ADAPTIVE STREAMING AND CONVERGED MANAGEMENT STRATEGY IN MULTISCREEN VIDEO SERVICE IMPLEMENTATION

Duncan Potter, Goran Appelquist
Edgeware AB

Abstract

With the massive proliferation of both video services and number of devices capable of consuming high quality video, the question of whether or not a cable operator should deploy a multiscreen environment is now one of survival rather than just opportunity.

This paper addresses 2 main areas of complexity:

1/ The use of HTTP Adaptive Streaming frameworks and their associated protocols in a distributed network environments

2/ The scaling of a management architecture to support this huge increase in both sessions, and traffic, and possible routes to further monetization

INTRODUCTION

Traditionally, telco and cable operator‟s video services have been distributed over managed access lines where the bandwidth required for a good quality of experience has been provisioned and is suitably robust. However, there are now a huge range of Internet connected devices available which are capable of high quality video playback. These include laptops and home media centers, smartphones such as the Apple iPhone, Blu-ray devices, and gaming consoles. These devices are typically connected to unmanaged access networks such as 3G, home networks and Wi-Fi hot spots.

In addition, video content owners are increasingly choosing to make their content available directly on the Internet via massively popular services such as the BBC iPlayer and Hulu™. Delivery of these services is typically handled by Content Delivery Networks (CDNs) such as Akamai and Limelight Networks® that deliver “Over The Top” of operator networks, leading to the description “OTT video services”. Although CDNs optimize delivery over the transit network, these services are all affected by varying degrees of congestion when they reach the local operator‟s network – the so called “middle and last mile”.

However, demand from consumers to watch video anytime, anywhere has led to an urgent requirement for operators and CDNs to be able to deliver video services to these devices with a high Quality of Experience (QoE). A number of leading companies have developed HTTP Adaptive Streaming technologies to specifically enable this including Microsoft®, Apple and most recently Adobe®.

In order to scale a system for providing services to this plethora of devices there are many components required.

1. HTTP ADAPTIVE STREAMING FRAMEWORKS AND PROTOCOLS

Issues Addressed by HTTP Adaptive Streaming

Network Connectivity and Traversal Assurance Requirement

When attempting network connectivity in an unmanaged network, there are a lot of unknowns. The unknowns include the potential existence of routers, firewalls as well as which ports are open. In a home network, there are personal firewalls, possible routers and security software scanning port activity.

In a Wi-Fi hot spot, the port access can be extremely limited due to security concerns.

This is a well-known hurdle with network applications and is overcome by using the HTTP protocol for communication. HTTP uses port 80 for requests. Requests to this port are most likely to be allowed through any firewall or router as they are used for all web surfing. As HTTP uses a state-full TCP connection, any issues that can be incurred by NAT based networks are also overcome.

Bandwidth Management Requirement

Bandwidth consistency is a major issue. If a user is watching a video and someone else on the same network suddenly decides to perform a file transfer, the available bandwidth for the video can be severely impacted. In order to maintain a good Quality of Experience, content therefore needs to be encoded at different bit rates and the delivery protocol needs to be able to dynamically switch the bit rate with no interruption in playback or action by the user.

The HTTP protocol is a synchronous client-to-server protocol so only one request can be made for a video file. Switching bit rates on the fly is therefore not possible in the middle of an HTTP transaction. To overcome this problem, the video must be sliced up into "chunks." Each chunk is typically between two to ten seconds of video. The chunk sizes are such that the reference IFrame at the beginning of each chunk is synchronized.

The delivery server can host several different bit rate encodings of the same video content. Each bit rate encoding has a separate playlist, which is defined by a master playlist. These playlists are typically in an M3U format and contain the list of chunks in order. When the client detects either insufficient bandwidth or more available bandwidth, it can switch to either the lower or higher bit rate playlist and download the chunks in that list. Since each chunk is synchronized with the other bit rate streams, there is a seamless transition between them so that the video playback is not interrupted. This maintains a high quality user experience.

Multiple Clients and Resolutions

As more and more Internet connected devices appear on the market every month, with greater and greater capabilities for video browsing and playback, the number of resolutions and bit rates required to support these devices increases exponentially. Historically, many devices have also communicated via a proprietary protocol. It is not viable to have separate encoders, DRM systems and delivery servers for each device that needs to be supported.

