**Introduction:**

This document commences by examining the fundamental concept of the internet and the mechanisms that facilitate the transfer of data between parties. Subsequently, it delves into the intricacies of a typical server, its functions, and its significance in a network architecture. Furthermore, it explores the concept of a media server, analyzing how data is exchanged between parties and how a media file is transferred from a media server to a client device. The document then delves into the concept of streaming and how it is enabled by a streaming media server.

**Internet**

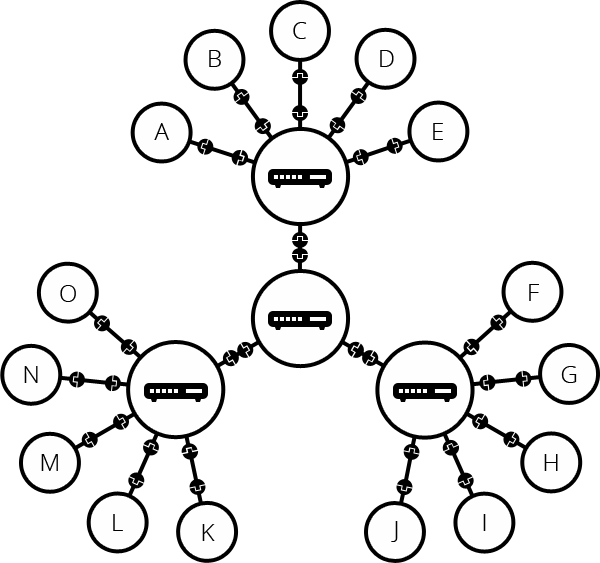
The internet serves as the backbone of the web, providing the technical infrastructure that makes it all possible. At its core, the internet is a vast network of interconnected computers that communicate with each other.

When two computers need to communicate, they can be linked either physically or wirelessly. While it is possible to connect two computers easily, adding more can quickly become complicated, as it requires many cables and significant management. To simplify this process, a router is typically used. The router allows multiple computers to connect and, if necessary, additional routers can be connected to accommodate more devices. A modem is also necessary, as it connects the router to the telephone infrastructure, allowing for data to be transmitted over long distances.

This intricate network of connected devices is what we refer to as the internet, and it enables information to be shared and exchanged around the world.

When transmitting a media file such as audio or video, it is possible that either the sender or the client may not have the required bandwidth to handle the full amount of data. As a result, the data may become corrupted during transmission, potentially causing issues with the quality of the audio or video.

In earlier years, media files were typically transferred from one location to another. However, in recent years, streaming technology has emerged as a popular alternative. With streaming, a client is no longer required to download the entire file before playing it. Instead, the media file is hosted on a server and can be accessed and played in real-time.



**Server**:

On the hardware side, a web server is a computer that stores web server software and a website's component files (for example, HTML documents, images, CSS stylesheets,). A web server connects to the Internet and supports physical data interchange with other devices connected to the web.

At the most basic level, whenever a browser needs a file that is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct (hardware) web server, the (software) HTTP server accepts the request, finds the requested document, and sends it back to the browser, also through HTTP.

A media server is a dedicated server running applications that store and share multimedia files (text, graphics, video, audio…etc.) on demand. It is a device that is designed specifically for the purpose of hosting and distributing media content, and typically includes multiple hard drives and a high-speed network connection to facilitate the streaming of media files to other devices.A media server can be any device having network access with suitable bandwidth for the sharing and saving of media. A server, PC, network-attached storage (NAS) or any other device with such storage capability can be used as a media server.

**Streaming**:

Streaming media is multimedia that is delivered and consumed in a continuous manner from a source, with little or no intermediate storage in network elements. Streaming refers to the delivery method of content, rather than the content itself.

Livestreaming is the real-time delivery of content during production, much as live television broadcasts content via television channels. Livestreaming requires a form of source media (e.g. a video camera, an audio interface, screen capture software), an encoder to digitize the content, a media publisher, and a content delivery network to distribute and deliver the content.

Streaming is most prevalent in video on demand and streaming television services. Other services stream music or video games.

The characteristics of streaming systems can reduce the server loading, and rapidly synchrony presented the media. The technology of streaming allows users or clients to view or hear media objects without having to wait until the entire media is downloaded, instead of downloading media files directly and entirely. Based on above characteristics, streaming technology would reduce the bandwidth requirement.

-YouTube is a platform that streams media to millions of end users every hour.

Live streaming workflows can vary immensely, but six key steps are often involved:

1)Capturing the stream with a camera or smart device.

2)Compressing the video and audio data using an encoder.

3)Ingesting that content into a streaming platform like Wowza for processing.

4)Transcoding and packaging the stream into its final format and protocol.

5)Delivering the live stream across the internet (often using a content delivery network).

6)Playing the content back on end-user devices.

**How this works?**

**Whats inside a media server?**

A physical media server can be made up of several components that work together to store, manage, and stream digital media files. Here are some of the key components that you might find in a physical media server:

CPU: The central processing unit (CPU) is the "brain" of the media server, and is responsible for processing data and managing system resources.

RAM: Random access memory (RAM) is used to temporarily store data that is being processed by the CPU. The amount of RAM in a media server can impact its performance when streaming or transcoding media files.

Hard drives: Media servers typically include multiple hard drives to store digital media files. These hard drives can be configured in various ways, such as in a RAID configuration for redundancy and data protection.

Network interface: The media server needs a network interface to communicate with other devices on the network and stream media files.

Power supply: A media server requires a power supply to provide electricity to its components.

Cooling system: Because media servers generate a lot of heat, they typically include a cooling system to keep the components from overheating.

Operating system: The media server requires an operating system (OS) to run the software that manages the media files. Common OS options include Windows, Linux, and macOS.

