(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 September 2012 (27.09.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/129549 A1

(51) International Patent Classification:
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/US2012/030469

(22) International Filing Date:
23 March 2012 (23.03.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/466,630 23 March 2011 (23.03.2011) US
13/429,266 23 March 2012 (23.03.2012) US

(71) Applicant *(for all designated States except US)*: **GENERAL INSTRUMENT CORPORATION** [US/US]; 101 Tournament Drive, Horsham, Pennsylvania 19044 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **MORONEY, Paul** [US/US]; 7357 Fay Avenue, La Jolla, California 92037 (US). **SHAMSAASEF, Rafie** [US/US]; 10253 Prairie Fawn, San Diego, California 92127 (US).

(74) Agents: **PAYNE, Susan** et al.; 600 North US Highway 45, Libertyville, Illinois 60048 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

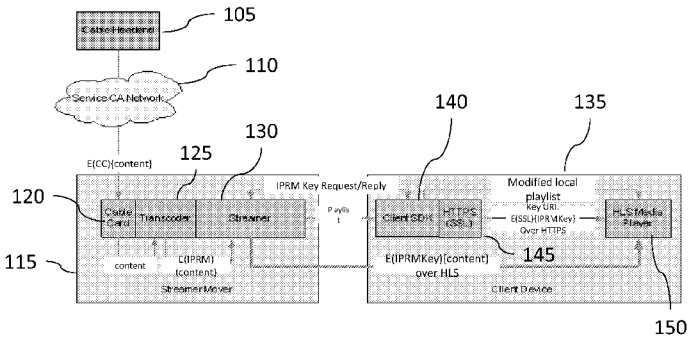
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(54) **Title**: SYSTEM AND METHODS FOR PROVIDING LIVE STREAMING CONTENT USING DIGITAL RIGHTS MANAGEMENT-BASED KEY MANAGEMENT



100

FIG. 1

(57) **Abstract**: In the present disclosure, a DRM (in this case IPRM) system may be used to deliver media content keys to a player device in a live streaming environment and take advantage of all DRM related functionalities that come with it, such as proximity control, copy protection enforcement and rights verification. A playlist may be used to deliver a key identifier for encrypted live streaming content.

WO 2012/129549 A1

SYSTEM AND METHODS FOR PROVIDING LIVE STREAMING CONTENT USING DIGITAL RIGHTS MANAGEMENT-BASED KEY MANAGEMENT

STATEMENT OF RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 61/466,630, filed March 23, 2011, which is incorporated herein in its entirety.

BACKGROUND

[0002] HyperText Transfer Protocol (HTTP) Live Streaming (also known as HLS) is an HTTP-based media streaming communications protocol. It works by breaking the overall stream into a sequence of small HTTP-based file downloads, each download loading one short chunk of an overall potentially unbounded transport stream. As the stream is played, the client may select from a number of different alternate streams containing the same material encoded at a variety of data rates, allowing the streaming session to adapt to the available data rate. At the start of the streaming session, it downloads an extended M3U playlist containing the metadata for the various sub-streams which are available.

[0003] In a typical live streaming scheme, the content key that is used to decrypt the media content on the player device is delivered over secure HTTP (HTTPS), which is a combination of HTTP with Secure Sockets Layer (SSL), a cryptographic protocol that provides communication security over the internet.

[0004] Most Digital Rights Management (DRM) systems provide services such as proximity checking or copy protection enforcement that go beyond what is provided by SSL. HTTPS is deemed not to be effective enough to implement live streaming in a DRM-based system. Therefore, the above problem calls for a DRM-based system to securely deliver media content keys to player devices.

SUMMARY

[0005] Disclosed is a method for providing live streaming content using DRM based key management via a server device. In one embodiment, live streaming content is encrypted using a key established by DRM-based key management. The encrypted content is segmented. A uniform resource identifier (URI) of a DRM-based live streaming content key is created. The URI indicates usage of a DRM key management protocol. The URI is sent to a client device in a playlist. The encrypted content is sent to the client device.

[0006] The playlist is sent to an application of the client device. When Advanced Encryption Standard 128 bit Cypher Block Chaining (AES-128 CBC) encryption is used, the encrypted content is sent to a media player of the client device. When Advanced Encryption Standard 128 bit Electronic Code Book (AES-128 ECB) MPEG-2 Transport Stream (MP2TS) encryption is used, the encrypted content is sent to the application of the client device.

