

Internet Technology
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No #36
Streaming Multimedia Applications

In this lecture we would be talking about a very interesting application which we see quite frequently in the internet environment, namely streaming multimedia applications. Now I am sure many of you are familiar with some kind of streaming applications over the internet. In this lecture we will try to talk about the basic technology behind this kind of streaming applications and how we can guarantee or achieve the desired performance or the so called quality of service parameters as demanded by the various applications.

(Refer slide time: 01:16)



So streaming multimedia applications, this is the topic today.

(Refer slide time: 01:21)



So we start by talking something about multimedia networking. How multimedia can be transported over network?

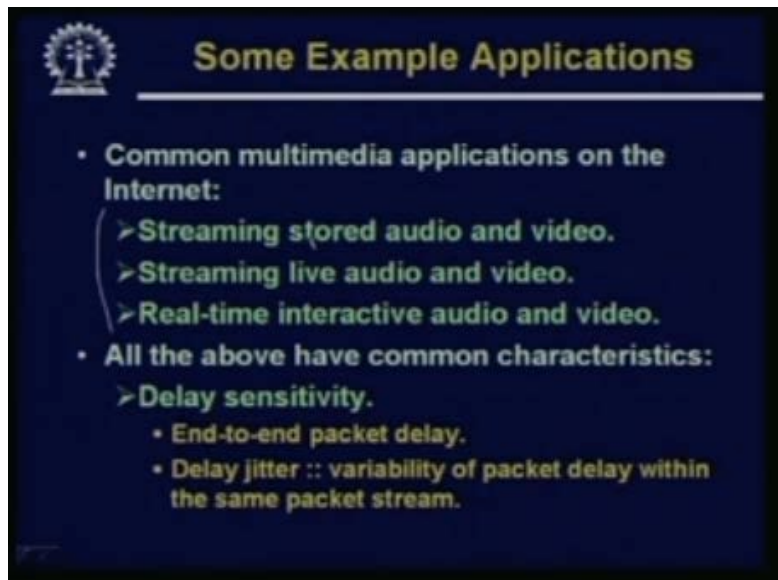
(Refer slide time: 01:30)



Now, first let us try to understand what do we mean by multimedia applications? Multimedia applications are some applications which deal with one or more of the following data types; text, images, audio, video. These are examples. Now some examples of such multimedia applications may be media player in which you are playing a video. Your browser is also a good example of multimedia application where you can have embedded video embedded audio, images, text, everything together. So your browser is a very powerful and flexible kind of a multimedia application. You say sometimes we will find that what the application in question. There are some kinds of

configurations you can make in order to improve the quality of the multimedia that is being showed or displayed or played on it. So we would basically try to address those points first. So as I have said this kind of multimedia information transmission is the most common scenario today. Whenever we browse the internet download a page they will invariably contain some multimedia content. So transmission processing and rendering of multimedia information over the network, this is the main challenge.

(Refer slide time: 03:06)

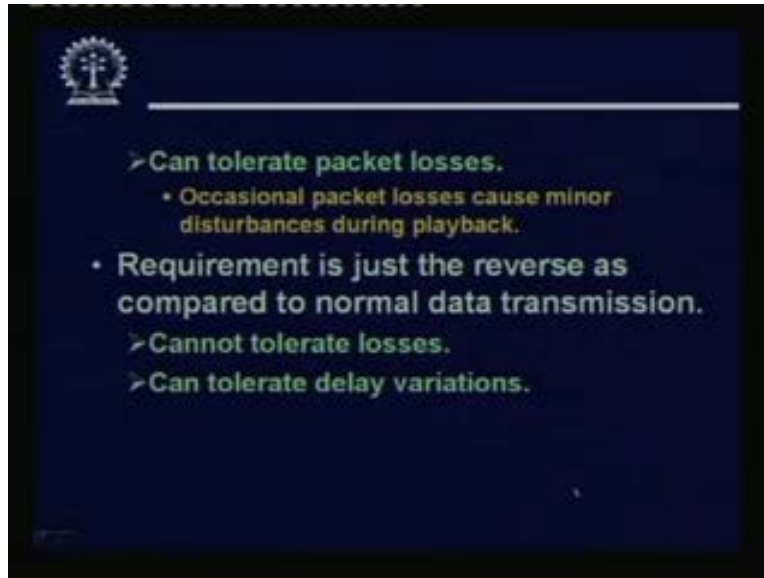


Some example applications are shown here. Some common multimedia applications seen by multimedia application I am not just telling that you are downloading a page and you are displaying it on a screen. I am talking something more demanding like you are downloading something which has a built in audio which has to be played it has a built in video clip which has to be played. These kinds of things I am talking about so the common applications seen in the internet scenario are streaming stored audio and video streaming live audio and video and real time interactive audio and video. Now we will be talking about the details of these three. But as the name implies the first one refers to something which is stored somewhere. Second one you are trying to broadcast something live and the third one there is an interactive nature of the communication. So these keywords actually signify the main differences in the way they will work.

But whichever application you talk of stored live or interactive whichever way all of the above share some common characteristics. Most important is the delay characteristics. Delay characteristics or the delay sensitivity says that the end to end packet delay. This end to end packet delay is important. Suppose I am playing a video clip or audio clip which I am downloading from the web server from somewhere else. If the end to end packet delay is very large or is variable then quality of playback will not be very good. So delay sensitivity one is of course packet delay. That is how long it takes to reach me and secondly variability of the delay. Well if all packets are equally delayed then possibly it will not be that much of a problem to me. But if the delay varies, then during playback I

may counter some breaks and jitters in between. This is called delay jitter. Jitters caused due to variability of packet delays.

(Refer slide time: 05:40)

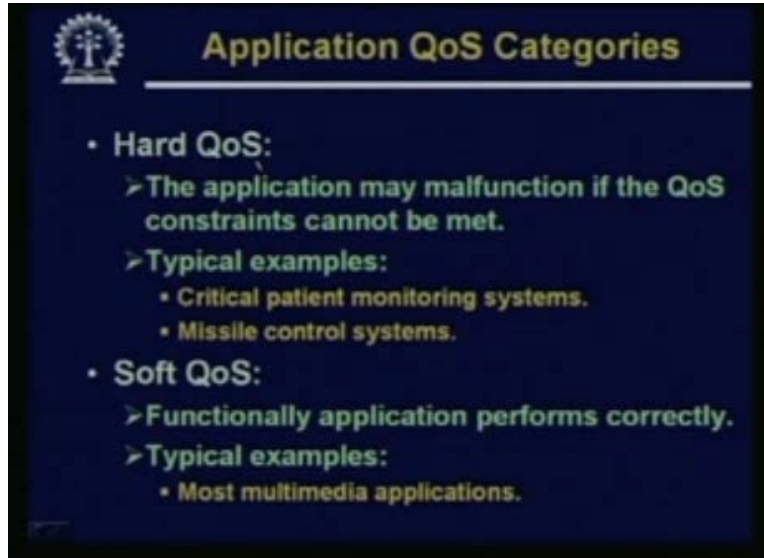


But this kind of applications, multimedia applications often can tolerate packet losses. Suppose when you are downloading a say a voice application a music streaming music you are downloading. The music packets are coming one after the other. You are playing suppose one packet in between is lost and if you do not take any special mechanism to recover that lost packet and straightaway going and playing whatever comes after that. Then with respect to the audio quality we will possibly feel a small glitch in between. Other than that you will not find any major cause of confusion or annoying.

Because there will be no gap in between. That means you need to wait for the correct packet to come back again you wait for that amount time. So those kinds of gaps or delays in between are typically not allowed. So those packets you can easily drop. These kinds of applications can tolerate few packet losses. As I said, small number of such losses can cause some small or minor disturbances during playback. If you look at the data transmission requirement when you transmit some data, transmit a file over the network, requirements are just the reverse there.

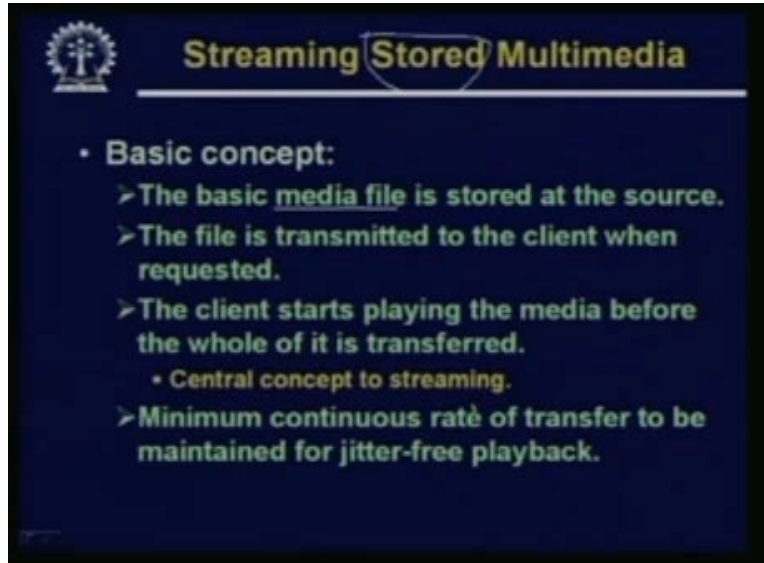
For data transmission you cannot tolerate any loss. Your data has to reach the other end without any loss. However here delay variations can be tolerated, so as you can see that the requirements are ultimately applications are orthogonally different from the requirements of a standard data applications. So and the point to notice that network as you see today that were built with the data applications in mind. But now that we have multimedia applications, we would have to do whatever we can do best on top of the network which was primarily built for data. This is something you should keep in mind.

