Increase Viewer Loyalty:
Best Practices for Ensuring a Quality OTT Experience

Authors:
John Councilman – Enterprise Architect, Global Consulting Services, Akamai Technologies
Karin Schade – Senior Enterprise Architect, Global Consulting Services, Akamai Technologies
**Introduction**

The media industry is evolving. As consumers increasingly turn to digital channels for entertainment and information, new opportunities are emerging and there is no place for low-quality video in any streaming business model. Workflow challenges can often cause the No. 1 user management pain point: rebuffering.

As broadcasters move to paid services, what can be done to ensure that customers remain loyal and receive the quality they are paying for?

Over-the-top (OTT) video does not have the luxury of a closed, private network. There are a few things that can be done to reduce rebuffering when network conditions are unknown or fluctuate.

**Publishing Matters**

Much thought and discussion can occur surrounding pushing content from encoders to end users, but content must be optimally prepared for many of these techniques to have a great outcome. Codecs, segment length, and resolution all play into the user experience.

**Optimize Segment Length**

Content segment length has a huge effect on rebuffering. A typical bitrate ladder for HLS (the most common streaming format today) begins around 145 KB/sec and tops out around 7 megabits/sec. In the past, larger segments were recommended due to the latency required to deliver smaller segments. Using modern delivery products, this is no longer the case. Apple, the DASH Forum, and the industry have moved on to recommend smaller segment sizes for several reasons.

Of course we want users to experience the highest-quality delivery, but fluctuating network conditions can sometimes make this impossible. Things as trivial as in-home Wi-Fi interference can affect streaming, and obviously there’s nothing a broadcaster can do to work around something like that. According to Akamai’s 2017 State of the Internet report, the average worldwide connection speed is roughly 7 megabits/sec.

In the past, 6-10 second MPEG segments were standard. If we continue to utilize this standard, one must wait that respective amount of time until an adjustment can be made. Using the highest bitrate of 7 megabits/sec per the ladder described above, the user is streaming at almost 7 megabits/sec and has the average 7-megabit connection, so there is little overhead for anything else. If another household member used some of the available bandwidth, the consumer must now wait 6 seconds to request the lower-quality version to continue streaming successfully.

Lowering segment sizes to 2-4 seconds has been shown to reduce rebuffering when network issues arise. The client is able to adjust for changing network conditions and utilize the many advantages of adaptive bitrate streaming.

Of course, there are always tradeoffs. We have the saying in IT, “Fast, Reliable, Cheap ... pick two.” When segment sizes are lowered, there are naturally more I-frames, which cannot be compressed as much as P- or B-frames.
Use at Least One IDR Frame Per Segment (Preferably at the Start)

In order to have fast startup and seek, standards dictate having an IDR frame in each segment. In H.264 video, IDR refers to Instantaneous Decoder Refresh, a special kind of I-frame. These indicate to the decoder that no frame occurring after an IDR frame depends on any frame that occurs before it. We recommend you have at least one IDR frame at the beginning of your segment, because when seeking or starting up partway through a live stream, the client needs an IDR frame to get started. If you put your IDR frames partway into the segment, the client cannot start until it finds an IDR frame. (Per Apple.com, use the `-start-segments-with-iframe` option with `mediafilesegmenter`.)

We recommend working with your encoder vendor to discover optimizations that can be accomplished during content preparation to prepare for smaller segment sizes. Some older encoders might not support smaller frames, or may require a software update to do so.

In the past, Apple and other vendors would recommend larger segments because of legacy protocols (HTTP 1.1), CDN support, and TCP Overhead. Many of these original issues are addressed by newer technologies. Persistent connections, QUIC, and HTTP/2 address the TCP Overhead issue. CDNs such as Akamai supporting these newer technologies are now better optimized to serve rapidly changing manifest files.

As segment sizes decrease, ingest latency becomes more and more important. To achieve broadcast-parity latency, the end-to-end publishing playback chain must be optimized. Akamai’s newest offerings address this workflow by offering a unique solution, IAS (Ingest Accelerated Service).

Here is a quick overview of key Image Manager functions that make up the service:

**Accelerate Ingest**

IAS involves an IAS and IAT component residing near the encoder and within Akamai’s network, respectively. The idea is for the encoder to be able to publish content via TCP to a local endpoint with the lowest latency, reducing TCP overhead, retries, and slow starts. This endpoint (IAS) will in turn publish to Akamai’s IAT endpoints utilizing an optimized UDP protocol specifically designed to decrease latency associated with TCP and the Internet.
Optimize In-Cloud Delivery
Akamai has a huge presence within most Internet points of presence. By its very nature, latency to an edge server from a client will be lower than CDNs utilizing centralized points of presence. In order to increase resiliency and lower latency throughout the delivery of streaming content, Akamai has the ability to utilize this distributed network to route around any issues that could cause buffering and stream failures.

