WG11/N19062 23090‑8 FDIS: "MPEG-I: Network-based Media Processing — Network-Based Media Processing Specification".

[40] 3GPP TS 26.247: "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)".[41] 3GPP TS 23.503: "Policy and charging control framework for the 5G System (5GS); Stage 2".

[42] 3GPP TS 29.514: "5G System; Policy Authorization Service; Stage 3".

[43] 3GPP TS 29.522: "5G System; Network Exposure Function Northbound APIs; Stage 3".

[44] 3GPP TS 29.122: "T8 reference point for Northbound APIs".

[45] 3GPP TS 29.512: "5G System; Session Management Policy Control Service; Stage 3".

[46] 3GPP TS 26.803: "5G Media Streaming (5GMS); Architecture extensions".

[47] 3GPP TS 23.558: "Architecture for enabling Edge Applications (EA)".

[48] 3GPP TS 23.288: "Architecture enhancements for 5G System (5GS) to support network data analytics services".

[49] Tdoc S4-210723: "Generic architecture for data collection and reporting", submission from BBC, Dolby Laboratories Inc., LM Ericsson and Qualcomm Incorporated to SA4#114-e, May 19-28, 2021.

[50] Tdoc S2-2103267: "Extension of Naf\_EventExposure for observed service experience data collection from UEs", CR from InterDigital to SA2#144e, Apr 12-16, 2021.

[51] 3GPP TS 26.114: "IP Multimedia Subsystem (IMS); Multimedia telephony; Media handling and interaction".

[52] Tdoc S2-2104496: "Extension of Naf\_EventExposure for observed service experience data collection from UEs", CR from Qualcomm Incorporated to SA2#145e, May 17-28, 2021.

[53] 3GPP TS 26.118: "Virtual Reality (VR) profiles for streaming applications".

[54] 3GPP TS 26.346: "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs".

[W] IETF RFC 822: “STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES”, August 13, 1982.

[X] IETF RFC 1521: "MIME (Multipurpose Internet Mail Extensions)”, September 1993.

[SRT-QUIC] Maxim Sharabayko and Mariam Sharabayko "Tunnelling SRT over QUIC”, draft-sharabayko-srt-over-quic-00, Work in Progress, Internet-Draft, 28 July, 2021.

[DASH-QUIC] A. Mondal and S. Chakraborty, "Does QUIC Suit Well With Modern Adaptive Bitrate Streaming Techniques?”, IEEE NETWORKING LETTERS, VOL. 2, NO. 2, JUNE 2020.

**===== CHANGE =====**

## 5.4 Additional/new transport protocols

### 5.4.1 Description

#### 5.4.1.1 General

Media streaming applications continue to use HTTP-based distribution protocols, but newer versions of HTTP such as HTTP/2 or HTTP/3 are introduced, see for example also TR 26.925 [5], clause 6.1.4. The architectural and performance impacts of such protocols for 5G-based media distribution is unclear and requires study. The study also considers how Media Players may use functionalities existing in new transport protocols, and also investigate the impact of new transport protocols on 5GMS usage and traffic identification (e.g. Service Data Flow Descriptions).

Based on SMTP (Simple Mail Transport Protocol) [W] and reusing MIME (Multipurpose Internet Mail Extensions) notation [X], the HTTP protocol (also known as web protocol), powers most websites, mobile apps, and videos. It was created by Tim Berners-Lee at CERN in 1989 and has been enhanced over the years to keep up with the ever-changing World Wide Web. Currently, the web is a mixture of HTTP/1.1 [3] and HTTP/2 [4] adoption. Most well-known websites are running HTTP/2, while smaller websites and late adopters plan to migrate to HTTP/2 in the near future as it is relatively easy to implement. HTTP/2 is used by about 45% of websites and supported by all major web browsers. HTTP/3 is only used by about 5% of websites now and not well-supported by web browsers yet. However. significant HTTP/3 deployments are emerging. For example, YouTube™ has for a long time been offering a pre-RFC draft version to any client that wants to use it, especially the Chrome™ browser. Other browsers are expected to follow soon after waiting for the QUIC and HTTP/3 RFCs to be published before mainlining that feature.

