Video Streaming (Real-Time Communication, Peer to Peer)

Sean Corzo | August 15, 2023

SDP:

WebRTC (Web Real-Time Communication) and SDP (Session Description Protocol) are closely related and work together to enable real-time communication and media streaming in web browsers.

SDP is a protocol used to describe the media capabilities and session details during the negotiation phase of establishing a communication session. It provides a standardized way for two devices to exchange information about their media capabilities, such as supported codecs, resolutions, and network addresses.

Relationship between WebRTC and SDP:

The process of establishing a WebRTC connection involves several steps, and SDP is a crucial component of this negotiation:

Offer/Answer Exchange:
The WebRTC communication starts with one device sending an "offer" to the other device. The offer includes the device's media capabilities, such as supported audio and video codecs, resolutions, and network addresses.

SDP Payload:
The offer and subsequent responses are encoded using SDP, which is then carried in the signaling messages between the two devices. The signaling channel can be established using various methods, such as WebSocket, HTTP, or a dedicated signaling server.

Remote Description:
When the other device receives the offer, it decodes the SDP payload and extracts the session details (media capabilities). It then creates an "answer" that includes its own media capabilities and sends it back to the first device.

Connection Establishment:
Both devices continue exchanging SDP payloads until they agree on a set of compatible media capabilities. This process is known as the "SDP negotiation" or "SDP exchange." Once they agree on the session details, the WebRTC connection is established, and media streaming can begin.

Peer-to-Peer Communication:
Once the WebRTC connection is established, the devices can communicate directly (peer-to-peer) without involving a centralized server. This direct communication enables real-time audio, video, and data exchange.

Video Streaming Upload Protocols/Technologies:

For live video streaming, real-time delivery and low latency are essential to ensure a smooth and interactive viewing experience for viewers. As a result, video streaming platforms commonly use Real-Time Messaging Protocol (RTMP) or WebRTC for uploading video content.

RTMP (Real-Time Messaging Protocol): RTMP is a streaming protocol designed for real-time transmission of audio, video, and data over the internet. It is commonly used for live video streaming as it allows for low-latency communication and real-time interaction between the publisher and the server.

WebRTC (Web Real-Time Communication):
WebRTC is a set of technologies that enable real-time communication directly between web browsers and devices. It is often used for browser-based live video streaming, allowing publishers to stream video directly from their web browsers without the need for additional plugins or software.

Streaming Upload Containerization

In the context of WebRTC, the individual RTP packets are not containerized in the same way as traditional video streaming protocols like MPEG-DASH or HLS. Instead, WebRTC relies on the underlying transport protocols (UDP or TCP) for packet delivery and reordering.

While WebRTC does not use a container format in the same sense as other streaming protocols, it does utilize protocols like SRTP (Secure Real-Time Transport Protocol) to provide encryption and security for the real-time data.

Video Formats: Video formats refer to the container or file format that holds the encoded video data along with audio, subtitles, and other metadata. Examples of video formats include MP4, AVI, MOV, and MKV. These formats define the structure and organization of the data within the file, including how the video and audio streams are stored and synchronized.

Codecs: Codecs, on the other hand, are the technologies used to encode and compress the video and audio data within the video format. Codecs are responsible for reducing the file size while maintaining acceptable video quality. They define algorithms for compressing and decompressing the data.

In some cases, video formats may also incorporate specific codecs as part of their specifications.

The formats and codecs used for video streaming and on-demand video platforms are generally similar. Here are the typical formats and codecs used for both scenarios:

Formats:
MP4 (H.264/AAC): MP4 is a widely used video container format for both streaming and on-demand video. It is compatible with most devices and browsers and supports the H.264 video codec and the AAC audio codec.

WebM (VP9/Opus): WebM is another popular video container format, primarily used for web-based content. It offers high-quality video compression with the VP9 video codec and Opus audio codec. It is well-supported by modern web browsers.

MPEG-DASH: MPEG-DASH (Dynamic Adaptive Streaming over HTTP) is a streaming format that uses fragmented MP4 (fMP4) as its media segment format. It is commonly used for adaptive bitrate streaming for both live and on-demand video.

Apple HTTP Live Streaming (HLS): HLS is an adaptive bitrate streaming format developed by Apple. It uses MPEG-2 Transport Stream (M2TS) segments in a .m3u8 playlist format and is commonly used for streaming to iOS devices and other platforms that support HLS.

Codecs:
H.264 (AVC): H.264 is one of the most widely used video codecs for both streaming and on-demand video. It provides a good balance of video quality and compression efficiency, making it suitable for various devices and network conditions.