[0007] In one embodiment, a channel change request from a client device is received and processed. A new DRM-based live streaming content key is derived. The

live streaming content is transcoded and encrypted using the DRM-based live streaming content key.

[0008] A channel change request from the client is received and processed. Content copy control information (CCI) is monitored to listen for changes. New CCI data is obtained. A new key is generated based in part on the change in CCI. Live streaming content is transcoded and encrypted using the new DRM-based live streaming content key.

[0009] In one embodiment, deriving the DRM-based live streaming content key further involves creating a rights data file and storing a subkey used to derive the DRM-based live streaming content key. In one embodiment, the URI provides a reference to the DRM-based live streaming content key.

[0010] In one embodiment, a change in a detected parameter is determined while currently tuned to a channel and using the DRM-based live streaming content key. A new DRM-based live streaming content key is derived is response to the detected parameter. Live streaming content is transcoded and encrypted using the new DRM-based live streaming content key. The encrypted content is segmented. A playlist is created. The playlist may have at least a URI of the new DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted segments. The URI indicates usage of a DRM key management protocol. In one embodiment, the detected parameter comprises changes in copy control information. In one embodiment, the detected parameter comprises an amount of time the previous key has been used. In one embodiment, the detected parameter comprises a program boundary.

[0011] Disclosed is a method for providing live streaming using DRM based key management via a client device. In one embodiment, a request for a playlist is sent from a media player of the client device. A playlist from a server is received at an application of the client device. The playlist has at least a Uniform Resource Identifier (URI) of a DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI indicates usage of a DRM key management protocol. The DRM key management protocol is used to establish the DRM-based live streaming content key.

[0012] In one embodiment, the DRM-based live streaming content key is sent from the application of the client device to a media player of the client device using Secure Socket Layer (SSL) encryption over Secure HyperText Transfer Protocol (HTTPS). The plurality of encrypted segments is decrypted, decoded, and presented by the media player.

[0013] In one embodiment, the application decrypts the plurality of encrypted segments using the DRM-based live streaming content key to produce un-encrypted content segments. The un-encrypted content segments are sent from the application to a media player of the client device.

[0014] In one embodiment, a tune channel command is sent from a client application of the client device to a server. The tune channel command may include a channel identifier. A playlist URI is received from the server. After establishing the DRM-based live streaming content key, a temporary Rights Data File that includes a subkey used to derive the content decryption key is stored. The playlist is parsed to determine an applied encryption type.

[0015] In one embodiment, the plurality of encrypted content segments is decrypted. A new playlist is generated for the media player with an encryption method set to none and having no key URI. The new playlist is presented to the media player.

[0016] In one embodiment, a format of the DRM-based live streaming content key is changed to match a media player of the client device. A new playlist is generated for the media player. The new playlist may include the applied encryption type with a key uniform resource locator (URI) pointed to a local Secure HyperText Transfer Protocol (HTTPS) server embedded within the client application. The new playlist is presented to the media player.

[0017] In one embodiment, a playlist is received from a server in response to the server changing the content encryption key. The playlist may have at least a URI of a new DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI indicates usage of a DRM key management protocol. A key exchange protocol is performed with the server under the indicated DRM key management system to establish the new key. A temporary Rights Data File that includes a subkey used to derive the new decryption key is stored. The playlist is parsed to determine an applied encryption type.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] So that the manner in which the above recited features of the present invention are attained and can be understood in detail, a more particular description of the invention, briefly summarized above, may be had by reference to the embodiments thereof which are illustrated in the appended drawings.

[0019] It is to be noted, however, that the appended drawings illustrate only typical embodiments of this invention and are therefore not to be considered limiting of its scope, for the invention may admit to other equally effective embodiments.