Media server software: To manage the media files and stream them to other devices, a media server typically runs specialized software such as Plex, Emby, or Kodi.

Overall, a media server is designed to be a reliable and powerful device that can store and distribute large amounts of digital media files.

In general, the principle is simple.

A Media server stores all the media files inside it in a remote location. User requests the file either audio or video through Internet, the server plays the requires file as response, thereby eliminating the need to store the data.This complex process involves n number of protocols, i.e a set of rules on how its done. so lets see the common protocols that are usually used in content streaming.

Streaming protocol, also known as a broadcast protocol, is a standardized method of delivering different types of media (usually video or audio) over the internet.

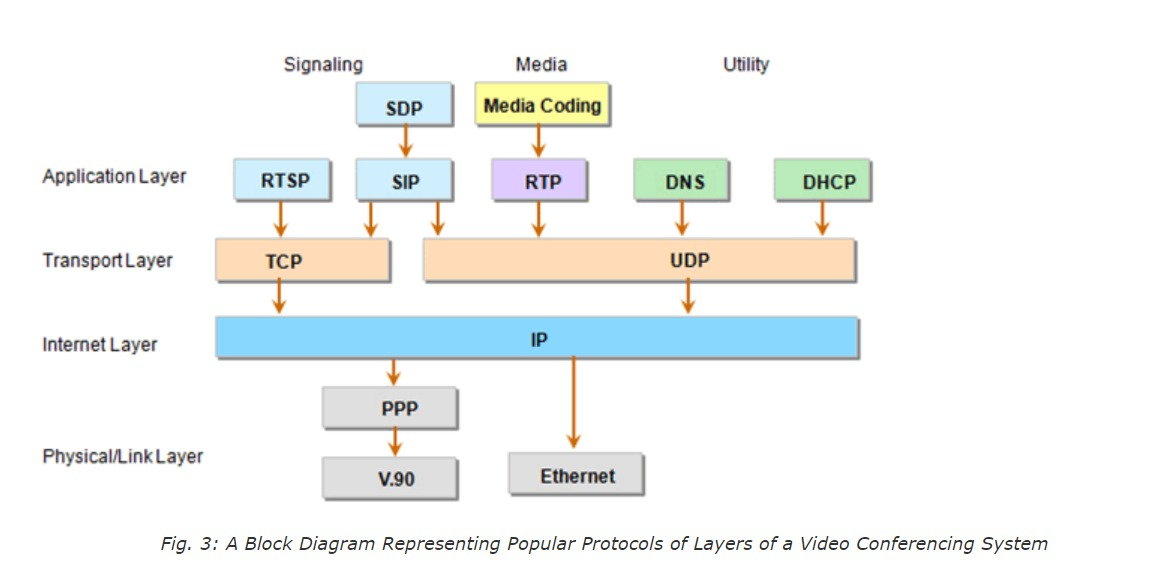
Essentially, a video streaming protocol sends “chunks” of content from one device to another. It also defines the method for “reassembling” these chunks into playable content on the other end.

That points toward one important aspect of streaming protocols: both the output device and the viewer have to support the protocol in order for it to work.

For example, if you’re sending a stream in MPEG-DASH, but the video player on the device to which you’re streaming doesn’t support MPEG-DASH, your stream won’t work.

For this reason, standardization is important. There are currently a few major media streaming protocols in widespread use, which we’ll look at in detail in a moment. Six common protocols include:

* HTTP Live Streaming (HLS)
* Session Initiation Protocol
* RTP
* RTCP
* RTSP
* RSVP
* Dynamic Adaptive Streaming over HTTP (DASH)
* and others Internet protocols.



**HTTP Live Streaming (HLS)**

HLS streaming is an alternative protocol developed by Apple. HLS stands for HTTP Live Streaming, and today it is the most widely used streaming protocol on the internet. However, this was not always the case because when Flash was still around, the top streaming protocol was RTMP.

HLS is an adaptive bitrate protocol and also uses HTTP servers. This protocol is an evolving specification, as Apple continually adds features and regularly improves HLS.

HLS is one of the protocols that Dacast uses. Dacast has also added support for HLS ingest, which is still relatively new. Keep in mind that very few streaming platforms at the moment support HLS ingest.

HTTP Live Streaming (HLS) is a widely used streaming protocol that allows streaming of audio and video content over the internet. It was developed by Apple Inc. and is used to deliver media content to Apple devices, as well as other devices that support the protocol.

The HLS protocol works by dividing the media content into small segments that are then delivered to the client over the internet. The client requests these segments from the server and plays them back as they are received. This allows the client to start playing the content almost immediately, without having to download the entire file first. The client also has the ability to adapt to changes in the network conditions, such as bandwidth fluctuations, and request the appropriate quality of the content.

The HLS protocol uses standard HTTP transactions to deliver the media content. This makes it easy to deploy, as it does not require any special server software or network configuration. It also allows content providers to use standard web servers and CDNs to deliver the content, which can help to reduce costs.

One of the key features of HLS is its support for adaptive bitrate streaming. This allows the client to automatically adjust the quality of the content based on the current network conditions. For example, if the network connection is slow, the client can request a lower quality version of the content to ensure smooth playback. If the network connection improves, the client can request a higher quality version of the content. This helps to ensure that the content is always delivered at the best possible quality, while also minimizing buffering and interruptions.

HLS also supports multiple audio and subtitle tracks, which can be switched by the user during playback. This allows content providers to offer content in multiple languages, as well as provide accessibility features for the hearing impaired.