(Refer slide time: 08:03)



Some quality of service categories with respect to the applications you can classify between hard QoS and soft QoS. For hard quality of service the application may malfunction if the constraints cannot be met. Some typical examples are critical patient monitoring systems or a missile control system. For a patient monitoring system it depends on what you are monitoring if you cannot take corrective action within a defined period of time then the life of the patient may be under threat. Similarly for a missile control system you know that you have certain defined period of time between the detection of the missile which is coming towards you and the retaliatory action to stop and intercept it because if you cannot act within that time possibly the missile will be hitting you. So these are some application where the quality of service parameters is very hard. But there are soft QoS even if it changes a little bit here and there. Users really do not care that much. Functionally applications will perform correctly. There will be some variations in the quality. Typical examples are multimedia application. So, multimedia applications usually fall under the soft QoS category.

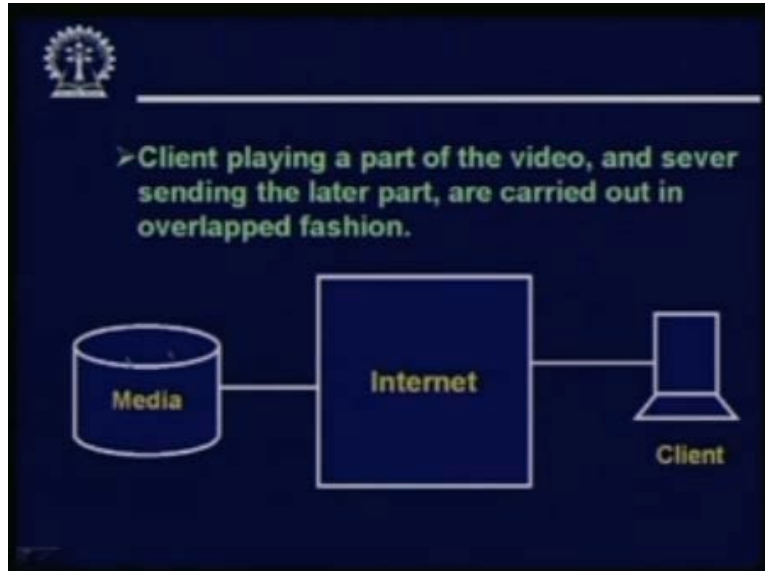
(Refer slide time: 09:41)



Now let us look at the three different kinds of streaming multimedia types we had talked about. The first one we talking about streaming stored multimedia. As I said that the multimedia is stored in some place. The basic concept is this. Here you are trying to place a media file in the network. The basic media file it can be audio, video is assumed to be stored at the source. When you want to play it over the network the file has to be transmitted to the client. The idea is like this. I am the client I want to play or view a video clip the video clip is stored somewhere in other server. I download the file from there. It is a stored video clip on that server. But the difference is that in a normal data download I first complete downloading my file then I start playing it.

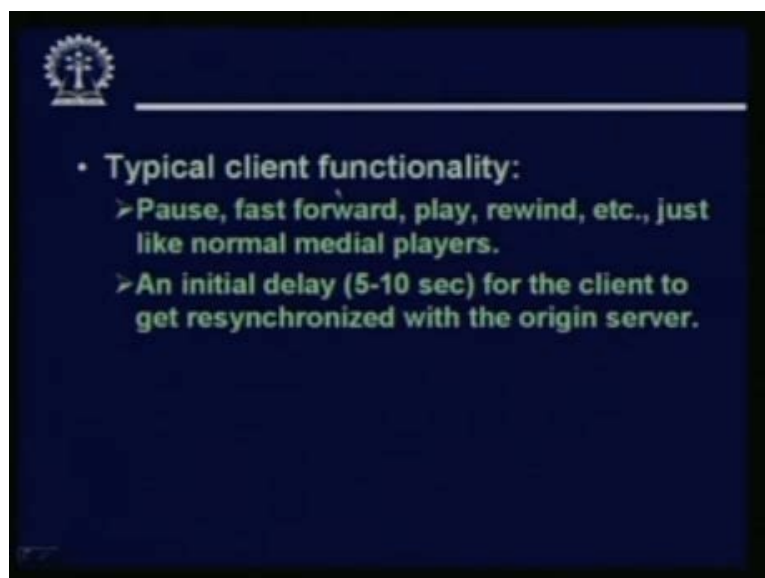
In contrast streaming multimedia the video starts get playing as soon as the first few packets of the video have started to arrive. So as the packets are coming I am continuously going on playing. This is something of a continuous playing as the data is coming mode. So that I am not waiting for the entire time for the file to get downloaded that time I am saving. This is the central concept of streaming. Client starts playing the media before the whole of it is transferred. But in order to have a good quality of playback minimum continuous rate of transfer has to be ensured. If you cannot ensure this, then there will be jitters in the playback. There will be disturbance there will be some gaps or delays.

(Refer slide time: 12:03)



So the picture looks like this for this mode of transmission. The client is out here the media is stored on some server out here and you have the internet in between. The client sends the request through the internet and the media will be sent or transmitted as data packets over the internet and as the packets come they get played. So the client is playing a part of the video and the server is sending the next part of the video. These two are carried out in an overlapping fashion. So as the client, I will get a part of the video file and start playing it and while this being played the remaining portion is coming continuously. So retrieving the data and playing back the part already come has already arrived. These two things are carried out in a overlap fashion, there is an overlap.

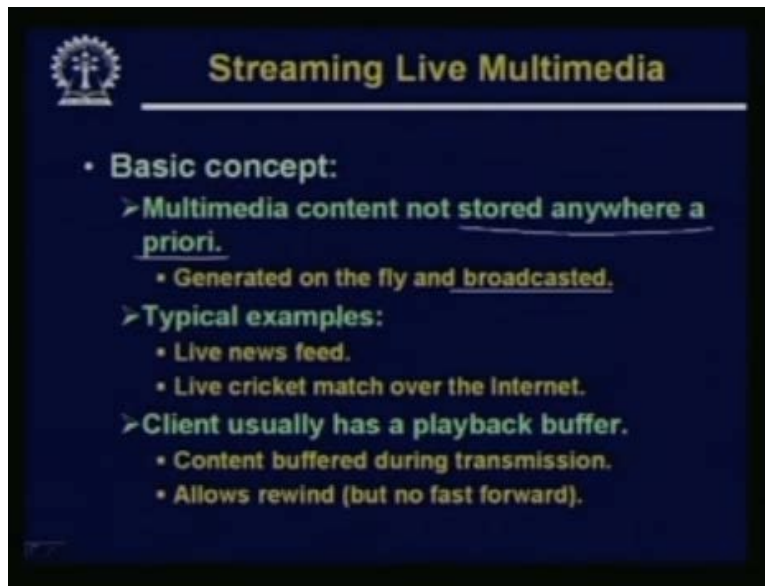
(Refer slide time: 13:04)



And in this mode the typical functionality that is supported by the client are pause. I can pause, playback, fast forward. Well fast forward, what you can do is that I can say that you start playing from one minute after minute. Well here you can implement it. But you will have to wait till that portion of the file has reached it before you can start the playback again. But you can have this fast forward feature. But there is some other way also you can send an explicit request to the server that you start sending from this off set onward. If that support you have you can start playing much earlier.

Play, rewind all these things are there. Play of course I have mentioned rewind is since the data as it is coming are getting stored you can also go back and play a portion of the clip again. So these facilities are available which are available in all normal media players. But however whenever you are doing a fast forward pause play some initial delay for the client to get resynchronized with the server. Because, the client will have to get to the point from where the playback has to start again. That initial delay to be there before the playback again starts.

(Refer slide time: 14:37)



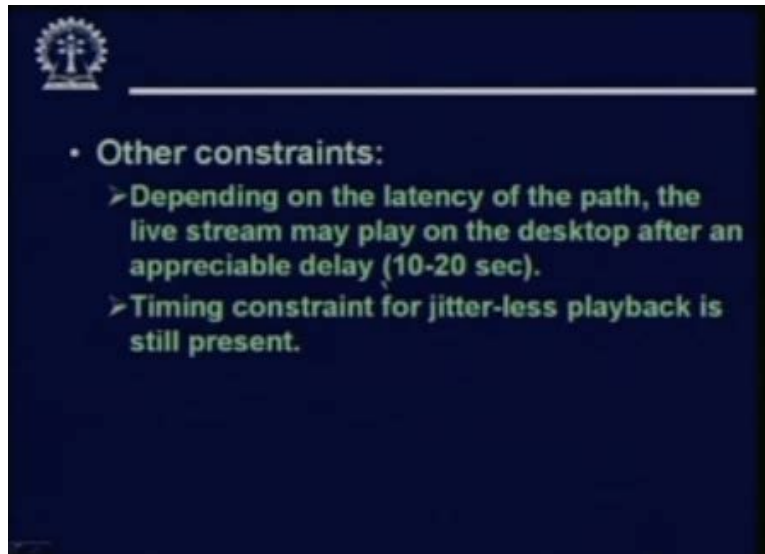
Streaming Live Multimedia

- **Basic concept:**
 - > **Multimedia content not stored anywhere a priori.**
 - **Generated on the fly and broadcasted.**
 - > **Typical examples:**
 - **Live news feed.**
 - **Live cricket match over the Internet.**
 - > **Client usually has a playback buffer.**
 - **Content buffered during transmission.**
 - **Allows rewind (but no fast forward).**

Next let us come to streaming live multimedia. Now in streaming live multimedia the concept is similar. But difference is that it is live which means the multimedia content we are trying to play it is not stored anywhere beforehand apriori. The multimedia content is generated on the fly and broadcast over the network. Some typical examples you might have seen. This example, this couple of examples I have shown here live news feed. There are many websites through which you can get online and live video clips that broadcast news or live cricket matches. These are something which is not prerecorded. These are coming to live. Cricket match is going on in the field. You can see it on your screen the cricket match is going on. These are examples of live streaming video feeds. So conceptually live and stored are similar in stored mechanism. The file is stored somewhere from there the file is getting transferred. In live there is a camera.