[0020] FIG. 1 illustrates one embodiment of a system 100, according to one embodiment;

[0021] FIG. 2 illustrates one embodiment of a system 200, according to one embodiment;

[0022] FIG. 3 illustrates one embodiment of a method 300 for providing DRM-based live streaming using a server, according to one embodiment;

[0023] FIG. 4 illustrates one embodiment of a method 400 for providing DRM-based live streaming using a server, according to one embodiment;

[0024] FIG. 5 illustrates one embodiment of a method 500 for providing DRM-based live streaming using a server, according to one embodiment;

[0025] FIG. 6 illustrates one embodiment of a method 600 for providing DRM-based live streaming using a client device, according to one embodiment;

[0026] FIG. 7 illustrates one embodiment of a method 700 for providing DRM-based live streaming using a client device, according to one embodiment;

[0027] FIG. 8 illustrates one embodiment of a method 800 for providing DRM-based live streaming using a client device, according to one embodiment;

[0028] FIG. 9 illustrates a server device 900, according to one embodiment; and

[0029] FIG. 10 illustrates an end-user device 1000, according to one embodiment.

DETAILED DESCRIPTION

[0030] The following disclosure describes DRM, e.g., Internet Protocol Rights Management (IPRM), media protection and key change mechanism as it applies to HLS. This disclosure explores how different protection modes are supported and how key change occurs as content travels from a streamer to client devices.

[0031] In the present disclosure, the main idea is to use a DRM, e.g., IPRM, system to deliver media content keys to the player device in the live streaming environment and take advantage of all DRM related functionalities that come with it, such as proximity control, copy protection enforcement and rights verification. With IPRM a content key seed is securely delivered to the player with additional data to locally derive a content decryption key while preserving and binding copy control information (CCI) and other rights data with the content decryption key.

[0032] IPRM key change is signaled by HLS key File URI changes in the Playlist. Every time IPRM detects a rule change that results in deriving a new encryption key, a new key file URI is created in the playlist so that the HLS client device gets the correct key. There are at least two key change events: One at any channel change event, and another one when CCI/rights data of incoming streams change during viewing of a channel. The key URI in this case is constructed as follows:

URI = ?iprm:////KeyID.txt?

where KeyID represents an Odd/Even key tag or sequential numeric key tag. Key tags such as KeyID are used so that the client device knows which key to request for which content chunk, e.g., segment. There may also be a home server name, e.g., home gateway device, or over-the-top server name prefixed to the uniform resource locator (URL). A key URI may in this instance be constructed as follows:

URI = iprm://<Streamer Domain Name or IP Address>/channel-keyID.txt

where key segments are signaled by different keyIDs.

[0033] The encryption method that is used by live streaming can also be configured and signaled to the player device and is not limited to the Advanced Encryption Standard 128 bit Cypher Block Chaining (AES-128 CBC) described in the Internet Engineering Task Force (IETF) draft HLS specification. If the media content is for example already encrypted using AES-128 Electronic Code Book (ECB) and encapsulated inside a Motion Picture Experts Group Phase 2 (MPEG-2) transport stream, there is no need for re-encryption to AES-128 CBC of the entire content stream. Rather, the live streaming playlist file, e.g., a manifest file, can signal an Advanced Encryption Standard 128 bit Electronic Code Book MPEG-2 Transport Stream (AES-128 ECB-MP2TS) method via its key tag. The IPRM system at the client device side determines an encryption type and using DRM-based methods, acquires keys using information received from the server in a uniform resource identifier (URI). Using AES-128 ECB-MP2TS encryption in this manner allows for the support of existing DVR boxes and home media server devices, where MPEG-2 transport is commonly used.

[0034] Use of the key URI to indicate Odd/Even key usage or sequential numeric key usage is signaled by the key tag in the URI. As keys and rights do not change very quickly, use of only two key URIs at any given time, e.g., in any given playlist file, is sufficient to indicate Odd and Even key usage. In one embodiment, the key URI, or a small portion of it, with key tag inside the IPRM protocol Digital Object Identifier (DOI) object is sent as part of key request message from a client device to a server device.

[0035] FIG. 1 illustrates a system 100 for providing DRM-based live streaming. In this embodiment, AES-128 CBC encryption is used to encrypt the live stream. Streamer 115 supports tuning to live channels, transcoding in real-time and streaming the transcoded media content to a client device 135 using HLS. Streamer 115 may also be referred to as a tuner streamer or simply, a server. Incoming live media to streamer 115 is either protected by a CableCard or MediaCipher protection system from a CA provider network. The content flows thru CableCard interface 120 where the content is decrypted. The content is then transcoded in real-time by transcoder 125. The content is then re-encrypted using a DRM-based live streaming content key by transcoder 125, for protection between streamer 130 and client device 135. The streamer 130 then establishes an HLS session with client devices 135 to stream the encrypted media. The encrypted content is segmented, e.g. by streamer 130. Different keys and different CCI rights may be associated with each segment.