VP9: VP9 is an open-source video codec developed by Google. It offers superior compression efficiency compared to H.264, making it ideal for high-resolution video streaming, especially in scenarios where bandwidth is limited.

AV1: AV1 is another open-source video codec that provides even better compression efficiency than VP9. It is gaining popularity for video streaming platforms, especially for 4K and higher resolution content, although it may require more computational resources for encoding and decoding.

AAC (Advanced Audio Codec): AAC is a widely used audio codec for both streaming and on-demand video platforms. It offers high-quality audio at relatively low bitrates, making it suitable for various devices and bandwidths.

Resolution Choices:

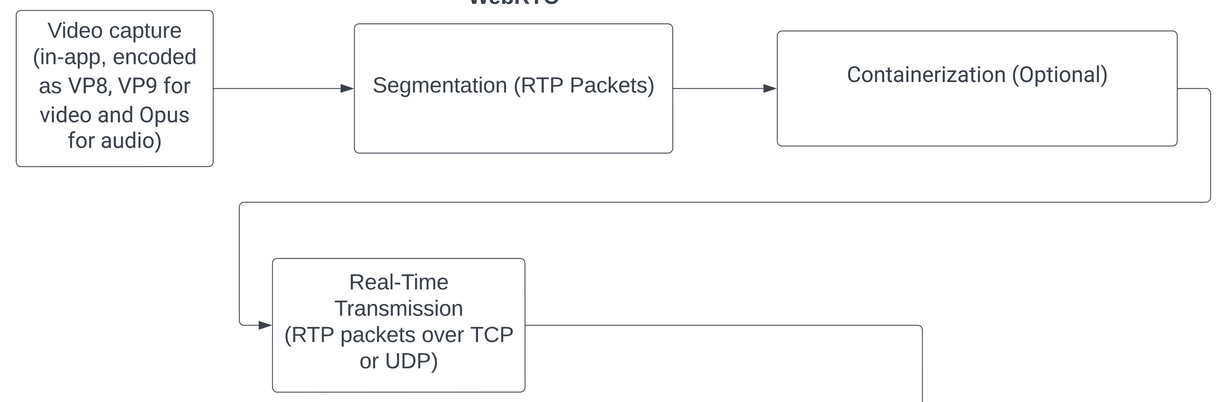
Both video streaming and on-demand platforms offer multiple resolution options to accommodate viewers with varying screen sizes and network capabilities. Common resolution choices include:

SD (Standard Definition): Common resolutions for SD content include 480p (854x480) and 360p (640x360). These are suitable for smaller screens and viewers with lower bandwidth.

HD (High Definition): HD resolutions commonly include 720p (1280x720) and 1080p (1920x1080). They offer higher quality and are ideal for larger screens and viewers with better internet connections.

4K (Ultra High Definition): 4K (3840x2160) provides even higher quality for large displays and viewers with high-speed internet connections. It is becoming more popular as 4K-capable devices become prevalent.

WebRTC



Transcoding considerations (streaming vs. on-demand)