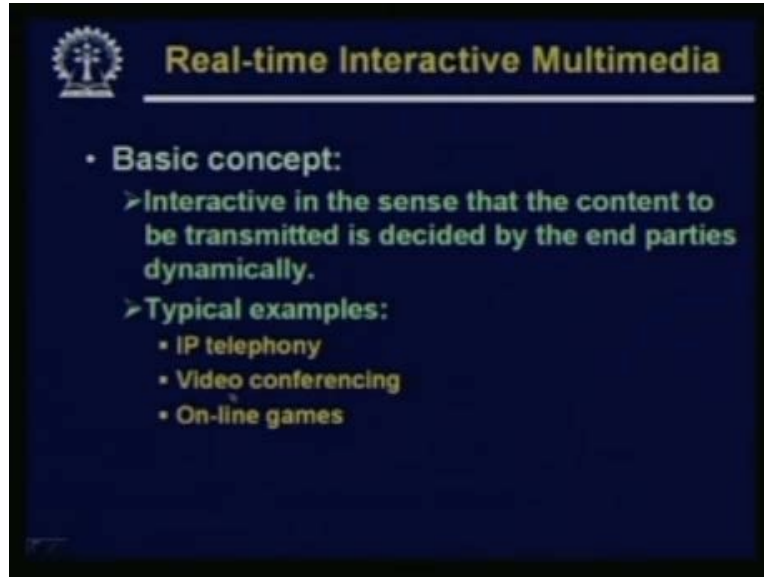
Suppose if we just if we talk about video there is a camera which is capturing the video images online and is digitizing them into packets online and sending it to the prospective customers or the viewer online. So the data is coming as there being generated after the initial delay, the delay it takes for the data to reach from the source to the destination after that the delay the video starts playing. Now the client here usually has a playback buffer because whatever comes is stored in a buffer the content is buffered. Here since we cannot move into the future, here you can have rewind. But now fast forward because the content of fast forward on a live feed does not make any sense. Because you are in the present and whatever you are seeing is present, you cannot go into the future because you only do a rewind because it is already stored on your computer. You can go back and see what was there. But you cannot do a fast forward.

(Refer slide time: 17:16)



And a few other constraints, depending on the latency of the path, the live stream may play on the desktop. After an initial delay this delay can be as large as 10 to 20 seconds. It depends on the delay of the path through which the data packets will come to the final destination. That delay can be appreciable, but that delay can be the same for all packets. So once it starts coming there will be a continuous flow of packets and the other constraints are still here. All the constraints for timing because there should not be too much variability in the delay of the packets to achieve jitter free playback all this are there.

(Refer slide time: 18:04)



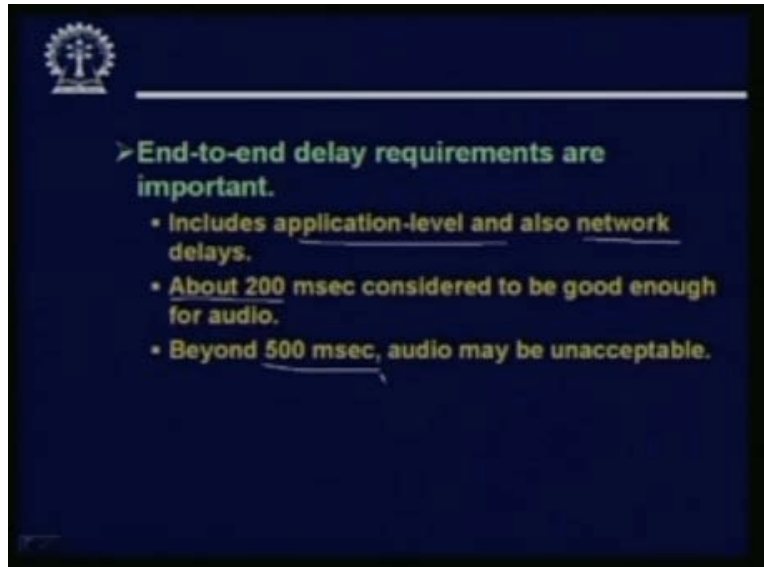
Third type I talked about, this is the real time interactive multimedia. So here the concept is like this. Here there is the concept of interactiveness. Interactive in the sense that the content to be transmitted is decided by the end parties dynamically. Some examples I am showing while just explaining with this IP telephony, video conferencing, online games. These three examples I have given. All these three examples, some kind of multimedia content are transferred to and fro between the two parties. For IP telephony I speak in my microphone the other person on the other end also speaks in his or her microphone. Data packets flow over the network. Now this is interactive because the way we normally talk about telephony after I finish talking, the other person will start talking we usually do not talk together.

So exactly when I should start talking will depend on exactly when the other person stops talking. So this is some kind of interactive. So if the other person continues to talk, does not stop I will not get chance to talk. So this is interactive in that means exactly when the data has to be sent it depends on some kind of interactive agreement or protocol. Talking about video conferencing the concept is the same. Suppose I have a video conferencing system through which I am conducting a class. Well I am sitting in a small studio with some controls with a computer in front of me. I can see a screen and there is a big screen on a big class room with a camera over there where all the students are sitting and watching.

Now there is a mechanism in which a student can ask a question and I can see it on my screen and I can control the camera from my controls and I can respond to that question. Depending on the question I will stop my lecture and respond to the answer. Now here you see it is a two way video transmission because its video conferencing that is going on. So when the transmission will go on when it will stop, when it will pause it depends again on some interactive agreement between the two parties and lastly for online games you can easily understand. Typically there are multi party online games which are played

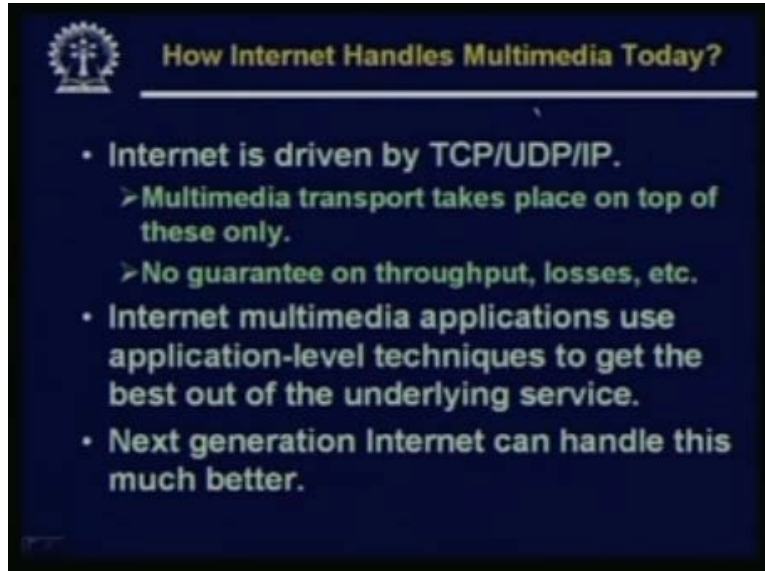
on the internet. Depending on your next move the other person will decide what to do next. So in many such games again some multimedia contents are being exchanged. These are some typical examples.

(Refer slide time: 21:09)



So for these kinds of real time applications the end to end delay requirements are very important because there are two parties who are interacting among themselves. And when you say delay this delay actually refers to the whole or the total delay. It will include the application level delay as well as the network delay. As a very rough rule of thumb if the delay is above 200 milliseconds it is considered to be good enough for interactive audio for making a call internet call. But if it is more than 500 milliseconds then it may be very annoying and the audio quality may be unacceptable. Quality means I talk the other person when he or she starts talking, there will be a gap of about half seconds before I can hear. So I really do not know whether the other person has started talking that I need to pause and wait till that voice comes back to me. So if there is a long delay because ultimately this latency is there whatever I am talking is going to the other end with that particular delay. If that delay is too much it may become quite annoying during the communication. So usually less than 200 millisecond is considered to be very good. 200 to 400 is considered to be acceptable.

(Refer slide time: 22:48)



More than that is usually considered to be unacceptable. How internet handles multimedia today? See before trying to address this question, there are a few issues you need to look at. Internet is driven by TCP, UDP and IP protocols. These are the protocols which drive the internet. All applications that you see they uses this protocols as the underlying support for transmission of the products. Multimedia transport can take only on top of these because these are only the transport systems available. In these protocols there is no guarantee on throughput, losses, etcetera which are important from the point of view of multimedia applications. Which means if there is some multimedia application sitting on top of TCP, UDP and IP multimedia application, whatever they want from the network this TCP, UDP, IP may not be able to provide that? So there is a gap.