[0036] A playlist is created by streamer 130 of server 115. The playlist has at least a URI of a DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted content segments. The URI also indicates usage of a DRM key management protocol. As described earlier, a URI form can be shown as:

URI = ?iprm:////KeyID.txt?

The iprm: portion indicates the protocol type, i.e. the usage of a DRM key management protocol.

[0037] The playlist is sent to client device 135. In one embodiment, the playlist is sent to an application 140 of the client device. Application 140 may be a client Software Development Kit (SDK).

[0038] The encrypted content is sent to client device 135. The encrypted content may be sent to different elements of the client device depending on the type of encryption used to deliver the content key. When AES-128 CBC encryption is used, the encrypted content is sent to a media player 150 of the client device.

[0039] A request for a playlist may be sent from media player 150 of client device 135. In response to the request, a playlist from the server, e.g. server 115 is received at an application of the client device. The playlist has at least a URI of the DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI also indicates usage of a DRM key management protocol.

[0040] Client application 140 generates a new or modified playlist. This modified playlist is used to provide at least encryption and key URI information to media player 150 of client device 135. When AES-128 CBC encryption is used, client application 140 generates a playlist for the locally native HLS player, e.g. media player 150 that lists AES-128 CBC as the encryption method and the key URI pointed to a local HTTPS server 145 embedded within the client application, e.g. client SDK 140. The DRM-

based live streaming content key URI is sent from application 140 to media player 150 using SSL encryption over HTTPS. In this embodiment, the plurality of encrypted segments is received from streamer 130 of server 115 by media player 150 of client device 135.

[0041] FIG. 2 illustrates a system 200 for providing DRM-based live streaming. In this embodiment, AES-128 ECB MP2TS encryption is used to encrypt the live stream. Streamer 215 supports tuning to live channels, transcoding in real-time and streaming the transcoded media to client device 135 using HLS. Streamer 215 may also be referred to as a tuner streamer or simply, a server. Incoming live media to streamer 215 is either protected by a CableCard or Media Cipher protection system from a CA provider network. The content flows thru CableCard interface 220 where the content is decrypted. The content is then transcoded in real-time by transcoder 225. The content is then re-encrypted using a DRM-based live streaming content key by transcoder 225, for protection between streamer 230 and client device 235. The streamer 230 then establishes an HLS session with client devices 235 to stream the encrypted media. The encrypted content is segmented, e.g. by streamer 230. Different keys and different CCI rights may be associated with each segment.

[0042] A playlist is created by streamer 230 of server 215. The playlist has at least a URI of a DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted content segments. The URI also indicates usage of a DRM key management protocol. The playlist is sent to client device 235. In one embodiment, the playlist is sent to an application 240 of the client device. Application 240 may be a client SDK.

[0043] The encrypted content is sent to client device 235. The encrypted content may be sent to different elements of the client device depending on the type of encryption used to deliver the content key. When AES-128 ECB MP2TS encryption is used, the encrypted content is sent to application 240 of the client device.

[0044] A request for a playlist may be sent from media player 250 of client device 235. A playlist from the server, e.g. server 215, is received at application 240. The playlist has at least a URI of the DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI also indicates usage of a DRM key management protocol.

[0045] Client application 240 generates a new or modified playlist. This modified playlist is used to provide at least encryption and key URI information to media player 250 of client device 235.

[0046] When AES-128 ECB MP2TS encryption is used, client application 240 generates a playlist for the locally native HLS player, e.g. media player 250. The playlist lists NONE as the encryption method and has no key URI. In this embodiment, the plurality of encrypted segments is received from streamer 230 of server 215 by client application 240. Application 240 decodes the plurality of encrypted segments using the DRM-based live streaming content key to produce un-encoded content segments. The un-encoded content segments are sent from application 240 to media player 250 of client device 235.

