WIDEVINE®

CYPHER FOR DIGITAL MEDIA

OVERVIEW OF WIDEVINE

VERSION: 1.3

Widevine Technologies

901 5th Ave, Suite 3400

Seattle, WA 98164 USA

www.widevine.com

206.254.3000 voice

206.254.3001 fax

sales@widevine.com

Revision History

Version	Date	Description	By
1.0	10/14/2010	Initial 4.5.0 release	Alex Lee
1.1	10/26/2010	Clarification on HLS support	Alex Lee
1.2	02/24/2011	Revision	Alex Lee
1.3	02/27/2011	Revision	Alex Lee

CONTENTS

1. PURPOSE

The purpose of this document is to provide an introduction to the Widevine system, its operation and products.

Specifically, describing the operation of the Widevine VOD and Live solutions for video delivery over the Internet.

2. Terms and Abbreviations

- **Asset** – A single video program protected by the Cypher system
- **CCI** – Copy Control Information – A data field specifying restrictions on how content may be used.
- **Client Device** – Equipment used to view or otherwise use protected video content
- **Container** – The data file format in which the protected content representing an asset is stored
- **Content Owner** – The creator or owner of rights to the video program comprising an asset
- **Cypher Activation Server** – Server responsible for client device and account management
- **Cypher CA** – Cypher Certificate Authority is a server responsible for license distribution and system security management
- **Cypher Encryptor** – Server responsible for packaging and protecting content
- **Deployment** – The collection of Cypher servers and other components that enables merchandising and distribution of protected content (assets) using the Cypher technology
- **Deployment Operator** – The entity responsible for operating the servers in a deployment and managing subscribers and devices.
- **ECM** – Entitlement Control Message – A protected data structure holding the cryptographic key(s) that protect the content for an asset stored in a container and the original usage rules that govern the asset's use. A client device must have obtained an EMM for the asset in order to be able to decrypt the asset's ECM.
- **ECMg** – ECM Generator – The component of a Cypher Encryptor that securely generates ECMs
- **EMM** – Entitlement Management Message – A protected data structure holding the cryptographic key(s) that protect an asset and, optionally, updated usage rules to govern the asset's use. Each EMM is cryptographically protected so that it can be used only by a single client device.
- **EMMg** – EMM Generator – The component of a Cypher CA that securely generates EMMs
- **MEMF** – Multipart Encrypted Media File – A container format capable of transporting multiple versions of the same content encoded in different ways and/or at multiple bit rates to enable adaptive streaming and trick play. The format of this file is a MPEG2-PS.
- **Metadata** – Information, supplied by the content owner, in a container that describes the content in the container and the usage rules that apply to it.
- **Secure SoC** – Secure System on Chip – An SoC that contains hardware security features suitable for establishing a trusted domain for processing protected content
- **SoC** - System on Chip – A single-chip microcontroller that provides the full set of processing capabilities and interfaces necessary to implement a client device.
- **Widevine Adaptive Container** – see MEMF.

3. Overview

Widevine provides a comprehensive secure digital content delivery solution in 3 categories.

3.1. Digital Rights Management

Widevine's proprietary DRM technology is studio approved and DECE certified. It is sufficiently robust for both SD and HD content delivery methods.

3.2. Video optimization

This covers a range of features built into every Widevine client on every supported platform.

3.2.1. Adaptive streaming

The ability to continue content playback in fluctuating network conditions, via reducing or increasing the quality of the displayed content. The adjustments and transitions are seamless to the end user without additional overhead to the content servers.

3.2.2. Fast startup on playback

Online streaming playback is normally within 2-3 seconds (a combination of available bandwidth and adaptive streaming specifications).

3.2.3. Trickplay

Provide performance of fast forward or rewind actions, similar to the end-user experience with a physical disc (DVD, Blu-ray).

3.2.4. Content delivery method

Widevine packaged content is delivered via HTTP protocol for maximum compatibility and network penetration. The only requirement is to comply with HTTP 1.1 standards.

3.3. Device and platform support

The Widevine client is available on a wide range of devices from TVs, Bluray players to mobile phones.

4. INTRODUCTION TO DRM

The following are excerpts from Widevine's Theory of Operations document.

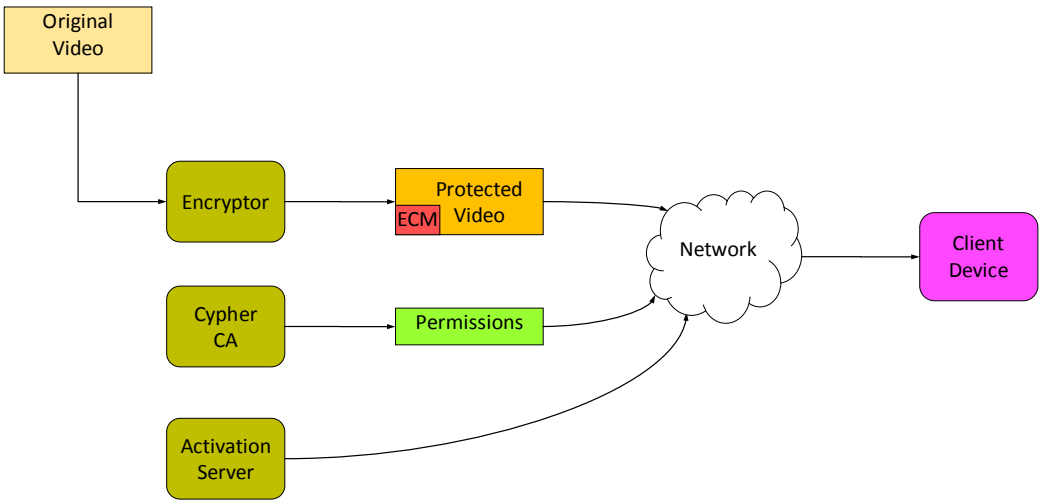

FIGURE 1 – WIDEVINE CYPHER SYSTEM OVERVIEW

Figure 1 shows the major Widevine Cypher components[1] and data flows in the system. Video content is encrypted and packaged by a Cypher Encryptor. The encrypted content is distributed, by streaming or download technology, to a secure client *device*. The device obtains *permissions* from a Cypher CA security appliance, and renders the video for user interaction. The device, which is a secure client for video presentation, is *activated* through interactions with a Cypher secure server. This figure does not show the content distribution and vending infrastructure into which the Widevine solution is integrated, as that can have quite different structures in different deployments.

As is the case with any content protection system, the client device security is at the heart of the Widevine system, since a client device may be in the hands of an adversary attempting to extract the content, violate business model rules, or otherwise harm the system. Client device security is based on a security architecture described later in the Theory of Operations document.

[1] Note that this diagram shows only the Widevine components, not the components operated by the deployment operator to enable merchandising and distribution of content.

Different classes of client devices provide different levels of security. Typically, client devices in the Cypher system are dedicated video devices, such as set-top boxes, video disc players, HDTVs, etc. Such dedicated devices typically have robust hardware security capabilities and can support all types of off-line operation. General-purpose open platform such as Windows and Macintosh are also supported as Cypher clients, but because their security is less robust, there may be classes of content or operations that are not permitted. However, although technically less robust, DRM implementations on open platforms are currently considered robust enough for the majority of commercial content. A broad category of quasi-open devices, such as game consoles, smart phones, etc., is also supported. The relatively closed nature of the hardware and software environment of these devices, plus (in some cases) their inherent hardware security capabilities, allows them to provide better protection than completely open platforms.

