Video Streaming Platform

Some applications that use video streaming are YouTube, Netflix, Amazon Prime, Zoom, Play, Facebook video features, Hotstar, Instagram video features, Learning management system also has video streaming (LMS).

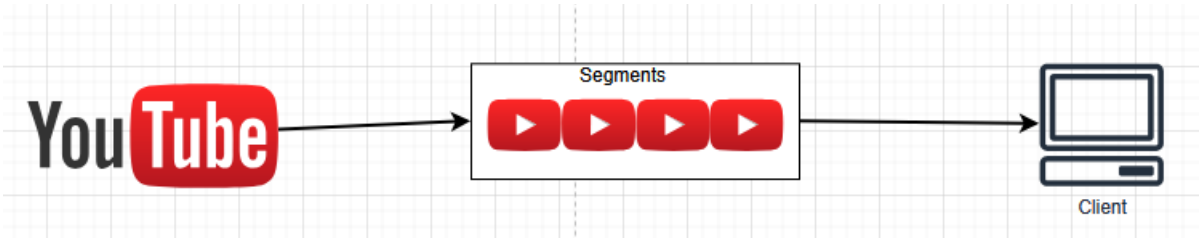
In all the applications, streaming plays a huge role. Before, we used to transfer files using Bluetooth or download video files. If transfer is interrupted then we see only part of the file content. Now we have video streaming and no video downloading like the old days. Point is, we had to download entire video before watching it, otherwise, we have to try over and over again.

Video is a sequential collection of images + audio
- 30 frame per second, 60 fps. It means we are dividing the second into 30 frames. 60Fps is going to be smoother and heavier than 30fps.

In older days, the size of the video used to be 1GB for a 480p but now a 5 min 4K video for YouTube is 1.5-2GB. A 5 min 4K video is consuming 1.5-2GB of data. A 1-1.5 hour video is going to take > 12GB size.
Lets say we have to download a 100Mb video file first before watching it, on the client machine. For a 100Mb download, lets say it is not going to take approximately 10 seconds.
What issues can happen during the 10seconds?
- Buffering
- Wait based on the size
- Video size can be too large

Lets say on YouTube, we are processing a video of 2Gb and we don't have any streaming application. In that case, we have to download the entire video before starting to see the video. It is not worth downloading the entire video beforehand. Lets say we want to watch only first 10seconds of the video, it is not *worth downloading the entire video* beforehand. We don't want waste of resources thats why we need Streaming.

What's Streaming?
Streaming comes with two new protocols: we already have TCP to handle sequential. Whatever we are sending will be in a sequence only. Two protocols: RTMP (Real-Time Messaging Protocol), RTSP (Real-Time Streaming Protocol). For Streaming, we cannot use HTTP.

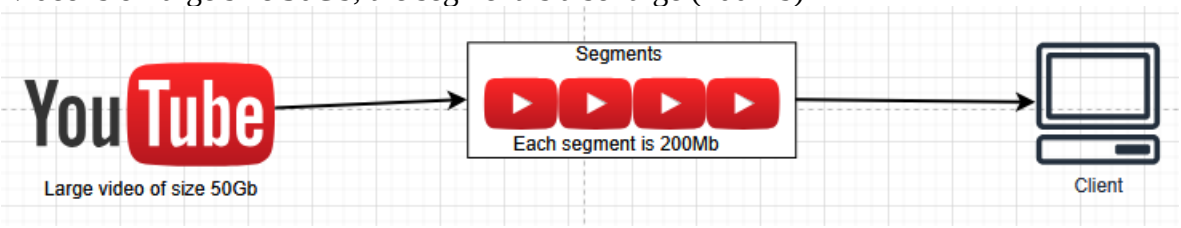
We 

cannot send the entire video in one go, we divide video into chunks (or segments). Our client picks one segment at a time while watching like Segment1, Segment2 and so on as they watch the video. We don't *download the entire video* to watch a video. First we download one segment then another segment later. Streaming allows breaking a large video into multiple chunks / segments and a particular segment can be pulled from the client depending upon network.

Advantages of Streaming:
- It solved Latency issue,
- Allows us to watch streaming live
- Efficiently we can use the bandwidth (we download the video once we are in the application, once we are done with the application, downloads also stop automatically. Download only when watching not other times.

