                    Operational Considerations for Streaming Media
                        draft-ietf-mops-streaming-opcons-09

Abstract

   This document provides an overview of operational networking issues
   that pertain to quality of experience when streaming video and other
   high-bitrate media over the Internet.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   This document examines networking issues as they relate to quality of
   experience in Internet media delivery, especially focusing on
   capturing characteristics of streaming video delivery that have
   surprised network designers or transport experts who lack specific
   video expertise, since streaming media highlights key differences
   between common assumptions in existing networking practices and
   observations of video delivery issues encountered when streaming
   media over those existing networks.

   This document specifically focuses on streaming applications and

defines streaming as follows:

*   Streaming is transmission of a continuous media from a server to a
    client and its simultaneous consumption by the client.

*   Here, "continuous media" refers to media and associated streams
    such as video, audio, metadata, etc. In this definition, the
    critical term is "simultaneous", as it is not considered streaming
    if one downloads a video file and plays it after the download is
    completed, which would be called download-and-play.

This has two implications.

*   First, the server's transmission rate must (loosely or tightly)
    match to client's consumption rate in order to provide
    uninterrupted playback. That is, the client must not run out of
    data (buffer underrun) or accept more data than it can buffer
    before playback (buffer overrun) as any excess media that cannot
    be buffered is simply discarded.

*   Second, the client's consumption rate is limited not only by
    bandwidth availability,but also media availability. The client
    cannot fetch media that is not available from a server yet.

This document contains

*   A short description of streaming video characteristics in
    Section 2, to set the stage for the rest of the document,

*   General guidance on bandwidth provisioning (Section 3) and latency
    considerations (Section 4) for streaming video delivery,

*   A description of adaptive encoding and adaptive delivery
    techniques in common use for streaming video, along with a
    description of the challenges media senders face in detecting the
    bitrate available between the media sender and media receiver, and
    collection of measurements by a third party for use in analytics
    (Section 5),

*   A description of existing transport protocols used for video
    streaming, and the issues encountered when using those protocols,
    along with a description of the QUIC transport protocol [RFC9000]

that we expect to be used for streaming media ([Section 6](#)),

   * A description of implications when streaming encrypted media
      ([Section 7](#)), and

   * A number of useful pointers for further reading on this rapidly
      changing subject ([Section 8](#)).

   Making specific recommendations on operational practices aimed at
   mitigating the issues described in this document is out of scope,
   though some existing mitigations are mentioned in passing. The
   intent is to provide a point of reference for future solution
   proposals to use in describing how new technologies address or avoid
   existing observed problems.

1.1. Notes for Contributors and Reviewers

   Note to RFC Editor: Please remove this section and its subsections
   before publication.

   This section is to provide references to make it easier to review the
   development and discussion on the draft so far.

1.1.1. Venues for Contribution and Discussion

   This document is in the Github repository at:

   https://github.com/ietf-wg-mops/draft-ietf-mops-streaming-opcons
   (https://github.com/ietf-wg-mops/draft-ietf-mops-streaming-opcons)

   Readers are welcome to open issues and send pull requests for this
   document.

   Substantial discussion of this document should take place on the MOPS
   working group mailing list (mops@ietf.org).

   * Join: https://www.ietf.org/mailman/listinfo/mops
      (https://www.ietf.org/mailman/listinfo/mops)

   * Search: https://mailarchive.ietf.org/arch/browse/mops/
      (https://mailarchive.ietf.org/arch/browse/mops/)

[2](#). Our Focus on Streaming Video

   As the internet has grown, an increasingly large share of the traffic
   delivered to end users has become video. Estimates put the total
   share of internet video traffic at 75% in 2019, expected to grow to
   82% by 2022. This estimate projects the gross volume of video
   traffic will more than double during this time, based on a compound
   annual growth rate continuing at 34% (from [Appendix D](#) of [[CVNI](#)]).

   A substantial part of this growth is due to increased use of
   streaming video, although the amount of video traffic in real-time
   communications (for example, online videoconferencing) has also grown
   significantly. While both streaming video and videoconferencing have
   real-time delivery and latency requirements, these requirements vary
   from one application to another. For example, videoconferencing
   demands an end-to-end (one-way) latency of a few hundreds of
   milliseconds whereas live streaming can tolerate latencies of several
   seconds.

   In many contexts, video traffic can be handled transparently as
   generic application-level traffic. However, as the volume of video
   traffic continues to grow, it is becoming increasingly important to
   consider the effects of network design decisions on application-level
   performance, with considerations for the impact on video delivery.

   Much of the focus of this document is on reliable media using HTTP
   over TCP, which is widely used because

   * support for HTTP is widely available in a wide range of operating
      systems,

   * HTTP is also used in a wide variety of other applications,

   * HTTP has been demonstrated to provide acceptable performance over
      the open Internet,

   * HTTP includes state of the art standardized security mechanisms,
      and

   * HTTP can make use of already-deployed caching infrastructure.

Unreliable media delivery using RTP and other UDP-based protocols is also discussed in Section 4.1, Section 5.7, Section 6.1, and Section 7.2, but it is difficult to give general guidance for these applications. For instance, when loss occurs, the most appropriate response may depend on the type of codec being used.

3. Bandwidth Provisioning

3.1. Scaling Requirements for Media Delivery

3.1.1. Video Bitrates

Video bitrate selection depends on many variables including the resolution (height and width), frame rate, color depth, codec, encoding parameters, scene complexity and amount of motion. Generally speaking, as the resolution, frame rate, color depth, scene complexity and amount of motion increase, the encoding bitrate increases. As newer codecs with better compression tools are used, the encoding bitrate decreases. Similarly, a multi-pass encoding generally produces better quality output compared to single-pass encoding at the same bitrate, or delivers the same quality at a lower bitrate.

Here are a few common resolutions used for video content, with typical ranges of bitrates for the two most popular video codecs [Encodings].

Name	Width x Height	H.264	H.265
DVD	720 x 480	1.0 Mbps	0.5 Mbps
720p (1K)	1280 x 720	3-4.5 Mbps	2-4 Mbps
1080p (2K)	1920 x 1080	6-8 Mbps	4.5-7 Mbps

```
+-----------+---------------+-----------+------------+
| 2160p (4k) | 3840 x 2160   | N/A       | 10-20 Mbps |
+-----------+---------------+-----------+------------+
```

Table 1

3.1.2. Virtual Reality Bitrates

The bitrates given in Section 3.1.1 describe video streams that
provide the user with a single, fixed, point of view - so, the user
has no "degrees of freedom", and the user sees all of the video image
that is available.

Even basic virtual reality (360-degree) videos that allow users to
look around freely (referred to as "three degrees of freedom", or
3DoF) require substantially larger bitrates when they are captured
and encoded as such videos require multiple fields of view of the
scene. Yet, due to smart delivery methods such as viewport-based or
tiled-based streaming, we do not need to send the whole scene to the
user. Instead, the user needs only the portion corresponding to its
viewpoint at any given time ([Survey360o]).

In more immersive applications, where limited user movement ("three
degrees of freedom plus", or 3DoF+) or full user movement ("six
degrees of freedom", or 6DoF) is allowed, the required bitrate grows
even further. In this case, immersive content is typically referred
to as volumetric media. One way to represent the volumetric media is
to use point clouds, where streaming a single object may easily
require a bitrate of 30 Mbps or higher. Refer to [MPEGI] and [PCC]
for more details.

3.2. Path Bandwidth Constraints

Even when the bandwidth requirements for video streams along a path
are well understood, additional analysis is required to understand
the contraints on bandwidth at various points in the network. This
analysis is necessary because media servers may react to bandwith
constraints using two independent feedback loops:

   * Media servers often respond to application-level feedback from the

media player that indicates a bottleneck link somewhere along the
path, by adjusting the amount of media that the media server will
send to the media player in a given timeframe. This is described
in greater detail in [Section 5](#).

* Media servers also typically implement transport protocols with
   capacity-seeking congestion controllers that probe for bandwidth,
   and adjust the sending rate based on transport mechanisms. This
   is described in greater detail in [Section 6](#).

The result is that these two (potentially competing) "helpful"
mechanisms each respond to the same bottleneck with no coordination
between themselves, so that each is unaware of actions taken by the
other, and this can result in a quality of experience for users that
is significantly lower than what could have been achieved.

In one example, if a media server overestimates the available
bandwidth to the media player,

* the transport protocol detects loss due to congestion, and reduces
   its sending window size per round trip,

* the media server adapts to application-level feedback from the
   media player, and reduces its own sending rate,

* the transport protocol sends media at the new, lower rate, and
   confirms that this new, lower rate is "safe", because no
   transport-level loss is occuring, but

* because the media server continues to send at the new, lower rate,
   the transport protocol's maximum sending rate is now limited by
   the amount of information the media server queues for
   transmission, so

* the transport protocol can't probe for available path bandwidth by
   sending at a higher rate.

In order to avoid these types of situations, which can potentially
affect all the users whose streaming media traverses a bottleneck
link, there are several possible mitigations that streaming operators
can use, but the first step toward mitigating a problem is knowing
when that problem occurs.

3.2.1. Know Your Network Traffic

There are many reasons why path characteristics might change
suddenly, for example,

   * "cross traffic" that traverses part of the path, especially if
      this traffic is "inelastic", and does not, itself, respond to
      indications of path congestion.

   * routing changes, which can happen in normal operation, especially
      if the new path now includes path segments that are more heavily
      loaded, offer lower total bandwidth, or simply cover more
      distance.

   Recognizing that a path carrying streaming media is "not behaving the
   way it normally does" is fundamental. Analytics that aid in that
   recognition can be more or less sophisticated, and can be as simple
   as noticing that the apparent round trip times for media traffic
   carried over TCP transport on some paths are suddenly and
   significantly longer than usual. Passive monitors can detect changes
   in the elapsed time between the acknowledgements for specific TCP
   segments from a TCP receiver, since TCP octet sequence numbers and
   acknowledgements for those sequence numbers are "carried in the
   clear", even if the TCP payload itself is encrypted. See Section 6.2
   for more information.

   As transport protocols evolve to encrypt their transport header
   fields, one side effect of increasing encryption is that the kind of
   passive monitoring, or even "performance enhancement" ([RFC3135])
   that was possible with the older transport protocols (UDP, described
   in Section 6.1 and TCP, described in Section 6.2) is no longer
   possible with newer transport protocols such as QUIC (described in
   Section 6.3). The IETF has specified a "latency spin bit" mechanism
   in Section 17.4 of [RFC9000] to allow passive latency monitoring from
   observation points on the network path throughout the duration of a
   connection, but currently chartered work in the IETF is focusing on
   end-point monitoring and reporting, rather than on passive
   monitoring.