Because the protected video content is encrypted, it can be stored and distributed on unsecure media and networks. When it is processed by a client device, it can be decrypted only if the client device has obtained appropriate permissions. The Widevine client software and the client device robustness rules are responsible for ensuring that the decrypted content cannot be accessed directly, and that it is processed in accordance with any specified restrictions (such as limitations on copying or video output).

Operation of the server components in the system (encryptor, Cypher CA, etc.) is also security-critical. These components are protected because they run in physically protected environments and are configured to take full advantage of their platforms' security capabilities. Secure, encrypted, communication protocols (including the Widevine-specific Secure Message Manager protocol) are used for communication with these components, providing protection for data traversing potentially untrusted (or less trusted) networks.

Although client devices must interact with servers to obtain permissions initially, server interactions are not always required for the system to operate. For example, a device with appropriate security capabilities (typically found in any dedicated video player device) can store permissions locally and use them without further interactions. It is important to note that in certain cases the server may actually be another locally connected device such as a USB flash memory device or SD card. Since content is implicitly protected, it can be stored at a client device regardless of device security capabilities.

The Cypher system includes integration APIs and components for creation and management of encrypted content, for creation and distribution of permissions, and for integration with client device software.

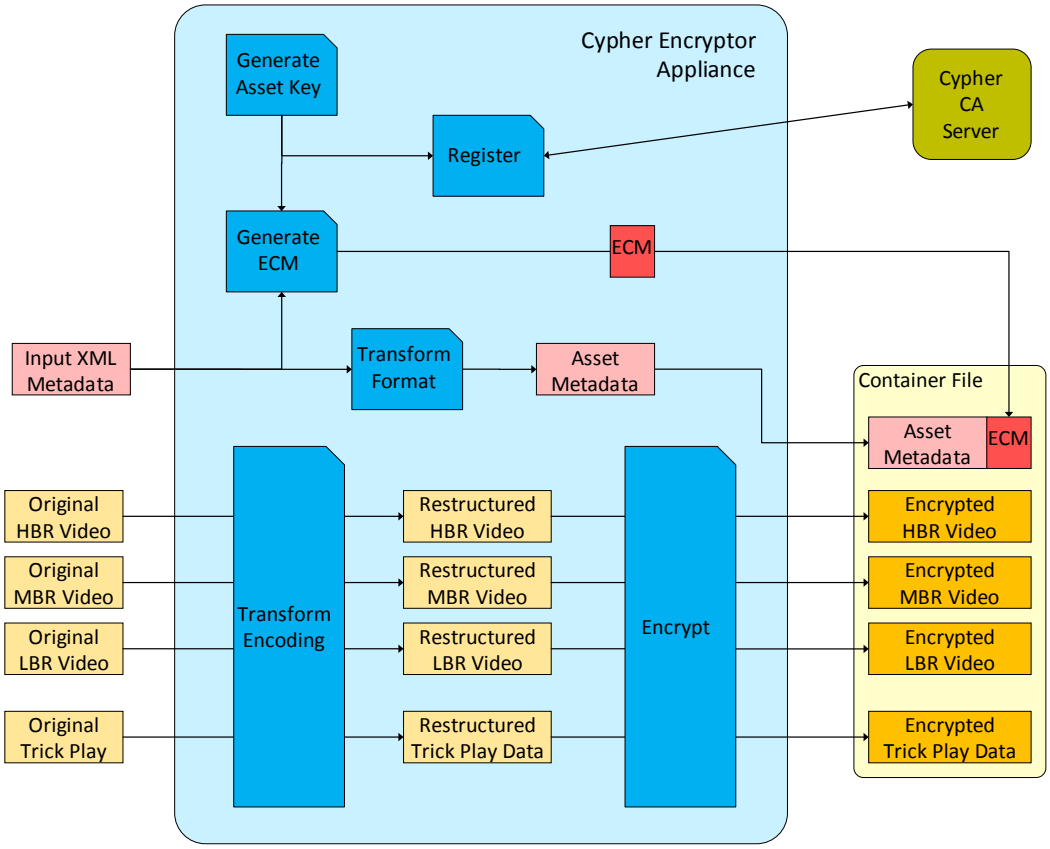

FIGURE 2 – VOD ENCRYPTION

To create a container file, the Cypher Encryptor first generates the keys that are used to protect the video/audio asset that the container represents. The encryptor generates an *asset* key that protects the entire asset, and one or more *content* keys that are used to encrypt the video data itself. Each content key is placed into an entitlement control message (ECM) that is encrypted with the asset key and placed into the container file. The asset key is *registered* with a Cypher CA server, which will use it later to create permissions.

The input metadata is transformed from the XML representation received as input into binary data for the container file. Some of the input metadata (license and other restrictions) is encoded in the ECM.

The primary security-critical data in the container consists of entitlement control message (ECM) packets. Each ECM contains the asset ID, policies for content usage (e.g., CCI, DCP; region restrictions; see section 3.3), and the relevant content key, encrypted with a unique *asset key*. In addition, to ensure that ECMs cannot be modified (and thus, for example, change the policy rules they specify), each ECM is digitally signed. In typical streaming and download scenarios,

there is one ECM for the container file; in other operational scenarios (such as broadcast) there may be multiple ECMs embedded throughout the content streams.

In cases where appropriate, the Widevine MEMF is used as the container. The MEMF file is a proprietary format based on the standard MPEG-2 Program Stream (PS), which facilitates selection and sequencing of content from multiple streams. It incorporates Cypher-specific metadata in a manner compatible with MPEG-2 standards and practices.

The Widevine MEMF is often used for on-demand applications, which need the benefit of Widevine's video optimization solutions. It also may be use to simplify content packaging, distribution and storage.

Refer to the VOD section in this document for additional clarification.