When video is of large size 50Gb, the segment is also large (200Mb)

There are



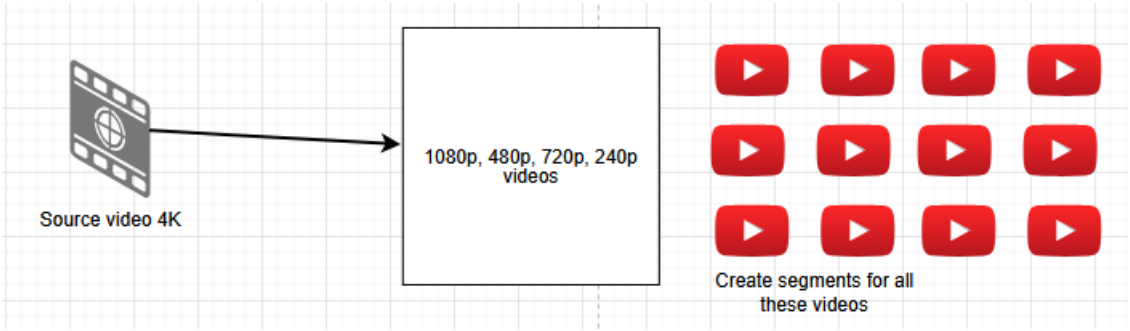
different types of devices with different screens:
- TV, Desktop, Laptop, Mobile, Fridge smartscreen, Watch, Tablets.

Not in all the different screens, we don't have to watch a video in 8K. Based on the type of screen or network, we can adjust the resolution of the video also. We have multiple resolutions: 8K, 4K, 1080p, 480p, 240p, 144p. If the network is slow, then we can change the resolution to 240p.

<u>Adaptive Bitrate Streaming:</u>
We are viewing any video on 300Mbps network and we are watching it on TV. What if network drops to 30Mbps. We have two options: 1. download the video still in 4K quality, using Buffering, 2. reduce the video quality or bitrate of streaming video, with no buffering. Once network speeds is back, again we can switch back to 4K. The video streaming platform will create different versions of the same video in different resolutions. Those videos will be chunked into multiple segments.

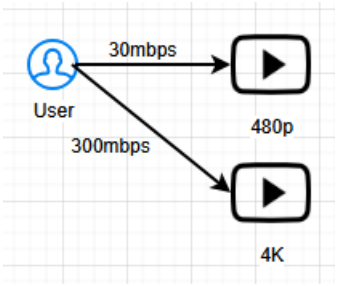
Source video in 4K ==> converted into 1080p, 480p, 720p, 240p videos.

Index file



maintains which segment do we need to show, which format, what should be the next segment to show and in what format, which screen we are watching the video.

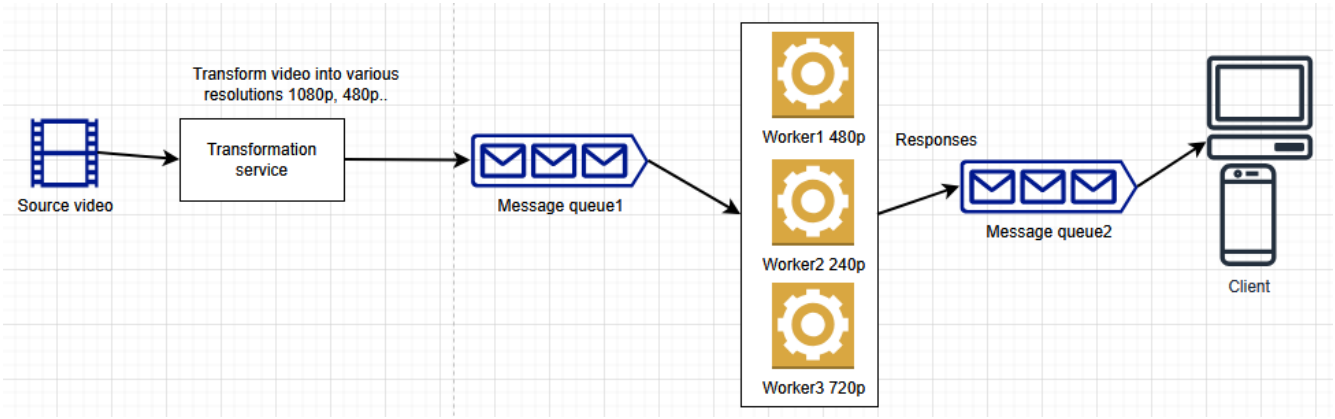
If the client network is very slow (30mbps), I give a 480p segment and if the speed improves (300mbps) after sometime, I give a 4K segment accordingly



We change the segment based on resolution.

It's a costly setup. If our network throughput better than bitrate of downloaded segment, we will pick a higher resolution segment. If our network throughput is lower then bitrate of downloaded segment, then we will go for lower resolution.