HTTP Adaptive Streaming enables delivery of multiple resolutions and bit rates over a common, open protocol. This enables consolidation of the encoder, encryption and delivery server infrastructure to a single manageable system. New devices can be added simply via a new encoding profile.

Content Security

The traditional broadcast method of restricting access to live content is achieved by encrypting the transmission stream. This method has been successfully applied to delivery of live video over the Internet: content is encrypted between the encoder and the server and then again between the server and the client. However, the risk with this approach is that content is being temporarily written to cache in the server before immediate retransmission and again is written to the client cache before immediate presentation and deletion. During these short periods, content is sitting „in the clear" and this presents a security risk. Also, for VOD and catch-up TV services, content will be stored on servers in the network or client

devices and must remain protected to avoid piracy.

HTTP Adaptive Streaming enables encryption of the individual chunks of content for both live streaming and VOD / catch-up TV downloads. The content itself is encrypted during or immediately after encoding and remains so during transmission across networks and when stored on servers or client devices.

HTTP Adaptive Streaming Ecosystem and Architecture

Support within an Asset Caching & Propagation System

Microsoft® Silverlight®, Apple Quicktime and Adobe® Flash® are the leading frameworks for the creation and playback of Rich Internet Applications incorporating video. These include server programming tools for the creation of the presentation portals, tools for the creation of encoders to compress and format video for Internet delivery and client programming tools for creation of video players on different Internet connected devices.

All frameworks now support variants of HTTP Adaptive Streaming. The following describes how an asset caching and propagation system might support these variants:

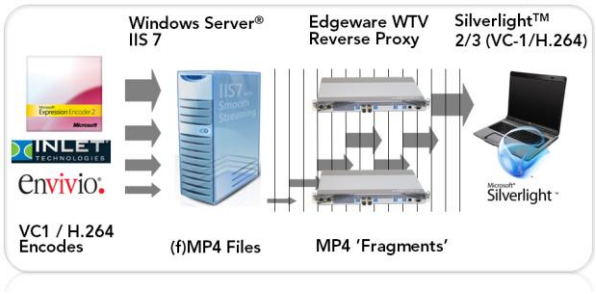
Microsoft® Smooth Streaming

The video delivery server acts as a reverse (server side) proxy to a Windows Server® running Internet Information Service 7 with the Media Service 3.0 extension. When running in this mode, caching replaces the need for proprietary asset propagation and this is therefore disabled.

The following is a description of the Smooth ecosystem and content flow:
Microsoft® or 3rd party encoders generate a single contiguous file per bit rate according to the ISO/IEC 14496-12 ISO Base Media File Format (MP4) specification. The files have the ***.ismv** extension and contain MP4 video fragments (and audio fragments if the video source also contains audio).

An XML-based server manifest file and a client manifest file is also generated. These describe available bit rates and other information required by the IIS 7 server and the Silverlight clients

For on-demand Smooth Streaming, following encoding, all these files are copied to the IIS 7 server or are published if WebDAV is enabled. For Live content, rather than storing the fragments in MP4 containers, encoders deliver the fragments directly to Live Smooth Streaming publishing points on the IIS 7 server. The server itself then generates a manifest for clients, based on information provided to it by the encoder.



Smooth Ecosystem and Content Flow

Requests for on-demand content stored on the IIS 7 server or for live content that is being delivered to a Live Smooth Streaming publishing point on the IIS 7 server should be directed to the distributed video server E.g. a request for an on-demand asset stored on the IIS 7 server with the following URL *http://iismedia7/BigBuckBunny/default.html* should be directed to the IP address of the Edgeware server.

In addition, the Client Cache Settings of the IIS 7 server should be enabled to specify the Cache-Control header. This is used in the IIS

7 server to specify any intermediate caches the conditions and restrictions for caching of content.

Requests for live and on-demand Smooth streams on the IIS 7 server are proxied by the distributed video server, ingested via HTTP and cached according to the specified cache Control directives. For a fragment that has been cached, subsequent requests are served directly from the distributed video server.