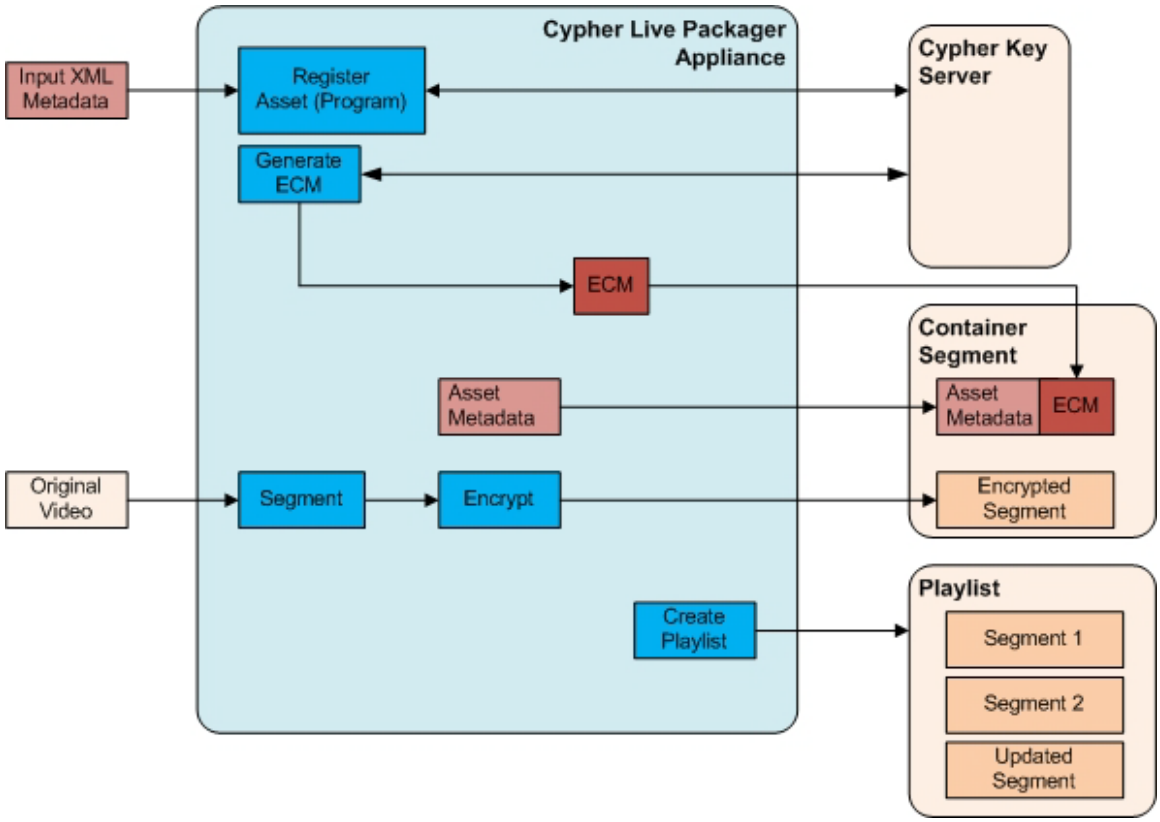

FIGURE 3 – LIVE STREAMING ENCRYPTION

Refer to the Live Streaming section of this document for additional clarification.

5. VOD OVERVIEW

The diagram below illustrates the relevant flows from content encryption to decryption.

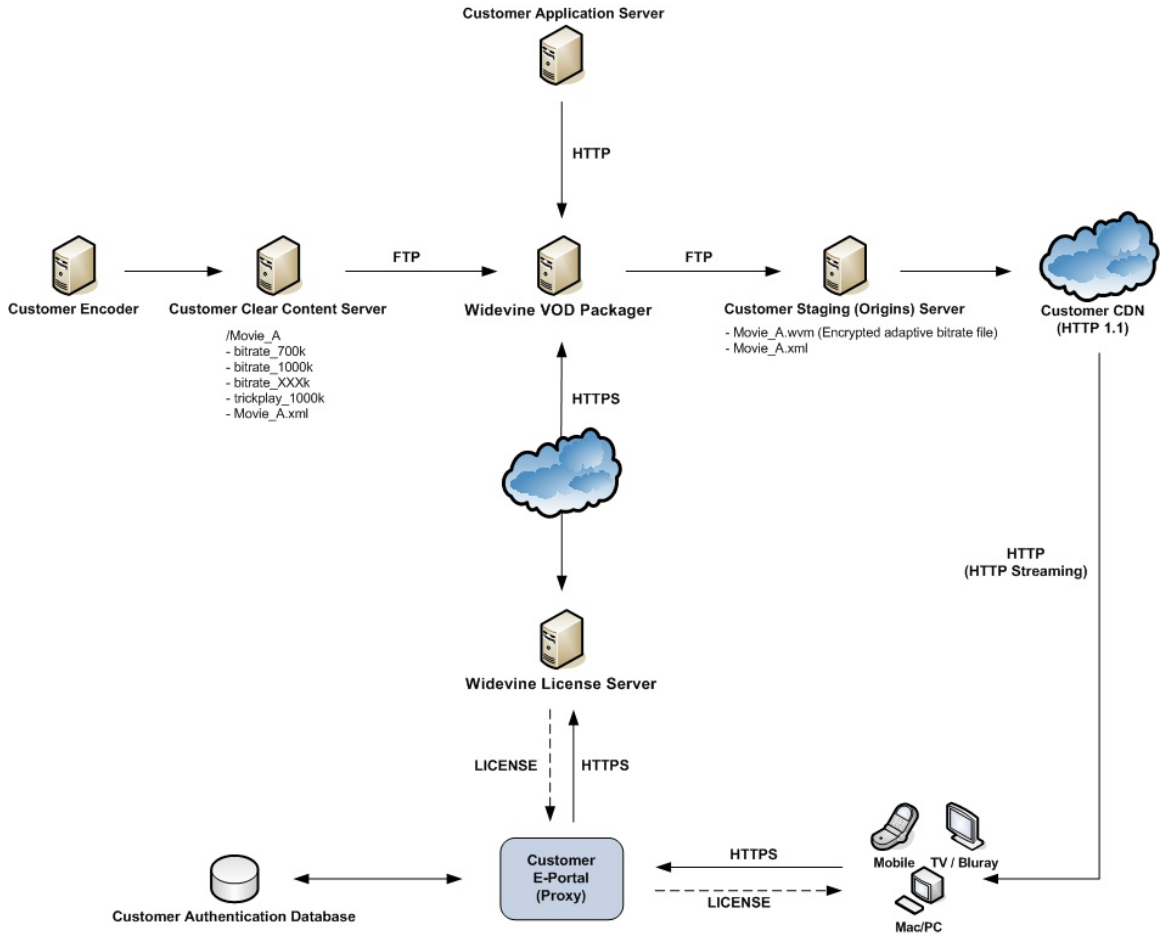

FIGURE 4 – VOD OVERVIEW

5.1. CONTENT ENCRYPTION

Content generated according to Widevine specifications (see relevant Encoding Specification document), is processed by a VOD Packager process.

5.1.1. SINGLE BITRATE (NON-ADAPTIVE)

This legacy packaging method is employed for selected platforms where the use of the adaptive format is untenable.

The single bitrate MP4 file is encrypted 1:1 into a resulting encrypted MP4 file. The overhead added to the MP4 headers vary, normally in the 200 byte to 1kb range.

The drawback with streaming of the MP4 container format reduces the impact of the video optimization technologies employed by Widevine.

5.1.2. ADAPTIVE

One or more bitrates are packaged and encrypted into the Widevine Adaptive Container (MPEG2-PS) with the file extension (.wvm).

All specified bitrates are concatenated into the single file format. The sequence of the concatenation is determined by the bitrate order during package encryption.

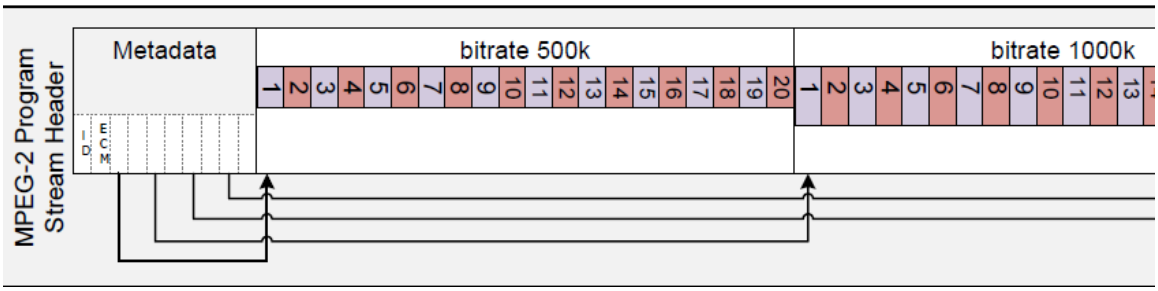

FIGURE 5 – WIDEVINE CONTAINER (BEGINNING)

The metadata located at the beginning of the file describes the available bitrates as well as other information identifying the asset.