[0047] FIG. 3 illustrates one embodiment of a method 300 for providing live streaming content using DRM-based key management via a server device, e.g. 115, 215. At block 305, live streaming content is encrypted using a key established by DRM-

based key management. The encryption may be implemented using an IPRM DRM system that derives a DRM-based live streaming content key. The live streaming content may be encrypted by transcoder 125, 225. At block 310, the encrypted content is segmented, e.g. by streamer 130,230.

[0048] At block 315, a URI of a DRM-based live streaming content key is created. The URI indicates usage of a DRM key management protocol. At block 320, the URI is sent to client device 135, 235 in a playlist. The playlist has at least a uniform resource identifier (URI) of a DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted content segments. In one embodiment, the playlist is sent to an application 140, 240 of client device 135, 235. Application 140, 240 may be a client SDK. The URI does not point to the actual key. The URI is an identifier the DRM system uses to find out what key is required.

[0049] At block 325, the encrypted content is sent to the client 135, 235. The encrypted content may be sent to different elements of client device 135, 235 depending on the type of encryption used to deliver the content key. When AES-128 CBC encryption is used, the encrypted content is sent to a media player 150 of the client device. When AES-128 ECB MP2TS encryption is used, the encrypted content is sent to application 240 of the client device.

[0050] FIG. 4 illustrates a method 400 for providing live streaming content using DRM-based key management via a server device. In this embodiment, the server 115, 215 implements an IPRM key change for a channel change event. Method 400 may begin at block 405 or proceed from block 325 after an initial URI has been sent to the client device 135, 235 in a playlist. At block 405, a channel change request from a

client device 135, 235 is received and processed. In one embodiment, the channel change request may be a representational state transfer (RESTful) channel change request or a Digital Living Network Alliance Digital Media Server (DLNA DMS) request for content browsing. Server 115, 215 monitors CCI data to listen for changes, e.g. Cable Card or Entitlement Control Message (ECM) CCI change events. New CCI data is obtained and live streaming content is transcoded and encrypted using the new DRM-based live streaming content key. In this manner, DRM rights to the content can be verified before access is given to the content key, e.g. providing a URI for the DRM-based live streaming content key. This rights verification cannot be performed in current systems using SSL to deliver content keys.

[0051] At block 410 a new DRM-based, e.g. IPRM, live streaming content key is derived. A rights data file is created and a subkey used to derive the new DRM-based live streaming content key is stored. When using IPRM or any DRM system, the content key is derived on the client device side on the fly rather than signaling the actual key through a URI as is done in present HLS systems. Therefore, the live streaming content key is never exposed during server-client communication. At block 415, the live streaming content is transcoded and encrypted using the new DRM-based live streaming content key.

[0052] The encrypted content is segmented, e.g. by streamer 130, 230. A URI of a DRM-based live streaming content key is created. The URI indicates usage of a DRM key management protocol. The key URI provides a reference to the DRM-based live streaming content key. In one embodiment, the URI provides a reference to this key. A playlist is also created. The playlist has at least the URI of the DRM-based live

streaming content key, an encryption method, and a list of a plurality of the encrypted content segments. The playlist and the encrypted content are sent to a client device, e.g. client device 135, 235. The Playlist file is created and the key URI is inserted as follows:

- Add #EXT-X-KEY tag with following parameters:
  METHOD = AES-128-ECB-MP2TS or AES-128

  URI = "iprm://<Streamer Domain Name or IP Address>/<Channel Name or ID>/channelkey.txt"

- This key URI is a symbolic URI used between a streamer and a client SDK to trigger IPRM key exchange message
- For example:
  URI="iprm://Streamer123.mot.com/abc/channelkey.txt"

In the above #EXT-X-KEY tag example, Streamer Domain Name or IP Address could be replaced by a Home Media Gateway Device Name or IP Address instead. Likewise, Channel Name or ID could also be replaced by a digital video recorder (DVR) recoded file name or ID.