In order to create an HLS stream, the content provider must first encode the media into the appropriate format. The most commonly used format for HLS is the MPEG-2 Transport Stream (TS), which is a container format that can contain video, audio, and other data. The encoded media is then divided into small segments, typically 2 to 10 seconds in length, and an index file is created that contains information about the segments and their location on the server.

The index file is typically a .m3u8 file, which is a playlist file that contains URLs for each of the segment files. When a client requests the content, it first downloads the index file and then requests the individual segment files. The index file also contains information about the available bitrates and resolutions for the content, which allows the client to choose the appropriate quality based on the network conditions.

HLS has become a popular choice for streaming media content, particularly for live events such as sports and concerts. This is because it allows for a large number of users to access the content simultaneously, without requiring a large amount of server resources. It also provides a high level of reliability, as the content is delivered over standard HTTP transactions that are widely supported by network infrastructure.

In conclusion, HTTP Live Streaming is a popular and effective streaming protocol that allows for the delivery of high-quality media content over the internet. Its support for adaptive bitrate streaming and multiple audio and subtitle tracks make it a versatile solution for content providers. Its use of standard HTTP transactions and playlist files also make it easy to deploy and integrate into existing web infrastructure.

If you want to connect with viewers who use Apple devices, HLS streaming is one of the best protocols for live streaming.

**Session Initiation Protocol (SIP):**

Session Initiation Protocol (SIP) is a signaling protocol used for initiating, maintaining, modifying and terminating real-time communications sessions between Internet Protocol (IP) devices. SIP enables voice, messaging, video and other communications applications and services between two or more endpoints on IP networks.

SIP was developed by the Internet Engineering Task Force in 1996 and standardized in 1999.

SIP addresses the evolving needs of IP-based communications. Native support for mobility, interoperability and multimedia was among the drivers behind SIP development. SIP complements other communications protocols, such as Real-Time Transport Protocol (RTP) and Real-Time Streaming Protocol, used in IP-based sessions.

SIP features:

The SIP communications protocol determines five attributes when establishing and terminating multimedia sessions:

* User location
* User availability
* User capabilities
* Session setup
* Session management

SIP sessions can include internet telephony, video conferencing and other forms of unified communications. The protocol can be used to invite participants to unicast or multicast sessions that do not necessarily involve the initiator.

SIP does not provide communication services. Instead, it defines interoperable implementations of SIP features, called primitives, which are used to facilitate different services. Primitives enable additional information to be embedded in a SIP message, such as linking a user's photo to directory information to enhance the user's caller ID.

SIP also supports name mapping and redirection services, which are two ways the protocol enables mobility. Users and endpoints are detected with a single identifier, or Uniform Resource Identifier (URI), which is independent of their network location. URIs are alphanumeric, using a syntax that looks more like an email address than a phone number or IP address. Other SIP features are available through application programming interfaces.

In addition to real-time services, SIP is used for asynchronous event notifications, such as automatic callbacks, message-waiting indicators and buddy lists based on presence.

How does SIP protocol work?

SIP operates similarly to, and incorporates parts of, Hypertext Transfer Protocol (HTTP) and Simple Mail Transfer Protocol (SMTP). Like HTTP or SMTP, SIP works in the application layer of the Open Systems Interconnection communications model. It is supported by IPv4 and IPv6.

SIP can be thought of as a client-server architecture. SIP will also work in tandem with other protocols, namely Session Description Protocol (SDP), which is contained in SIP messages. SDP is used to describe multimedia communication to sessions for invitations, announcements and parameter negotiations.

Also, SIP is a text-based protocol, like HTTP, which means its content is in a readable format. This makes SIP easier to read and debug compared with similar signaling protocols, like H.323.

SIP is a request-response protocol. Requests and responses are the names message protocols send between devices to communicate. SIP receives requests from clients and responses from servers. Requests can be sent through any transport protocol, such as User Datagram Protocol, Stream Control Transmission Protocol or Transmission Control Protocol.

Devices using SIP communicate with each other directly via a SIP proxy server. The proxy acts as an intermediary system to offload tasks that would otherwise be handled by SIP.

SIP determines the endpoint used for a session, the communication media and media parameters, and whether the called party agrees to communicate. Then, SIP establishes call parameters at either end of the communication, also handling call transfer and termination.

**RTP**

The Real-time Transport Protocol (RTP) is a network protocol for delivering audio and video over IP networks. RTP is used in communication and entertainment systems that involve streaming media, such as telephony, video teleconference applications including WebRTC, television services and web-based push-to-talk features.

RTP typically runs over User Datagram Protocol (UDP). RTP is used in conjunction with the RTP Control Protocol (RTCP). While RTP carries the media streams (e.g., audio and video), RTCP is used to monitor transmission statistics and quality of service (QoS) and aids synchronization of multiple streams. RTP is one of the technical foundations of Voice over IP and in this context is often used in conjunction with a signaling protocol such as the Session Initiation Protocol (SIP) which establishes connections across the network.

RTP was developed by the Audio-Video Transport Working Group of the Internet Engineering Task Force (IETF) and first published in 1996 as RFC 1889 which was then superseded by RFC 3550 in 2003.

RTP is designed to work in conjunction with the auxiliary control protocol RTCP to get feedback on quality of data transmission and information about participants in the on-going session.

Development:

Attempts to send voice over networks began in early 70's. Several patents on packet transmission of speech, time stamp and sequence numbering were granted in 70's and 80's. In 1991, a series of voice experiments were completed on DARTnet. In August 1991, the Network Research Groupof Lawrence Berkeley National Laboratoryreleased an audio conference tool vatfor DARTnet use. The protocol used was referred later as RTP version 0.