   One example is the "qlog" mechanism [I-D.ietf-quic-qlog-main-schema],
   a protocol-agnostic mechanism used to provide better visibility for
   encrypted protocols such as QUIC ([I-D.ietf-quic-qlog-quic-events])
   and for HTTP/3 ([I-D.ietf-quic-qlog-h3-events]).

3.3. Path Requirements

   The bitrate requirements in Section 3.1 are per end-user actively
   consuming a media feed, so in the worst case, the bitrate demands can
   be multiplied by the number of simultaneous users to find the
   bandwidth requirements for a router on the delivery path with that
   number of users downstream. For example, at a node with 10,000

downstream users simultaneously consuming video streams,
approximately 80 Gbps might be necessary in order for all of them to

get typical content at 1080p resolution.

However, when there is some overlap in the feeds being consumed by
end users, it is sometimes possible to reduce the bandwidth
provisioning requirements for the network by performing some kind of
replication within the network. This can be achieved via object
caching with delivery of replicated objects over individual
connections, and/or by packet-level replication using multicast.

To the extent that replication of popular content can be performed,
bandwidth requirements at peering or ingest points can be reduced to
as low as a per-feed requirement instead of a per-user requirement.

3.4. Caching Systems

When demand for content is relatively predictable, and especially
when that content is relatively static, caching content close to
requesters, and pre-loading caches to respond quickly to initial
requests is often useful (for example, HTTP/1.1 caching is described
in [I-D.ietf-httpbis-cache]). This is subject to the usual
considerations for caching - for example, how much data must be
cached to make a significant difference to the requester, and how the
benefits of caching and pre-loading caches balances against the costs
of tracking "stale" content in caches and refreshing that content.

It is worth noting that not all high-demand content is "live"
content. One relevant example is when popular streaming content can
be staged close to a significant number of requesters, as can happen
when a new episode of a popular show is released. This content may
be largely stable, so low-cost to maintain in multiple places
throughout the Internet. This can reduce demands for high end-to-end
bandwidth without having to use mechanisms like multicast.

Caching and pre-loading can also reduce exposure to peering point
congestion, since less traffic crosses the peering point exchanges if
the caches are placed in peer networks, especially when the content
can be pre-loaded during off-peak hours, and especially if the
transfer can make use of "Lower-Effort Per-Hop Behavior (LE PHB) for
Differentiated Services" [RFC8622], "Low Extra Delay Background

Transport (LEDBAT)" [RFC6817], or similar mechanisms.

All of this depends, of course, on the ability of a content provider to predict usage and provision bandwidth, caching, and other mechanisms to meet the needs of users. In some cases (Section 3.5), this is relatively routine, but in other cases, it is more difficult (Section 3.6, Section 3.7).

And as with other parts of the ecosystem, new technology brings new challenges. For example, with the emergence of ultra-low-latency streaming, responses have to start streaming to the end user while still being transmitted to the cache, and while the cache does not yet know the size of the object. Some of the popular caching systems were designed around cache footprint and had deeply ingrained assumptions about knowing the size of objects that are being stored, so the change in design requirements in long-established systems caused some errors in production. Incidents occurred where a transmission error in the connection from the upstream source to the cache could result in the cache holding a truncated segment and transmitting it to the end user's device. In this case, players rendering the stream often had the video freeze until the player was reset. In some cases the truncated object was even cached that way and served later to other players as well, causing continued stalls at the same spot in the video for all players playing the segment delivered from that cache node.

3.5. Predictable Usage Profiles

Historical data shows that users consume more videos and at a higher bit rate than they did in the past on their connected devices. Improvements in the codecs that help with reducing the encoding bitrates with better compression algorithms could not have offset the increase in the demand for the higher quality video (higher resolution, higher frame rate, better color gamut, better dynamic range, etc.). In particular, mobile data usage has shown a large jump over the years due to increased consumption of entertainment as well as conversational video.

3.6. Unpredictable Usage Profiles

Although TCP/IP has been used with a number of widely used
applications that have symmetric bandwidth requirements (similar
bandwidth requirements in each direction between endpoints), many
widely-used Internet applications operate in client-server roles,
with asymmetric bandwidth requirements. A common example might be an
HTTP GET operation, where a client sends a relatively small HTTP GET
request for a resource to an HTTP server, and often receives a
significantly larger response carrying the requested resource. When
HTTP is commonly used to stream movie-length video, the ratio between
response size and request size can become arbitrarily large.

For this reason, operators may pay more attention to downstream
bandwidth utilization when planning and managing capacity. In
addition, operators have been able to deploy access networks for end
users using underlying technologies that are inherently asymmetric,
favoring downstream bandwidth (e.g. ADSL, cellular technologies,

most IEEE 802.11 variants), assuming that users will need less
upstream bandwidth than downstream bandwidth. This strategy usually
works, except when it fails because application bandwidth usage
patterns have changed in ways that were not predicted.

One example of this type of change was when peer-to-peer file sharing
applications gained popularity in the early 2000s. To take one well-
documented case ([RFC5594]), the Bittorrent application created
"swarms" of hosts, uploading and downloading files to each other,
rather than communicating with a server. Bittorrent favored peers
who uploaded as much as they downloaded, so that new Bittorrent users
had an incentive to significantly increase their upstream bandwidth
utilization.

The combination of the large volume of "torrents" and the peer-to-
peer characteristic of swarm transfers meant that end user hosts were
suddenly uploading higher volumes of traffic to more destinations
than was the case before Bittorrent. This caused at least one large
internet service provider (ISP) to attempt to "throttle" these
transfers in order to to mitigate the load that these hosts placed on
their network. These efforts were met by increased use of encryption
in Bittorrent, similar to an arms race, and set off discussions about
"Net Neutrality" and calls for regulatory action.

Especially as end users increase use of video-based social networking

applications, it will be helpful for access network providers to
watch for increasing numbers of end users uploading significant
amounts of content.

3.7. Extremely Unpredictable Usage Profiles

The causes of unpredictable usage described in Section 3.6 were more
or less the result of human choices, but we were reminded during a
post-IETF 107 meeting that humans are not always in control, and
forces of nature can cause enormous fluctuations in traffic patterns.

In his talk, Sanjay Mishra [Mishra] reported that after the CoViD-19
pandemic broke out in early 2020,

*   Comcast's streaming and web video consumption rose by 38%, with
    their reported peak traffic up 32% overall between March 1 to
    March 30,

*   AT&T reported a 28% jump in core network traffic (single day in
    April, as compared to pre stay-at-home daily average traffic),
    with video accounting for nearly half of all mobile network
    traffic, while social networking and web browsing remained the
    highest percentage (almost a quarter each) of overall mobility
    traffic, and

*   Verizon reported similar trends with video traffic up 36% over an
    average day (pre COVID-19)}.

We note that other operators saw similar spikes during this time
period. Craig Labowitz [Labovitz] reported

*   Weekday peak traffic increases over 45%-50% from pre-lockdown
    levels,

*   A 30% increase in upstream traffic over their pre-pandemic levels,
    and

* A steady increase in the overall volume of DDoS traffic, with
  amounts exceeding the pre-pandemic levels by 40%. (He attributed
  this increase to the significant rise in gaming-related DDoS
  attacks ([LabovitzDDoS]), as gaming usage also increased.)

Subsequently, the Internet Architecture Board (IAB) held a COVID-19
Network Impacts Workshop [IABcovid] in November 2020. Given a larger
number of reports and more time to reflect, the following
observations from the draft workshop report are worth considering.

* Participants describing different types of networks reported
  different kinds of impacts, but all types of networks saw impacts.

* Mobile networks saw traffic reductions and residential networks
  saw significant increases.

* Reported traffic increases from ISPs and internet exchange points
  (IXP) over just a few weeks were as big as the traffic growth over
  the course of a typical year, representing a 15-20% surge in
  growth to land at a new normal that was much higher than
  anticipated.

* At DE-CIX Frankfurt, the world's largest Internet Exchange Point
  in terms of data throughput, the year 2020 has seen the largest
  increase in peak traffic within a single year since the IXP was
  founded in 1995.

* The usage pattern changed significantly as work-from-home and
  videoconferencing usage peaked during normal work hours, which
  would have typically been off-peak hours with adults at work and
  children at school. One might expect that the peak would have had
  more impact on networks if it had happened during typical evening
  peak hours for video streaming applications.

* The increase in daytime bandwidth consumption reflected both
  significant increases in "essential" applications such as
  videoconferencing and virtual private networks (VPN), and
  entertainment applications as people watched videos or played

games.

* At the IXP-level, it was observed that port utilization increased.
  This phenomenon is mostly explained by a higher traffic demand
  from residential users.

4. Latency Considerations

Streaming media latency refers to the "glass-to-glass" time duration,
which is the delay between the real-life occurrence of an event and
the streamed media being appropriately displayed on an end user's
device. Note that this is different from the network latency
(defined as the time for a packet to cross a network from one end to
another end) because it includes video encoding/decoding and
buffering time, and for most cases also ingest to an intermediate
service such as a CDN or other video distribution service, rather
than a direct connection to an end user.

Streaming media can be usefully categorized according to the
application's latency requirements into a few rough categories:

* ultra low-latency (less than 1 second)

* low-latency live (less than 10 seconds)

* non-low-latency live (10 seconds to a few minutes)

* on-demand (hours or more)

4.1. Ultra Low-Latency

Ultra low-latency delivery of media is defined here as having a
glass-to-glass delay target under one second.

Some media content providers aim to achieve this level of latency for
live media events. This introduces new challenges relative to less-
restricted levels of latency requirements because this latency is the

---

same scale as commonly observed end-to-end network latency variation
(for example, due to effects such as bufferbloat ([CoDel]), Wi-Fi
error correction, or packet reordering). These effects can make it
difficult to achieve this level of latency for the general case, and

may require tradeoffs in relatively frequent user-visible media
artifacts. However, for controlled environments or targeted networks
that provide mitigations against such effects, this level of latency
is potentially achievable with the right provisioning.

Applications requiring ultra low latency for media delivery are
usually tightly constrained on the available choices for media
transport technologies, and sometimes may need to operate in
controlled environments to reliably achieve their latency and quality
goals.

Most applications operating over IP networks and requiring latency
this low use the Real-time Transport Protocol (RTP) [RFC3550] or
WebRTC [RFC8825], which uses RTP for the media transport as well as
several other protocols necessary for safe operation in browsers.

Worth noting is that many applications for ultra low-latency delivery
do not need to scale to more than a few users at a time, which
simplifies many delivery considerations relative to other use cases.

Recommended reading for applications adopting an RTP-based approach
also includes [RFC7656]. For increasing the robustness of the
playback by implementing adaptive playout methods, refer to [RFC4733]
and [RFC6843].

Applications with further-specialized latency requirements are out of
scope for this document.