[0053] FIG. 5 illustrates a method 500 for providing live streaming content using DRM-based key management via a server device. In this embodiment, the server 115, 215 implements a DRM, e.g., IPRM, key change upon determining a detected event. Method 500 may begin at block 505, proceed from block 325 after an initial URI has been sent to the client device 135, 235 in a playlist, or proceed from block 425 after a new URI has been sent to client device 135, 235 in a playlist in response to a channel change. At block 505, a change in a detected parameter is determined while currently tuned to a channel and using a previous DRM-based live streaming content key. At block 510, a new DRM-based live streaming content key is derived in response to the detected parameter. At block 515, live streaming content is transcoded and encrypted

using the new DRM-based live streaming content key. At block 520, the encrypted content is segmented. At block 525, a playlist is created. The playlist has at least a URI of the new DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted content segments. This key URI serves as an identifier to the DRM content key and indicates usage of a DRM key management protocol. The playlist and the encrypted content are sent to a client device, e.g. client device 135, 235.

[0054] In one embodiment, the detected parameter is changes in CCI. In one embodiment, the detected parameter is an amount of time the previous key has been used. In one embodiment, the detected parameter is a program boundary.

[0055] The key change, e.g. key rotation, may happen while tuned to a channel but each program within the same channel has different key or CCI data therefore has to be encrypted with different key. All media content of the same channel may be encrypted using the same key, or new keys may be required at intervals or special events such as CCI changes. According to HLS specification, the theoretical limit is one key per media file, but because each media key adds a file request and transfer to the overhead for presenting the following media segments, changing to a new key periodically is less likely to impact system performance than changing keys for each segment.

[0056] The key change signal in HLS is similar to a key URI change in a playlist file. The key change trigger point for streamer 115, 215 is when an incoming live stream has new CCI data detected by Cable Card or ECM. The sequence of operations shown for key change at the channel change event in the previous section is also applicable

here with the following extra steps of detecting CCI changes in the middle of channel and by creating playlist with new key URIs every time that happens:

- Streamer, e.g. streamer 115, 215, derives a new key every time CCI change is detected
- Streamer also sets the new key into the hardware to start encrypting the incoming live stream
- Streamer generates a new key URI (called program key here) in the playlist file every time CCI changes. For instance:

Key1: URI="iprm://Streamer123.mot.com/abc/programkey1.txt"

Key2: URI="iprm://Streamer123.mot.com/abc/programkey2.txt"

Key3: URI="iprm://Streamer123.mot.com/abc/programkey3.txt"

[0057] FIG. 6 illustrates one embodiment of a method 400 for providing live streaming content using DRM based key management via a client device, e.g. client device 135, 235. At block 605, a request for a playlist is sent from a media player, e.g. media player 150, 250 of the client device. At block 610, the playlist from the server, e.g. server 115, 215, is received at an application, e.g. 140, 240 of the client device. The playlist has at least a URI of the DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI indicates usage of a DRM key management protocol. At block 615, the DRM key management protocol is used to establish the DRM-based live streaming content key. The IPRM protocol uses a security protocol to authenticate a client device with a streamer, e.g. server. The client device uses a key request message to request for a

content key from the server. Once a key reply message is returned to the client device, the client device uses a sub-key communicated from the server and other data such as CCI from the message to create a local rights data file, and derive the actual content key in use.

[0058] When AES-128 CBC encryption is used, client application 140 generates a playlist for the locally native HLS player, e.g. media player 150 that lists AES-128 CBC as the encryption method and the key URI pointed to a local HTTPS server 145 embedded within the client application, e.g. client SDK 140. The DRM-based live streaming content key is sent from application 140 to media player 150 using SSL encryption over HTTPS. In this embodiment, the plurality of encrypted segments is received from streamer 130 of server 115 by media player 150 of client device 135. The plurality of encrypted segments is decrypted, decoded, and presented by the media player.

[0059] When AES-128 ECB MP2TS encryption is used, client application 240 generates a playlist for the locally native HLS player, e.g. media player 250. The playlist lists NONE as the encryption method and has no key URI. In this embodiment, the plurality of encrypted segments is received from streamer 230 of server 215 by client application 240. Application 240 decrypts the plurality of encrypted segments using the DRM-based live streaming content key to produce un-encrypted content segments. The un-encrypted content segments are sent to media player 250 by application 240.