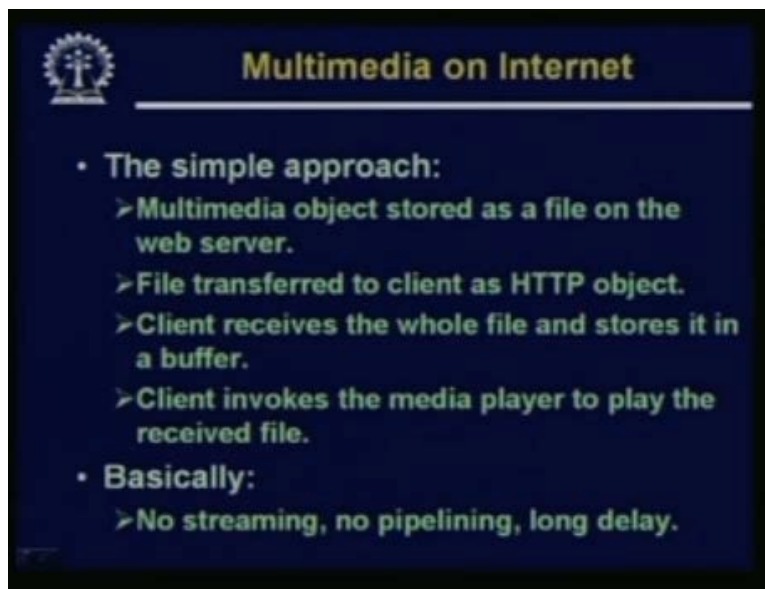
What the applications want and what the network can provide? So to the extent possible internet multimedia applications use application level techniques. Because at the network level there is not much to do. The protocols which are running on the network does not provide much help. So use as much application level techniques possible to get the best out of the underlying network or the service. However next generation internet which is due to come up can handle this much better. Because in the next generation internet the network infrastructure will be such that quality of service guarantee can be provided to the communicating parties. But in today's internet there is no guarantee about the same. Most of the routers which are there in between they do not support real time traffic or quality of service parameters.

(Refer slide time: 25:00)



So now let us talk about streaming multimedia. How these are typically implemented?

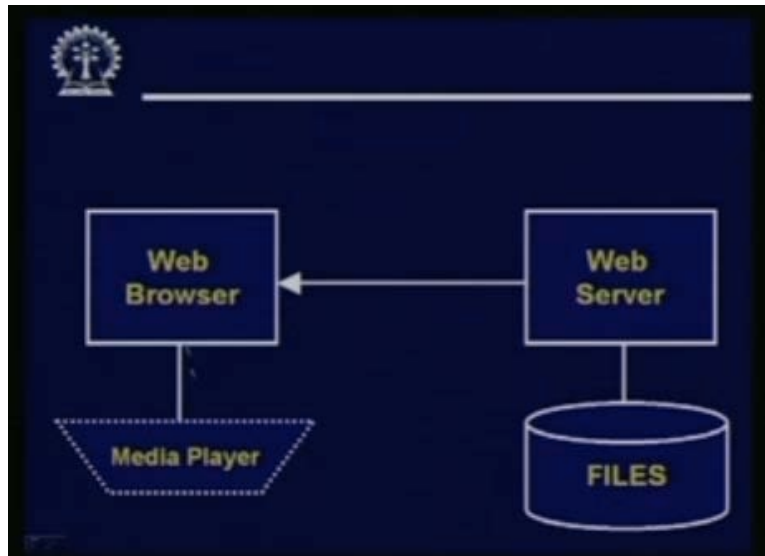
(Refer slide time: 25:09)



Multimedia on internet. So there are several variations. First let us look at simple approach. In the simple approach what we say the multimedia object is stored as a file on the web server. The client sends an http request and the server sends the file back to the client as an http object. Client will receive the whole file and store in a buffer and then it will invoke the media player to play the received file. This is what we still use today for many media types which do not support streaming. When you want to run a particular media file which is stored somewhere. You send a request, we get it back, we save it on

our hard disk, then we call our media player and open that file that file starts playing. So here basically it is a download and play mode. There is no streaming no overlapping or pipelining and typically the delay is long. You need to download the whole file before the playback can start.

(Refer slide time: 26:28)



So the picture is like this. Pictorially the file is stored here attached to the web server. So web server will be transferring or transmitting that file to the web browser. The file will get stored in the folder on the web browser machine and then the media player will be invoked to play the file. This is the typical flow for scenario in this kind of an environment.

(Refer slide time: 27:05)

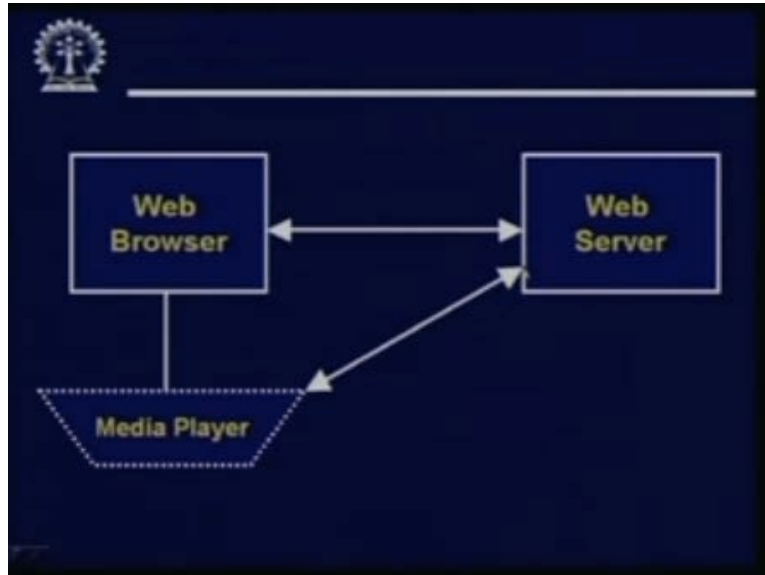


Now let us come to the so called streaming approach. Now in this approach I am downloading the whole file and then I am playing. In the streaming approach I should start playing as soon as the data starts coming to me. But you see if you want to have this you can have it in a similar way. But there are a few other things we looked into. See when I am requesting to view a streaming multimedia content on my desktop, there are some issues. How I am connected to the internet? Is it through a low speed dial up line? Is it through a medium speed internet link, leased line? Or is it that entire thing is happening in a LAN which is high speed? So I have a media server in my LAN itself and I am trying to access some media from that LAN, so speed is important.

The kind of media the server will try to send me should very much be dependent on the speed of the link the mechanism bandwidth is supported. So in the streaming approach typically the flow is like this. Browser first requests for a metafile from the web server. See metafile is something I will give an example later. Metafile contains some information about the media I am trying to play. Suppose I am trying to play music I request to the web server please send me the metafile for this particular music. In the metafile I may have several different things stored like this. That if I have a low speed link if I want to play it in the low resolution or low fidelity mode, I should download this file this link.

If I want Hi-fi then I should download this link. In this way depending on exactly what I want or if I do not want, only the audio I want to play, the whole video clip that contains that audio song, then I may provide with another link. See all these things depend on the kind of speed the kind of bandwidth I can provide. So first the browser gets the metafile then it launches or invokes the media player and passes the metafile to it. Now the browser is nowhere. The media player now starts directly interacting with the web server from using the information stored in the metafile. Server streams audio video object in its http response to the media player. This works but is usually considered unsatisfactory because you do not have much control here on the playback and it is non-interactive.

(Refer slide time: 30:33)



Pictorially this is shown like this. First web browser and web server interacts the web browser will be sending some request to the server. The server will be sending back the metafile. After getting back the metafile web browser will be invoking the media player. Then the media player will be directly interacting with the web server. So this is how the streaming mode works.

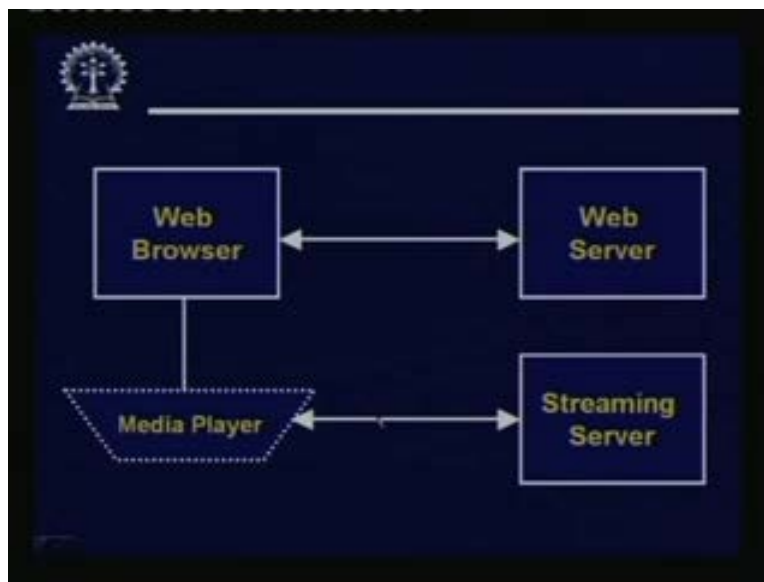
(Refer slide time: 31:06)

- Using a separate streaming server:
 - Provides the best performance.
 - This architecture can use non-HTTP (may be proprietary) protocols between server and media player.
 - Can also use UDP instead of TCP.
 - For better response.

Now the next alternative to streaming we say that we do not burden the web server with everything the method. We have just talked about there we have said that all the media files are also stored as http object in the web server. They are requested by http. They are

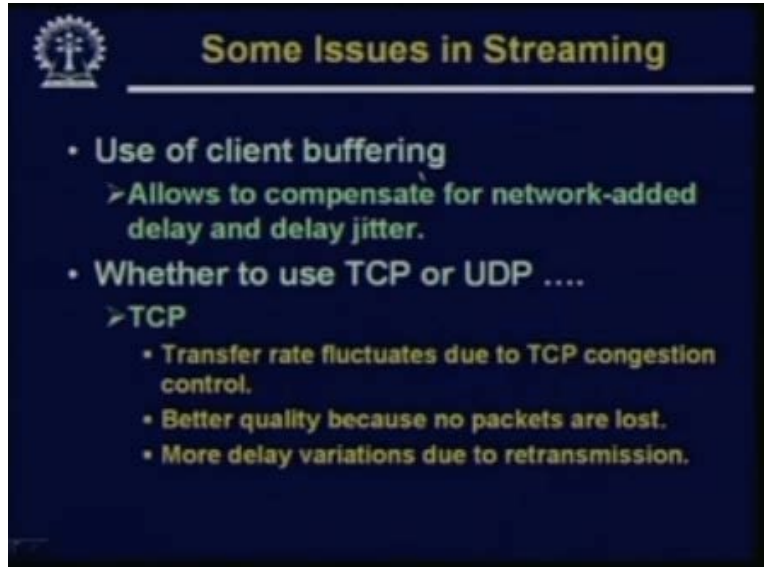
sent back as http objects. But because of constraints in the web server, quality is often not that good or acceptable, Now what we say is that let us have a separate streaming server in addition to the web server, there will be another server where only the streaming contents will be stored. This usually provides the best performance. Best performance why? Because here you can use non http protocols. These may be proprietary which may be tailor made for this type of real time media downloads and moreover http uses TCP. Here you can choose to use UDP instead of TCP for better response. So here there are two things. Number one, you are using separate streaming server. Secondly interaction between the media player and the streaming server can be using any non standard or proprietary protocol that something which you are allowing.