4.2. Low-Latency Live

Low-latency live delivery of media is defined here as having a glass-
to-glass delay target under 10 seconds.

This level of latency is targeted to have a user experience similar
to traditional broadcast TV delivery. A frequently cited problem
with failing to achieve this level of latency for live sporting
events is the user experience failure from having crowds within
earshot of one another who react audibly to an important play, or
from users who learn of an event in the match via some other channel,
for example social media, before it has happened on the screen
showing the sporting event.

Applications requiring low-latency live media delivery are generally
feasible at scale with some restrictions. This typically requires
the use of a premium service dedicated to the delivery of live video,
and some tradeoffs may be necessary relative to what is feasible in a
higher latency service. The tradeoffs may include higher costs, or
delivering a lower quality video, or reduced flexibility for adaptive
bitrates, or reduced flexibility for available resolutions so that
fewer devices can receive an encoding tuned for their display. Low-
latency live delivery is also more susceptible to user-visible
disruptions due to transient network conditions than higher latency
services.

Implementation of a low-latency live video service can be achieved
with the use of low-latency extensions of HLS (called LL-HLS)
[I-D.draft-pantos-hls-rfc8216bis] and DASH (called LL-DASH)
[LL-DASH]. These extensions use the Common Media Application Format
(CMAF) standard [MPEG-CMAF] that allows the media to be packaged into
and transmitted in units smaller than segments, which are called
chunks in CMAF language. This way, the latency can be decoupled from
the duration of the media segments. Without a CMAF-like packaging,
lower latencies can only be achieved by using very short segment
durations. However, shorter segments means more frequent intra-coded
frames and that is detrimental to video encoding quality. CMAF
allows us to still use longer segments (improving encoding quality)
without penalizing latency.

While a LL-HLS client retrieves each chunk with a separate HTTP GET
request, a LL-DASH client uses the chunked transfer encoding feature
of the HTTP [CMAF-CTE] which allows the LL-DASH client to fetch all
the chunks belonging to a segment with a single GET request. An HTTP
server can transmit the CMAF chunks to the LL-DASH client as they
arrive from the encoder/packager. A detailed comparison of LL-HLS
and LL-DASH is given in [MMSP20].

4.3. Non-Low-Latency Live

Non-low-latency live delivery of media is defined here as a live
stream that does not have a latency target shorter than 10 seconds.

This level of latency is the historically common case for segmented
video delivery using HLS [RFC8216] and DASH [MPEG-DASH]. This level
of latency is often considered adequate for content like news or pre-
recorded content. This level of latency is also sometimes achieved
as a fallback state when some part of the delivery system or the
client-side players do not have the necessary support for the
features necessary to support low-latency live streaming.

This level of latency can typically be achieved at scale with
commodity CDN services for HTTP(s) delivery, and in some cases the
increased time window can allow for production of a wider range of
encoding options relative to the requirements for a lower latency
service without the need for increasing the hardware footprint, which
can allow for wider device interoperability.

4.4. On-Demand

On-Demand media streaming refers to playback of pre-recorded media
based on a user's action. In some cases on-demand media is produced
as a by-product of a live media production, using the same segments
as the live event, but freezing the manifest after the live event has
finished. In other cases, on-demand media is constructed out of pre-
recorded assets with no streaming necessarily involved during the
production of the on-demand content.

On-demand media generally is not subject to latency concerns, but
other timing-related considerations can still be as important or even
more important to the user experience than the same considerations
with live events. These considerations include the startup time, the
stability of the media stream's playback quality, and avoidance of
stalls and video artifacts during the playback under all but the most
severe network conditions.

In some applications, optimizations are available to on-demand video
that are not always available to live events, such as pre-loading the
first segment for a startup time that doesn't have to wait for a
network download to begin.

5. Adaptive Encoding, Adaptive Delivery, and Measurement Collection

5.1. Overview

A simple model of video playback can be described as a video stream
consumer, a buffer, and a transport mechanism that fills the buffer.
The consumption rate is fairly static and is represented by the
content bitrate. The size of the buffer is also commonly a fixed
size. The fill process needs to be at least fast enough to ensure
that the buffer is never empty, however it also can have significant

complexity when things like personalization or ad workflows are
introduced.

The challenges in filling the buffer in a timely way fall into two
broad categories: 1. content selection and 2. content variation.
Content selection comprises all of the steps needed to determine
which content variation to offer the client. Content variation is
the number of content options that exist at any given selection

point. A common example, easily visualized, is Adaptive BitRate
(ABR), described in more detail below. The mechanism used to select
the bitrate is part of the content selection, and the content
variation are all of the different bitrate renditions.

ABR is a sort of application-level response strategy in which the
streaming client attempts to detect the available bandwidth of the
network path by observing the successful application-layer download
speed, then chooses a bitrate for each of the video, audio, subtitles
and metadata (among the limited number of available options) that
fits within that bandwidth, typically adjusting as changes in
available bandwidth occur in the network or changes in capabilities
occur during the playback (such as available memory, CPU, display
size, etc.).

5.2. Adaptive Encoding

Media servers can provide media streams at various bitrates because
the media has been encoded at various bitrates. This is a so-called
"ladder" of bitrates, that can be offered to media players as part of
the manifest that describes the media being requested by the media
player, so that the media player can select among the available
bitrate choices.

The media server may also choose to alter which bitrates are made
available to players by adding or removing bitrate options from the
ladder delivered to the player in subsequent manifests built and sent
to the player. This way, both the player, through its selection of
bitrate to request from the manifest, and the server, through its
construction of the bitrates offered in the manifest, are able to
affect network utilization.

5.3. Adaptive Segmented Delivery

ABR playback is commonly implemented by streaming clients using HLS
[RFC8216] or DASH [MPEG-DASH] to perform a reliable segmented
delivery of media over HTTP. Different implementations use different
strategies [ABRSurvey], often relying on proprietary algorithms
(called rate adaptation or bitrate selection algorithms) to perform
available bandwidth estimation/prediction and the bitrate selection.

Many server-player systems will do an initial probe or a very simple
throughput speed test at the start of a video playback. This is done
to get a rough sense of the highest video bitrate in the ABR ladder
that the network between the server and player will likely be able to
provide under initial network conditions. After the initial testing,
clients tend to rely upon passive network observations and will make
use of player side statistics such as buffer fill rates to monitor
and respond to changing network conditions.

The choice of bitrate occurs within the context of optimizing for
some metric monitored by the client, such as highest achievable video
quality or lowest chances for a rebuffering event (playback stall).

5.4. Advertising

A variety of business models exist for producers of streaming media.
Some content providers derive the majority of the revenue associated
with streaming media directly from consumer subscriptions or one-time
purchases. Others derive the majority of their streaming media
associated revenue from advertising. Many content providers derive
income from a mix of these and other sources of funding. The
inclusion of advertising alongside or interspersed with streaming
media content is therefore common in today's media landscape.

Some commonly used forms of advertising can introduce potential user
experience issues for a media stream. This section provides a very

brief overview of a complex and evolving space, but a complete
coverage of the potential issues is out of scope for this document.

The same techniques used to allow a media player to switch between
renditions of different bitrates at segment or chunk boundaries can
also be used to enable the dynamic insertion of advertisements
(herafter referred to as "ads").

Ads may be inserted either with Client Side Ad Insertion (CSAI) or
Server Side Ad Insertion (SSAI). In CSAI, the ABR manifest will
generally include links to an external ad server for some segments of
the media stream, while in SSAI the server will remain the same
during advertisements, but will include media segments that contain
the advertising. In SSAI, the media segments may or may not be
sourced from an external ad server like with CSAI.

In general, the more targeted the ad request is, the more requests
the ad service needs to be able to handle concurrently. If
connectivity is poor to the ad service, this can cause rebuffering
even if the underlying video assets (both content and ads) are able
to be accessed quickly. The less targeted, the more likely the ad
requests can be consolidated and can leverage the same caching
techniques as the video content.

In some cases, especially with SSAI, advertising space in a stream is
reserved for a specific advertiser and can be integrated with the
video so that the segments share the same encoding properties such as
bitrate, dynamic range, and resolution. However, in many cases ad
servers integrate with a Supply Side Platform (SSP) that offers
advertising space in real-time auctions via an Ad Exchange, with bids
for the advertising space coming from Demand Side Platforms (DSPs)
that collect money from advertisers for delivering the
advertisements. Most such Ad Exchanges use application-level
protocol specifications published by the Interactive Advertising
Bureau [IAB-ADS], an industry trade organization.

This ecosystem balances several competing objectives, and integrating
with it naively can produce surprising user experience results. For
example, ad server provisioning and/or the bitrate of the ad segments
might be different from that of the main video, either of which can
sometimes result in video stalls. For another example, since the
inserted ads are often produced independently they might have a
different base volume level than the main video, which can make for a
jarring user experience.

Additionally, this market historically has had incidents of ad fraud
(misreporting of ad delivery to end users for financial gain). As a
mitigation for concerns driven by those incidents, some SSPs have
required the use of players with features like reporting of ad
delivery, or providing information that can be used for user
tracking. Some of these and other measures have raised privacy
concerns for end users.

In general this is a rapidly developing space with many
considerations, and media streaming operators engaged in advertising
may need to research these and other concerns to find solutions that
meet their user experience, user privacy, and financial goals. For
further reading on mitigations, [BAP] has published some standards
and best practices based on user experience research.

5.5. Bitrate Detection Challenges

   This kind of bandwidth-measurement system can experience trouble in
   several ways that are affected by networking issues. Because
   adaptive application-level response strategies are often using rates
   as observed by the application layer, there are sometimes inscrutable
   transport-level protocol behaviors that can produce surprising
   measurement values when the application-level feedback loop is
   interacting with a transport-level feedback loop.

   A few specific examples of surprising phenomena that affect bitrate
   detection measurements are described in the following subsections.

As these examples will demonstrate, it is common to encounter cases that can deliver application level measurements that are too low, too high, and (possibly) correct but varying more quickly than a lab-tested selection algorithm might expect.

These effects and others that cause transport behavior to diverge from lab modeling can sometimes have a significant impact on bitrate selection and on user quality of experience, especially where players use naive measurement strategies and selection algorithms that don't account for the likelihood of bandwidth measurements that diverge from the true path capacity.

5.5.1. Idle Time between Segments

When the bitrate selection is chosen substantially below the available capacity of the network path, the response to a segment request will typically complete in much less absolute time than the duration of the requested segment, leaving significant idle time between segment downloads. This can have a few surprising consequences:

*   TCP slow-start when restarting after idle requires multiple RTTs to re-establish a throughput at the network's available capacity. When the active transmission time for segments is substantially shorter than the time between segments, leaving an idle gap between segments that triggers a restart of TCP slow-start, the estimate of the successful download speed coming from the application-visible receive rate on the socket can thus end up much lower than the actual available network capacity. This in turn can prevent a shift to the most appropriate bitrate. [RFC7661] provides some mitigations for this effect at the TCP transport layer, for senders who anticipate a high incidence of this problem.