HTTP/2 introduces the "Streams" concept at HTTP level and each stream can have different priorities. All objects can from a web-page can be multiplexed in single long-lived TCP connection. Also, HTTP/uses header compression (HPACK) to avoid verbose/clear text. Also, HTTP/2 pseudo-mandates TLS to prevent “middle boxes” from messing up with the content. However, HTTP/2 does not remove the drawbacks of TCP’s head-of-line blocking - packet loss on one stream will block all other streams until recovery even if packets for all other streams are correctly received.

HTTP/2 testing shows [2] that the delivery of large objects over HTTP/2 can be slower than over HTTP/1.1 when there is packet loss. This is because HTTP/2 uses a single TCP connection, versus about six connections which most web browsers open over HTTP/1.1. In addition, the TCP congestion control algorithms reduce the TCP congestion window size, resulting in fewer bytes sent over the wire when using just one TCP connection.

HTTP/2 provides on average a 5% to 15% performance improvement on page load times over HTTP/1.1 [2]. HTTP/1.1 allows persistent TCP connections, but requests still had to be serialized, resulting in the well-known "HTTP head of queue blocking". In order to improve downloads, many TCP flows still needed to be parallelized to speed up delivery.

The solution to this problem is to use HTTP/2 over a different transport protocol that provides more efficient congestion control. One option would be to upgrade and modify TCP, but modifying TCP implementations is viewed as an impossible task. For example, middle boxes such as NAT, Firewalls, and Load balancers are problematic, because they get rarely upgraded which prevents any updates to TCP. TCP is also hard to evolve as it is almost always implemented as part of operating system kernels, requiring an updated operating system as part of TCP updates. Hence, it was considered easier to introduce transport functions on top of UDP, outside the operating system kernel, in the user space – referred to as QUIC.

That, in essence, is what HTTP/3 [5] is: HTTP/2 over User Datagram Protocol (UDP) based on IETF QUIC. HTTP/3 is a thin layer on top of QUIC [32] including QPACK header compression [31]. The main QUIC functions are connection and stream multiplexing [32], fast startup[32], loss recovery, in-order delivery (within stream) [32], flow control [32]. TLS1.3 (handshake) [33], loss recovery and congestion control [34].

By multiplexing multiple concurrent logical streams over a single UDP-based transport association, and by giving each stream its own independent byte offset numbering space, packet loss in one stream does not block progress on other logical streams in the same QUIC connection. (However, the affected stream will still block when packets are lost, so as to guarantee in-order delivery of payloads to the application.).



Figure 5.4-1: HTTP/2 and HTTP/3 protocol stacks

For an entertaining introduction to QUIC and HTTP/3, please check <https://www.youtube.com/watch?v=B1SQFjIXJtc>.

However, using QUIC for adaptive streaming still requires study as under certain circumstances, the quality using QUIC may even degrade for DASH-based streaming than it would increase [6]. The evaluation results show that using the unmodified DASH algorithms on top of QUIC may not provide the anticipated performance boost when compared to the standard DASH over TCP.

The main expected benefit of QUIC is being able to multiplex requests for all Adaptation Sets onto the same transport association, and then to manage the network QoS on that aggregate connection. This has a valuable operational benefit to a CDN operator (including the 5GMS AS) in reducing the number of UDP ports that a server needs to keep open. Another benefit is being able to migrate connections from one IP address to another with minimal interruption to either client or server. This is useful when the client moves, but it is also useful when the server changes (e.g. in edge computing relocation Use Cases).

Because HTTP/3 and IETF QUIC are new protocols, there are several questions about performance and management that need to be investigated during this study.

#### 5.4.1.2 Performance Considerations for HTTP/3 using IETF QUIC over 5G Networks

The IETF specifications for HTTP/3 [5][31] and the core QUIC functions [32, 33, 34] are now approved in the IETF, and have been broadly deployed by a number of browser vendors and content providers, since the IETF QUIC working group has focused on specification, implementation, and, after the specifications were sufficiently stable enough, deployment, all in parallel. The performance of HTTP/3 over IETF QUIC in environments that have not been encountered during deployments to date is still an open question. Of greatest interest for this study, is the performance of HTTP/3 over IETF QUIC in 5G networks. Although deployment of 5G networks has begun, most deployment experience with HTTP/3 over IETF QUIC in mobile networks over the past few years has been in non-5G networks.

When end users have used HTTP/3 over QUIC to access servers outside the 5G core network, 3GPP terminal mobility was handled transparently by the 3GPP network, and the UE’s IP address(es) didn’t change. If a server’s IP addresses changed, this was often not visible to the user, due to the widespread deployment of CDNs and load balancers in data centers. If edge computing resource IP addresses change in relocation use cases, QUIC connection migration could be used to reduce the impact on user experience, but this needs to be analyzed carefully.