In December 1992 Henning Schulzrinne, GMD Berlin, published RTP version 1. It went through several states of Internet Drafts and was finally approved as an Proposed Standard on November 22, 1995 by the IESG. This version was called RTP version 2 and was published as

* RFC 1889, RTP: A Transport Protocol for Real-Time Applications
* RFC 1890, RTP Profile for Audio and Video Conferences with Minimal Control

On January 31, 1996, Netscapeannounced "Netscape LiveMedia" based on RTP and other standards. Microsoft claims that their NetMeeting Conferencing Software supports RTP. The latest extensions have been made by an industry alliance around Netscape Inc., who uses RTP as the basis of the Real Time Streaming Protocol RTSP.

How does RTP work?

As discussed in the first section, Internt is a shared datagram network. Packets sent on the Internet have unpredictable delay and jitter. But multimedia applications require appropriate timing in data transmission and playing back. RTP provides timestamping, sequence numbering, and other mechanisms to take care of the timing issues. Through these mechanisms, RTP provides end-to-end transport for real-time data over datagram network.

Timestamping is the most important information for real-time applications. The sender sets the timestamp according to the instant the first octet in the packet was sampled. Timestamps increase by the time covered by a packet. After receiving the data packets, the receiver uses the timestamp to reconstruct the original timing in order to play out the data in correct rate. Timestamp is also used to synchronize different streams with timing properties, such as audio and video data in MPEG. However, RTP itself is not responsible for the synchronization. This has to be done in the application level.

UDP does not deliver packets in timely order, so sequence numbers are used to place the incoming data packets in the correct order. They are also used for packet loss detection. Notice that in some video format, when a video frame is split into several RTP packets, all of them can have the same timestamp. So just timestamp is not enough to put the packets in order.

The payload type identifier specifies the payload format as well as the encoding/compression schemes. From this payload type identifier, the receiving application knows how to interpret and play out the payload data. Default payload types are defined in RFC 1890. Example specifications include PCM, MPEG1/MPEG2 audio and video, JPEG video, Sun CellB video, H.261 video streams, et al. More payload types can be added by providing a profile and payload format specification. At any given time of transmission, an RTP sender can only send one type of payload, although the payload type may change during transmission, for example, to adjust to network congestion.

Another function is source identification.It allows the receiving application to know where the data is coming from. For example, in an audio conference, from the source identifier a user could tell who is talking.

**Real-time Transport Control Protocol (RTCP)**

Real-time Transport Control Protocol (RTCP) is a protocol used in conjunction with the Real-time Transport Protocol (RTP) to monitor and control the quality of service (QoS) of multimedia transmissions over a network. It is responsible for providing feedback to the sender about the quality of the transmission, and also for managing the transmission statistics.

RTCP works by sending periodic control packets, called RTCP packets, from the receiver back to the sender. These packets contain information about the quality of the received data, including packet loss, delay, jitter, and bandwidth. The sender can then use this information to adjust the transmission parameters to optimize the quality of the transmission.

RTCP is the control protocol designed to work in conjunction with RTP. It is standardized in RFC 1889 and 1890. In an RTP session, participants periodically send RTCP packets to convey feedback on quality of data delivery and information of membership. RFC 1889 defines five RTCP packet types to carry control information. These five types are:

* RR: receiver report. Receiver reports are generated by participants that are not active senders. They contain reception quality feedback about data delivery, including the highest packets number received, the number of packets lost, inter-arrival jitter, and timestamps to calculate the round-trip delay between the sender and the receiver.
* SR: sender report. Sender reports are generated by active senders. In addition to the reception quality feedback as in RR, they contain a sender information section, providing information on inter-media synchronization, cumulative packet counters, and number of bytes sent.
* SDES: source description items. They contains information to describe the sources.
* BYE: indicates end of participation.
* APP: application specific functions. It is now intended for experimental use as new applications and new features are developed.

Through these control information packets, RTCP provides the following services

QoS monitoring and congestion control

This is the primary function of RTCP. RTCP provides feedback to an application about the quality of data distribution. The control information is useful to the senders, the receivers and third-party monitors. The sender can adjust its transmission based on the receiver report feedback. The receivers can determine whether a congestion is local, regional or global. Network managers can evaluate the network performance for multicast distribution.

Source identification

In RTP data packets, sources are identified by randomly generated 32-bit identifiers. These identifiers are not convenient for human users. RTCP SDES (source description) packets contain textual information called canonical names as globally unique identifiers of the session participants. It may include user's name, telephone number, email address and other information.

Inter-media synchronization

RTCP sender reports contain an indication of real time and the corresponding RTP timestamp. This can be used in inter-media synchronization like lip synchronization in video.

control information scaling

RTCP packets are sent periodically among participants. When the number of participants increases, it is necessary to balance between getting up-to-date control information and limiting the control traffic. In order to scale up to large multicast groups, RTCP has to prevent the control traffic from overwhelming network resources. RTP limits the control traffic to at most 5% of the overall session traffic. This is enforced by adjusting the RTCP generating rate according to the number of participants.

RTP features

RTP provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. But RTP itself does not provide any mechanism to ensure timely delivery. It needs support from lower layers that actually have control over resources in switches and routers. RTP depends on RSVP to reserve resources and to provide the requested quality of service.

RTP doesn't assume anything about the underlying network, except that it provides framing. RTP is typically run on the top of UDP to make use of its multiplexing and checksum service, but efforts have been made to make RTP compatible with other transport protocols, such as ATM AAL5 and IPv6.

Unlike usual data transmission, RTP does not offer any form of reliability or flow/congestion control. It provides timestamps, sequence numbers as hooks for adding reliability and flow/congestion control, but how to implement is totally left to the application.

RTP is a protocol framework that is deliberately not complete. It is open to new payload formats and new multimedia software. By adding new profile and payload format specifications, one can tailor RTP to new data formats and new applications.