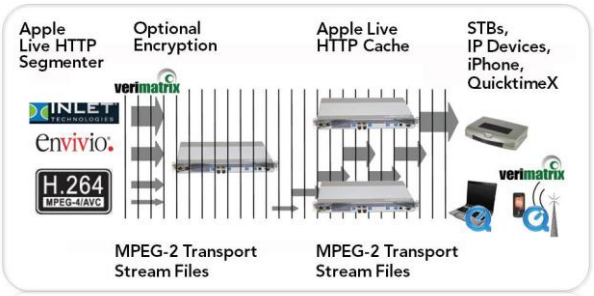
Apple Live HTTP Streaming

There are two modes of operation when supporting Apple Live HTTP streaming with distributed delivery servers: one for non-encrypted content and one for encrypted content.

For encrypted content, the distributed video server does not act as a reverse proxy but instead receives segmented content pushed directly from an encoder or content store. The architecture is very similar to Smooth Streaming except the (f)MP4 files are replaced with MPEG-2 Transport Streams (*.ts extension).

A segmenter creates and maintains an index file (*.M3U8 extension) containing a list of the media files. Apple provides a software segmenter but typically this is included as part of the encoder functionality. The index file and the stream segments are all pushed to the first server using WebDAV.

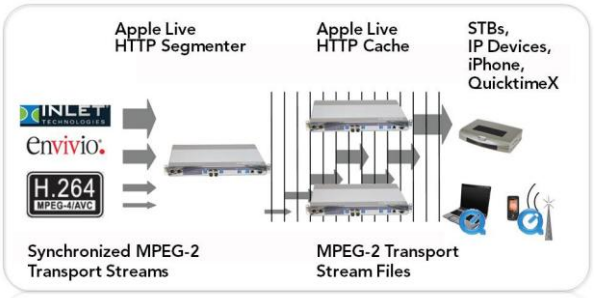
Additional distributed video servers can be added as edge caches to this first server and distributed asset propagation system can be enabled to download transport stream segments according to their popularity.



Mode 1 – Encrypted Apple HTTP Ecosystem and Content Flow

For non-encrypted content, encoders deliver a single Transport Stream per bit rate without segmentation but with synchronized IFrames. These can be ingested by the Web TV server via FTP, HTTP or even scheduled UDP multicast for live transmissions. This option significantly reduces core network bandwidth, especially if live content is to be distributed to multiple servers.

A centralized server can then perform the segmentation and create the index file. Additional Servers can be added as edge caches to this first server and asset propagation systems can be enabled to download transport stream segments according to their popularity. Alternatively, each edge server can ingest complete transport streams via scheduled multicast or according to popularity via the asset propagation and it can then perform segmentation on the fly.



Mode 2 – Non-Encrypted Apple HTTP Ecosystem and Content Flow

Requests for Apple Live HTTP files should be directed to the IP address of the appropriate server. Client software first reads the index file, based on a URL identifying the stream. This index specifies the location of the available media files, decryption keys (if applicable), and any alternate streams available. For the selected stream, the client downloads each available media file in sequence.

This process continues until the client encounters the #EXT-X-ENDLIST tag in the index file. If no #EXT-X-ENDLIST tag is encountered, the index file is part of an ongoing broadcast. The client loads a new version of the index file periodically. The client looks for new media files and encryption keys in the updated index and adds these URLs to its queue.

Conditional Access Solution for Enhanced Apple HTTP Live Streaming

The Apple HTTP Live Streaming Protocol provides optional encryption of the video chunks using the AES-128-CBC block encryption algorithm. During encryption, a 128-bit key is generated and placed in a "key file" on a server so it can be downloaded by the client. The location of the key file is given in the playlist. A new key can be given at any time. When a key file is encountered in the playlist, this key must be used to decrypt each subsequent chunk until another key file is encountered.

The URL provided to the client to retrieve the key file is HTTPS. This ensures that the connection to the server will at least be encrypted, even if it does not have server-side authentication. However, the protocol does not provide a way to ensure mutual authentication of the HTTPS session. There is no client-side certificate provisioning built into the protocol and there is no mechanism for the reporting of a unique client identifier from the device to the server. The lack of either of these features does not allow for enforceable entitlements against the video content encryption key.