If a trickplay file is specified during the encryption process, the I-frames are extracted and stored within the Widevine Adaptive Container for fast forward and rewind.

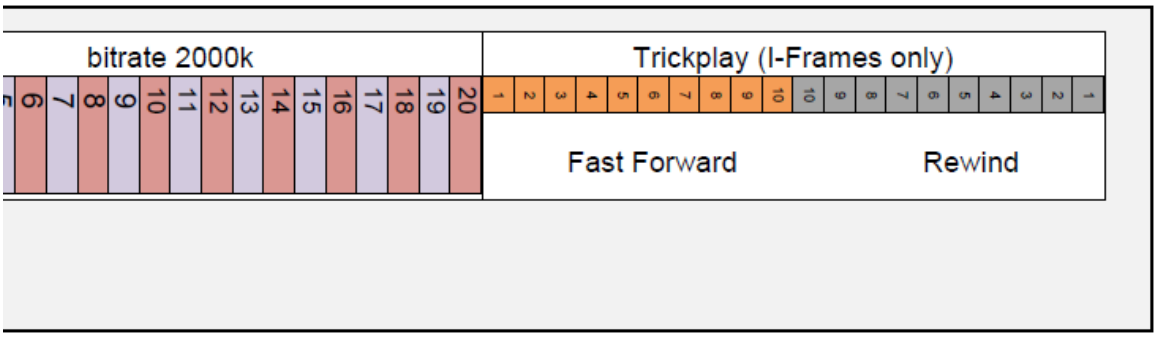

FIGURE 6 – WIDEVINE CONTAINER (END)

The encrypted package size is approximately the sum total of all bitrates (not including trickplay file) + 1%.

5.2. MULTIPLE VOD PACKAGERS

Multiple VOD Packagers can operate in parallel for scalability and improved processing.

If package duplicate checking or querying is required across packagers, the use of the Widevine multicast 233.126.125.17 is required.

Assuming all VOD Packagers reside on the same subnet or network with multicast permitted using the Widevine multicast IP, every package API action (notify, query and so on) will prompt the selected VOD Packager to send a multicast message to its counterparts for verification.

Normally, a customer-provided load balancer is placed in front of a VOD Packager cluster to distribute the processing evenly across all nodes.

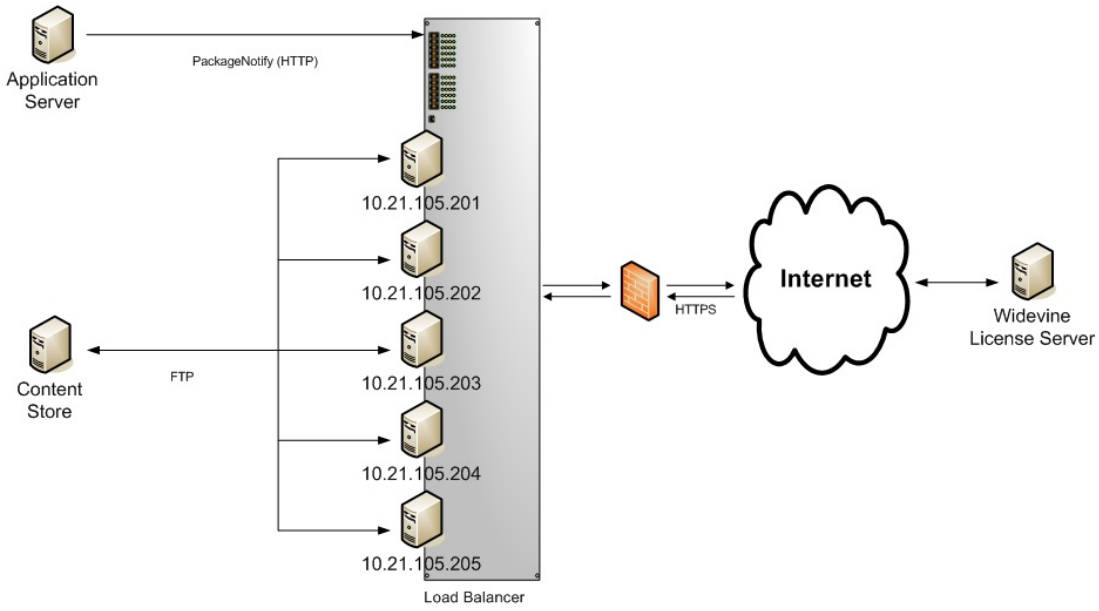

FIGURE 7 – LOAD BALANCED EXAMPLE

5.3. Content Staging

The encrypted VOD assets are propagated from a staging (or origins) server for HTTP streaming delivery to the client devices.

5.4. Content Management

The management of the VOD Packager is via the UI installed on the same system. This user interface manages both assets and policies.

5.5. Content Decryption

See section below on license acquisition.

5.5.1. Single bitrate (non-adaptive)

After the decryption of the content within the Widevine client (library), the content is delivered to the platform's media player via an internal HTTP loopback.

This is transparent to the application and end user.

The figure below illustrates the operation of the HTTP loopback within the iOS client.

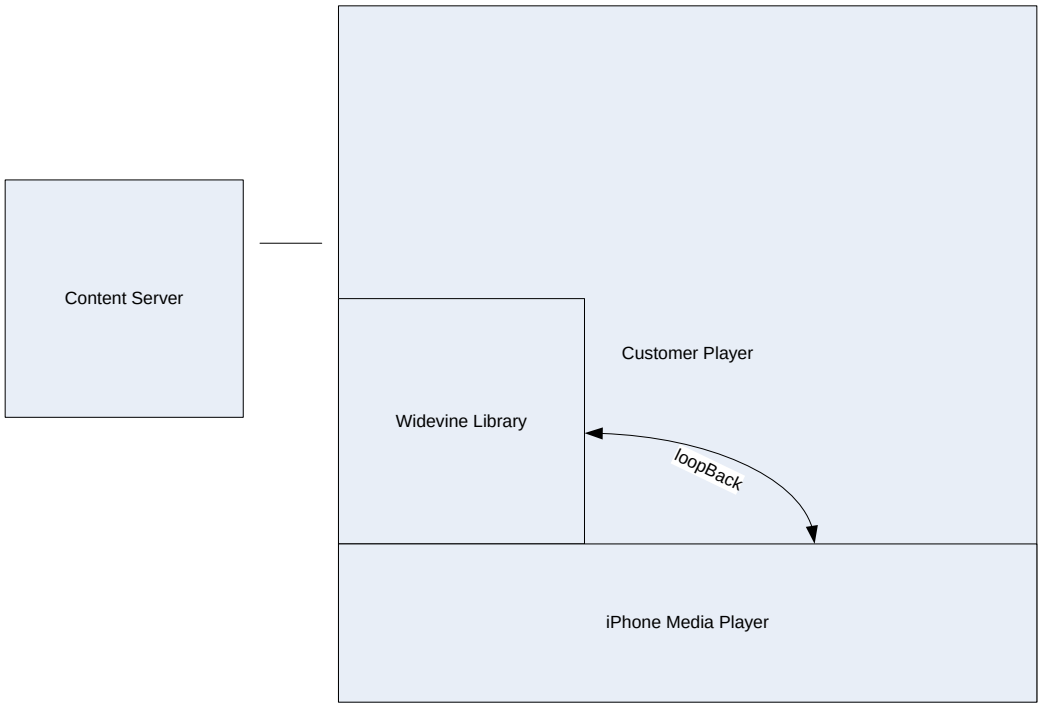

FIGURE 8 –IOS CONTENT VIA HTTP LOOPBACK

5.5.2. ADAPTIVE

The delivery mechanism of both the content and the license (to decrypt the content) to a client device is over standard HTTP (version 1.1) or HTTPS connections.