(Refer slide time: 32:48)



Pictorially it is like this. Now on this side there are two servers. Web server, streaming server. Web browser sends a request gets back invokes the media player. Media player now connects to the streaming server and starts the downloading.

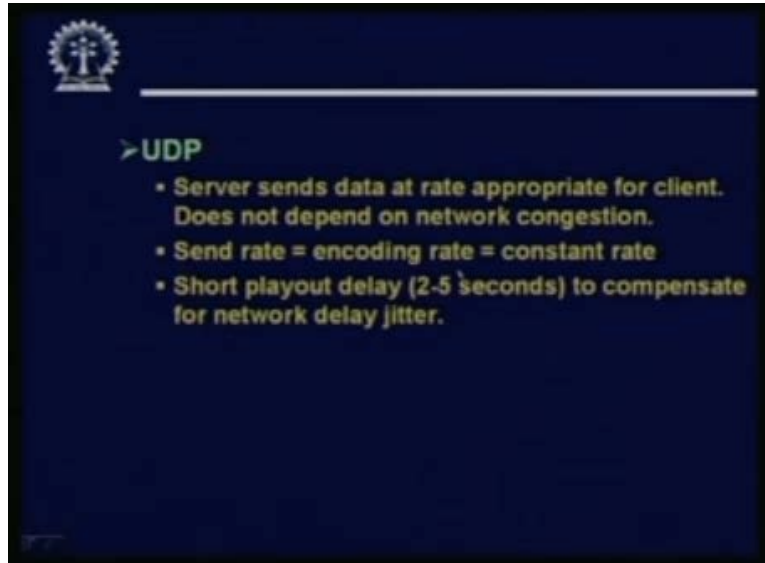
(Refer slide time: 33:12)



Now here there are some issues. Normally we have client buffering. Client buffering says that as soon as the packets starts arriving at the client side do not start playing immediately. Keep a buffer; start playing only after certain numbers of packets have already arrived in the buffer. So that after you have started playing, if there is some delay, in some future packet that delay you can compensate by using that buffer. So this buffering helps to control network added variable delay and jitter to some extent. But of course if the variability is too much and continuous over long periods of time then even by using buffering you cannot do much and there will be jitters in the playback. The second thing is that your choice of TCP or UDP means for communicating between the media player and the streaming server. Do you want your TCP or UDP? Now if you choose to use TCP there are a few points here.

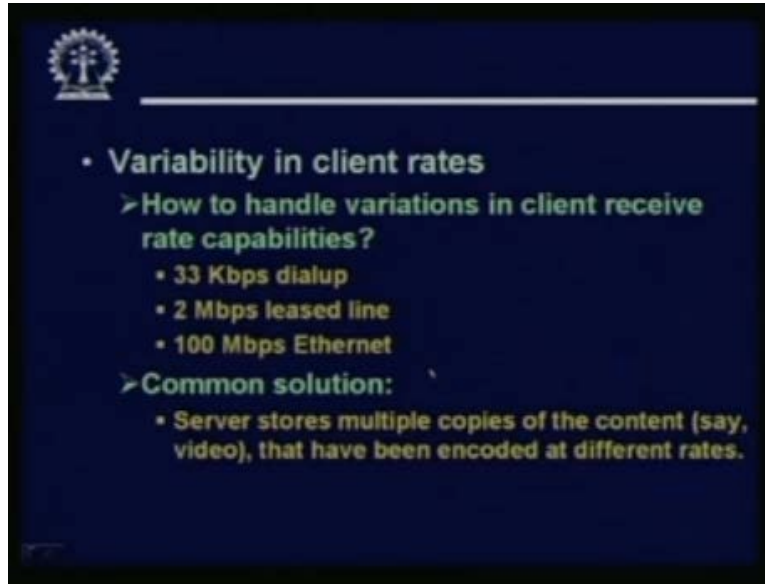
In TCP you know TCP tries to provide a reliable link. So if actually there is a packet which has not arrived TCP will send back an explicit request for the packet wait for the packet to come back and once the entire message has come back in its entirety, then only it will be forwarded to the application on top. But the application here is the streaming media player. The application has to wait for so long. A request going in the packet and again coming back correctly receiving all the pieces and then only it will get the next junk of data. So the transfer rate fluctuates due to TCP congestion control. Delays of the packets is variable and the transfer rates therefore fluctuate. Quality is better means once you have received your quality will be better because there is no packet loss. All packets have arrived and you can get exact playback. Just the case I have said that because of retransmission because of non arrival or error in the packets there can be more delay variations in TCP.

(Refer slide time: 35:46)



However if you choose to use UDP the server can send the data at an appropriate rate for a client, UDP does not depend on network congestion. Depending on the present network situation the UDP packets can find any arbitrary load and you will reach the final destination. TCP there is a congestion control mechanism due to which there are lots of overheads and there can be delay variations. When UDP is a very light weight protocol there is no such congestion control mechanism here. So typically the data rate will remain approximately the same. The sending rate of the data will be the rate at which you will be encoding the data into digital form and this is will be approximately constant. But there will be network delay jitter. There will be some variability. So you can have a short play out delay in the range of 2 to 5 seconds to compensate for this. So in general UDP works better than TCP. When you are having a streaming kind of an application this is what you should remember.

(Refer slide time: 37:07)



And the variability in client rate is which I was talking about that how to handle variations in client receive rate capabilities. As I said some user may be having a dialup connection only 33 Kbps bandwidth. Some may be having leased line 2 Mbps. Some may be having the streaming server as well as the client on the same line. So there can be 100 Mbps Ethernet connections. See if you have these three different kinds of connections, these three examples I have taken. So the rate at which you can pump data on to the desktop will be very much different. And it will not be you can say a good idea to choose the worst scenario that means assume the link to be slowest. Accordingly do some encoding of audio or video so that you can transmit data at a slow rate at a low quality because a person having 100 Mbps connectivity with the server will not like that quality of playback. They will want very high quality. So the solution goes like this that you can have multiple copies of the same media encoded in several different formats stored in the streaming server like I will give an example.

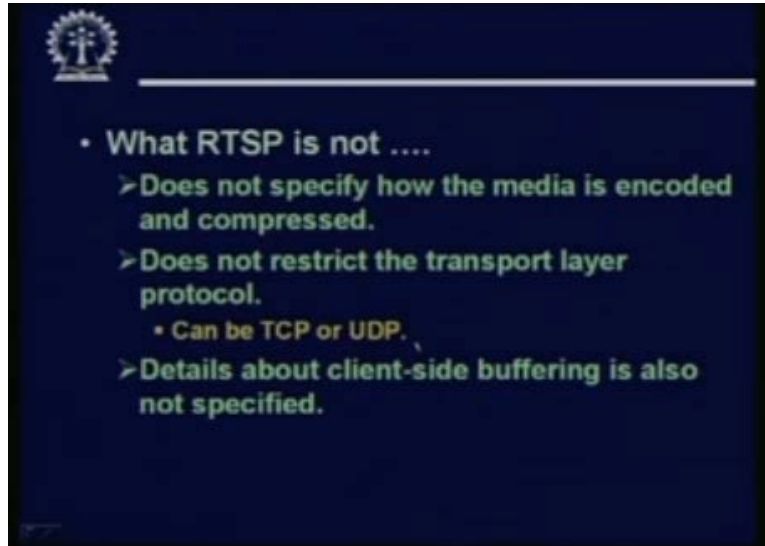
Suppose our national anthem. Our national anthem jana gana mana you can store in a very high resolution Hi-fi format that of course will take more number of megabytes for storage. You can store it in a medium reservation format that can take 1 or 2 megabytes. You can store it in a low reservation format which will not of course be DVD quality audio which will take possibly a few kilo bytes of storage. So depending on the requirement of the client depending on the capability of the client, the client will send a request for the appropriate type of media. The streaming server will be trying to send the correct kind of media to suit the bandwidth constraints. So just what I have said the common solution here is server will store multiple copies of the same content which are encoded at different rates. And hence will occupy different sizes on the disk and will occupy or and will also require different bit rates for real time transmission on the client desktops.