To overcome these security limitations and enable HTTP lives streaming video delivery suitable for a secured pay-TV service, the distributed delivery supplier must have confirmed interoperability with a security solution such as that provided by Verimatrix. Using such a system, the 128-bit keys are managed and selectively distributed to the encoder and authorized clients only and can support HTTP live streaming in a standalone OTT service configuration, or can be utilized as part of a unified security head-end supporting multi-screen deployments pairing OTT delivery alongside IPTV and DVB content distribution. Currently, to support this additional security, the encoder must perform the segmenting of the chunks prior to encryption and distribution to distributed video delivery servers (Mode 1).

Adobe® HTTP Dynamic Streaming

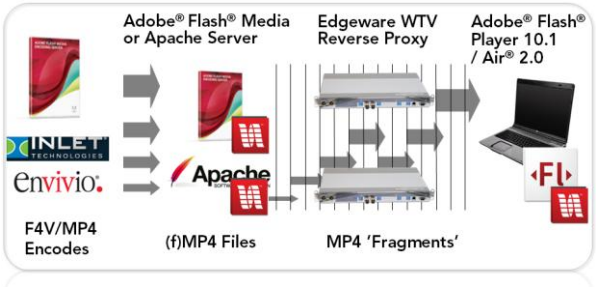
The Web TV server acts as a reverse (server side) proxy to an Apache Server running the Adobe® Origin Module (VOD) or an Adobe® Flash® Media Server 4.0. When running in this mode, caching replaces the need for asset propagation system and this is therefore disabled.

The following is a description of the Adobe® HTTP Dynamic Streaming ecosystem and content flow:

Adobe® or 3rd party encoders generate a single contiguous file per bit rate using the following file types: F4V/MP4 compatible files and FLV. For VOD content, an Adobe® File Packager command line tool is used to parse the file and translate it into fragments. This tool can also be used to encrypt files for use with Adobe® Flash® Access 2.0 DRM. Once created (and encrypted if used) fragments are written to files with the ***.f4f**

extension. Each file can contain multiple fragments and quantity and duration of fragments per file can be optimized.

An XML-based client manifest file is created with the filename of the input file. This contains information about the codec, resolution and the availability of multi-bitrate files. In addition, a server index file is also generated. This contains information on the specific location of fragments within a file for the Origin Module to translate to byte range requests.



Adobe® HTTP Dynamic Streaming Ecosystem and Content Flow

For VOD, requests for content stored on the Apache Origin server should be directed to the distributed video delivery server. E.g. a request for an on-demand asset stored on the server with the following URL http://localhost/media/webplayer.html should be directed to the IP address of the Edgeware server.

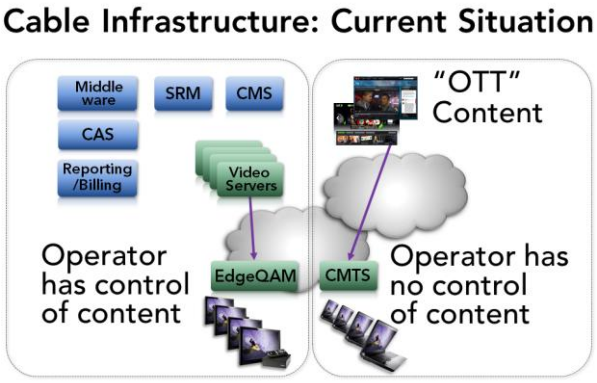
In addition, the Client Cache Settings of the Apache server should be enabled to specify the Cache-Control header. This is used in the Apache server to specify any intermediate caches the conditions and restrictions for caching of content.

Requests for on demand streams on the Apache Origin server are proxied by the distributed video delivery server, ingested via HTTP and cached according to the specified cache Control directives. For a fragment that has been cached, subsequent requests are served directly from the distributed video delivery server.

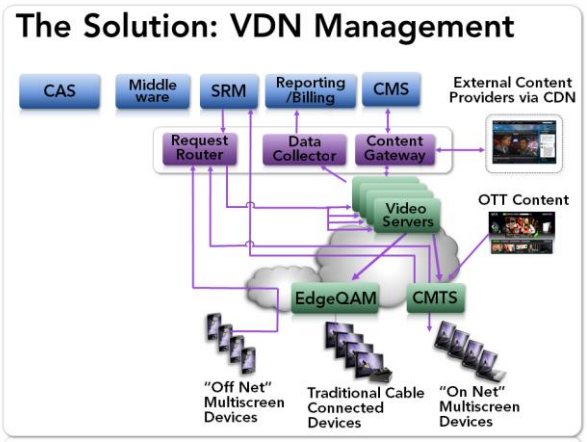
For live streams, the Apache Origin is incorporated into the Flash® Media Interactive Server (FMIS). Live streams are ingested over Real Time Messaging Protocol (RTMP) and segmented into *.f4f files. The FMIS server has a built in Apache HTTP Server and uses the Origin Module to deliver the live content over HTTP. Requests for live streams are proxied by the distributed video delivery server and cached in the same way as on demand streams.