*   Mobile flow-bandwidth spectrum and timing mapping can be impacted by idle time in some networks. The carrier capacity assigned to a link can vary with activity. Depending on the idle time characteristics, this can result in a lower available bitrate than would be achievable with a steadier transmission in the same

network.

Some receiver-side ABR algorithms such as [ELASTIC] are designed to
try to avoid this effect.

Another way to mitigate this effect is by the help of two
simultaneous TCP connections, as explained in [MMSys11] for Microsoft
Smooth Streaming. In some cases, the system-level TCP slow-start
restart can also be disabled, for example as described in
[OReilly-HPBN].

5.5.2. Head-of-Line Blocking

In the event of a lost packet on a TCP connection with SACK support
(a common case for segmented delivery in practice), loss of a packet
can provide a confusing bandwidth signal to the receiving
application. Because of the sliding window in TCP, many packets may
be accepted by the receiver without being available to the
application until the missing packet arrives. Upon arrival of the
one missing packet after retransmit, the receiver will suddenly get
access to a lot of data at the same time.

To a receiver measuring bytes received per unit time at the
application layer, and interpreting it as an estimate of the
available network bandwidth, this appears as a high jitter in the
goodput measurement, presenting as a stall, followed by a sudden leap
that can far exceed the actual capacity of the transport path from
the server when the hole in the received data is filled by a later
retransmission.

It is worth noting that more modern transport protocols such as QUIC
have mitigation of head-of-line blocking as a protocol design goal.
See Section 6.3 for more details.

5.5.3. Wide and Rapid Variation in Path Capacity

As many end devices have moved to wireless connectivity for the final
hop (Wi-Fi, 5G, or LTE), new problems in bandwidth detction have
emerged from radio interference and signal strength effects.

Each of these technologies can experience sudden changes in capacity
as the end user device moves from place to place and encounters new
sources of interference. Microwave ovens, for example, can cause a

throughput degradation of more than a factor of 2 while active
[Micro]. 5G and LTE likewise can easily see rate variation by a
factor of 2 or more over a span of seconds as users move around.

These swings in actual transport capacity can result in user
experience issues that can be exacerbated by insufficiently
responsive ABR algorithms.

5.6. Measurement Collection

In addition to measurements media players use to guide their segment-
by-segment adaptive streaming requests, streaming media providers may
also rely on measurements collected from media players to provide
analytics that can be used for decisions such as whether the adaptive
encoding bitrates in use are the best ones to provide to media
players, or whether current media content caching is providing the
best experience for viewers. To that effect, the Consumer Technology
Association (CTA) who owns the Web Application Video Ecosystem (WAVE)
project has published two important specifications.

5.6.1. CTA-2066: Streaming Quality of Experience Events, Properties and Metrics

[CTA-2066] specifies a set of media player events, properties,
quality of experience (QoE) metrics and associated terminology for
representing streaming media quality of experience across systems,
media players and analytics vendors. While all these events,
properties, metrics and associated terminology is used across a
number of proprietary analytics and measurement solutions, they were
used in slightly (or vastly) different ways that led to
interoperability issues. CTA-2066 attempts to address this issue by
defining a common terminology as well as how each metric should be
computed for consistent reporting.

5.6.2. CTA-5004: Common Media Client Data (CMCD)

Many assume that the CDNs have a holistic view into the health and
performance of the streaming clients. However, this is not the case.
The CDNs produce millions of log lines per second across hundreds of
thousands of clients and they have no concept of a "session" as a
client would have, so CDNs are decoupled from the metrics the clients
generate and report. A CDN cannot tell which request belongs to
which playback session, the duration of any media object, the
bitrate, or whether any of the clients have stalled and are
rebuffering or are about to stall and will rebuffer. The consequence
of this decoupling is that a CDN cannot prioritize delivery for when
the client needs it most, prefetch content, or trigger alerts when
the network itself may be underperforming. One approach to couple

the CDN to the playback sessions is for the clients to communicate
standardized media-relevant information to the CDNs while they are
fetching data. [CTA-5004] was developed exactly for this purpose.

5.7. Unreliable Transport

In contrast to segmented delivery, several applications use
unreliable UDP or SCTP with its "partial reliability" extension
[RFC3758] to deliver Media encapsulated in RTP [RFC3550] or raw MPEG
Transport Stream ("MPEG-TS")-formatted video [MPEG-TS], when the
media is being delivered in situations such as broadcast and live
streaming, that better tolerate occasional packet loss without
retransmission.

Under congestion and loss, this approach generally experiences more
video artifacts with fewer delay or head-of-line blocking effects.
Often one of the key goals is to reduce latency, to better support
applications like videoconferencing, or for other live-action video
with interactive components, such as some sporting events.

The Secure Reliable Transport protocol [SRT] also uses UDP in an
effort to achieve lower latency for streaming media, although it adds
reliability at the application layer.

Congestion avoidance strategies for deployments using unreliable
transport protocols vary widely in practice, ranging from being
entirely unresponsive to congestion, to using feedback signaling to
change encoder settings (as in [RFC5762]), to using fewer enhancement
layers (as in [RFC6190]), to using proprietary methods to detect
"quality of experience" issues and turn off video in order to allow
less bandwidth-intensive media such as audio to be delivered.

More details about congestion avoidance strategies used with
unreliable transport protocols are included in Section 6.1.

6. Evolution of Transport Protocols and Transport Protocol Behaviors

Because networking resources are shared between users, a good place
to start our discussion is how contention between users, and
mechanisms to resolve that contention in ways that are "fair" between
users, impact streaming media users. These topics are closely tied
to transport protocol behaviors.

As noted in Section 5, ABR response strategies such as HLS [RFC8216] or DASH [MPEG-DASH] are attempting to respond to changing path characteristics, and underlying transport protocols are also attempting to respond to changing path characteristics.

For most of the history of the Internet, these transport protocols, described in Section 6.1 and Section 6.2, have had relatively consistent behaviors that have changed slowly, if at all, over time. Newly standardized transport protocols like QUIC [RFC9000] can behave differently from existing transport protocols, and these behaviors may evolve over time more rapidly than currently-used transport protocols.

For this reason, we have included a description of how the path characteristics that streaming media providers may see are likely to evolve over time.

6.1. UDP and Its Behavior

For most of the history of the Internet, we have trusted UDP-based applications to limit their impact on other users. One of the strategies used was to use UDP for simple query-response application protocols, such as DNS, which is often used to send a single-packet request to look up the IP address for a DNS name, and return a single-packet response containing the IP address. Although it is possible to saturate a path between a DNS client and DNS server with DNS requests, in practice, that was rare enough that DNS included few mechanisms to resolve contention between DNS users and other users (whether they are also using DNS, or using other application protocols).

In recent times, the usage of UDP-based applications that were not simple query-response protocols has grown substantially, and since UDP does not provide any feedback mechanism to senders to help limit impacts on other users, application-level protocols such as RTP [RFC3550] have been responsible for the decisions that TCP-based applications have delegated to TCP - what to send, how much to send, and when to send it. So, the way some UDP-based applications interact with other users has changed.

It is also worth pointing out that because UDP has no transport-layer
feedback mechanisms, UDP-based applications that send and receive
substantial amounts of information are expected to provide their own
feedback mechanisms. This expectation is most recently codified in
Best Current Practice [RFC8085].

RTP relies on RTCP Sender and Receiver Reports [RFC3550] as its own
feedback mechanism, and even includes Circuit Breakers for Unicast
RTP Sessions [RFC8083] for situations when normal RTP congestion
control has not been able to react sufficiently to RTP flows sending
at rates that result in sustained packet loss.

The notion of "Circuit Breakers" has also been applied to other UDP
applications in [RFC8084], such as tunneling packets over UDP that
are potentially not congestion-controlled (for example,
"Encapsulating MPLS in UDP", as described in [RFC7510]). If
streaming media is carried in tunnels encapsulated in UDP, these
media streams may encounter "tripped circuit breakers", with
resulting user-visible impacts.

6.2. TCP and Its Behavior

For most of the history of the Internet, we have trusted TCP to limit
the impact of applications that sent a significant number of packets,
in either or both directions, on other users. Although early
versions of TCP were not particularly good at limiting this impact
[RFC0793], the addition of Slow Start and Congestion Avoidance, as
described in [RFC2001], were critical in allowing TCP-based
applications to "use as much bandwidth as possible, but to avoid
using more bandwidth than was possible". Although dozens of RFCs
have been written refining TCP decisions about what to send, how much
to send, and when to send it, since 1988 [Jacobson-Karels] the
signals available for TCP senders remained unchanged - end-to-end
acknowledgements for packets that were successfully sent and
received, and packet timeouts for packets that were not.

The success of the largely TCP-based Internet is evidence that the
mechanisms TCP used to achieve equilibrium quickly, at a point where
TCP senders do not interfere with other TCP senders for sustained
periods of time, have been largely successful. The Internet

continued to work even when the specific mechanisms used to reach
equilibrium changed over time. Because TCP provides a common tool to
avoid contention, as some TCP-based applications like FTP were
largely replaced by other TCP-based applications like HTTP, the
transport behavior remained consistent.

In recent times, the TCP goal of probing for available bandwidth, and
"backing off" when a network path is saturated, has been supplanted
by the goal of avoiding growing queues along network paths, which
prevent TCP senders from reacting quickly when a network path is
saturated. Congestion control mechanisms such as COPA [COPA18] and
BBR [I-D.cardwell-iccrg-bbr-congestion-control] make these decisions
based on measured path delays, assuming that if the measured path
delay is increasing, the sender is injecting packets onto the network
path faster than the receiver can accept them, so the sender should
adjust its sending rate accordingly.

Although TCP behavior has changed over time, the common practice of
implementing TCP as part of an operating system kernel has acted to
limit how quickly TCP behavior can change. Even with the widespread

use of automated operating system update installation on many end-
user systems, streaming media providers could have a reasonable
expectation that they could understand TCP transport protocol
behaviors, and that those behaviors would remain relatively stable in
the short term.

6.3. The QUIC Protocol and Its Behavior

The QUIC protocol, developed from a proprietary protocol into an IETF
standards-track protocol [RFC9000], turns many of the statements made
in Section 6.1 and Section 6.2 on their heads.

Although QUIC provides an alternative to the TCP and UDP transport
protocols, QUIC is itself encapsulated in UDP. As noted elsewhere in
Section 7.1, the QUIC protocol encrypts almost all of its transport
parameters, and all of its payload, so any intermediaries that
network operators may be using to troubleshoot HTTP streaming media
performance issues, perform analytics, or even intercept exchanges in
current applications will not work for QUIC-based applications
without making changes to their networks. Section 7 describes the
implications of media encryption in more detail.