(Refer slide time: 40:08)



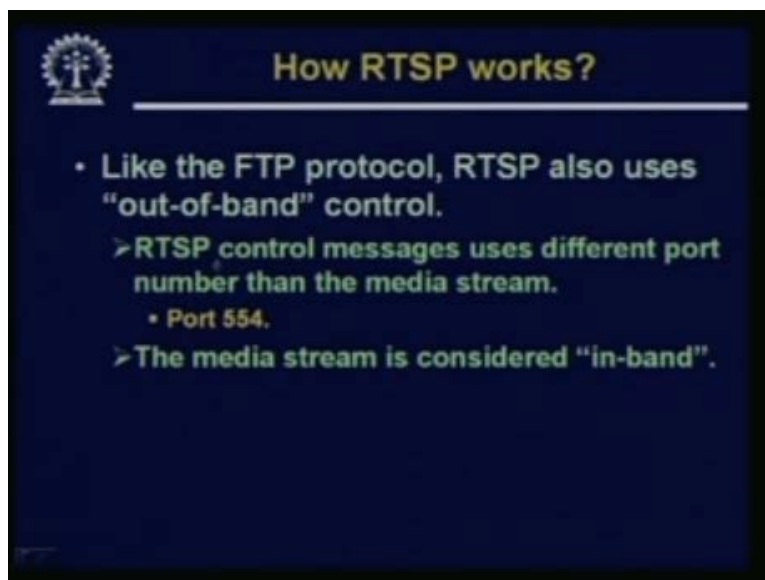
Some protocols have been designed at the application layer level to give some breathing space to the multimedia applications. One such is the real time streaming protocol RTSP. Now a real time streaming protocol is as I said, this is a protocol which is working at the application layer level under TCP, IP, UDP. Whatever is there, it is there, RTSP tries to provide some facilities which are very much relevant to multimedia streaming applications. This RTSP allows users to have much better control over streaming media. Now RTSP is basically a client server protocol. The kinds of controls that it provides to the users are: pause, play, rewind, forward, repositioning. So you can move the play cursor forward and backward. You can play pause you can play whatever you want and this RTSP protocols has the feature to handle all these needs. Now as I said when you have RTSP when you, for example do a fast forward RTSP protocol will be sending a request to the streaming server that from now on you start sending the packet from the requested position so that wherever I have pulled a pointer from there playback will start. So this is what RTSP is.

(Refer slide time: 41:58)



But RTSP what it does not do. It does not specify the encoding mechanism. It does not specify how the media is encoded how it is compressed. This is entirely left open. It also does not tell you whether you use TCP or UDP it is your choice you can use both. And how much buffer client has to have for jitter feed playback for controlling jitters that is also not specified. Because these are something which can very widely depending upon the kind of network, the kind of users, the kind of multimedia streaming on to view and so on. So this RTSP does not impose any kind of constraint on these kind of variations.

(Refer slide time: 42:52)



Let us see how RTSP works in general. Now you remember in the in the FTP protocol. There was one port number for the control connection one for the data connection. RTSP

does the same thing and this is technical name, this is out of band control. In RTSP also there is one connection through which the control requests and acknowledgements come; there is another connection through which the streaming data are transmitted. There are two channels. The RTSP control messages as I mentioned uses different port number than media stream. It uses a port number of 554 and the media stream is considered to be in-band and control message is considered to be out band. This is the convection.

(Refer slide time: 43:51)



Now a typical scenario out here will be like this. Well here also we have the concept of a metafile. The file will contain the relevant information as I said about the different versions of the same media which is stored. They are encoding formats and so on and the respective name of the file. So the metafile is first sent to the web browser over http. The browser launches the media player. Now the media player will be setting up an RTSP control connection. This is the so called that proprietary connection I was talking about. Of course RTSP has a standard, but at the beginning this was a proprietary protocol. This was not TCP not UDP.

So the windows media player will be setting up RTCP, RTSP connection with the server, the cut end server or the streaming server and a data connection also. So these two connections will allow the media player to communicative with the server and request for the appropriate media content. Here the two servers in existence a web server and a streaming server. So just to summarize the browser will first get the metafile the browser will launch the media player. Media player will initiate control connection to the server. It will initiate an RTSP connection for the data stream with the streaming server and then the transactions will go on. There will be some simple, I will give an example how the typical commands are.

(Refer slide time: 45:57)



```
<title>Trailer</title>
<session>
  <group language=en>
    <switch>
      <track type=audio
        e="PCMU/8000/1"
        src = "rtsp://stream.com/trailer/audio.en/lofi">
      <track type=audio
        e="DVI4/16000/2" pt="90 DVI4/8000/1"
        src="rtsp://stream.com/trailer/audio.en/hi-fi">
    </switch>
    <track type="video/jpeg"
      src="rtsp://stream.com/trailer/video.en">
  </group>
</session>
```

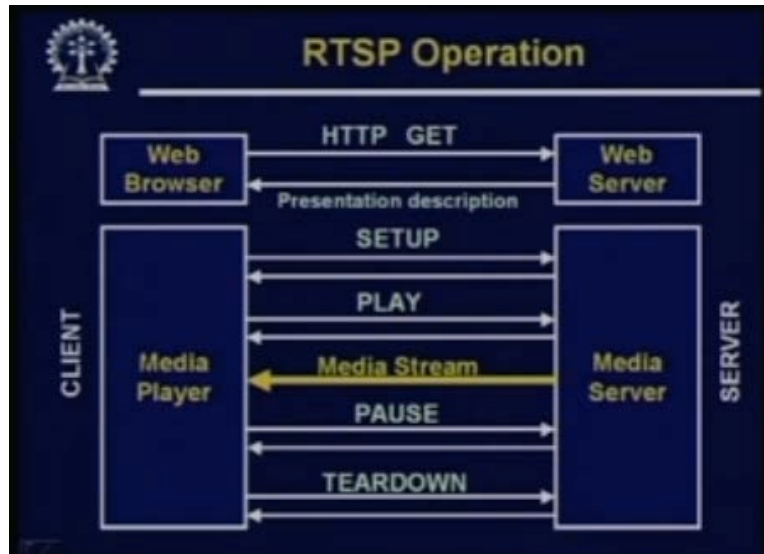
Now this diagram shows you how a typical metafile will look like. This is a typical metafile. This has been coded in a xml like language. There are some tags you can see which are these are html like tags. It starts with a title. There is a tag called session. This indicates the present session that what we want to do in this present session. There are several things which you can group together this begin group, end group. This indicates group and in the group there are a number of optional parameters. Like here we have mentioned language should be English en. There is a switch statement inside group. See inside group you see there are two statements. One is a switch statement, other is a track statement. Now inside the switch statement there are two tracks.

Switch statement actually means say inside a switch statement you can put more than one track statements and you will be selecting one of the track statements. And if we have two statements inside a group they are all considered to be together you have to do both. So this actually tells this is the example of a trailer. This tells you that in the group there are two things. In the first one there is an alternative in the switch for two tracks. First track says type equal to audio e equal to some code. Here this actually tells you what kind of encoding you have used and source exactly what is the name of the file. Second one says again if type is audio and the encoding is different. There are additional parameters you can specify.

This actually indicates a hi-fi audio stream DVD quality, then you contact or try to download this particular file right this one. So in the first switch statement depending on the capabilities of the media player media player will be asking for one of these two. And in addition there will also be another track which contains video of type video RTSP stream com some URL is again here. So actually this metafile tells you that whatever you are downloading it consists of one audio file. Here you have an alternate you either download the low fidelity or high fidelity audio file and you have a video file you download both and play it on your media player. So essentially what to get you will be

viewing the trailer on it you will be seeing the video clip. You will also be hearing the, audio.

(Refer slide time: 49:14)



So let us look at a typical RTSP operation scenario. This diagram actually shows you what kind of commands and responses go between the web servers, browsers and the servers. You see here there are four entities. First one is the web browser then we have the web server. You have the media player you have the streaming server or the media server whatever you call. First the web server must send an http get request to the web server to get back the metafile. So here we are showing we are just mentioning it as the presentation description. This is the metafile. After getting the metafile as I have said web browser will be forwarding the metafile to the media player so that now the media player will take up the issues with the media server. Here in an entire session there are several phases as this diagram shows. There is a setup phase. Media player sends a request media server sends back a response.

This is the setup play media player sends a request media server sends back a response this is the play. After this media stream will be continuously downloaded this yellow arrow gets that. This will be the most you can say time consuming in this entire session media stream download. Media player can pause the playback. So pause and pause acknowledge, not only play back downloading can be paused. When a pause request is sent to the media server, then downloading of the media stream is blocked. And finally you can totally close down the connection by sending a so called teardown message. So the connection is tore down and the connection gets broken after getting back an acknowledgement from the server. So these are the different phases or the different message exchanges that goes on between the web browser or the client and the server. Server as I said there is a web server there is a streaming media server. So between one of these two this kind of communication or transaction goes on.

(Refer slide time: 51:57)

A slide titled "RTSP Exchange Example" with a logo in the top left corner. The slide displays a sequence of RTSP messages between a client (C) and a server (S).

```
C: SETUP rtsp://stream.com/trailer/audio RTSP/1.0
  Transport: rtp/udp; compression; port=3056; mode=PLAY

S: RTSP/1.0 200 OK
  Session: 4231

C: PLAY rtsp://stream.com/trailer/audio.en/lofi RTSP/1.0
  Session: 4231
  Range: npt=0-

C: PAUSE rtsp://stream.com/trailer/audio.en/lofi RTSP/1.0
  Session: 4231
  Range: npt=37
```

Now let us look at a typical message exchange scenario in RTSP. C indicates clients, S indicates the servers. So I am assuming in this diagram again that this initial downloading is already done of the metafile is done. Metafile has been given to the media player and media player starts communication with the meta-server. First is the setup command from the metafile. The media player can find out which particular file it wants to download. So in setup it specifies the name of that file and some RTSP version. There is a particular syntax of this command and some additional data that you want transport to be done over RTP, UDP, you want compression port number to be used 3056 and mode is PLAY which means you are wanting to play the media downloaded.