MPEG-DASH has provided years of good user experience running over HTTP/1.1. As use cases arise which require very low latency, it is reasonable to better understand how, and when, MPEG-DASH might take advantage of HTTP/3 and analyze whether this has any implications for this study.

[6] raised the issue that MPEG-DASH performance might be lower over HTTP/3 than over HTTP/2. This reference was comparing Google’s pre-standardization QUIC implementation to highly optimized HTTP/2 over TCP implementations, and is about five years old, as of this writing, but the point remains – we need to know more about unmodified MPEG-DASH over standardized HTTP/3 implementations.

In particular, in the downlink media streaming Use Case, the server chooses the algorithm but the client media player decides whether to use a long-lived connection or to drop and reconnect when changing representation/rendition or adaptation/switching set, and the decision on connection (re)use is often delegated by the client application to a library, and this library might not offer an API to influence connection reuse. QUIC optimisations for reducing connection setup time (0-RTT or 1-RTT handshake (as described in [32]) mitigate this suboptimal use of connections to some extent, but the if the client, directly or indirectly by delegating the decision to the library, chooses to close and open connections, any QUIC congestion control algorithm begins probing for available bandwidth with no information about path characteristics. The connection handling characteristics of common APIs need to be analyzed as part of this study.

#### 5.4.1.3 Performance Considerations for IETF QUIC over 5G networks

The standardized QUIC congestion control and recovery procedures in [34] are chosen to emulate TCP’s standardized behaviors ([35], plus extensions). These are quite conservative, and not match current work on delay-based congestion control and recovery mechanisms, which have also seen wide deployment in QUIC implementations. In principle, delay-based congestion control and recovery mechanisms should improve user experience for streaming media applications, but this isn’t known yet, and this needs to be carefully analyzed.One of the core functions of QUIC is the capability to migrate connections without application involvement when endpoint IP addresses change, rather than requiring the detection of a connection failure, teardown of that connection, and setup of a new connection. Connection migration is one of the key QUIC functions that we do not have a great deal of experience with – implementers in the IETF said they were concentrating on performance for a connection, and many had not completely implemented or tested connection migrations at scale (data point is from October 2020).

#### 5.4.1.4 Management Considerations for HTTP/3 and QUIC in 5G networks

One of the biggest distinctions between HTTP/2 over TCP and HTTP/3 over QUIC has been the encryption of almost all transport-level information in QUIC. This information, which was not encrypted in TCP even when it was carrying encrypted payloads, was often used in network management to identify and troubleshoot performance problems on the Internet. In most of experience with HTTP/3 over QUIC deployments to date, content providers (e.g. Google/YouTube, Facebook, etc.) have terminated at least one end of the encrypted end-to-end connections, allowing them to identify problems at the QUIC transport level. That might be true in 5G deployments, or it might not be the case. If that is not the case, it would be very useful to consider the guidance in [36] as part of this study. It is also likely that operator deployments which relied on split-TCP connections to improve performance over radio links will require reconsideration for QUIC-based t, since QUIC transport-level information is not available unless a device has a security context for the encrypted QUIC connection. It is likely that we need to enhance the 3GPP QoS framework, and that if an application is using QUIC, 5-tuples are not sufficient for per-flow QoS.

Although QUIC can be implemented as part of operating system kernels, it is commonly implemented in user space, allowing frequent updates to congestion control and recovery procedures, including introduction of entirely new procedures (e.g. BBR, "Bottleneck Bandwidth and Round-trip propagation time" congestion control [37]). It is likely that the performance characteristics of HTTP/3 applications will vary more dynamically than HTTP/1.1 implementations that have been used by MPEG-DASH in the past. This may also have implications for existing applications if they migrate from MPEG-DASH over HTTP/1.1 or HTTP/2, to MPEG-DASH over HTTP/3.

### 5.4.2 Collaboration Scenarios

The QUIC protocol [32] is a general-purpose transport protocol, although most current deployments have been in conjunction with HTTP/3 [5]. Even when limiting discussion to HTTP/3 over QUIC, this capability can be relevant to multiple key topics described in the present document, as well as the basic procedures for 5G Media Streaming described in TS 26.501 [15].