RTP/RTCP provides functionality and control mechanisms necessary for carrying real-time content. But RTP/RTCP itself is not responsible for the higher-level tasks like assembly and synchronization. These have to be done at application level.

The flow and congestion control information of RTP is provided by RTCP sender and receiver reports.Real-Time Streaming Protocol (RTSP) is a network control protocol designed for use in entertainment and communication systems to control streaming media servers. It was developed by the Internet Engineering Task Force (IETF) as a standard protocol for streaming audio, video, and other multimedia content over the internet.

**RTSP**

The primary function of RTSP is to facilitate communication between the client and the server for the delivery of real-time data, such as audio and video streams. RTSP operates in a client-server model, with the client sending requests to the server and the server sending responses. The RTSP protocol is designed to work on top of the User Datagram Protocol (UDP) or Transmission Control Protocol (TCP) to ensure reliable and efficient data transfer.

The protocol works by dividing the multimedia content into streams and assigning a unique URL to each stream. The client can then request a specific stream using its URL, and the server will deliver that stream to the client. The RTSP protocol also supports features such as play, pause, and stop, which allow the client to control the playback of the stream.

RTSP supports various video and audio codecs, including H.264, MPEG-4, and AAC. It can also handle different container formats, such as MPEG Transport Stream (MPEG-TS) and Real-time Transport Protocol (RTP).

One of the significant advantages of RTSP is its ability to handle live streaming events. The protocol can deliver real-time data from the server to the client with minimal latency, making it ideal for live video broadcasts and teleconferencing applications. Additionally, RTSP can handle multiple simultaneous streams, making it possible to deliver multiple video and audio feeds to different clients simultaneously.

RTSP operations and methods

RTSP establishes and controls streams of continuous audio and video media between the media servers and the clients. A media server provides playback or recording services for the media streams while a client requests continuous media data from the media server. RTSP is the "network remote control" between the server and the client. It provides the following operations:

Retrieval of media from media server: The client can request a presentation description, and ask the server to setup a session to send the requested data.

Invitation of a media server to a conference: The media server can be invited to the conference to play back media or to record a presentation.

Adding media to an existing presentation: The server or the client can notify each other about any additional media becoming available.

RTSP aims to provide the same services on streamed audio and video just as HTTP does for text and graphics. It is designed intentionally to have similar syntax and operations so that most extension mechanisms to HTTP can be added to RTSP.

In RTSP, each presentation and media stream is identified by an RTSP URL. The overall presentation and the properties of the media are defined in a presentation description file, which may include the encoding, language, RTSP URLs, destination address, port, and other parameters. The presentation description file can be obtained by the client using HTTP, email or other means.

But RTSP differs from HTTP in several aspects. First, while HTTP is a stateless protocol, an RTSP server has to maintain "session states" in order to correlate RTSP requests with a stream. Second, HTTP is basically an asymmetric protocol where the client issues requests and the server responds, but in RTSP both the media server and the client can issue requests. For example the server can issue an request to set playing back parameters of a stream.

In the current version, the services and operations are supported through the following methods:

* OPTIONS:The client or the server tells the other party the options it can accept.
* DESCRIBE: The client retrieves the description of a presentation or media object identified by the request URL from the server.
* ANNOUNCE: When sent from client to server, ANNOUNCE posts the description of a presentation or media object identified by the request URL to a server. When sent from server to client, ANNOUNCE updates the session description in real-time.
* SETUP: The client asks the server to allocate resources for a stream and start an RTSP session.
* PLAY: The client asks the server to start sending data on a stream allocated via SETUP.
* PAUSE: The client temporarily halts the stream delivery without freeing server resources.
* TEARDOWN: The client asks the server to stop delivery of the specified stream and free the resources associated with it.
* GET\_PARAMETER: Retrieves the value of a parameter of a presentation or a stream specified in the URI.
* SET\_PARAMETER: Sets the value of a parameter for a presentation or stream specified by the URI.
* REDIRECT: The server informs the clients that it must connect to another server location. The mandatory location header indicates the URL the client should connect to.
* RECORD: The client initiates recording a range of media data according to the presentation description.

Note that some of these methods can be sent either from the server to the client or from the client to the server, but others can only be sent in one direction. Not all these methods are necessary in a fully functional server. For example, a media server with live feeds may not support the PAUSE method.

RTSP requests are usually sent on a channel independent of the data channel. They can be transmitted in persistent transport connections, or as a one connection per request/response transaction, or in connectionless mode.

RTSP features

* RTSP is an application level protocol with syntax and operations similar to HTTP, but works for audio and video. It uses URLs like those in HTTP.
* An RTSP server needs to maintain states, using SETUP, TEARDOWN and other methods.
* RTSP messages are be carried out-of-band. The protocol for RTSP may be different from the data delivery protocol.
* Unlike HTTP, in RTSP both servers and clients can issue requests.
* RTSP is implemented on multiple operating system platforms, it allows interoperability between clients and servers from different manufacturers.

**RSVP --- Resource Reservation Protocol**

RSVP is the network control protocol that allows data receiver to request a special end-to-end quality of service for its data flows. Real-time applications use RSVP to reserve necessary resources at routers along the transmission paths so that the requested bandwidth can be available when the transmission actually takes place. RSVP is a main component of the future Integrated Services Internet which can provide both best-effort and real-time service.