[0060] FIG. 7 illustrates a method 700 for providing live streaming content using DRM-based key management via a client device. In this embodiment, the client device

135, 235 implements a DRM-based, e.g. IPRM, key change for a channel change event. An application (not shown) on the client device initiates a channel change and calls a client SDK 140, 240 tune channel API. Method 700 may start at block 705 or proceed from block 615, after an initial key has been derived. At block 705, a tune channel command is sent from a client application 140, 240 of the client device 135, 235 to a server, e.g. streamer 115, 215. The tune channel command may be a representational state transfer (RESTful) command or a Digital Living Network Alliance Digital Media Server (DLNA DMS) command for content browsing. The RESTful command may include at least a channel identifier. At block 710, a playlist URI is received from the server, e.g. from streamer 130, 230.

[0061] The IPRM application, e.g. client SDK 140, 240, on the client device performs key exchange using the channel name or channel ID with the IPRM application on the streamer 130, 230. At block 715, a temporary Rights Data File that includes a subkey used to derive the content decryption key is stored by the IPRM application after establishing the DRM-based live streaming content key.

[0062] At block 720, the playlist is parsed to determine an applied encryption type. The behavior of the client SDK depends on the content encryption method/type indicated in the HLS playlist.

[0063] AES-128 EBC MP2TS encryption is specific to IPRM and is not currently supported by any standard HLS server or player. Therefore, the decryption has to happen outside the HLS player and within the client SDK. The clear media content, e.g. un-encrypted media content, is sent from the client SDK to the media player. The HLS player, in this case treats the incoming stream as clear without a key URI in the playlist.

When AES-128 EBC MP2TS is the applied encryption type, the client SDK 240 decrypts the plurality of encrypted content segments. IPRM decrypts the content and provides a clear buffer to client SDK 240. Client SDK 240 sends the buffer to media player 250, e.g. a native HLS player, of client device 235. A new playlist is generated for the media player with an encryption method set to none and having no key URI. The new playlist is presented by the client SDK 240 to the HLS player 250.

[0064] AES-128 CBC mode decryption is supported by a native HLS player, e.g. media player 150 as long as the key URI is securely presented to the native HLS player. When AES-128 CBC encryption is the applied encryption type, a format of the DRM-based live streaming content key is changed to match the media player of the client device. The HLS key provided to media player 150 by HTTPS server 145 is an AES-128 key that is 16-octet keys. The format of the Key file is simply a packed array of these 16 octets in binary format. The key value must be interpreted as a 128-bit hexadecimal number and must be pre-fixed with 0x or 0X.

[0065] A new or modified playlist is generated for the media player 150. The new playlist includes the applied encryption type with a key URI pointed to a local HTTPS server embedded within the client application 140. The client SDK presents the new playlist to the media player.

[0066] The #EXT-X-KEY tag in the new playlist must be as follows:

METHOD = AES-128

URI = "https://localhost/<Channel Name or ID>/channelkey.txt"

For example:

URI="https://localhost/abc/channelkey.txt"

[0067] HLS player 150 gets the new or modified playlist file from client SDK 140. When AES-128-ECB-MP2TS encryption is used, the HLS player 250 gets clear (un-encrypted) media and renders the media. When AES-128 CBC is used, player 150 gets encrypted media, decrypts the media using the key URI and renders the media.

[0068] FIG. 8 illustrates a method 800 for providing live streaming content using DRM-based key management via a server device. In this embodiment, the server 115, 215 implements an IPRM key change in response to a detected parameter. The detected parameter may be a program boundary, an amount of time a previous key has been used, or a CCI change event. Method 800 may begin at block 805, proceed from block 615, after an initial key has been derived, or block 730, after a key has been derived in connection with a channel change. At block 805, a playlist is received from the server 115, 215 in response to the server changing the content encryption key. The playlist has at least a URI of a new DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments. The URI indicates usage of a DRM key management protocol.

[0069] At block 810, a key exchange protocol is performed with the server under the indicated DRM key management system to establish the new key. In one embodiment, the IPRM application, e.g. client SDK 140, 240, on the client device performs key exchange using the URI information to identify the key of interest with the IPRM application on the streamer 130, 230. At block 815, a temporary Rights Data File that includes a subkey used to derive the new decryption key is stored by the IPRM application.

[0070] At block 820, the playlist is parsed to determine an applied encryption type. The behavior of the client SDK depends on the content encryption method/type indicated in the HLS playlist.