For this key topic, the discussion will focus on the collaboration scenario where a 5GMS Application Provider runs an adaptive media streaming service between an HTTP/3-enabled 5GMSd AS and an HTTP/3-enabled 5GMSd Client using downlink media streaming over M2d and M4d. This is Collaboration Scenario 2, from clause A.2 in the present document.

### 5.4.3 Deployment Architectures

The substitution of HTTP/3 for HTTP/2, or even for HTTP/1.1, has very little effect on high-level deployment architectures.

In the following discussion, the term *HTTP/3 client* is used for the entity that opens an HTTP/3 connection, and the term *HTTP/3 server* is used for the entity that the HTTP/3 client wishes to communicate with.

- If HTTP/3 is used at reference point M4d in Figure A.2-1, the 5GSMd Client acts as an HTTP/3 client, and the 5GSMd AS acts as an HTTP/3 server.

- If the 5GSMd AS and 5GSMd Application Provider are using HTTP/3 over the M2d interface, either entity might act as an HTTP/3 client and initiate communication with the other, as described below.



1. In order to open an HTTP/3 connection and retrieve a resource, an HTTP/3 client possesses a target URL providing a *scheme*, which needs to be "https:", a *location*, which must be a resolvable DNS name, and a *path* on the location to the resource. This target URL may be obtained in a number of ways (configuration, dynamic lookup when the HTTP/3 client begins operation, or even from a resource that the HTTP/3 client has already retrieved).

2. The location is resolved to an IP address, by looking up a DNS name, using HTTPS, or other mechanisms that would be used to resolve any other URL location.

3. Once in possession of the IP address for the HTTP/3 server, the HTTP/3 client attempts to open an HTTP/3 connection to the HTTP/3 server. In order to use HTTP/3, the HTTP/3 client communicates with the HTTP/3 server using the underlying QUIC protocol, sent over UDP. The HTTP/3 client and HTTP/3 server perform a TLS 1.3 handshake, and when this is complete, both HTTP/3 client and HTTP/3 server have validated the connection.

4. When connection validation is complete, the HTTP/3 client can begin performing normal HTTP requests over the HTTP/3 connection. In the case of M4d interactions, for example, the HTTP/3 client issues an HTTP GET request.

5. When the HTTP/3 server receives an HTTP request, it responds using normal HTTP status codes, and then performs the requested operation if the request has succeeded.

There are many more details involved, but this description gives a sense of how little changes at the application layer when 5GMS begins using HTTP/3 as its application protocol.

### 5.4.4 Mapping to 5G Media Streaming and High-Level Call Flows

#### 5.4.4.1 General

Using HTTP/3 for 5GMS functions that currently use earlier versions of HTTP (either HTTP/1.1 or HTTP 2/) can provide performance improvements, such as:

1. “Single-threading” HTTP requests and responses – HTTP/1.1 and HTTP/2 don’t have any way of disambiguating response packets when multiple requests are outstanding on a connection, so the server sends responses as “next request received, next response returned”, even if the HTTP client has “pipelined” multiple requests without waiting for its outstanding requests to be responded to.

2. Avoiding TCP “Head-of-Line” (“front-of-queue”) blocking that blocks all further response packets from being delivered to the application until a missing response packet has been detected and successfully retransmitted.

3. Setting up connections in at most one “Round-Trip Time”, or RTT, as opposed to multiple RTTs, as is required to set up a TCP connection with a three-way handshake, and then to perform a TLS 1.3 handshake that requires at least one more RTT, and potentially more than one.

But beyond these performance improvements, there are very few impacts of adding the capability to use HTTP/3 to high-level call flows in 5GMS environments.

### 5.4.5 Potential open issues

#### 5.4.5.1 Introduction

Most of the potential questions previously identified in clause 5.4.1 don’t appear to be continuing concerns, as HTTP/3 and QUIC are being deployed on the Internet at scale. A few open issues are worth noting here.

#### 5.4.5.2 Streaming Protocols taking advantage of HTTP/3 capabilities

Because so many media providers have used HTTP-based transport mechanisms, they have been forced to provide “workarounds” to overcome the TCP-related deficiencies that HTTP implementations were forced to deal with. Confirming with the predominant implementors of HLS, MPEG-DASH, SRT, and even RTP what their timeframes are for versions of these protocols that take advantage of HTTP/3 capabilities would be useful, but these versions are already starting to appear. “Tunnelling SRT over QUIC” has already been submitted as an Internet-Draft [SRT-QUIC], and even proposals for RTP over QUIC are also under discussion – some using HTTP/3 as a framework, and others running directly over QUIC.