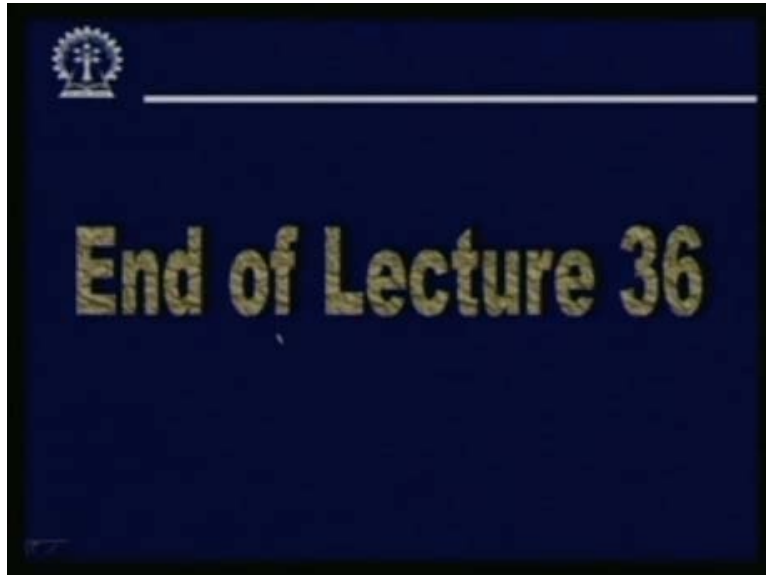
Server responds by its own version and some message code and session identifier. Now after getting back this response client sends a request for the explicit file which particular file it wants to play. First command was the general command trailer audio. Now here trailer audio LOF identity. Now here client also sends some additional information like the session ID and some range I am not going into detail of these parameters. But it basically tells that exactly what bit rate is to be followed and it is the range of this media it wants to play. Zero dash means, from zero it wants to play till the end. Then some time later see the clients can and send a pause message pause the name of the file. And it says that you pause at an index of 35, 37, go there and pause.

(Refer slide time: 54:16)



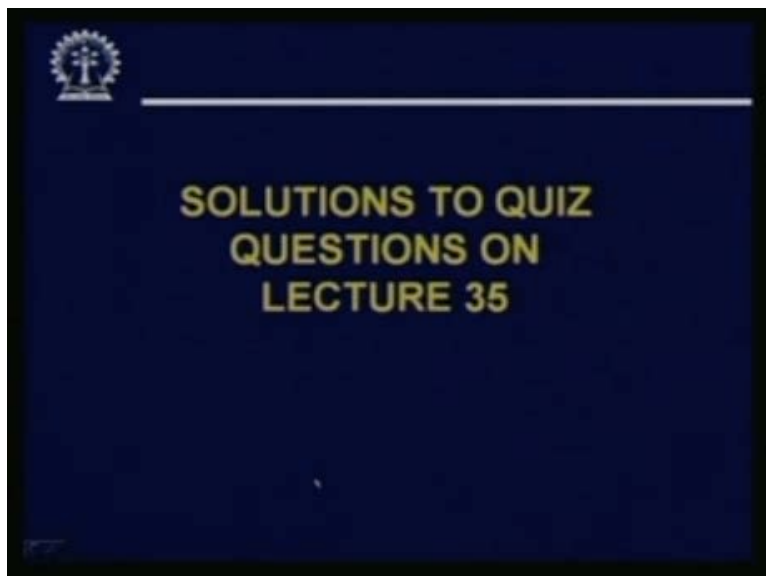
Finally the client can say TEARDOWN. TEARDOWN the name of the file again the name of the session name here and the session name is specified here. And finally the server responds to this mess and the connection breaks. Actually what I was mentioning is that this is, these are the low level commands which flow between the media player and the media server, depending on what the user has clicked on the media player. User may click on pause play fast forward. Whatever if it is forward then the media player will know exactly where the point has been set. So a pause message will go to the streaming server specifying the exact relative location which frame number has to be the next one being downloaded. So in this way using the simple protocol. See this protocol is specifically meant for downloading and controlling multimedia messages. So using this protocol you can get much better performance as compared to doing everything over http. This is in that sense or so called proprietary protocol running on top of UDP or TCP.

(Refer slide time: 55:37)



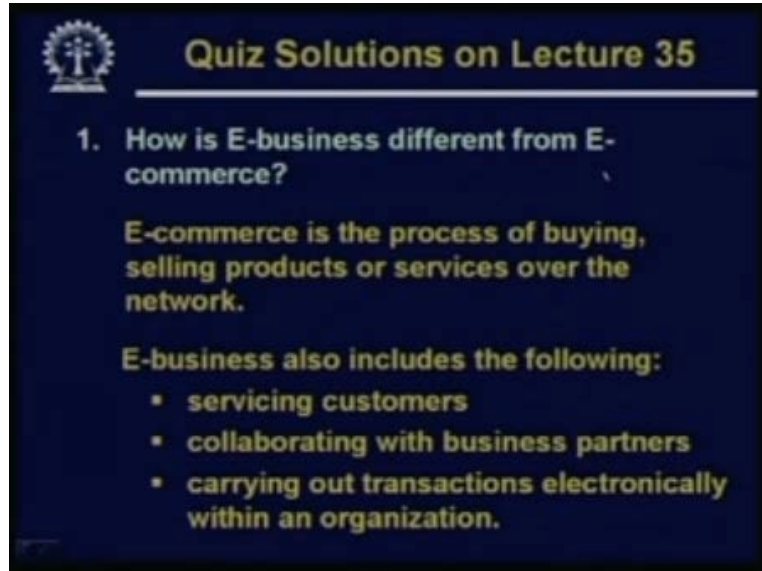
So with this we come to the end of lecture number 36.

(Refer slide time: 55:48)



First we see that where the answers to the questions posted during our last lecture.

(Refer slide time: 55:51)



Quiz Solutions on Lecture 35

1. **How is E-business different from E-commerce?**

E-commerce is the process of buying, selling products or services over the network.

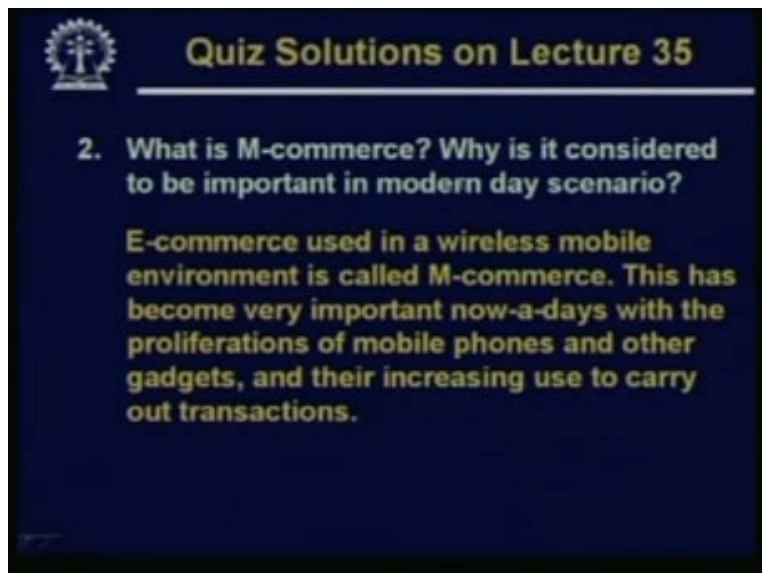
E-business also includes the following:

- **servicing customers**
- **collaborating with business partners**
- **carrying out transactions electronically within an organization.**

How is E-business different from E-commerce?

Well E-commerce as I said is the process of buying selling products or services over the network. E-business includes all these things. In addition it is including servicing customers collaborating with other business partners or carrying out transactions electronically within an organization. These also fall under the preview of E-business.

(Refer slide time: 56:24)



Quiz Solutions on Lecture 35

2. **What is M-commerce? Why is it considered to be important in modern day scenario?**

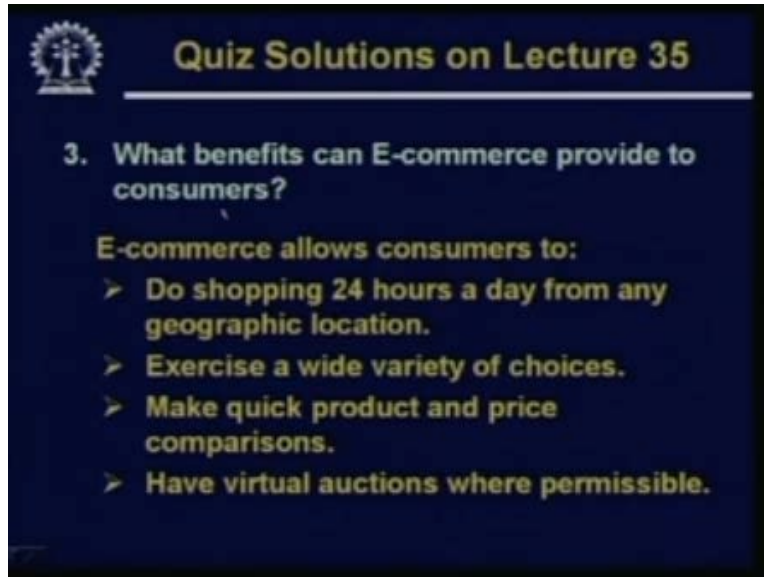
E-commerce used in a wireless mobile environment is called M-commerce. This has become very important now-a-days with the proliferations of mobile phones and other gadgets, and their increasing use to carry out transactions.

What is M-commerce? Why is it considered to be important in modern day scenario?

E-commerce used in a wireless mobile environment is called M-commerce. Basically this has become very important nowadays with the proliferations of mobile phones and other

hand held gadgets and their increasing use to carry out transactions. This is the main motivating factor behind having this M-commerce applications being developed. And it is considered to be one of the hottest applications to hit the market in the near future.

(Refer slide time: 57:00)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 35" is written in yellow at the top. Below the title, the question "3. What benefits can E-commerce provide to consumers?" is displayed in white. The answer is presented in yellow text, starting with "E-commerce allows consumers to:" followed by four bullet points, each preceded by a yellow right-pointing arrow.