While QUIC is designed as a general-purpose transport protocol, and
can carry different application-layer protocols, the current
standardized mapping is for HTTP/3 [I-D.ietf-quic-http], which
describes how QUIC transport features are used for HTTP. The
convention is for HTTP/3 to run over UDP port 443 [Port443] but this
is not a strict requirement.

When HTTP/3 is encapsulated in QUIC, which is then encapsulated in
UDP, streaming operators (and network operators) might see UDP
traffic patterns that are similar to HTTP(S) over TCP. Since earlier
versions of HTTP(S) rely on TCP, UDP ports may be blocked for any
port numbers that are not commonly used, such as UDP 53 for DNS.
Even when UDP ports are not blocked and HTTP/3 can flow, streaming
operators (and network operators) may severely rate-limit this
traffic because they do not expect to see legitimate high-bandwidth
traffic such as streaming media over the UDP ports that HTTP/3 is
using.

As noted in Section 5.5.2, because TCP provides a reliable, in-order
delivery service for applications, any packet loss for a TCP
connection causes "head-of-line blocking", so that no TCP segments
arriving after a packet is lost will be delivered to the receiving
application until the lost packet is retransmitted, allowing in-order
delivery to the application to continue. As described in [RFC9000],
QUIC connections can carry multiple streams, and when packet losses
do occur, only the streams carried in the lost packet are delayed.

A QUIC extension currently being specified ([I-D.ietf-quic-datagram])
adds the capability for "unreliable" delivery, similar to the service
provided by UDP, but these datagrams are still subject to the QUIC
connection's congestion controller, providing some transport-level
congestion avoidance measures, which UDP does not.

As noted in Section 6.2, there is an increasing interest in transport
protocol behaviors that respond to delay measurements, instead of
responding to packet loss. These behaviors may deliver improved user
experience, but in some cases have not responded to sustained packet
loss, which exhausts available buffers along the end-to-end path that
may affect other users sharing that path. The QUIC protocol provides
a set of congestion control hooks that can be used for algorithm
agility, and [RFC9002] defines a basic algorithm with transport

behavior that is roughly similar to TCP NewReno [RFC6582]. However,
QUIC senders can and do unilaterally choose to use different
algorithms such as loss-based CUBIC [RFC8312], delay-based COPA or
BBR, or even something completely different.

We do have experience with deploying new congestion controllers
without melting the Internet (CUBIC is one example), but the point
mentioned in Section 6.2 about TCP being implemented in operating
system kernels is also different with QUIC. Although QUIC can be
implemented in operating system kernels, one of the design goals when
this work was chartered was "QUIC is expected to support rapid,
distributed development and testing of features", and to meet this
expectation, many implementers have chosen to implement QUIC in user
space, outside the operating system kernel, and to even distribute
QUIC libraries with their own applications.

The decision to deploy a new version of QUIC is relatively
uncontrolled, compared to other widely used transport protocols, and
this can include new transport behaviors that appear without much
notice except to the QUIC endpoints. At IETF 105, Christian Huitema
and Brian Trammell presented a talk on "Congestion Defense in Depth"
[CDiD], that explored potential concerns about new QUIC congestion
controllers being broadly deployed without the testing and
instrumentation that current major content providers routinely
include. The sense of the room at IETF 105 was that the current
major content providers understood what is at stake when they deploy
new congestion controllers, but this presentation, and the related
discussion in TSVAREA minutes from IETF 105 ([tsvarea-105], are still
worth a look for new and rapidly growing content providers.

It is worth considering that if TCP-based HTTP traffic and UDP-based
HTTP/3 traffic are allowed to enter operator networks on roughly
equal terms, questions of fairness and contention will be heavily
dependent on interactions between the congestion controllers in use
for TCP-based HTTP traffic and UDP-based HTTP/3 traffic.

More broadly, [I-D.ietf-quic-manageability] discusses manageability

of the QUIC transport protocol, focusing on the implications of
QUIC's design and wire image on network operations involving QUIC
traffic. It discusses what network operators can consider in some
detail.

7. Streaming Encrypted Media

   "Encrypted Media" has at least three meanings:

   * Media encrypted at the application layer, typically using some
      sort of Digital Rights Management (DRM) system, and typically
      remaining encrypted "at rest", when senders and receivers store
      it.

   * Media encrypted by the sender at the transport layer, and
      remaining encrypted until it reaches the ultimate media consumer
      (in this document, referred to as "end-to-end media encryption").

   * Media encrypted by the sender at the transport layer, and
      remaining encrypted until it reaches some intermediary that is
      not the ultimate media consumer, but has credentials allowing
      decryption of the media content. This intermediary may examine
      and even transform the media content in some way, before
      forwarding re-encrypted media content (in this document referred
      to as "hop-by-hop media encryption").

   Both "hop-by-hop" and "end-to-end" encrypted transport may carry
   media that is, in addition, encrypted at the application layer.

   Each of these encryption strategies is intended to achieve a
   different goal. For instance, application-level encryption may be
   used for business purposes, such as avoiding piracy or enforcing
   geographic restrictions on playback, while transport-layer encryption
   may be used to prevent media steam manipulation or to protect
   manifests.

   This document does not take a position on whether those goals are
   "valid" (whatever that might mean).

In this document, we will focus on media encrypted at the transport layer, whether encrypted "hop-by-hop" or "end-to-end". Because media encrypted at the application layer will only be processed by application-level entities, this encryption does not have transport-layer implications.

Both "End-to-End" and "Hop-by-Hop" media encryption have specific implications for streaming operators. These are described in Section 7.2 and Section 7.3.

7.1. General Considerations for Media Encryption

The use of strong encryption does provide confidentiality for encrypted streaming media, from the sender to either an intermediary or the ultimate media consumer, and this does prevent Deep Packet Inspection by any intermediary that does not possess credentials allowing decryption. However, even encrypted content streams may be vulnerable to traffic analysis. An intermediary that can identify an encrypted media stream without decrypting it, may be able to "fingerprint" the encrypted media stream of known content, and then match the targeted media stream against the fingerprints of known content. This protection can be lessened if a media provider is repeatedly encrypting the same content. [CODASPY17] is an example of what is possible when identifying HTTPS-protected videos over TCP transport, based either on the length of entire resources being transferred, or on characteristic packet patterns at the beginning of a resource being transferred.

If traffic analysis is successful at identifying encrypted content and associating it with specific users, this breaks privacy as certainly as examining decrypted traffic.

Because HTTPS has historically layered HTTP on top of TLS, which is in turn layered on top of TCP, intermediaries do have access to unencrypted TCP-level transport information, such as retransmissions, and some carriers exploited this information in attempts to improve transport-layer performance [RFC3135]. The most recent standardized version of HTTPS, HTTP/3 [I-D.ietf-quic-http], uses the QUIC protocol [RFC9000] as its transport layer. QUIC relies on the TLS 1.3 initial handshake [RFC8446] only for key exchange [RFC9001], and encrypts almost all transport parameters itself, with the exception of a few invariant header fields. In the QUIC short header, the only transport-level parameter which is sent "in the clear" is the Destination Connection ID [RFC8999], and even in the QUIC long header, the only transport-level parameters sent "in the clear" are the Version, Destination Connection ID, and Source Connection ID. For these reasons, HTTP/3 is significantly more "opaque" than HTTPS with HTTP/1 or HTTP/2.

7.2. Considerations for "Hop-by-Hop" Media Encryption

   Although the IETF has put considerable emphasis on end-to-end
   streaming media encryption, there are still important use cases that
   require the insertion of intermediaries.

   There are a variety of ways to involve intermediaries, and some are
   much more intrusive than others.

   From a content provider's perspective, a number of considerations are
   in play. The first question is likely whether the content provider
   intends that intermediaries are explicitly addressed from endpoints,
   or whether the content provider is willing to allow intermediaries to
   "intercept" streaming content transparently, with no awareness or
   permission from either endpoint.

   If a content provider does not actively work to avoid interception by
   intermediaries, the effect will be indistinguishable from
   "impersonation attacks", and endpoints cannot be assumed of any level
   of privacy.

   Assuming that a content provider does intend to allow intermediaries
   to participate in content streaming, and does intend to provide some
   level of privacy for endpoints, there are a number of possible tools,
   either already available or still being specified. These include

   * Server And Network assisted DASH [MPEG-DASH-SAND] - this
      specification introduces explicit messaging between DASH clients
      and network elements or between various network elements for the
      purpose of improving the efficiency of streaming sessions by
      providing information about real-time operational characteristics
      of networks, servers, proxies, caches, CDNs, as well as DASH
      client's performance and status.

   * "Double Encryption Procedures for the Secure Real-Time Transport
      Protocol (SRTP)" [RFC8723] - this specification provides a
      cryptographic transform for the Secure Real-time Transport
      Protocol that provides both hop-by-hop and end-to-end security
      guarantees.

   * Secure Media Frames [SFRAME] - [RFC8723] is closely tied to SRTP,
      and this close association impeded widespread deployment, because
      it could not be used for the most common media content delivery

mechanisms. A more recent proposal, Secure Media Frames [SFRAME],
also provides both hop-by-hop and end-to-end security guarantees,
but can be used with other transport protocols beyond SRTP.

If a content provider chooses not to involve intermediaries, this
choice should be carefully considered. As an example, if media
manifests are encrypted end-to-end, network providers who had been
able to lower offered quality and reduce on their networks will no
longer be able to do that. Some resources that might inform this
consideration are in [RFC8825] (for WebRTC) and
[I-D.ietf-quic-manageability] (for HTTP/3 and QUIC).

7.3. Considerations for "End-to-End" Media Encryption

"End-to-end" media encryption offers the potential of providing
privacy for streaming media consumers, with the idea being that if an
unauthorized intermediary can't decrypt streaming media, the
intermediary can't use Deep Packet Inspection (DPI) to examine HTTP
request and response headers and identify the media content being
streamed.

"End-to-end" media encryption has become much more widespread in the
years since the IETF issued "Pervasive Monitoring Is an Attack"
[RFC7258] as a Best Current Practice, describing pervasive monitoring
as a much greater threat than previously appreciated. After the
Snowden disclosures, many content providers made the decision to use
HTTPS protection - HTTP over TLS - for most or all content being
delivered as a routine practice, rather than in exceptional cases for
content that was considered "sensitive".