Tuning of In-Cloud Connections
We will speak later about last-mile TCP optimizations, but many of these are entirely relevant within the Akamai CDN cloud. The Akamai media team have spent countless hours amassing best practices surrounding internal networking related to streaming. Network initialization timeouts, cascading timeouts, and edge timeouts all are highly dependent on the nature of the content being served. In the case of timeouts within the Akamai network, these must be tuned appropriately to match MPEG segment sizes. If segment sizes have been changed in accordance with recommendations within this document, it’s extremely important to have an Akamai engineer investigate and tune internal timeouts to appropriate values in order to prevent buffering.

CH (Cache Hierarchy) Failover
The premise behind CH failover is to react to changing network conditions between edge server(s) and parent server(s) as quickly as possible. Akamai and the majority of other CDNs utilize the same public Internet that end users traffic, except nodes are placed in highly reliable, well-connected data centers. However, it’s entirely possible to have issues affect one of these locations. In the case that a parent region is unable to respond to a request with an appropriate response, Akamai has a feature which can re-route the request instantly to an alternate parent map in real time. A map is Akamai terminology for a subset of nodes able to serve traffic. Though CDN networks are self-healing, it can typically take up to 5 minutes for this to happen. This can cause major issues especially during live events.

Implementing this particular failover behavior is not trivial, as we must fully understand the underlying CDN architecture so that a failover does not occur to the same troublesome region. Akamai architects can properly implement this strategy for a media property needing this sort of resiliency. Some of the largest events broadcast recently have used this approach with amazing results, resulting in greater than 99% availability.
Pre-Fetching of Segments

Segmented media delivery typically serves content with individual filenames that follow an incremental pattern. Due to this unique happenstance, we are able to actually predict with certainty the next piece of content the user will request and have it available in cache before it’s requested. As mentioned earlier, having content close to users greatly reduces latency, thus increasing download speeds and response times. An origin server might not always be responsive enough to serve live content without rebuffering, but with pre-fetch, we are able accommodate for this.

For Akamai’s Media Services Live product (version 4.0 and higher), pre-fetching has been proven to be very low risk. It’s worth noting, though, that when pulling live stream content from a non-Akamai origin server, the next segment might not always be available, causing pre-fetching to lose any benefit and instead generate unnecessary noise via 404 error (“file not found”) messages.

Example of Pre-Fetching:

A client (playback device) requests MPEG segment WalkingDeadEpisode1-1.ts. We know that the client will shortly request WalkingDeadEpisode1-2.ts, so we instruct the Akamai Edge node to request that segment (and possibly additional ones) during the initial client request of WalkingDeadEpisode1-1.ts. This workflow is further broken down below:

Pre-fetching is currently not built into any of Akamai’s product offerings, but can easily be added by Professional Services to an existing streaming configuration if it will benefit startup time and align with the end-to-end architecture.
The following diagrams illustrate the performance benefit of enabling pre-fetching for a media customer using HLS VOD.

**Utilize SureRoute with Cache Parents**

Akamai constantly maps various connections used throughout the Internet to optimize content for end users. Having this sort of data, we are able to harness all of this information to make intelligent decisions regarding routing traffic.

If there are connection issues such as slowness or even a fiber-optic loss between parent and origin, we can utilize SureRoute to route around those in real time. Putting in SureRoute between one of the CH tiers and origin server is a great solution to address connection issues between tiers.

Streaming is by nature very time sensitive. In the case that an origin server is located in the United States with a viewer in Stockholm, a packet might take one of several routes to receive the content needed. Akamai (as well as anyone on the Internet) routes via their uplink’s BGP routing table. BGP is usually configured to take cost of a link into account first, then speed of the link second. If X ISP peers for free with Y and pays for transit with Z, they would prefer to route traffic via Y instead of utilizing the faster paid link with Z.

SureRoute polls the origin server routes at a regular basis, making note of an optimal route at that particular time. Using this data, it can route traffic via our own nodes instead, providing a much more direct route.

Many readers are familiar with SureRoute pertaining to dynamic web content. This can usually be self-provisioned within Ion products, but it is usually utilized for optimizing edge-to-origin routing. SureRoute within a cache hierarchy works in a similar fashion, but optimizes routing from mid-tier to origin. It’s advisable to work with Akamai media consultants during design of a SureRoute-enabled hierarchy, as origin IPs and test objects must be known during implementation.
SureRoute does not have the ability to determine fastest path on a request-by-request basis, so depending on requirements, a quick retry implementation might be advisable as well. Using this, slowness can be detected based on very short sample times, and an alternate route can be utilized as needed.

Note: Since this article is mostly about lowering latency, introducing SureRoute with a two-tier CH and a live origin product might be not be beneficial for streaming services, because a three-tier design is then introduced, adding to overall latency. If utilizing live origin, please contact a member of your Akamai consulting team to discuss best practices.