#### 5.4.5.3 3GPP-specific impediments to HTTP/3-QUIC deployment

One well-recognized impediment to HTTP/3-QUIC deployment in the broader Internet is that the QUIC protocol is a well-behaved transport protocol, but it runs over the UDP protocol, which is not understood to be well-behaved. Traditional UDP protocols have been query–response protocols, such as DNS domain name resolution ("what is the IP address for this domain name?" followed by "This is the IP address for that domain name"), and normally use well-known port numbers (if you send a UDP packet to port 53, network firewalls will assume that it is a DNS query, and unlikely to cause congestion).

HTTP/3, like any other protocol running over QUIC, will look like a highly encrypted UDP protocol, and it **might** use the UDP port number that matches the TCP port number of the same protocol running over TCP, but that is not a requirement, so network operators often investigate UDP packets being sent to an unfamiliar port number, especially if that traffic does not seem to be query-response traffic.

If their investigation is not reassuring, they may block UDP packets being sent to an unfamiliar port number at an unfamiliar IP address, and even if they don’t block that traffic, they may rate-limit the traffic to prevent their network links being overwhelmed by unknown traffic that might not respond to congestion indications. So, on the Internet, HTTP applications that attempt to use HTTP/3 are prepared to fall back to HTTP/2 over TCP, which is more reassuring for network operators.

#### 5.4.5.4 Adaptive Streaming clients operating on top of HTTP/3 capabilities

Adaptive streaming clients are implemented to overcome the TCP-related deficiencies that HTTP implementations were forced to deal with. Today, typically the adaptive streaming client is even unaware that it operates on top of HTTP/3 and QUIC instead of HTTP/1.1 and TCP. A DASH client as documented in TS 26.512, clause 13, included some typical functions that may be impacted by the operation on top of HTTP/3, in particular:

- Throughput estimation: estimates the throughput from the 5GMSd Application Server. The throughput is typically computed as the object size divided by the download time where the download time is the time difference between the first and the last bytes received for that object. According to [DASH-QUIC], an adaptive streaming client typically measures the multiplexing of audio and video streams over a single UDP socket results in additional response latency for the audio segments, which are not captured during the calculation of channel throughput.

- Request Scheduling: The adaptive streaming client schedules requests under the assumption of TCP operation. It typically operates for example audio and video on two separate sockets for the two HTTP streams, each of the sockets maintains its own socket buffer. For HTTP/3, both the streams and uses a single UDP socket having a single socket buffer. Therefore, the HTTP responses from both the streams interfere, and higher response rate at one stream affects the queuing delay for the response at the other stream.

Based on these operations, it is important that the adaptive streaming client would

1. be aware of the operation on top of HTTP/3
2. adapts its operation based on HTTP/3 properties

Details are for further study.

#### 5.4.5.5 5GMS Operation taking advantage of HTTP/3 capabilities

5G Media Streaming provides the ability to support regular OTT media streaming by providing additional and auxiliary information between the Media Session Handler and the 5GMSd AF. Supported functions in Rel-16 include telco CDN, network assistance and for example metrics reporting. It would be appropriate to adjust 5GMS function to HTTP/3 based delivery. As an example, certain DASH metrics are designed for TCP based streaming and would preferably be updated to take into account HTTP/3 based delivery.

Editor’s Note: That’s on the broader Internet. Can we do better on 5G networks?

### 5.4.6 Candidate Solutions

ffs

#### 5.4.6.1 Streaming Protocol

An important question we may face in further discussion is what direction(s) we expect to go on HTTP-based and QUIC-based media streaming protocols, even in the best understood scenarios. The following table contains some of the known alternatives, and these aren’t mutually exclusive. They could **all** coexist in 5GMS service offerings.

|  |  |  |
| --- | --- | --- |
| Scenario | Media Streaming protocol | Uses HTTP or QUIC? |
| 1 | MPEG-DASH | HTTP/3 over QUIC |
| 2 | HLS – HTTP Live Streaming | HTTP/3 over QUIC |
| 3 | SRT – Secure Reliable Transport [SRT-QUIC] | QUIC |
| 4 | RTP – Realtime Transport Protocol | Proposals for both usages |
| 5 | RUSH – Reliable (unreliable) streaming protocol | QUIC |

**===== END CHANGES =====**