Development

RSVP design has been a joint effort of Xerox Corp.'s Palo Alto Research Center (PARC), MIT, and Information Sciences Institute of University of California (ISI). The RSVP specification was submitted to the Internet Engineering Steering Group (IESG)for consideration as a Proposed RFC in November 1994. In September 1997, RSVP Version 1 Functional Specificationand several other related internet drafts were approved as Proposed Standards They are:

* RFC 2205, Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification
* RFC 2206, RSVP Management Information Base using SMIv2 (RFC 2206)
* RFC 2207, RSVP Extensions for IPSEC Data Flows
* RFC 2208, RSVP Version 1 Applicability Statement Some Guidelines on Deployment
* RFC 2209, RSVP Version 1 Message Processing Rules
* The RSVP working group of the IETF is developing other protocols to be used with RSVP.

How does RSVP work?

RSVP is used to set up reservations for network resources. When an application in a host (the data stream receiver) requests a specific quality of service (QoS) for its data stream, it uses RSVP to deliver its request to routers along the data stream paths. RSVP is responsible for the negotiation of connection parameters with these routers. If the reservation is setup, RSVP is also responsible for maintaining router and host states to provide the requested service.

Each node capable of resource reservation has several local procedures for reservation setup and enforcement (see Figure 1). Policy control determines whether the user has administrative permission to make the reservation. In the future, authentication, access control and accounting for reservation will also be implemented by policy control. Admission controlkeeps track of the system resources and determines whether the node has sufficient resources to supply the requested QoS.

The RSVP daemon checks with both procedures. If either check fails, the RSVP program returns an error notification to the application that originated the request. If both checks succeed, the RSVP daemon sets parameters in the packet classifier and packet scheduler to obtain the requested QoS. The packet classifierdetermines the QoS class for each packet and the packet schedulerorders packet transmission to achieve the promised QoS for each stream.

RSVP daemon also communicate with the routing process to determine the path to send its reservation requests and to handle changing memberships and routes.

This reservation procedure is repeated at routers along the reverse data stream path until the reservation merges with another reservation for the same source stream.

Reservations are implemented through two types of RSVP messages: PATH and RESV. The PATH messages are sent periodically from the sender to the multicast address. A PATH message contains flow spec to describe sender template (data format, source address, source port) and traffic characteristics. This information is used by receivers to find the reverse path to the sender and to determine what resources should be reserved. Receivers must join the multicast group in order to receive PATH messages.

RESV messages are generated by the receivers and contains reservation parameters including flow spec and filter spec. The filter spec defines what packets in the flow should be used by the packet classifier. The flow spec is used in packet scheduler and its content depends on the service. RESV messages follow the exact reverse path of PATH messages, setting up reservations for one or more senders at every node.

The reservation states RSVP builds at the routers are soft states. The RSVP daemon needs to send refresh messages periodically to maintain the reservation states. The absence of refresh message within a certain time will destroy the reservation state. By using soft states, RSVP can easily handle changing memberships and routes.

The reservation requests are initiated by the receivers. They do not need to travel all the way to the source of the sender. In stead, it travels upstream until it meets another reservation request for the same source stream, then merges with that reservation.

This reservation merging leads to the primary advantage of RSVP: scalability---a large number of users can be added to a multicast group without increasing the data traffic significantly. So RSVP can scale to large multicast groups and the average protocol overhead decreases as the number of participants increases.

The reservation process does not actually transmit the data and provide the requested quality of service. But through reservation, RSVP guarantees the network resources are available when the transmission actually takes place.

Although RSVP sits on top of IP in the protocol stack, it is not a routing protocol, but rather an internet control protocol. Actually, RSVP relies on the underlying routing protocols to find where it should deliver the reservation requests. RSVP is also intended to cooperate with unicast and multicast routing protocols. When the RSVP-managed flow changes its path, the routing module will notify the RSVP module of the route changes. Therefore, RSVP can quickly adjust the resource reservation to new routes.

The delivery of reservation parameters is different from the determination of these parameters. How to set the connection parameters to achieve the requested QoS is the task of QoS control devices, the role of RSVP is just a general facility to distribute these parameters. Since different applications may have different QoS control devices, RSVP is designed to treat these QoS parameters as opaque data to be delivered to and interpreted by the control modules at the routers. This logical separation of QoS control devices and distribution facility simplifies the design of RSVP and makes it more adaptive to new network technologies and applications.

RSVP Features

* RSVP flows are simplex.
* RSVP distinguishes senders and receivers. Although in many cases, a host can act both as a sender and as a receiver, one RSVP reservation only reserves resources for data streams in one direction.
* RSVP supports both multicast and unicast, and adapts to changing memberships and routes.
* RSVP is designed for both multicast and unicast. Since the reservations are initiated by the receivers and the reservation states are soft, RSVP can easily handle changing memberships and routes. A host can send IGMP (Internet Group Management Protocol) messages to join a multicast group. Reservation merging enables RSVP to scale to large multicast groups without causing heavy overhead for the sender.
* RSVP is receiver-oriented and handles heterogeneous receivers.
* In heterogeneous multicast groups, receivers have different capacities and levels of QoS. The receiver oriented RSVP reservation requests facilitate the handling of heterogeous multicast groups. Receivers are responsible for choosing its own level of QoS, initiating the reservation and keeping it active as long as it wants. The senders divide traffic in several flows, each is a separate RSVP flow with different level of QoS. Each RSVP flow is homogeneous and receivers can choose to join one or more flows. This approach makes it possible for heterogeneous receivers to request different QoS tailored to their particular capacities and requirements.
* RSVP has good compatibility.
* Efforts have been made to run RSVP over both IPv4 and IPv6. It provides opaque transport of traffic control and policy control messages in order to be more adaptive to new technologies. It also provides transparent operation through non-supporting regions.