Unfortunately, as noted in [RFC7258], there is no way to prevent
pervasive monitoring by an "attacker", while allowing monitoring by a
more benign entity who "only" wants to use DPI to examine HTTP
requests and responses in order to provide a better user experience.
If a modern encrypted transport protocol is used for end-to-end media
encryption, intermediary streaming operators are unable to examine
transport and application protocol behavior. As described in
Section 7.2, only an intermediary streaming operator who is
explicitly authorized to examine packet payloads, rather than
intercepting packets and examining them without authorization, can

continue these practices.

[RFC7258] said that "The IETF will strive to produce specifications that mitigate pervasive monitoring attacks", so streaming operators should expect the IETF's direction toward preventing unauthorized monitoring of IETF protocols to continue for the forseeable future.

8. Further Reading and References

   Editor's note: This section is to be kept in a living document where future references, links and/or updates to the existing references will be reflected. That living document is likely to be an IETF-owned Wiki: https://tinyurl.com/streaming-opcons-reading

8.1. Industry Terminology

   * SVA Glossary: https://glossary.streamingvideoalliance.org/

   * Datazoom Video Player Data Dictionary:
      https://help.datazoom.io/hc/en-us/articles/360031323311

   * Datazoom Video Metrics Encyclopedia: https://help.datazoom.io/hc/
      en-us/articles/360046177191

8.2. Surveys and Tutorials

8.2.1. Encoding

   The following papers describe how video is encoded, different video encoding standards and tradeoffs in selecting encoding parameters.

   * Overview of the Versatile Video Coding (VVC) Standard and its
      Applications (https://ieeexplore.ieee.org/document/9503377)

   * Video Compression - From Concepts to the H.264/AVC Standard
      (https://ieeexplore.ieee.org/document/1369695)

* Developments in International Video Coding Standardization After
      AVC, With an Overview of Versatile Video Coding (VVC)
      (https://ieeexplore.ieee.org/document/9328514)

   * A Technical Overview of AV1 (https://ieeexplore.ieee.org/
      document/9363937)

   * CTU Depth Decision Algorithms for HEVC: A Survey
      (https://arxiv.org/abs/2104.08328)

8.2.2. Packaging

   The following papers summarize the methods for selecting packaging
   configurations such as the resolution-bitrate pairs, segment
   durations, use of constant vs. variable-duration segments, etc.

   * Deep Reinforced Bitrate Ladders for Adaptive Video Streaming
      (https://dl.acm.org/doi/10.1145/3458306.3458873)

   * Comparing Fixed and Variable Segment Durations for Adaptive Video
      Streaming: a Holistic Analysis (https://dl.acm.org/
      doi/10.1145/3339825.3391858)

8.2.3. Content Delivery

   The following links describe some of the issues and solutions
   regarding the interconnecting of the content delivery networks.

   * Open Caching: Open standards for Caching in ISP Networks:
      https://www.streamingvideoalliance.org/working-group/open-caching/

   * Netflix Open Connect: https://openconnect.netflix.com

8.2.4. ABR Algorithms

   The two surveys describe and compare different rate-adaptation
   algorithms in terms of different metrics like achieved bitrate/
   quality, stall rate/duration, bitrate switching frequency, fairness,
   network utilization, etc.

   * A Survey on Bitrate Adaptation Schemes for Streaming Media Over

HTTP (https://ieeexplore.ieee.org/document/8424813)

   * A Survey of Rate Adaptation Techniques for Dynamic Adaptive
      Streaming Over HTTP (https://ieeexplore.ieee.org/document/7884970)

8.2.5. Low-Latency Live Adaptive Streaming

   The following papers describe the peculiarities of adaptive streaming
   in low-latency live streaming scenarios.

   * Catching the Moment with LoL+ in Twitch-like Low-latency Live
      Streaming Platforms (https://ieeexplore.ieee.org/document/9429986)

   * Data-driven Bandwidth Prediction Models and Automated Model
      Selection for Low Latency (https://ieeexplore.ieee.org/document/9154522)

   * Performance Analysis of ACTE: A Bandwidth Prediction Method for
      Low-latency Chunked Streaming (https://dl.acm.org/doi/10.1145/3387921)

   * Online Learning for Low-latency Adaptive Streaming
      (https://dl.acm.org/doi/10.1145/3339825.3397042)

   * Tightrope Walking in Low-latency Live Streaming: Optimal Joint
      Adaptation of Video Rate and Playback Speed (https://dl.acm.org/doi/10.1145/3458305.3463382)

   * Content-aware Playback Speed Control for Low-latency Live
      Streaming of Sports (https://dl.acm.org/doi/10.1145/3458305.3478437)

8.2.6. Server/Client/Network Collaboration

   The following papers explain the benefits of server and network
   assistance in client-driven streaming systems. There is also a good
   reference about how congestion affects video quality and how rate
   control works in streaming applications.

* Manus Manum Lavat: Media Clients and Servers Cooperating with
  Common Media Client/Server Data (https://dl.acm.org/
  doi/10.1145/3472305.3472886)

* Common media client data (CMCD): initial findings
  (https://dl.acm.org/doi/10.1145/3458306.3461444)

* SDNDASH: Improving QoE of HTTP Adaptive Streaming Using Software
  Defined Networking (https://dl.acm.org/
  doi/10.1145/2964284.2964332)

* Caching in HTTP Adaptive Streaming: Friend or Foe?
  (https://dl.acm.org/doi/10.1145/2578260.2578270)

* A Survey on Multi-Access Edge Computing Applied to Video
  Streaming: Some Research Issues and Challenges
  (https://ieeexplore.ieee.org/document/9374553)

* The Ultimate Guide to Internet Congestion Control
  (https://www.compiralabs.com/ultimate-guide-congestion-control)

8.2.7. QoE Metrics

The following papers describe various QoE metrics one can use in
streaming applications.

* QoE Management of Multimedia Streaming Services in Future
  Networks: a Tutorial and Survey (https://ieeexplore.ieee.org/
  document/8930519)

* A Survey on Quality of Experience of HTTP Adaptive Streaming
  (https://ieeexplore.ieee.org/document/6913491)

* QoE Modeling for HTTP Adaptive Video Streaming-A Survey and Open
  Challenges (https://ieeexplore.ieee.org/document/8666971)

8.2.8. Point Clouds and Immersive Media

The following papers explain the latest developments in the immersive
media domain (for video and audio) and the developing standards for
such media.

* A Survey on Adaptive 360o Video Streaming: Solutions, Challenges and Opportunities (https://ieeexplore.ieee.org/document/9133103)

* MPEG Immersive Video Coding Standard (https://ieeexplore.ieee.org/document/9374648)

* Emerging MPEG Standards for Point Cloud Compression (https://ieeexplore.ieee.org/document/8571288)

* Compression of Sparse and Dense Dynamic Point Clouds--Methods and Standards (https://ieeexplore.ieee.org/document/9457097)

* MPEG Standards for Compressed Representation of Immersive Audio (https://ieeexplore.ieee.org/document/9444109)

* An Overview of Omnidirectional MediA Format (OMAF) (https://ieeexplore.ieee.org/document/9380215)

* From Capturing to Rendering: Volumetric Media Delivery with Six Degrees of Freedom (https://ieeexplore.ieee.org/document/9247522)

8.3. Open-Source Tools

* 5G-MA: https://www.5g-mag.com/reference-tools

* dash.js: http://reference.dashif.org/dash.js/latest/samples/

* DASH-IF Conformance: https://conformance.dashif.org

* ExoPlayer: https://github.com/google/ExoPlayer

* FFmpeg: https://www.ffmpeg.org/

* GPAC: https://gpac.wp.imt.fr/

* hls.js: https://github.com/video-dev/hls.js

* OBS Studio: https://obsproject.com/

* Shaka Player: https://github.com/google/shaka-player

* Shaka Packager: https://github.com/google/shaka-packager

   * Traffic Control CDN: https://trafficcontrol.apache.org/

   * VideoLAN: https://www.videolan.org/projects/

   * video.js: https://github.com/videojs/video.js

8.4. Technical Events

   * ACM Mile High Video (MHV): https://mile-high.video/

   * ACM Multimedia Systems (MMSys): https://acmmmsys.org

   * ACM Multimedia (MM): https://acmmm.org

   * ACM NOSSDAV: https://www.nossdav.org/

   * ACM Packet Video: https://packet.video/

   * Demuxed and meetups: https://demuxed.com/ and https://demuxed.com/
      events/

   * DVB World: https://www.dvbworld.org

   * EBU BroadThinking: https://tech.ebu.ch/events/broadthinking2021

   * IBC Conference: https://show.ibc.org/conference/ibc-conference

   * IEEE Int. Conf. on Multimedia and Expo (ICME)

   * Media Web Symposium: https://www.fokus.fraunhofer.de/de/go/mws

   * Live Video Stack: https://sh2021.livevideostack.com

   * Picture Coding Symp. (PCS)

   * SCTE Expo: https://expo.scte.org/

8.5. List of Organizations Working on Streaming Media

   * 3GPP SA4: https://www.3gpp.org/specifications-groups/sa-plenary/
      sa4-codec

   * 5G-MAG: https://www.5g-mag.com/

   * AOM: http://aomedia.org/

   * ATSC: https://www.atsc.org/

   * CTA WAVE: https://cta.tech/Resources/Standards/WAVE-Project

   * DASH Industry Forum: https://dashif.org/

   * DVB: https://dvb.org/

   * HbbTV: https://www.hbbtv.org/

   * HESP Alliance: https://www.hespalliance.org/

   * IAB: https://www.iab.com/

   * MPEG: https://www.mpegstandards.org/

   * Streaming Video Alliance: https://www.streamingvideoalliance.org/

   * SCTE: https://www.scte.org/

   * SMPTE: https://www.smpte.org/

   * SRT Alliance: https://www.srtalliance.org/

   * Video Services Forum: https://vsf.tv/

   * VQEG: https://www.its.bldrdoc.gov/vqeg/vqeg-home.aspx

   * W3C: https://www.w3.org/

8.6. Topics to Keep an Eye on

8.6.1. 5G and Media

   5G new radio and systems technologies provide new functionalities for
   video distribution. 5G targets not only smartphones, but also new
   devices such as augmented reality glasses or automotive receivers.
   Higher bandwidth, lower latencies, edge and cloud computing
   functionalities, service-based architectures, low power consumption,
   broadcast/multicast functionalities and other network functions come
   hand in hand with new media formats and processing capabilities
   promising better and more consistent quality for traditional video
   streaming services as well as enabling new experiences such as
   immersive media and augmented realities.