Quiz Solutions on Lecture 35

3. What benefits can E-commerce provide to consumers?

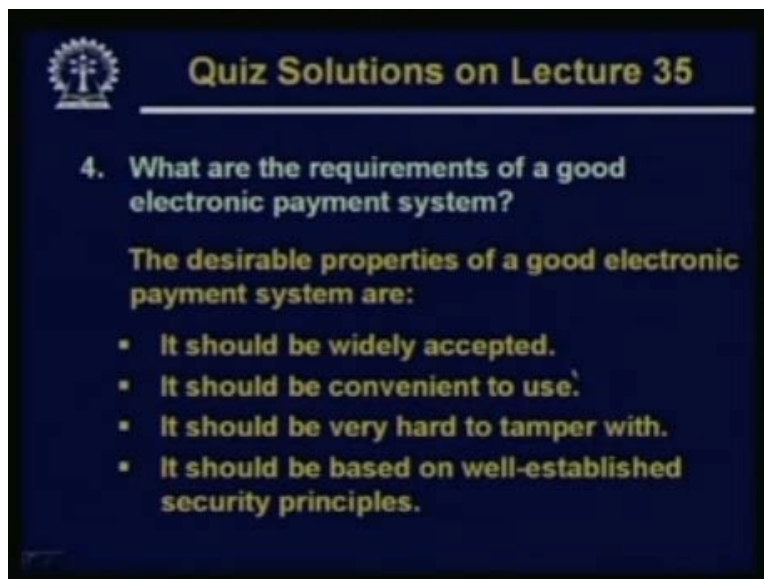
E-commerce allows consumers to:

- Do shopping 24 hours a day from any geographic location.
- Exercise a wide variety of choices.
- Make quick product and price comparisons.
- Have virtual auctions where permissible.

What benefits can E-commerce provide to consumers?

E-commerce allows consumers to do shopping 24 hours a day from any geographic location. Exercise a wide variety of choices makes quick product and price comparisons have virtual auctions wherever permissible.

(Refer slide time: 57:22)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 35" is written in yellow at the top. Below the title, the question "4. What are the requirements of a good electronic payment system?" is displayed in white. The answer is presented in yellow text, starting with "The desirable properties of a good electronic payment system are:" followed by four bullet points, each preceded by a yellow square bullet.

Quiz Solutions on Lecture 35

4. What are the requirements of a good electronic payment system?

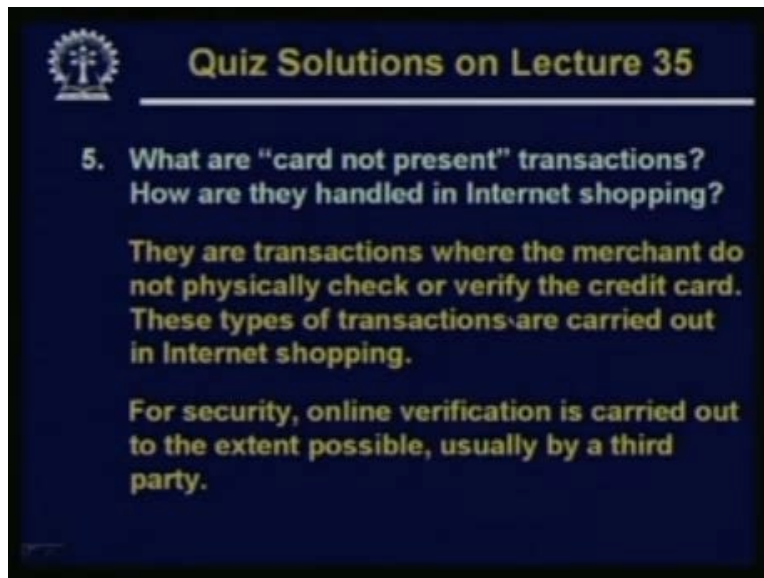
The desirable properties of a good electronic payment system are:

- It should be widely accepted.
- It should be convenient to use.
- It should be very hard to tamper with.
- It should be based on well-established security principles.

What are the requirements of a good electronic payment system?

The desirable properties are as follows as I mentioned during the last class. It should be widely accepted, convenient to use, very hard to tamper with and should be based on well-established security principles. Well if you say, today that I have developed a new security algorithm. I am using it. No one will trust you because no one really knows how good or how bad your security algorithm is.

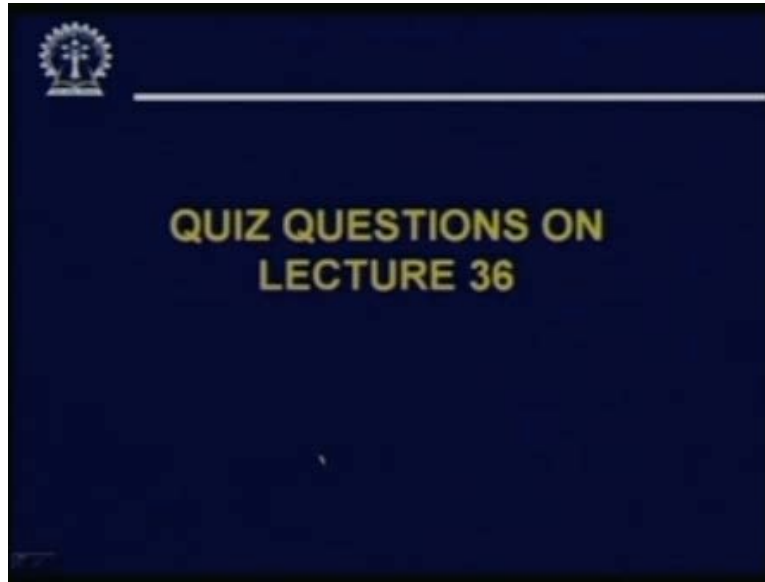
(Refer slide time: 57:55)



What are card not present transactions? How are they handled in internet shopping?

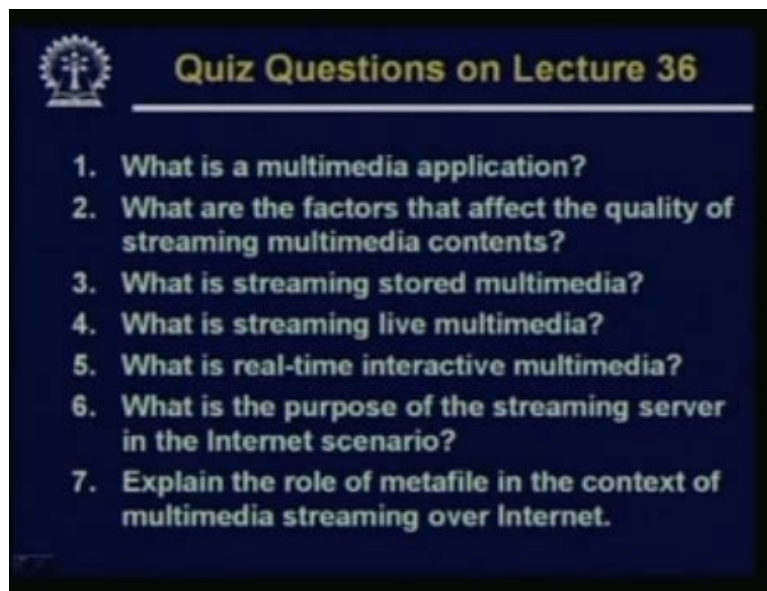
They are transactions where the merchant do not physically check or verify the credit card. Because they are not physically present there. These types of transactions are carried out in internet shopping. As I mentioned again for security online verification is essential and this online verification is typically done by a trusted third party.

(Refer slide time: 58:29)



Now some questions from today's lecture.

(Refer slide time: 58:34)



What is a multimedia application?

What are the factors that affect the quality of streaming multimedia contents?

What is streaming stored multimedia?

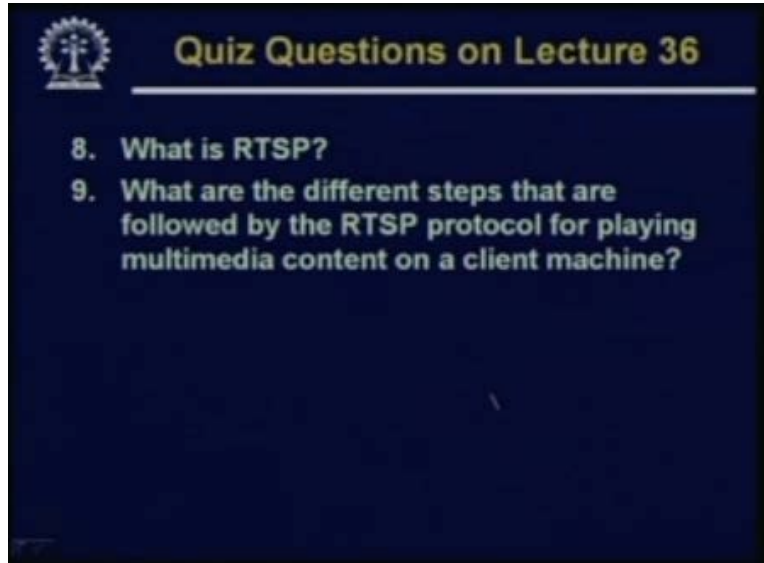
What is streaming live multimedia?

What is real time interactive multimedia?

These three different types

What is the purpose of the streaming server in the internet scenario?
Explain the role of metafile in the context of multimedia streaming over internet?

(Refer slide time: 59:11)



What is RTSP?

What are the different steps that are followed by the RTSP protocol for playing multimedia content on a client machine?

So with this we come to the end of today's lecture. In our next lecture we shall be continuing our discussion on multimedia in real time contents over the internet. We shall be talking of some standards. We shall be talking about IP telephony and some other kind of application which can be built using the standards that we would be talking about. Thank you.