2. CONVERGED SCALABLE MANAGEMENT ARCHITECTURE

In many cases the current cable environment provides sophisticated management of the video infrastructure but little opportunity to extend access to the video assets outside the closed cable environment. The challenge is not necessarily to provide access to the video assets but to provide access without building completely parallel infrastructure and management systems. To achieve this, it is important to identify the key management components of an existing system and then to propose the minimum changes to the architecture to provide integrated management of both traditional cable and multiscreen environments



In the new converged management environment, an additional layer is inserted that will be referred to as the Video Delivery Network (VDN) management layer.



It is made up of 3 main components:

Request Router (a.k.a. Session gateway)

The request router is responsible for receiving the request from one of 3 potential sources:

Existing Session Resource Manager (SRM)

An existing SRM system would forward a request from an existing Set Top Box (STB) using a standard protocol such as Real Time Streaming Protocol (RTSP) or (in the US) Streaming Control Protocol/ Lightweight Stream Control Protocol (SCP /LSCP).

"On Net" Multiscreen web device

On the operators own IP network, a device would access the video services through a middleware interface but in effect would connect to the video services through the Request Router (RR) using an HTTP request. The RR would then redirect the device to the nominated video delivery server, which would provide the requested content.

"Off Net" Multiscreen web device

There is also a possibility that devices connected to an alternative network such as a wireless operators network would also want to access the video services. Assuming that peering agreements are in place, the RR would also be responsible for ensuring access to the content for off net devices.

Decisions on which video delivery server to connect to could be made via a range of selection criteria, such as geographic lookup, IP range or service group depending on the operators own criteria.

Content Gateway

The content gateway is responsible for the interface between the Video Delivery Network subsystems, the existing CMS system and also providing an interface to external content providers such as CDNs.
It is responsible for content ingest for both LiveTV and VOD content

NPVR functionality is also provided by the content gateway with middleware providing the user interface to allow users to schedule their own recording schedules, which are then provided in a standard format to the content gateway.

Data Collection and Reporting system

The Data Collection and Reporting system is vital to the revenue generation element of the system. It is responsible for 4 main elements:

Usage statistics

This is the collection of statistics showing which assets are being viewed, by which subscribers, at what time, for how long and through which viewer.

Performance monitoring

In an adaptive streaming environment it is vital that not only viewing behavior can be monitored but also the delivered quality is monitored as well. Unlike a traditional cable environment, adaptive streaming works by providing different quality video and the ability to automatically adjust based on

bandwidth availability. It is therefore essential to be able to understand the actual quality of the video being delivered by the video delivery system and the underlying network infrastructure.

Fault Monitoring

Monitoring of errors, faults and overall loss or degradation of the hardware and software components of the system.

Reporting

This part of the system is also responsible for providing all statistics either through a northbound interface to an ISA or NGOD compatible system, a proprietary API or through a provided graphical user interface.

Note: Security and Conditional Access

Security and conditional access falls outside the scope of this paper, being a significant subject in its own right. It is suggested that anyone looking in to this area must consider it as an integral part of the system and therefore should seek assurance of the interoperability of a system such as the Verimatrix VCAS [1] system with the chosen management framework

3. SUMMARY

This paper addresses two of the most common questions and challenges for a cable operator contemplating a multiscreen approach to serving new and existing subscribers. There are of course many other considerations that fall outside the scope of this paper.

Understanding of adaptive streaming protocols and their implementation is key to successfully planning an implementation. A converged management strategy is essential to maintain the lowest possible TCO and efficiency while launching, maintaining and growing profitable video services to an expanding subscriber base.

4. REFERENCES

[1] Verimatrix HTTP Live Streaming Interface Control Document (ICD).