   * 5G Multimedia Standardization (https://www.riverpublishers.com/
      journal_read_html_article.php?j=JICTS/6/1/8)

8.6.2. Ad Insertion

   Ads can be inserted at different stages in the streaming workflow, on
   the server side or client side. The DASH-IF guidelines detail
   server-side ad-insertion with period replacements based on
   manipulating the manifest. HLS interstitials provide a similar
   approach. The idea is that the manifest can be changed and point to
   a sub-playlist of segments, possibly located on a different location.
   This approach results in efficient resource usage in the network, as
   duplicate caching is avoided, but some intelligence at the player is
   needed to deal with content transitions (e.g., codec changes,
   timeline gaps, etc.). Player support for such content is gradually
   maturing. Other important technologies for ad insertion include
   signalling of ads and breaks that is still typically based on SCTE-35
   for HLS and SCTE-214 for DASH. Such signals provide useful
   information for scheduling the ads and contacting ad servers. The
   usage of SCTE-35 for ad insertion is popular in the broadcast
   industry, while the exact usage in the OTT space is still being
   discussed in SCTE. Another important technology is identification of
   ads, such as based on ad-id or other commercial entities that provide
   such services. The identification of the ad in a manifest or stream
   is usually standardized by SMPTE. Other key technologies for ad
   insertion include tracking of viewer impressions, usually based on
   Video Ad Serving Template (VAST) defined by IAB.

   * DASH-IF Ad Insertion Guidelines: https://dashif.org/docs/CR-Ad-
      Insertion-r7.pdf

   * SCTE-214-1: https://www.scte.org/standards-development/library/
      standards-catalog/ansiscte-214-1-2016/

   * RP 2092-1:2015 - SMPTE Recommended Practice - Advertising Digital
      Identifier (Ad-ID) Representations: https://ieeexplore.ieee.org/
      document/7291518

   * IAB Tech Lab Digital Video Studio: https://iabtechlab.com/audio-

video/tech-lab-digital-video-suite/

8.6.3. Contribution and Ingest

There are different contribution and ingest specifications dealing
with different use cases. A common case is contribution that
previously happened over satellite to a broadcast or streaming
headend. RIST and SRT are examples of such contribution protocols.
Within a streaming headend the encoder and packager/CDN may have an
ingest/contribution interface as well. This is specified by the
DASH-IF Ingest.

* DASH-IF Ingest: https://github.com/Dash-Industry-Forum/Ingest

* RIST: https://www.rist.tv/

* SRT: https://github.com/Haivision/srt

8.6.4. Synchronized Encoding and Packaging

Practical streaming headends need redundant encoders and packagers to
operate without glitches and blackouts. The redundant operation
requires synchronization between two or more encoders and also
between two or more packagers that possibly handle different inputs
and outputs, generating compatible inter-changeable output
representations. This problem is important for anyone developing a
streaming headend at scale, and the synchronization problem is
currently under discussion in the wider community. Follow the
developments at: https://sites.google.com/view/encodersyncworkshop/home

8.6.5. WebRTC-Based Streaming

WebRTC is increasingly being used for streaming of time-sensitive

content such as live sporting events. Innovations in cloud computing
allow implementers to efficiently scale delivery of content using
WebRTC. Support for WebRTC communication is available on all modern
web browsers and is available on native clients for all major
platforms.

   * DASH-IF WebRTC Discussions: https://dashif.org/webRTC/

   * Overview of WebRTC: https://webrtc.org/

9. IANA Considerations

   This document requires no actions from IANA.

10. Security Considerations

   Security is an important matter for streaming media applications and
   it was briefly touched on in Section 7.1. This document itself
   introduces no new security issues.

11. Acknowledgments

   Thanks to Alexandre Gouaillard, Aaron Falk, Chris Lemmons, Dave Oran,
   Glenn Deen, Kyle Rose, Leslie Daigle, Lucas Pardue, Mark Nottingham,
   Matt Stock, Mike English, Renan Krishna, Roni Even, Sanjay Mishra,
   and Will Law for very helpful suggestions, reviews and comments.

12. Informative References

   [ABRSurvey]
             Taani, B., Begen, A. C., Timmerer, C., Zimmermann, R., and
             A. Bentaleb et al, "A Survey on Bitrate Adaptation Schemes
             for Streaming Media Over HTTP", IEEE Communications
             Surveys & Tutorials , 2019,
             <https://ieeexplore.ieee.org/abstract/document/8424813>.

   [BAP]   "The Coalition for Better Ads", n.d.,
             <https://www.betterads.org/>.

   [CDiD]   Huitema, C. and B. Trammell, "(A call for) Congestion
               Defense in Depth", July 2019,
               <https://datatracker.ietf.org/meeting/105/materials/
               slides-105-tsvarea-congestion-defense-in-depth-00>.

   [CMAF-CTE] Law, W., "Ultra-Low-Latency Streaming Using Chunked-
               Encoded and Chunked Transferred CMAF", October 2018,
               <https://www.akamai.com/us/en/multimedia/documents/white-
               paper/low-latency-streaming-cmaf-whitepaper.pdf>.

   [CODASPY17]
               Reed, A. and M. Kranch, "Identifying HTTPS-Protected
               Netflix Videos in Real-Time", ACM CODASPY , March 2017,
               <https://dl.acm.org/doi/10.1145/3029806.3029821>.

   [CoDel]   Nichols, K. and V. Jacobson, "Controlling Queue Delay",
               Communications of the ACM, Volume 55, Issue 7, pp. 42-50 ,
               July 2012.

   [COPA18]   Arun, V. and H. Balakrishnan, "Copa: Practical Delay-Based
               Congestion Control for the Internet", USENIX NSDI , April
               2018, <https://web.mit.edu/copa/>.

   [CTA-2066] Consumer Technology Association, "Streaming Quality of
               Experience Events, Properties and Metrics", March 2020,
               <https://shop.cta.tech/products/streaming-quality-of-
               experience-events-properties-and-metrics>.

   [CTA-5004] CTA, "Common Media Client Data (CMCD)", September 2020,
               <https://shop.cta.tech/products/web-application-video-
               ecosystem-common-media-client-data-cta-5004>.

   [CVNI]   "Cisco Visual Networking Index: Forecast and Trends,
               2017-2022 White Paper", 27 February 2019,
               <https://www.cisco.com/c/en/us/solutions/collateral/
               service-provider/visual-networking-index-vni/white-paper-
               c11-741490.html>.

   [ELASTIC]   De Cicco, L., Caldaralo, V., Palmisano, V., and S.
               Mascolo, "ELASTIC: A client-side controller for dynamic
               adaptive streaming over HTTP (DASH)", Packet Video

Workshop , December 2013,
           <https://ieeexplore.ieee.org/document/6691442>.

   [Encodings]
           Apple, Inc, "HLS Authoring Specification for Apple
           Devices", June 2020,
           <https://developer.apple.com/documentation/
           http_live_streaming/
           hls_authoring_specification_for_apple_devices>.

   [I-D.cardwell-iccrg-bbr-congestion-control]
           Cardwell, N., Cheng, Y., Yeganeh, S. H., Swett, I., and V.
           Jacobson, "BBR Congestion Control", Work in Progress,
           Internet-Draft, draft-cardwell-iccrg-bbr-congestion-
           control-01, 7 November 2021,
           <https://datatracker.ietf.org/doc/html/draft-cardwell-
           iccrg-bbr-congestion-control-01>.

   [I-D.draft-pantos-hls-rfc8216bis]
           Pantos, R., "HTTP Live Streaming 2nd Edition", Work in
           Progress, Internet-Draft, draft-pantos-hls-rfc8216bis-10,
           8 November 2021, <https://datatracker.ietf.org/doc/html/
           draft-pantos-hls-rfc8216bis-10>.

   [I-D.ietf-httpbis-cache]
           Fielding, R. T., Nottingham, M., and J. Reschke, "HTTP
           Caching", Work in Progress, Internet-Draft, draft-ietf-
           httpbis-cache-19, 12 September 2021,
           <https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-
           cache-19>.

   [I-D.ietf-quic-datagram]
           Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable
           Datagram Extension to QUIC", Work in Progress, Internet-
           Draft, draft-ietf-quic-datagram-10, 4 February 2022,
           <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
           datagram-10>.

   [I-D.ietf-quic-http]
           Bishop, M., "Hypertext Transfer Protocol Version 3
           (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-
           quic-http-34, 2 February 2021,

          <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
          http-34>.

   [I-D.ietf-quic-manageability]
          Kuehlewind, M. and B. Trammell, "Manageability of the QUIC
          Transport Protocol", Work in Progress, Internet-Draft,
          draft-ietf-quic-manageability-14, 21 January 2022,
          <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
          manageability-14>.

   [I-D.ietf-quic-qlog-h3-events]
          Marx, R., Niccolini, L., and M. Seemann, "HTTP/3 and QPACK
          event definitions for qlog", Work in Progress, Internet-
          Draft, draft-ietf-quic-qlog-h3-events-00, 10 June 2021,
          <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
          qlog-h3-events-00>.

   [I-D.ietf-quic-qlog-main-schema]
          Marx, R., Niccolini, L., and M. Seemann, "Main logging
          schema for qlog", Work in Progress, Internet-Draft, draft-
          ietf-quic-qlog-main-schema-01, 25 October 2021,
          <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
          qlog-main-schema-01>.

   [I-D.ietf-quic-qlog-quic-events]
          Marx, R., Niccolini, L., and M. Seemann, "QUIC event
          definitions for qlog", Work in Progress, Internet-Draft,
          draft-ietf-quic-qlog-quic-events-00, 10 June 2021,
          <https://datatracker.ietf.org/doc/html/draft-ietf-quic-
          qlog-quic-events-00>.

   [IAB-ADS] "IAB", n.d., <https://www.iab.com/>.

   [IABcovid] Arkko, J., Farrel, S., Kühlewind, M., and C. Perkins,
          "Report from the IAB COVID-19 Network Impacts Workshop
          2020", November 2020, <https://datatracker.ietf.org/doc/
          draft-iab-covid19-workshop/>.

   [Jacobson-Karels]
          Jacobson, V. and M. Karels, "Congestion Avoidance and
          Control", November 1988,
          <https://ee.lbl.gov/papers/congavoid.pdf>.

[Labovitz]   Labovitz, C., "Network traffic insights in the time of
             COVID-19: April 9 update", April 2020,
             <https://www.nokia.com/blog/network-traffic-insights-time-
             covid-19-april-9-update/>.

[LabovitzDDoS]
             Takahashi, D., "Why the game industry is still vulnerable
             to DDoS attacks", May 2018,
             <https://venturebeat.com/2018/05/13/why-the-game-industry-
             is-still-vulnerable-to-distributed-denial-of-service-
             attacks/>.

[LL-DASH]   DASH-IF, "Low-latency Modes for DASH", March 2020,
             <https://dashif.org/docs/CR-Low-Latency-Live-r8.pdf>.

[Micro]   Taher, T. M., Misurac, M. J., LoCicero, J. L., and D. R.
             Ucci, "Microwave Oven Signal Interference Mitigation For
             Wi-Fi Communication Systems", 2008 5th IEEE Consumer
             Communications and Networking Conference 5th IEEE, pp.
             67-68 , 2008.