A client device refers to a Widevine-enabled platform with an application enabling playback of Widevine-encrypted content.

The diagram below graphically illustrates a playback sequence with adaptive bitrate switching due to fluctuating network conditions.

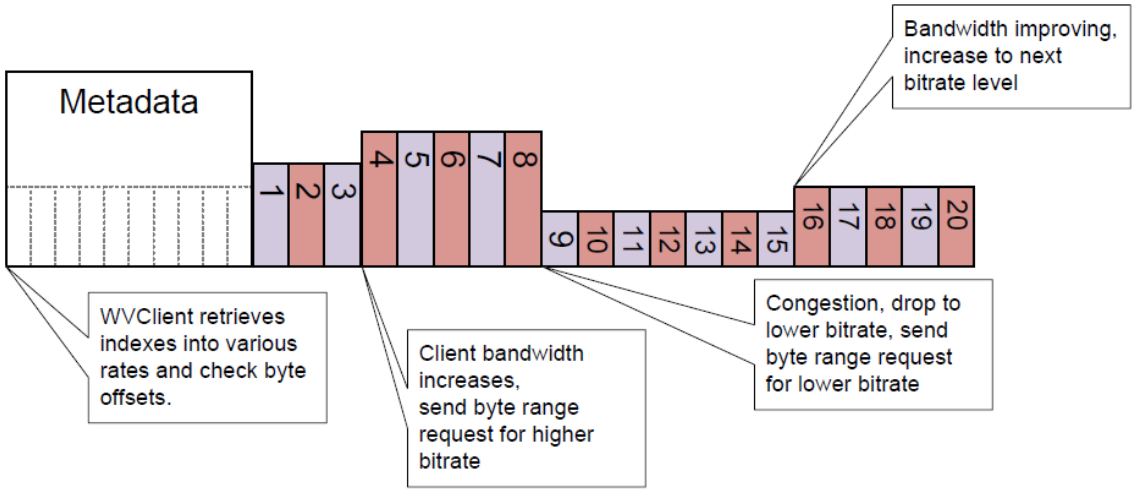

FIGURE 9 – INITIAL PLAYBACK

Upon initial playback, the client retrieves the information via a byte 0 request for the entire file.

The metadata is parsed, to perform the license request as well as determine available bitrates, chaptering information and trickplay capabilities (if available).

The Widevine client determines the highest possible bitrate for the current available network bandwidth and immediately performs a byte-range request to seek to the location of the bitrate and download the first set of data within the Widevine Adaptive Container.

Progressively, the Widevine client on the device will continue to execute byte-range requests to retrieve further sets of data. If the bandwidth fluctuates, the byte-range request will correspond to the matching bitrate at that moment.

In the case of trickplay, the Widevine client will skip to the trickplay section of the container, using a byte-range request. Immediately start playing the retrieved I-frames and skips to the appropriate bitrate once normal play resumes, via another byte-range request.

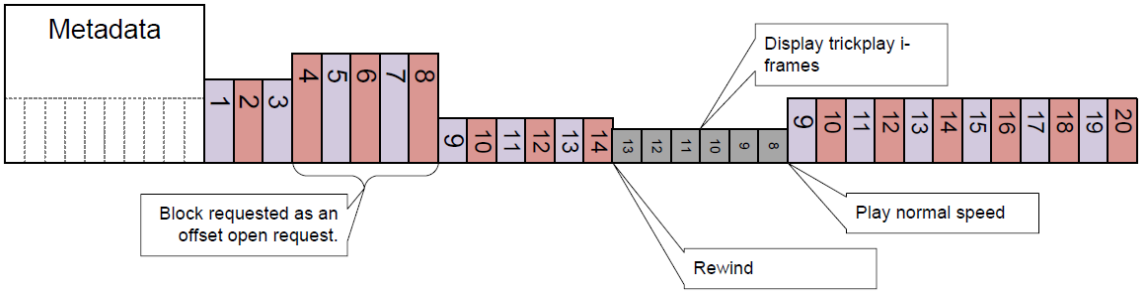

FIGURE 10 – TRICKPLAY

6. LIVE OVERVIEW

Widevine's Live Adaptive Streaming solution utilizes the HTTP Live Streaming (HLS) draft standard (http://tools.ietf.org/html/draft-pantos-http-live-streaming-02) .

The diagram below illustrates the relevant flows on content encryption to content decryption.

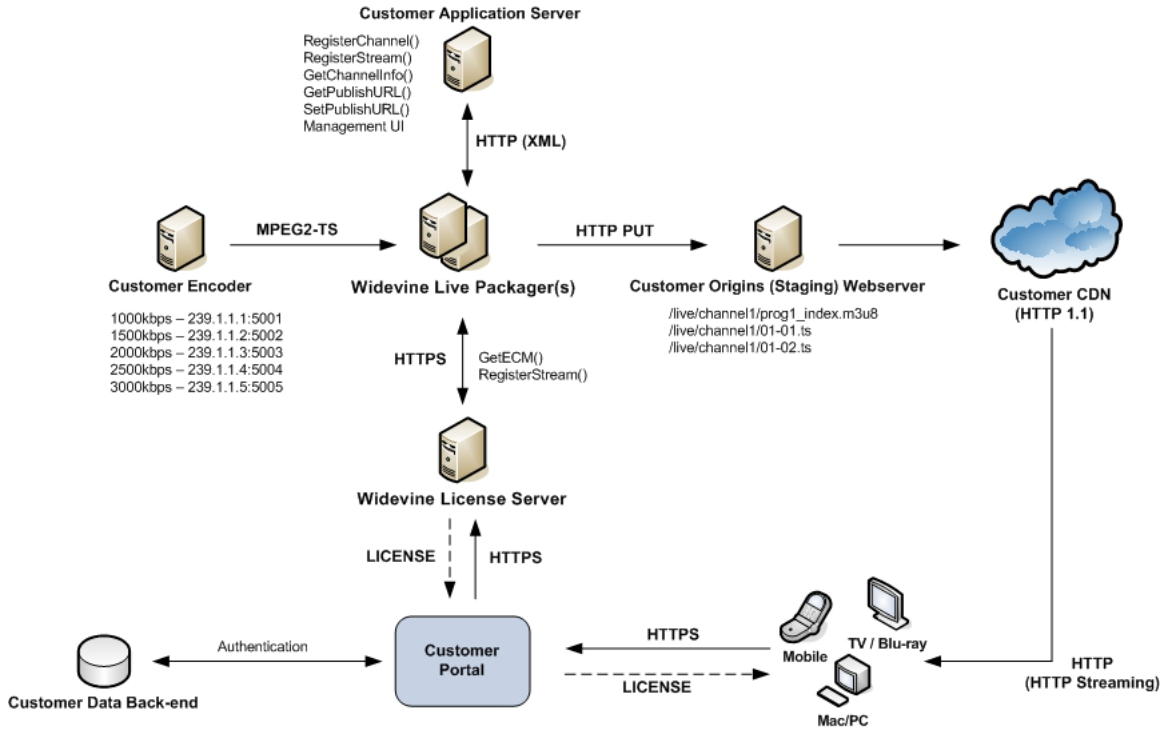

FIGURE 11 – LIVE STREAMING OVERVIEW

Depending on the purpose, the deployment of the Live Packager range from a consolidated single instance to a separate cluster of servers (Management User Interface and multiple Live Packagers).