**Dynamic Adaptive Streaming over HTTP (DASH)**

Dynamic Adaptive Streaming over HTTP (DASH), also known as MPEG-DASH, is an adaptive bitrate streaming technique that enables high quality streaming of media content over the Internet delivered from conventional HTTP web servers. Similar to Apple's HTTP Live Streaming (HLS) solution, MPEG-DASH works by breaking the content into a sequence of small segments, which are served over HTTP. An early HTTP web server based streaming system called SProxy was developed and deployed in the Hewlett Packard Laboratories in 2006.It showed how to use HTTP range requests to break the content into small segments. SProxy shows the effectiveness of segment based streaming, gaining best Internet penetration due to the wide deployment of firewalls, and reducing the unnecessary traffic transmission if a user chooses to terminate the streaming session earlier before reaching the end. Each segment contains a short interval of playback time of content that is potentially many hours in duration, such as a movie or the live broadcast of a sport event. The content is made available at a variety of different bit rates, i.e., alternative segments encoded at different bit rates covering aligned short intervals of playback time. While the content is being played back by an MPEG-DASH client, the client uses a bit rate adaptation (ABR) algorithm to automatically select the segment with the highest bit rate possible that can be downloaded in time for playback without causing stalls or re-buffering events in the playback.The current MPEG-DASH reference client dash.js offers both buffer-based and hybrid bit rate adaptation algorithms. Thus, an MPEG-DASH client can seamlessly adapt to changing network conditions and provide high quality playback with few stalls or re-buffering events.

MPEG-DASH is the first adaptive bit-rate HTTP-based streaming solution that is an international standard.MPEG-DASH should not be confused with a transport protocol — the transport protocol that MPEG-DASH uses is TCP. MPEG-DASH uses existing HTTP web server infrastructure that is used for delivery of essentially all World Wide Web content. It allows devices like Internet-connected televisions, TV set-top boxes, desktop computers, smartphones, tablets, etc. to consume multimedia content (video, TV, radio, etc.) delivered via the Internet, coping with variable Internet receiving conditions. Standardizing an adaptive streaming solution is meant to provide confidence to the market that the solution can be adopted for universal deployment, compared to similar but more proprietary solutions like Smooth Streaming by Microsoft, or HDS by Adobe. Unlike HDS, or Smooth Streaming, DASH is codec-agnostic, which means it can use content encoded with any coding format, such as H.265, H.264, VP9, etc.

Differences:

* SIP - Session Initiation Protocol - A protocol that is used to initiate the session and exchange call parameters, QoS etc., required to set-up the call. The packets are encapsulated using UDP transport layer protocol.
* RTP - The media (audio/video) itself is carried by Real-time Transport Protocol. This can be considered as transport for VoIP packets. Not to be confused with UDP transport headers.
* RTCP - RTP Control Protocol, also referred as Real-time Transport Control Protocol works together with RTP and is a control protocol to monitor the media (quality) in RTP session
* RSVP - This comes in the context of QoS and is a Resource ReSerVation Protocol, a network control protocol that allows receiver to request a special treatment for data transmission.
* RTSP - Can be considered a competitor to SIP. Real-time Streaming Protocol is yet another signalling protocol like SIP but more suited in client-server applications while SIP is primarily designed for peer-to-peer calling.
* VoIP - It is an umbrella protocol that encompasses several different technologies to transmit Voice over Internet Protocol aka Voice over Internet/Network. SIP is one such technology.

Other Internet Protocols:

### **TCP/IP:**

TCP/IP stands for Transmission Control Protocol/Internet Protocol and is a suite of communication protocols used to interconnect network devices on the internet. TCP/IP is also used as a communications protocol in a private computer network (an intranet or extranet).

# **User Datagram Protocol (UDP):**

User Datagram Protocol (UDP) is a Transport Layer protocol. UDP is a part of the Internet Protocol suite, referred to as UDP/IP suite. Unlike TCP, it is an unreliable and connectionless protocol. So, there is no need to establish a connection prior to data transfer. The UDP helps to establish low-latency and loss-tolerating connections establish over the network.The UDP enables process to process communication

**Codecs**

Codecs are the oxygen of the streaming media market; no codecs, no streaming media. From shooting video to editing to encoding our streaming media files for delivery, codecs are involved every step of the way. Many video producers also touch the DVD-ROM and Blu-ray markets, as well as broadcast, and codecs play a role there as well.

First we’ll cover the basics regarding how codecs work, then we’ll examine the different roles performed by various codecs. Next we’ll examine how H.264 became the most widely used video codec today, and finish with a quick discussion of audio codecs.

Basics of codecs:

Codecs are compression technologies and have two components, an encoder to compress the files, and a decoder to decompress. There are codecs for data (PKZIP), still images (JPEG, GIF, PNG), audio (MP3, AAC) and video (Cinepak, MPEG-2, H.264, VP8)

There are two kinds of codecs; lossless, and lossy. Lossless codecs, like PKZIP or PNG, reproduce the same exact file as the original upon decompression. There are some lossless video codecs, including the Apple Animation codec and Lagarith codec, but these can’t compress video to data rates low enough for streaming.

In contrast to lossless codecs, lossy codecs produce a facsimile of the original file upon decompression, but not the original file. Lossy codecs have one immutable trade-off–the lower the data rate, the less the decompressed file looks (or sounds) like the original. In other words, the more you compress, the more quality you lose.

Lossy compression technologies use two types of compression, intra-frame and inter-frame compression. Intra-frame compression is essentially still image compression applied to video, with each frame compressed without reference to any other. For example, Motion-JPEG uses only intra-frame compression, encoding each frame as a separate JPEG image. The DV codec also uses solely intra-frame compression, as does DVCPRO-HD, which essentially divides each HD frame into four SD DV blocks, all encoded solely via intra-frame compression.