[Mishra]   Mishra, S. and J. Thibeault, "An update on Streaming Video
             Alliance", April 2020,
             <https://datatracker.ietf.org/meeting/interim-2020-mops-
             01/materials/slides-interim-2020-mops-01-sessa-april-
             15-2020-mops-interim-an-update-on-streaming-video-
             alliance>.

[MMSP20]   Durak, K. and et al, "Evaluating the performance of
             Apple's low-latency HLS", IEEE MMSP , September 2020,
             <https://ieeexplore.ieee.org/document/9287117>.

[MMSys11]   Akhshabi, S., Begen, A. C., and C. Dovrolis, "An
             experimental evaluation of rate-adaptation algorithms in
             adaptive streaming over HTTP", ACM MMSys , February 2011,
             <https://dl.acm.org/doi/10.1145/1943552.1943574>.

[MPEG-CMAF]
             "ISO/IEC 23000-19:2020 Multimedia application format
             (MPEG-A) - Part 19: Common media application format (CMAF)
             for segmented media", March 2020,
             <https://www.iso.org/standard/79106.html>.

[MPEG-DASH]
            "ISO/IEC 23009-1:2019 Dynamic adaptive streaming over HTTP
            (DASH) - Part 1: Media presentation description and
            segment formats", December 2019,
            <https://www.iso.org/standard/79329.html>.

[MPEG-DASH-SAND]
            "ISO/IEC 23009-5:2017 Dynamic adaptive streaming over HTTP
            (DASH) - Part 5: Server and network assisted DASH (SAND)",
            February 2017, <https://www.iso.org/standard/69079.html>.

[MPEG-TS]   "H.222.0 : Information technology - Generic coding of
            moving pictures and associated audio information:
            Systems", 29 August 2018,
            <https://www.itu.int/rec/T-REC-H.222.0>.

[MPEGI]   Boyce, J. M. and et al, "MPEG Immersive Video Coding
            Standard", Proceedings of the IEEE , n.d.,
            <https://ieeexplore.ieee.org/document/9374648>.

[OReilly-HPBN]
            "High Performance Browser Networking (Chapter 2: Building
            Blocks of TCP)", May 2021,
            <https://hpbn.co/building-blocks-of-tcp/>.

[PCC]   Schwarz, S. and et al, "Emerging MPEG Standards for Point
            Cloud Compression", IEEE Journal on Emerging and Selected
            Topics in Circuits and Systems , March 2019,
            <https://ieeexplore.ieee.org/document/8571288>.

[Port443]   "Service Name and Transport Protocol Port Number
            Registry", April 2021, <https://www.iana.org/assignments/
            service-names-port-numbers/service-names-port-
            numbers.txt>.

[RFC0793]   Postel, J., "Transmission Control Protocol", STD 7,
            RFC 793, DOI 10.17487/RFC0793, September 1981,
            <https://www.rfc-editor.org/rfc/rfc793>.

[RFC2001]   Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast
            Retransmit, and Fast Recovery Algorithms", RFC 2001,
            DOI 10.17487/RFC2001, January 1997,
            <https://www.rfc-editor.org/rfc/rfc2001>.

   [RFC3135] Border, J., Kojo, M., Griner, J., Montenegro, G., and Z.
              Shelby, "Performance Enhancing Proxies Intended to
              Mitigate Link-Related Degradations", RFC 3135,
              DOI 10.17487/RFC3135, June 2001,
              <https://www.rfc-editor.org/rfc/rfc3135>.

   [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
              Jacobson, "RTP: A Transport Protocol for Real-Time
              Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550,
              July 2003, <https://www.rfc-editor.org/rfc/rfc3550>.

   [RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P.
              Conrad, "Stream Control Transmission Protocol (SCTP)
              Partial Reliability Extension", RFC 3758,
              DOI 10.17487/RFC3758, May 2004,
              <https://www.rfc-editor.org/rfc/rfc3758>.

   [RFC4733] Schulzrinne, H. and T. Taylor, "RTP Payload for DTMF
              Digits, Telephony Tones, and Telephony Signals", RFC 4733,
              DOI 10.17487/RFC4733, December 2006,
              <https://www.rfc-editor.org/rfc/rfc4733>.

   [RFC5594] Peterson, J. and A. Cooper, "Report from the IETF Workshop
              on Peer-to-Peer (P2P) Infrastructure, May 28, 2008",
              RFC 5594, DOI 10.17487/RFC5594, July 2009,
              <https://www.rfc-editor.org/rfc/rfc5594>.

   [RFC5762] Perkins, C., "RTP and the Datagram Congestion Control
              Protocol (DCCP)", RFC 5762, DOI 10.17487/RFC5762, April
              2010, <https://www.rfc-editor.org/rfc/rfc5762>.

   [RFC6190] Wenger, S., Wang, Y.-K., Schierl, T., and A.
              Eleftheriadis, "RTP Payload Format for Scalable Video
              Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011,
              <https://www.rfc-editor.org/rfc/rfc6190>.

   [RFC6582] Henderson, T., Floyd, S., Gurtov, A., and Y. Nishida, "The
              NewReno Modification to TCP's Fast Recovery Algorithm",
              RFC 6582, DOI 10.17487/RFC6582, April 2012,
              <https://www.rfc-editor.org/rfc/rfc6582>.

   [RFC6817] Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind,
              "Low Extra Delay Background Transport (LEDBAT)", RFC 6817,
              DOI 10.17487/RFC6817, December 2012,
              <https://www.rfc-editor.org/rfc/rfc6817>.

   [RFC6843] Clark, A., Gross, K., and Q. Wu, "RTP Control Protocol
              (RTCP) Extended Report (XR) Block for Delay Metric
              Reporting", RFC 6843, DOI 10.17487/RFC6843, January 2013,
              <https://www.rfc-editor.org/rfc/rfc6843>.

   [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an
              Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May
              2014, <https://www.rfc-editor.org/rfc/rfc7258>.

   [RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black,
              "Encapsulating MPLS in UDP", RFC 7510,
              DOI 10.17487/RFC7510, April 2015,
              <https://www.rfc-editor.org/rfc/rfc7510>.

   [RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and
              B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms
              for Real-Time Transport Protocol (RTP) Sources", RFC 7656,
              DOI 10.17487/RFC7656, November 2015,
              <https://www.rfc-editor.org/rfc/rfc7656>.

   [RFC7661] Fairhurst, G., Sathiaseelan, A., and R. Secchi, "Updating
              TCP to Support Rate-Limited Traffic", RFC 7661,
              DOI 10.17487/RFC7661, October 2015,
              <https://www.rfc-editor.org/rfc/rfc7661>.

   [RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control:
              Circuit Breakers for Unicast RTP Sessions", RFC 8083,
              DOI 10.17487/RFC8083, March 2017,
              <https://www.rfc-editor.org/rfc/rfc8083>.

   [RFC8084] Fairhurst, G., "Network Transport Circuit Breakers",
              BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017,
              <https://www.rfc-editor.org/rfc/rfc8084>.

   [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage
              Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085,
              March 2017, <https://www.rfc-editor.org/rfc/rfc8085>.

   [RFC8216] Pantos, R., Ed. and W. May, "HTTP Live Streaming",
              RFC 8216, DOI 10.17487/RFC8216, August 2017,
              <https://www.rfc-editor.org/rfc/rfc8216>.

   [RFC8312] Rhee, I., Xu, L., Ha, S., Zimmermann, A., Eggert, L., and
              R. Scheffenegger, "CUBIC for Fast Long-Distance Networks",
              RFC 8312, DOI 10.17487/RFC8312, February 2018,
              <https://www.rfc-editor.org/rfc/rfc8312>.

   [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol
              Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018,
              <https://www.rfc-editor.org/rfc/rfc8446>.

   [RFC8622] Bless, R., "A Lower-Effort Per-Hop Behavior (LE PHB) for
              Differentiated Services", RFC 8622, DOI 10.17487/RFC8622,
              June 2019, <https://www.rfc-editor.org/rfc/rfc8622>.

   [RFC8723] Jennings, C., Jones, P., Barnes, R., and A.B. Roach,
              "Double Encryption Procedures for the Secure Real-Time
              Transport Protocol (SRTP)", RFC 8723,
              DOI 10.17487/RFC8723, April 2020,
              <https://www.rfc-editor.org/rfc/rfc8723>.

   [RFC8825] Alvestrand, H., "Overview: Real-Time Protocols for
              Browser-Based Applications", RFC 8825,
              DOI 10.17487/RFC8825, January 2021,
              <https://www.rfc-editor.org/rfc/rfc8825>.

   [RFC8999] Thomson, M., "Version-Independent Properties of QUIC",
              RFC 8999, DOI 10.17487/RFC8999, May 2021,
              <https://www.rfc-editor.org/rfc/rfc8999>.

   [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based
              Multiplexed and Secure Transport", RFC 9000,
              DOI 10.17487/RFC9000, May 2021,

                  <https://www.rfc-editor.org/rfc/rfc9000>.

   [RFC9001]   Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure
               QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021,
               <https://www.rfc-editor.org/rfc/rfc9001>.

   [RFC9002]   Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection
               and Congestion Control", RFC 9002, DOI 10.17487/RFC9002,
               May 2021, <https://www.rfc-editor.org/rfc/rfc9002>.

   [SFRAME]   "Secure Media Frames Working Group (Home Page)", n.d.,
               <https://datatracker.ietf.org/doc/charter-ietf-sframe/>.

   [SRT]   Sharabayko, M., "Secure Reliable Transport (SRT) Protocol
               Overview", 15 April 2020,
               <https://datatracker.ietf.org/meeting/interim-2020-mops-
               01/materials/slides-interim-2020-mops-01-sessa-april-
               15-2020-mops-interim-an-update-on-streaming-video-
               alliance>.

   [Survey360o]
               Yaqoob, A., Bi, T., and G. Muntean, "A Survey on Adaptive
               360° Video Streaming: Solutions, Challenges and
               Opportunities", IEEE Communications Surveys & Tutorials ,
               July 2020, <https://ieeexplore.ieee.org/document/9133103>.

   [tsvarea-105]
               "TSVAREA Minutes - IETF 105", July 2019,
               <https://datatracker.ietf.org/meeting/105/materials/
               minutes-105-tsvarea-00>.

Authors' Addresses

   Jake Holland
   Akamai Technologies, Inc.
   150 Broadway
   Cambridge, MA 02144,
   United States of America
   Email: jakeholland.net@gmail.com

Ali Begen
Networked Media
Turkey
Email: ali.begen@networked.media

Spencer Dawkins
Tencent America LLC
United States of America
Email: spencerdawkins.ietf@gmail.com