



**streamingmedia.com** home of the streaming media industry!

## Streaming vs. Downloading Video: Understanding The Differences

One of the most frequently asked questions about delivering online video is "What's the difference between streaming video and downloading video?" Let's face it, as a user clicking a video link on a web page, you often won't know which method you're using, unless you poke around a little. But streaming and downloading are distinct methods of delivery, each with its own benefits and limitations. We'll take a look at the difference between the two methods, and make some suggestions about when you should choose one or the other for your projects.

7/22/2003 - By [Larry Bouthillier](#)

### **Delivering From A Web Server**

Delivering your video file using a web server is sometimes referred to as "progressive download" or "http streaming". In fact, it's not really streaming at all, but a very simple bulk download of the video file to the viewer's computer. Let's say you have a video file encoded at 200kbps. You place that file on your Web server, and put a link to the file on your web page.

The web server does not know or care that it's a 200kbps video file. It simply pushes the data out to the client as fast as it can. It may appear to be streaming since playback can begin almost immediately. The "progressive download" feature in most media players allows them to begin playing the file as soon as enough data has been downloaded. Of course, you can't fast-forward to the end of the file until the whole file arrives from the server.

If the actual network bandwidth is smaller than the 200kbps that the file is encoded at, then you may have to wait a while before you can begin playing it. But even on a 56kbps connection, the video will look great – you're essentially trading waiting time for video quality. The temporary file is saved to the user's computer, so they can play it again if they want to without having to download it again.

Web servers use HTTP (Hypertext Transport Protocol) to transfer files over the network. One of the features of HTTP is that it operates on top of TCP (Transport Control Protocol), which controls the actual transport of data packets over the network. TCP is optimized for guaranteed delivery of data, regardless of its format or size. For example, if your browser or media player realizes that it's missing a data packet from the server, it will request a resend of that packet. Resend requests take time, take up more bandwidth, and can increase the load on the server and if the network connection is sketchy, you could begin to use more bandwidth for resends than you're using for the video itself! TCP is not designed for efficient real time delivery or careful bandwidth control, but for accurate and reliable delivery of every bit.

### **Next Page: Delivering From A Streaming Server**

#### **Delivering From a Streaming Server**

A streaming media server is a specialized piece of software that accepts requests for video files, knows about the format, bandwidth, and structure of those files, and in many cases, pays attention to the performance of the player that's receiving the video. Streaming servers deliver just the amount of data necessary to play the video, at precisely the rate needed to play it.

Unlike the web server, which simply starts dumping as much video data onto the network as it can, the streaming server opens a conversation with the media player. There are two sides to this conversation – one to transfer the video and one for control messages between the player and the server. Because they continue to exchange these control messages with the player, streaming servers can adjust to changing network conditions as the video plays, improving the viewing

experience. The control messages also include user actions like play, pause, stop, and seeking to a particular part of the file. Since the server sends video data only as it's needed and at just the rate it's needed, it also allows you to have precise control over the number of streams you serve and the maximum bandwidth you consume.

If you've got a 56kbps connection to the network, you won't be able to receive that 200kbps video. You'll have to settle for a lower-quality version that's encoded for 56kbps connections. But streaming delivery of video data does have some advantages:

- You can skip ahead in a video, or begin playback at a point somewhere in the middle. This is a convenience to users, but also a boon to you as a provider. It enables interactive applications like video search and personalized playlists.
- It lets you monitor exactly what people are watching and for how long they are watching it.
- It makes more efficient use of bandwidth since only the part of the file that's watched gets transferred.
- The video file is not stored on the viewer's computer. The video data is played and then discarded by the media player, so you maintain more control over your content.

In a pinch, streaming servers can use HTTP and TCP to deliver video streams, but by default they use protocols more suited to streaming, such as RTSP (Real Time Streaming Protocol) and UDP (User Datagram Protocol). RTSP provides built-in support for the control messages and other features of streaming servers. UDP is a lightweight protocol that saves bandwidth by introducing less overhead than other protocols. It's more concerned with continuous delivery than with being 100% accurate – a feature that makes it well-suited to real time operations like streaming. Unlike TCP, it doesn't request resends of missing packets. With UDP, if a packet gets dropped on the way from the server to the player, the server just keeps sending data. The idea behind UDP is that it's better to have a momentary glitch in the audio or video than to stop everything and wait for the missing data to arrive.

Finally, a streaming server is necessary to deliver live webcasts and to use multicast. For networks that support it, multicast allows more than one client to tune in to a single stream, saving bandwidth at every part of the delivery chain.

## **Next Page: Conclusions**

### **Conclusions**

All of this adds up to a few simple rules of thumb for when to use streaming and when to use http downloading to deliver your video. The main reason for downloading video from your Web server is that it's simple and you can do it with infrastructure you already have. It's most useful when your videos are short, when you're more interested in delivering high-bitrate encodings than in delivering in real time, or when you want your viewers to be able to keep a copy of the video on their own computers.

Streaming is the better solution when your clips are more than a few minutes long, when you want to enable interactive applications like video search or linking deep into a file, or you want to collect statistics on what's actually being watched. Streaming is the way to go when you want to control the impact of video on your network, or when you need to support large numbers of viewers. And of course, it's the only way to do live webcasts and multicasting.

We'll be covering many of these topics in more detail over the coming months. If there are particular technical topics you'd like us to address, let me know at [larry@streamingmedia.com](mailto:larry@streamingmedia.com)