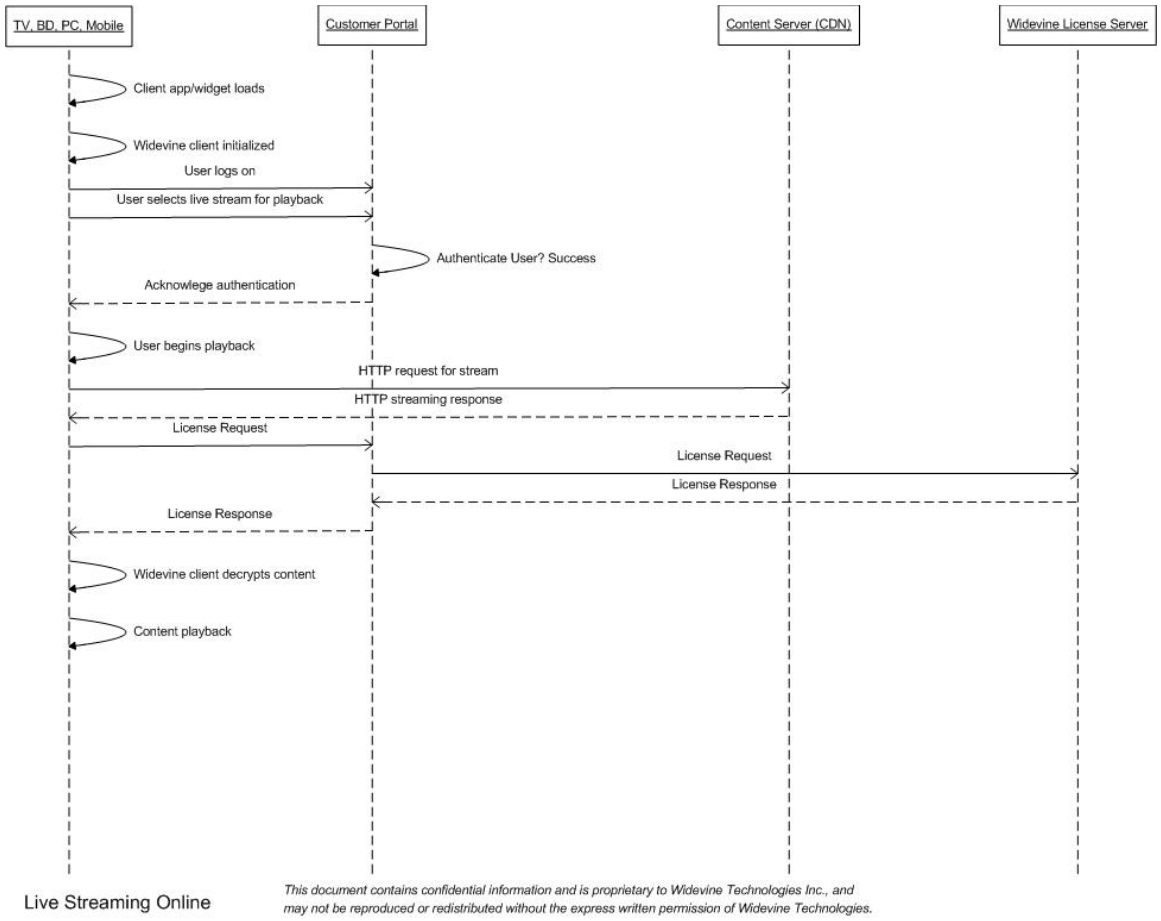

FIGURE 12 – LIVE STREAMING FLOW

The diagram above describes the sequence of live streaming content playback. The customer portal acts as the subscriber authorization service, hosting a Widevine license proxy component.

6.1. CONTENT ENCRYPTION

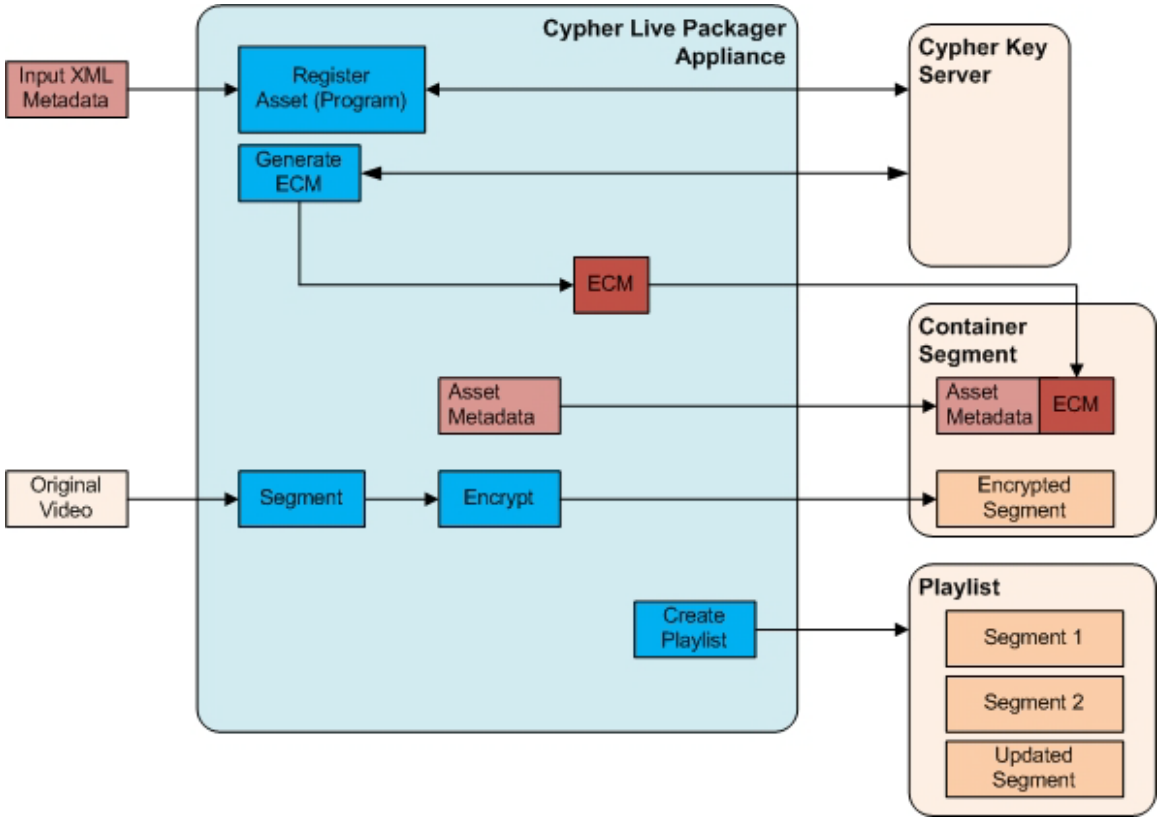

FIGURE 13 – LIVE PACKAGER ENCRYPTION FLOW

Content generated according to Widevine specifications (see relevant Encoding Specification document) is processed by a Live Packager.