number of proxy (bitrate) resolution versions

keyframe interval duration

compression optimizations

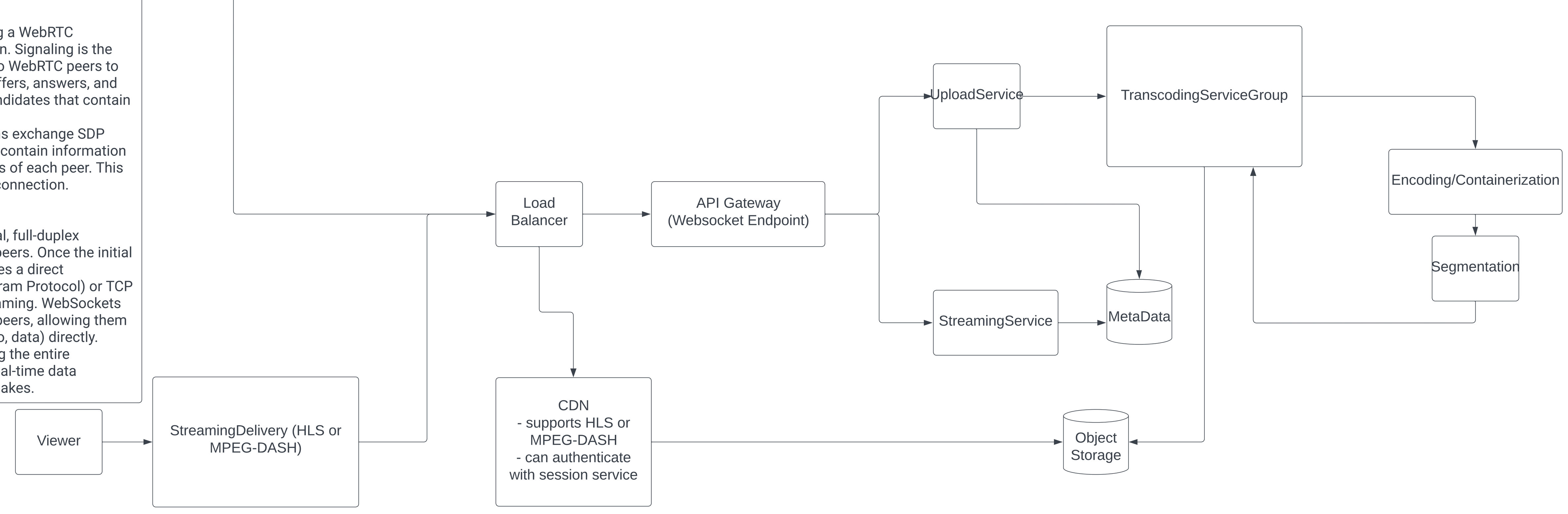
segment duration

dynamic vs. static storage solutions

WebRTC (Web Real-Time Communication) primarily uses both HTTP and WebSockets, but for different purposes within the WebRTC communication process.

HTTP:
HTTP is used in the initial stages of establishing a WebRTC connection for signaling and session negotiation. Signaling is the process of exchanging information between two WebRTC peers to set up the connection. It includes exchanging offers, answers, and ICE (Interactive Connectivity Establishment) candidates that contain network information.
During the signaling phase, WebRTC applications exchange SDP (Session Description Protocol) payloads, which contain information about the media capabilities and network details of each peer. This information is used to establish a peer-to-peer connection.

WebSockets:
WebSockets are used to establish a bidirectional, full-duplex communication channel between the WebRTC peers. Once the initial signaling phase is complete, WebRTC establishes a direct peer-to-peer connection using UDP (User Datagram Protocol) or TCP (Transmission Control Protocol) for media streaming. WebSockets facilitate real-time data exchange between the peers, allowing them to send and receive media streams (audio, video, data) directly.
The WebSocket connection remains open during the entire communication session, allowing continuous real-time data exchange without the need for repeated handshakes.



On-Demand and Streaming Delivery Protocols:

The two main HTTP-based adaptive streaming protocols used are:

HTTP Live Streaming (HLS): HLS is an adaptive streaming protocol developed by Apple. It breaks video content into small segments and serves them over HTTP. HLS uses .m3u8 playlist files that contain URLs to the video segments at different bitrates, allowing the media player to choose the appropriate bitrate based on the viewer's network speed and device capabilities. HLS is widely supported on various devices and platforms, making it a popular choice for video streaming, especially for iOS devices.

Dynamic Adaptive Streaming over HTTP (MPEG-DASH): MPEG-DASH is an adaptive streaming protocol that works similarly to HLS but is more platform-agnostic and widely supported across different browsers and devices. It uses .mpd manifest files to provide URLs to video segments at various bitrates, enabling adaptive streaming based on network conditions. MPEG-DASH is a standardized format, making it an attractive choice for cross-platform video delivery.

Both HLS and MPEG-DASH are based on the principles of adaptive bitrate streaming, which improves the viewing experience by automatically adjusting the video quality to match the viewer's internet connection speed and device capabilities. This approach helps to avoid buffering and ensures smoother playback.

Chunking (Segmentation):

After the video and audio are placed within the container format, they are divided into smaller segments or "chunks." Chunking is a process of breaking down the continuous video stream into smaller, fixed-size segments.

Each chunk typically has a fixed duration, such as 2 seconds, 4 seconds, or 6 seconds. The duration of the chunks can vary depending on the use case and requirements.

Segmented File Format:
The result of the chunking process is a series of smaller segmented files, each containing a portion of the original video and audio data. These segments are often referred to as "chunks" or "segments."

These segmented files are not new containers; rather, they are individual media files that are part of the original container format. Each segment retains the codec information and metadata necessary for playback.