**Take Control of the Last Mile**

With OTT video, many assume the last mile is out of control, but this is not the case. Through caching content close to end users and utilizing TCP optimizations, many of the issues that plagued video delivery over the Internet are now greatly reduced.

**Leverage Global Edge Network**

If streaming from a centralized endpoint, network latency increases proportionally to the distance of the consumer and number of network hops between the two. High bandwidth at the customer premise does not always equate to high transfer speeds.

Network latency greatly affects the available bitrate to stream content. Per [RFC 3390](http://www.ietf.org/rfc/rfc3390.txt), a realistic initial window size is 65535 bytes.

Using the formula:

\[ \frac{\text{TCP-window-size-in-bits}}{\text{Latency-in-seconds}} = \text{Bits-per-second-throughput} \]

We can see that with a 100-megabit link and 80-ms latency, the fastest we can expect to stream using an initial window size of 64 KB is 6.5 megabits. Until the sliding window increases, we will not have enough bandwidth to properly play back some HD content and all 4K UHD content. We will see lower-quality video and possible rebuffering until the window size increases.

By decreasing latency to 30 ms, we now are able to sustain 17 megabits during initial connection and play back all HD content and the majority of 4K content without rebuffering.

**What can be done to lower this latency?**

A highly distributed content distribution network such as Akamai’s eliminates the low transfer rates associated with high latency links. Other CDNs utilize “Super PoPs” as a solution, allowing an inflated number of nodes, but not solving the latency issue. By placing endpoints close to an end user and utilizing persistent connections, the slow start issue is solved. End users can optimally begin streaming content at the highest possible bitrate without having to climb the bitrate ladder as conditions slowly improve.
Optimize TCP

A huge factor in latency when requesting MPEG segments is the initial TCP handshake, which when utilized in HTTP connections can add around 300 ms to each request.

Many CDNs offer the ability for the client and server to maintain a persistent connection throughout the process, greatly reducing latency for each segment request. If a CDN is not used, the delay to the origin, plus the handshake for each connection, can sometimes add .5-1 second to each .ts request. When segments are 2 seconds long, this can greatly affect rebuffering.

As mentioned earlier, the initial TCP window is usually 64 KB. Creating a new session to an endpoint severely limits available startup bandwidth. Utilizing a CDN, it’s likely that a connection has already been established by a previous piece of content, so the connection window has already been increased and the connection is still active. Utilizing concepts such as FastTCP, Akamai can further optimize window sizing.

Utilize SDKs

In the case that many viewers reside in a similar location, such as a college campus or office complex, Akamai has published several SDKs available to accelerate delivery. Media Acceleration can utilize peer-to-peer technology to request portions of MPEG segments from nearby viewers with available content, thus keeping a great deal of traffic within a private network.

Additionally, Akamai provides SDKs to enable granular user behavior and stream metric tracking in the form of media analytics.

Akamai’s AMP (Adaptive Media Player) has many SDKs (such as Media Acceleration) pre-implemented. AMP maintains feature compatibility with many new Akamai products as they are released, ensuring that end users are able to easily experience all client-side optimizations as they are introduced. The player is easily integrated into modern browsers and mobile operating systems.

Additionally, developers will enjoy the ability to easily author AMP plugins and skins to add custom functionality or modify the look and feel.

Conclusion

Decreasing rebuffering and lowering latency is never a one-size-fits-all quick fix. There are many layers of the onion to peel back to ensure that a streaming offering is equal to or better than traditional broadcast methodologies. CableLabs and DVB have had years to optimize delivery, while real-time, large-scale Internet broadcasting is still maturing. Akamai and the industry as a whole have addressed many of the barriers preventing adoption of HD and 4K OTT content. Akamai is in the unique position to have worked with some of the largest VOD and live offerings to date, and our team of consultants (who can be reached at consulting@akamai.com) have the experience and know-how to make OTT video fast, reliable, and secure.
About the Authors

John Councilman is an enterprise architect at Akamai with years of experience in the video and Internet fields. Prior to Akamai, John worked on Ericsson’s Global Services team designing video solutions for some of the world’s largest media companies. At AT&T Labs, John was responsible for core network services design, including DNS, authentication, and lawful intercept. He has an extensive background in end-to-end solutions and is able to utilize this experience in a consultative manner. Along with his video experience, John carries a rich background of software development with him, creating many custom solutions for self-help services and back-end automation.

Karin is a senior enterprise architect at Akamai with 13 years of experience as a software developer, IT project manager, and IT architect. Before Akamai, Karin worked in Germany on multimillion-dollar budget software projects, and developed and designed software in different computer languages and frameworks. At Akamai she specializes in Digital Media and is working on the biggest streaming events. Karin is a German engineer that likes to be challenged with complex requirements or problems.