One or more bitrates are segmented into transport stream files and encrypted. A playlist is generated to maintain the ordered list of file playback.

The encrypted segment files and playlists are sent to a web server URL via the use of HTTP PUT and HTTP DELETE. This operation normally requires the use of a DAV-enabled web server.

The amount of storage required is dependant on the size of the segment and the number of segments kept in the playlist.

The publishing sequence is a HTTP PUT of a new transport stream file, followed by the updated playlist file.

The Live Packager will automatically delete old files from the publishing URL location (on the staging server). In the case of an unexpected software failure, the existing files in the publish URL will remain untouched. Orphan files can be cleaned up by checking timestamps.

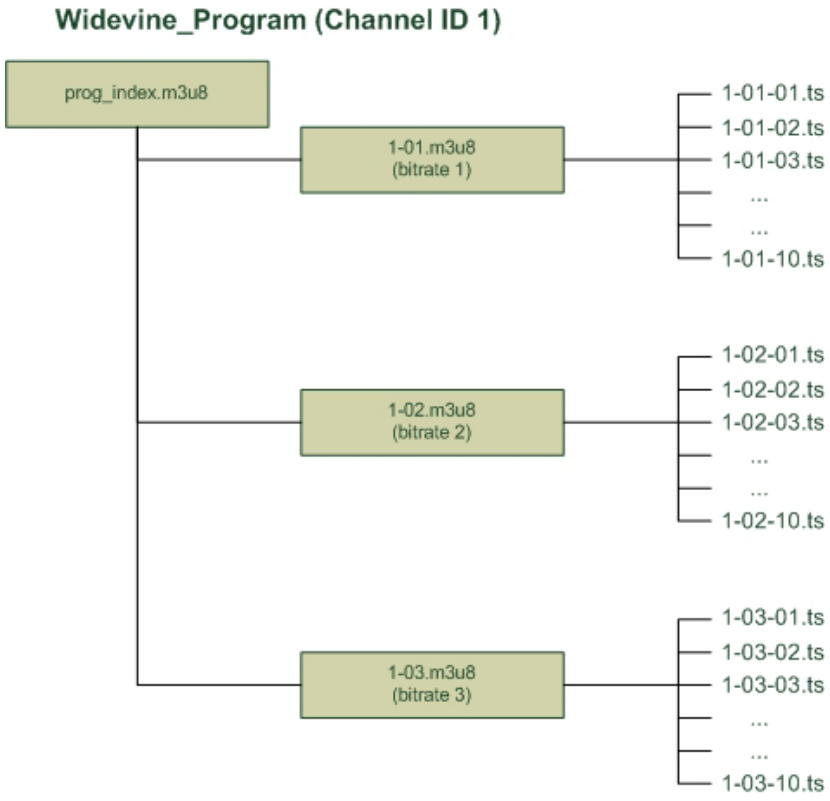

FIGURE 14 – OUTPUT FILE GENERATION

6.2. CONTENT STAGING

The generated playlists and encrypted transport stream segments are published to an origin server. This serves as a replication point to the CDN.

6.3. CONTENT DECRYPTION

See section below on license acquisition.

The Widevine client will read the master index playlist just once when playback is initiated. Next, it reads the playlist for the lowest bitrate, followed by the first transport stream file in the list.

Once each download is complete, the Widevine client will make a determination on which bitrate to use next. If it has the playlist for that bitrate, and the next transport stream segment is listed, the client will download the segment file. Otherwise, the Widevine client will reload the new bitrate playlist and checks for additional entries from where it left off. If there are no new entries, it waits 1 second and downloads the playlist again.

A sample sequence:

prog_index.m3u8
1-01.m3u8
1-01-101.ts
1-03.m3u8
1-03-102.ts
1-03-103.ts
1-03.m3u8
1-03-104.ts
1-03.m3u8
1-03.m3u8
1-03.m3u8
1-03-105.ts
1-03-106.ts
…

6.4. MANAGEMENT

The program/channel management of the Live Packagers can be one of the following:

a. A user interface per Live Packager
b. A separate Management User Interface server that manages multiple Live Packager servers.