On-Demand Video

Sean Corzo | August 15, 2023

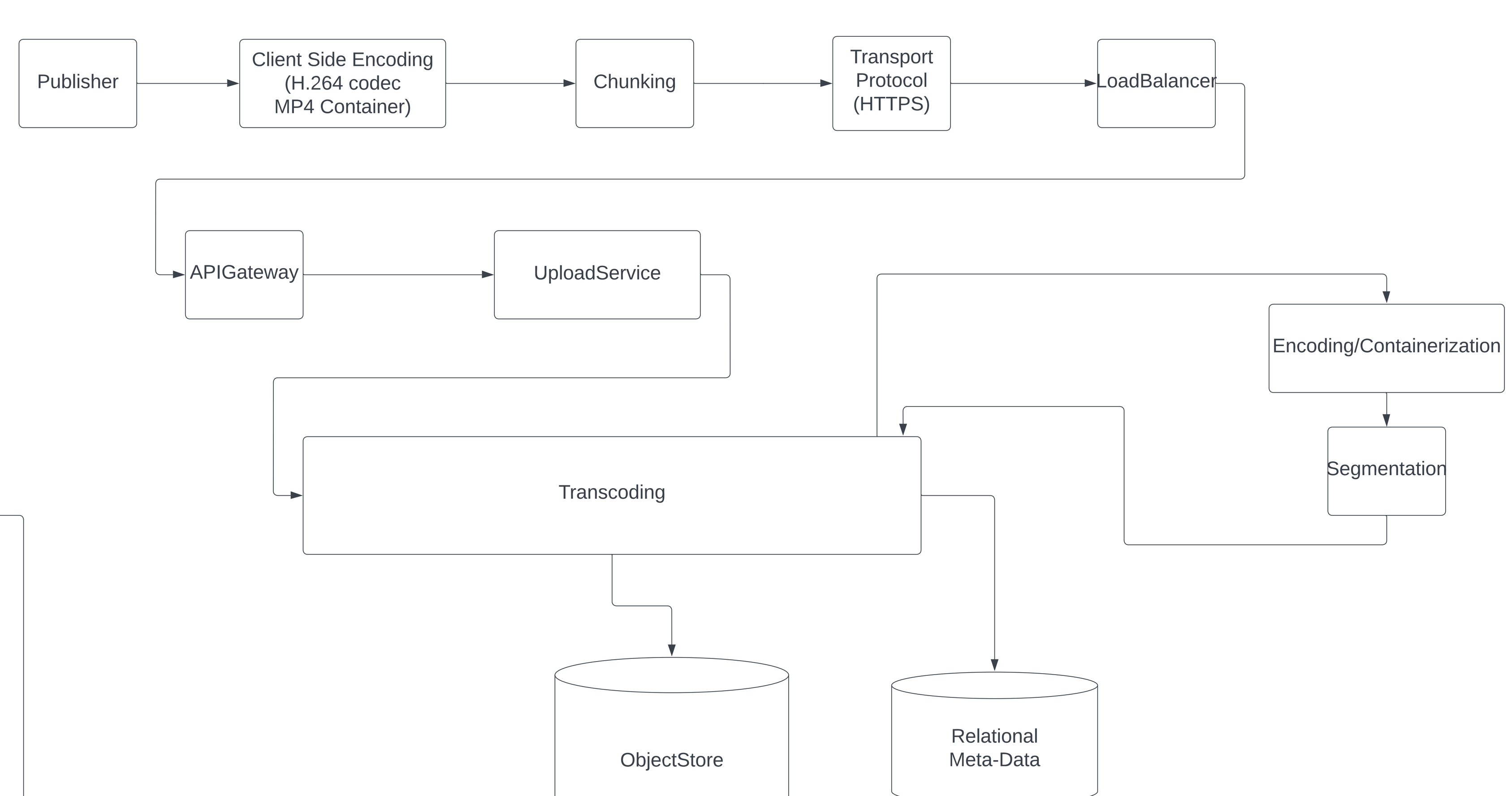
Client-Side Video Upload Pre-Processing

Format Conversion: The client may convert the video to a format compatible with YouTube's video streaming infrastructure. While YouTube supports various video formats, some formats may not be suitable for efficient streaming or may require additional processing, so the client may convert the video to a standard format supported by YouTube, such as MP4 with H.264 video codec and AAC audio codec.

Resolution and Bitrate Optimization: Depending on the original video's resolution and bitrate, the client may optimize these parameters to better match YouTube's recommended settings. YouTube has guidelines for video resolutions, bitrates, and other encoding parameters to ensure the best quality and performance during playback.

Metadata and Thumbnail Generation: The client may prompt you to provide metadata for the video, such as the title, description, tags, and category. It may also generate thumbnails or allow you to select a custom thumbnail for the video.