In contrast, inter-frame compression uses redundancies between frames to compress video. For example, in a talking head scenario, much of the background remains static. Inter-frame techniques store the static background information once, then store only the changed information in subsequent frames. Inter-frame compression is much more efficient than inter-frame compression, so most codecs are optimized to search for and leverage redundant information between frames.

Early CD-ROM based codecs like Cinepak and Indeo used two types of frames for this operation: key frames and delta frames. Key frames stored the complete frame and were compressed only with intra-frame compression. During encoding, the pixels in delta frames were compared to pixels in previous frames, and redundant information was removed. The remaining data in each delta frame is also compressed using intra-frame techniques as necessary to meet the target data rate of the file.

Some common audio and video codecs include:

1)H.264/AVC

The most used codec and encoding output used today are H.264 files. It is also referred to as Advanced Video Coding (AVC). This we developed by Telecommunications Union and the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) Moving Picture Experts Group.

H.264 has become so popular because it isn’t just used for streaming; it is used for Blu-ray disks and cable broadcasting. This is more than likely due to who developed it.

Plus, H.264 often works together with AAC audio codec, and can work with .mp4, .mov, .F4v, .3GP, and .ts containers.

The thing that makes H.264 so popular is that it can be used on virtually any device – any browser, any device – can play H.264 videos. It is widely supported, and publishers know how to use it.

According to Bitmovin’s 2022 Video Developer report, over 91% of those who took the survey use H.264.

The downside to this is that it doesn’t work well with 4K video or with HDR content. It works well with low-latency streaming. It is commonly used with HTTP-based and WebRTC-based applications. This format has been around the longest and looks to stay that way for a while, as it can work with virtually any device.

2)H.265/HEVC

The SO/IEV Moving Picture Experts Group made H.265 a successor to H.264. It is also called High-Efficiency Video Coding.

The big difference from H.264 is that it is made to create smaller files, uses less bandwidth, and supports high-resolution streaming. Although this is designed as a successor to H.264, it is only used by about 10% of encoded files.

There has been some confusion about its royalties and what developers will have to pay for using the codec, which has stopped its widespread adaption. Royalty confusion actually helped spur the development of the AV1 codec.

It is a good codec to use if you want to deliver 4K or HDR content.

3)AV1

Major tech players are Amazon, Netflix, Cisco, Microsoft, Google, and Mozilla came together to form the Alliance for Open Media. They then created AV1, which is a royalty-free and open-source alternative.

It is designed to be more efficient than H.264 and H.265. It works well with high-quality content but has long encoding times, making it more expensive to use.

Audio Codecs:

Finally, since most video is also captured with audio, the audio component must also be addressed. The most widely used audio format for acquisition and editing is PCM, which stands for Pulse-code Modulation, which is usually stored in either WAV or AVI format on Windows, or AIFF or MOV on the Mac. PCM is considered uncompressed, so it may be more properly characterized as a file format, rather than a codec. To preserve quality, most intermediate codecs simply pass through the uncompressed audio as delivered by the camcorder.

Most delivery formats have an associated lossy audio codec, like MPEG audio and AC-3 Dolby Digital compression on DVDs. Most early streaming technologies, like RealVideo and Windows Media, had proprietary audio components, so RealAudio accompanied RealVideo files, as did Windows Media Audio with Windows Media Video.

This dynamic changed most prominently when Adobe paired the VP6 codec with the MP3 audio codec for Flash distribution. The standards-based audio codec for H.264 video is the Advanced Audio Coding (AAC) codec, while WebM pairs the VP8 codec with the open-source Vorbis codec.

Where to go from here:

Designing a media server is a challenging task that requires careful consideration of various factors. However, it is a task that can be accomplished with the right skills and resources. In order to compete with existing media servers in the market, it is essential to identify key technologies that are critical to providing high-quality streaming services. One such technology is Adaptive Bitrate Streaming (ABR).

ABR is a critical component of modern media servers and is designed to provide users with a seamless streaming experience, regardless of their network conditions. Traditional streaming methods often send a fixed bitrate stream to the client, resulting in buffering and stuttering if the user's network is congested or slow. In contrast, ABR technology dynamically adjusts the quality of the media stream based on the user's available bandwidth, resulting in smooth playback and minimal buffering.

Implementing ABR requires the media server to encode multiple versions of the same media file at different bitrates. These versions are typically encoded using the latest video codecs such as H.264, H.265, and VP9, and can be delivered using popular streaming protocols such as HTTP Live Streaming (HLS), Dynamic Adaptive Streaming over HTTP (DASH), or Microsoft Smooth Streaming.

The ABR technology also requires sophisticated algorithms to determine which bitrate stream to serve to the client based on network conditions. These algorithms may take into account the client's current bandwidth, the current network conditions, and the available server resources. The server should also be capable of responding quickly to changes in network conditions to ensure that the client receives the best possible streaming experience.

In conclusion, ABR technology is a critical component of modern media servers and provides users with a seamless streaming experience. To compete with existing media servers in the market, it is essential to implement ABR technology effectively and efficiently. This requires careful consideration of encoding, delivery protocols, and sophisticated algorithms that can dynamically adjust the quality of the media stream based on the user's available bandwidth.

**Conclusion:**

In conclusion, the development of media server technology has revolutionized the way we consume and distribute media. With the availability of high-speed internet, it has become increasingly important to have a reliable and efficient media server that can handle the demands of users across different devices and platforms. From the protocols used for data transfer to the adaptive bit rate streaming technology, every aspect of media server design plays a critical role in providing a seamless and enjoyable media experience. It is important for media server designers to stay up-to-date with the latest technologies and constantly innovate to provide the best possible service to users. With the continued growth of the digital media industry, the importance of media servers will only continue to increase in the coming years.