6.5. LIVE PACKAGER HIGH AVAILABILITY

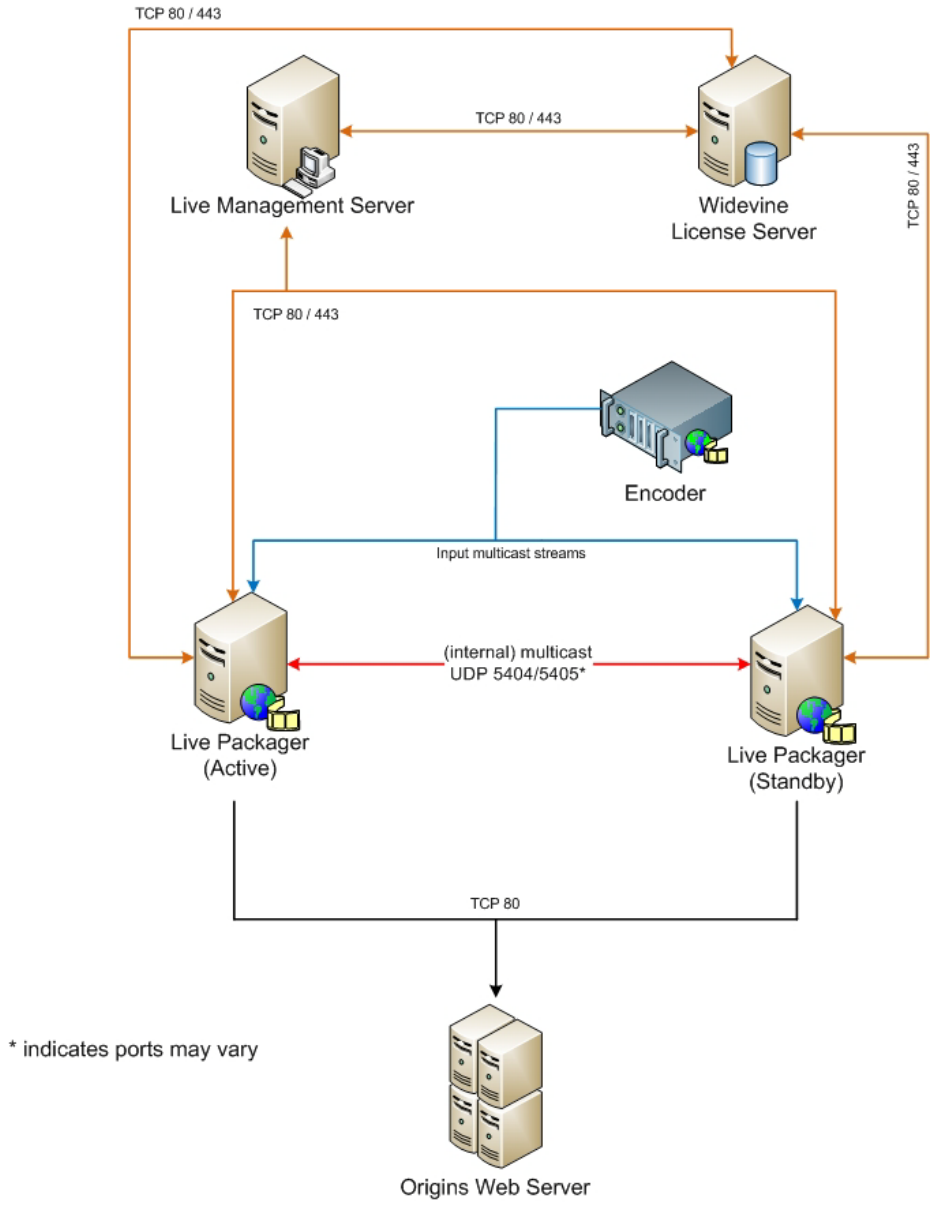

FIGURE 15 – HIGH AVAILABILITY BASIC LAYOUT

The high availability option requires the use of a dedicated Live Management Server along with a pair of Live Packagers.

The packagers are connected to the same ingress network (containing the multiple bitrate streams from the encoders). In addition, the packagers are connected to the same egress network (with connectivity to the origins server).

The solution requires the use of a non-conflicting multicast ports per HA pair. This internal monitoring messaging will occur over multicast 226.94.1.1 on these 2 ports on the egress network port of the packagers.

All Live Packagers and Live Management servers require direct HTTPS connectivity to the Widevine License Server for program (channel) management and key retrieval. In addition, the Management Server receives status and issues commands to the Live Packagers via HTTP and HTTPS.

7. License Acquisition (Request and Response)

There are 2 methods to manage the license acquisition flow.

1. Direct request to the Widevine License Server
    a. All properly formatted license requests will be accepted and a valid license response will be issued.
    b. This option has some risks regarding how to manage access to request a license. IP white-listing and geo-ip filtering are alternatives provided with this method.
2. The use of a license proxy service
    a. The Widevine license proxy is a customer-operated component that receives all incoming license requests, performs the necessary business logic to determine authorized use and forwards the request to the Widevine license server if access is permitted.
    b. The communication between the license proxy and the license server is trusted implicitly. A combination of the following factors establishes the trust:
        i. IP whitelisting to permit only license proxy IP addresses to request licenses.
        ii. Widevine-issued SSL certificates employed by the license proxy to verify the authenticity of the proxy.
        iii. Signed license requests from the proxy, validated by the license server prior to issuing the license response.
    c. The license proxy can be written in any language, as long as it conforms to the ability to receive and send licenses via HTTP POST. The backend authorization components will be different per customer integration, and entirely up to the customer for implementation.

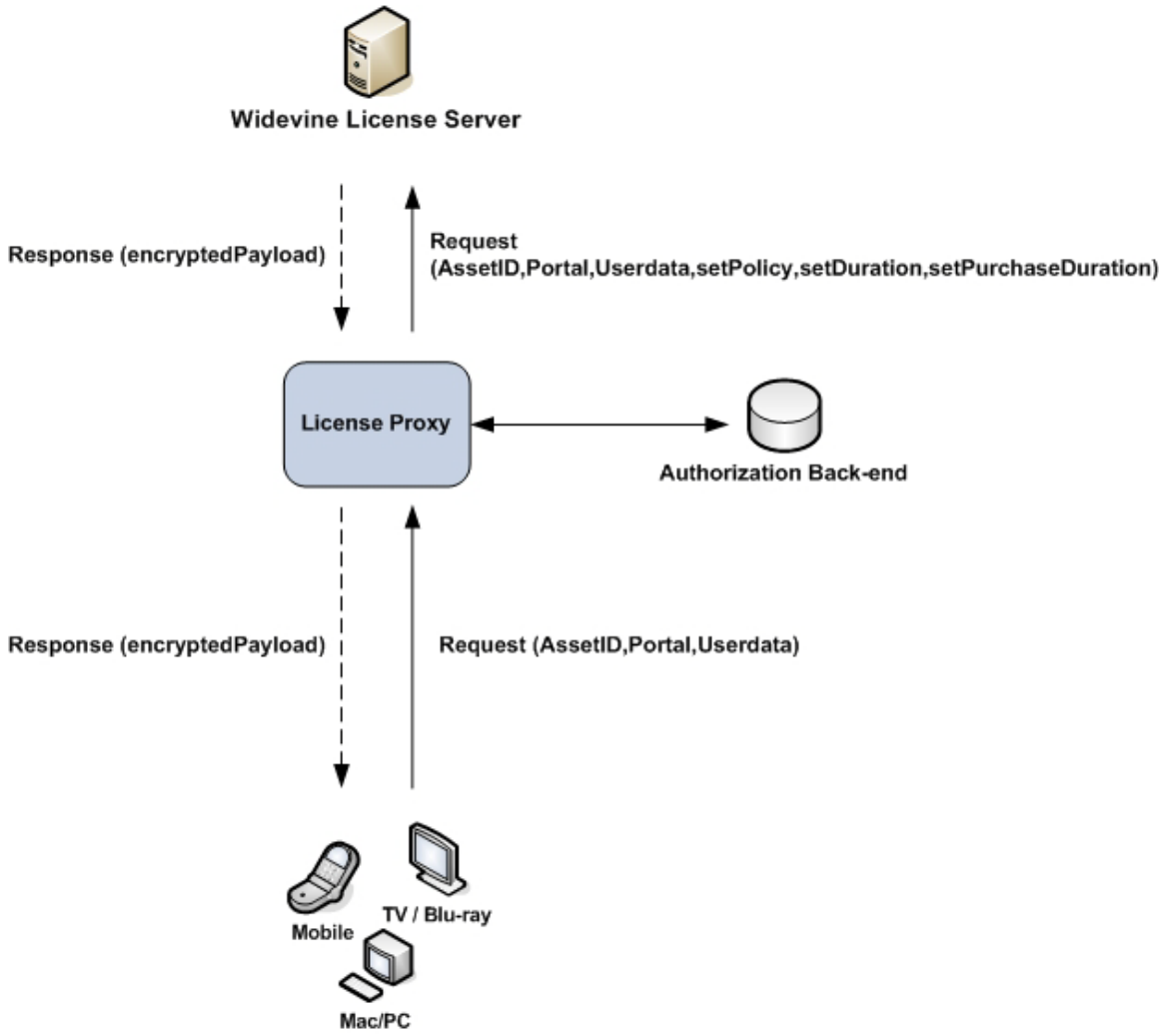

FIGURE 16 - LICENSE FLOW VIA PROXY

The license request is always generated by the Widevine client, inserting custom values when specified during application execution or initialization. A couple basic parameters include the AssetID, which is the Widevine unique id for the encrypted content and the Portal, which is the Widevine-assigned unique identifier for the customer.

The function of the license proxy is to obtain a license by forwarding the request to the license server. Since the communication between proxy and license server is trusted, a license response will return to the proxy and then delivered to the originating client. A license response cannot be modified once it has been generated by the license server.

At the proxy level, there are parameters that can be modified or added to the license request, prior to having it processed by the license server. These settings allow for maximum flexibility

in implementing business rules. Please refer to the proxy integration document for further details.