Upload Chunking: For large video files, the client may use a technique called "chunking," where the video is split into smaller segments or chunks for more efficient and reliable uploading.



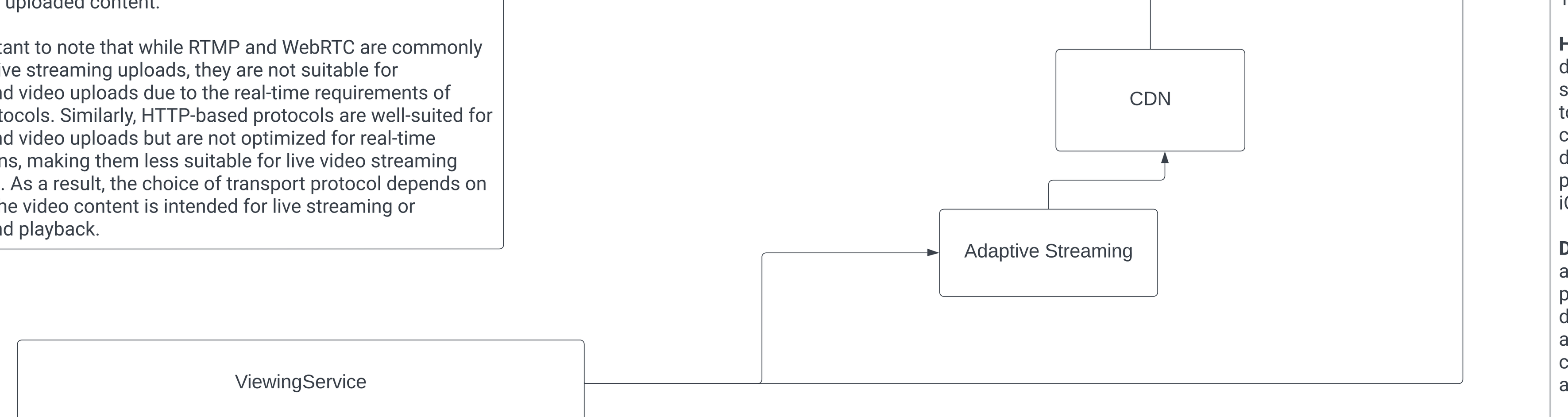
On-Demand Video Upload Protocols:

On-demand video platforms, like YouTube or Netflix, deal with pre-recorded videos, and real-time delivery is not a primary concern. As a result, standard HTTP-based protocols are commonly used for uploading on-demand video content.

HTTP (Hypertext Transfer Protocol): HTTP is the foundation of data communication on the World Wide Web and is used for transferring resources like video files from the client (uploader) to the server (video platform). It is a widely supported and reliable protocol for uploading pre-recorded video content.

HTTPS (Hypertext Transfer Protocol Secure): HTTPS is a secure version of HTTP, which encrypts data during transmission. Most on-demand video platforms use HTTPS to ensure the security and privacy of uploaded content.

It's important to note that while RTMP and WebRTC are commonly used for live streaming uploads, they are not suitable for on-demand video uploads due to the real-time requirements of these protocols. Similarly, HTTP-based protocols are well-suited for on-demand video uploads but are not optimized for real-time interactions, making them less suitable for live video streaming scenarios. As a result, the choice of transport protocol depends on whether the video content is intended for live streaming or on-demand playback.



On-Demand and Streaming Delivery Protocols:

The two main HTTP-based adaptive streaming protocols used are:

HTTP Live Streaming (HLS): HLS is an adaptive streaming protocol developed by Apple. It breaks video content into small segments and serves them over HTTP. HLS uses .m3u8 playlist files that contain URLs to the video segments at different bitrates, allowing the media player to choose the appropriate bitrate based on the viewer's network speed and device capabilities. HLS is widely supported on various devices and platforms, making it a popular choice for video streaming, especially for iOS devices.

Dynamic Adaptive Streaming over HTTP (MPEG-DASH): MPEG-DASH is an adaptive streaming protocol that works similarly to HLS but is more platform-agnostic and widely supported across different browsers and devices. It uses .mpd manifest files to provide URLs to video segments at various bitrates, enabling adaptive streaming based on network conditions. MPEG-DASH is a standardized format, making it an attractive choice for cross-platform video delivery.

Both HLS and MPEG-DASH are based on the principles of adaptive bitrate streaming, which improves the viewing experience by automatically adjusting the video quality to match the viewer's internet connection speed and device capabilities. This approach helps to avoid buffering and ensures smoother playback.