System design:
We take the source video, then transform into various resolutions. Worker is a background process that converts video into multiple resolution. Say Worker1 is for 480k, Worker2 is for 240p and so on.

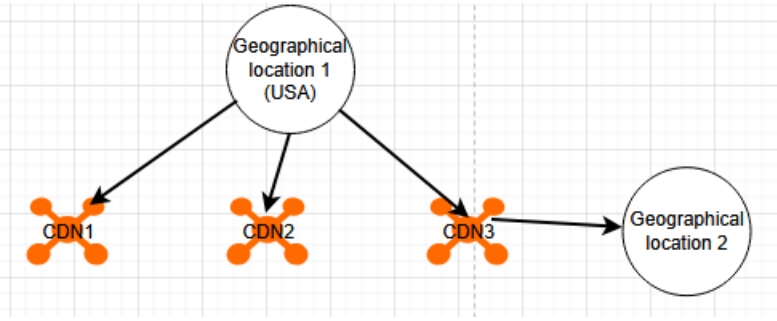


Source video goes through Transformation service then workers convert into different resolutions, then goes on message queue to client (any screen).

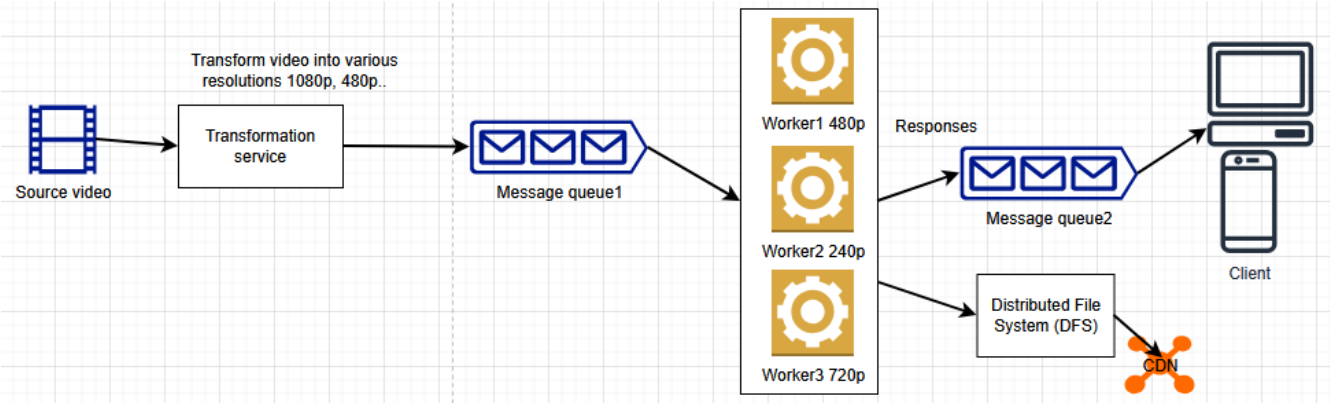
Sequence of video:
Sequence of the video segments are very important because we cant watch the final segment in the first. So the sequence is maintained by RTMP (Real-Time Messaging Protocol) and RTSP (Real-Time Streaming Protocol). Even the workers when the process the video they maintain the order as well. Message queue also maintains order in FIFO. It will also make sure no bit or data will be lost during the process. Two things are maintained: 1. sequence, 2. no data loss

Data storage:
Lets say we divide 50Gb video into 50Gb for 4K video segments + 20Gb for HD + 10Gb for 720p + 5Gb for 480p = 85Gb. Workers will store in a Distributed File System (DFS).

We need Distributed File System because not all can watch the video from the same Geographic location because that will add Latency. Thats why, we need CDNs (Content Delivery Network) closer to other geographic locations as well.



Some videos / movies are banned in some geographic locations, so that kind of things need to be taken care also.



Lets say we divide 50Gb video into 50Gb for 4K video segments + 20Gb for HD + 10Gb for 720p + 5Gb for 480p = 85Gb.
Number of users = 10,000
Lets say it is a 20min video. 20*60seconds = 1200seconds or 1200 segments of 1 second long
Size of 1 1080p segment = 1080p segment (20Gb / 1200segments) =
Size of 1 720p segment = 720p segment (10Gb / 1200segments) =
Size of 1 480p segment = 480p segment (5Gb / 1200segments) =
Size of 1 240p segment = 240p segment (2Gb / 1200segments) =

Lets say if size of 1 segment for all resolutions adds up to 500MB, that’s the storage we need for 1 user
For 1000 users = 500Mb * 1000 = 5,00,000Mb. We have to have in the CDNs

Homework:

When we move the cursor in a YouTube video, how are we getting some images while scrolling