[0071] AES-128 EBC MP2TS encryption is specific to IPRM and is not currently supported by any standard HLS server or player. Therefore, the decryption has to happen outside the HLS player, e.g. media player 250, and within the client SDK, 240. The clear media content, e.g. un-encrypted media content, is sent from client SDK 140 to media player 250. The HLS player, in this case treats the incoming stream as clear without a key URI in the playlist. When AES-128 EBC MP2TS is the applied encryption type, the client SDK 240 decrypts the plurality of encrypted content segments. IPRM is used to decrypt the content and provide a clear buffer to client SDK 240. Client SDK 240 sends the buffer to media player 250, e.g. a native HLS player, of client device 235. A new or modified playlist is generated for the media player with an encryption method set to none and having no key URI. The new playlist is presented by the client SDK 240 to the HLS player 250.

[0072] AES-128 CBC mode decryption is supported by a native HLS player as long as the key URI is securely presented to the native HLS player, e.g. player 150. When AES-128 CBC encryption is the applied encryption type, the DRM-based live streaming content key is established. A format of the DRM-based live streaming content key is changed to match the media player of the client device. The HLS key provided to media player 150 by HTTPS server 145 is an AES-128 key that is 16-octet keys. The format of the Key file is simply a packed array of these 16 octets in binary

format. The key value must be interpreted as a 128-bit hexadecimal number and must be pre-fixed with 0x or 0X.

[0073] A new playlist, e.g. modified playlist, is generated for the media player 150. The new playlist includes the applied encryption type with a modified key URI for program keys. The client SDK presents the new playlist to the media player.

[0074] The modified key URI may be as follows:

Key1: URI=" https://localhost/abc/programkey1.txt"

Key2: URI=" https://localhost/abc/programkey2.txt"

Key3: URI=" https://localhost/abc/programkey3.txt"

[0075] The *programKey n* field in the URI above is just an indicator of subsequent key changes due to either new CCI, a long elapsed time of usage for the prior key, or some other detected program parameter change. The *programKey n* field can also be a fixed name with a strictly sequential number to show the key changes across all usage scenarios. Thus tuning to channel 1 would show a "keyname1" then a new tuned channel would use "keyname2" then a CCI change on that channel would go to "keyname3" and so on. The field just needs to be explicit enough that client devices can establish the correct key. Another option is to use "keyname1" then "keyname2" then back to "keyname1" and so on, essentially odd and even key names forever. This option is also effective, even though all the referenced keys would be different, so long as the keys do not change very often.

[0076] HLS player 150 gets the new or modified playlist file with the modified key URI from client SDK 140. When AES-128-ECB-MP2TS encryption is used, the HLS

player 250 gets clear (un-encrypted) media and renders the media. When AES-128 CBC is used, player 150 gets encrypted media, decrypts the media using the modified key URI and renders the media.

[0077] IPRM currently supports AES-128 ECB mode for MPEG-2 Transport Streams. HLS, in addition to supporting clear content only supports AES-128 with CBC mode for encryption.

[0078] For an encryption method/mode of 'NONE', there is no encryption signal and there should be no key file associated with this method. The media file following this tag is in clear.

[0079] HLS currently only supports one encryption method. This supported encryption method is AES-128 CBC. The decryption type is signaled in the playlist with URI to key file. The native client device HLS player obtains the key and decrypts the encrypted live streaming content. The following rules apply to AES-128 CBC encryption:

- Individual media files or chunks must be encrypted in entirety and Cipher Block Chaining must not be applied across different media files;
- The initialization vector (IV) used for encryption and decryption must be either the sequence number of the media file or the value of the IV attribute of the EXT-X-KEY tag. The EXT-X-KEY tag contains information to decrypt media files that follow it.

[0080] AES-128 ECB MP2TS is a new proposed method to HLS in order to support IPRM encrypted MP2TS packets using AES-128 ECB mode. Since this is an IPRM specific mode, the native client device platform HLS player does not support this

mode and packets need to be decrypted before getting to the player. The playlist should be tagged with encryption METHOD of NONE while being presented to the player.

[0081] The AES-128 ECB MP2TS encryption method used by IPRM server follows these rules:

- The 4-byte TS packet header is in the clear and the transport_scrambling_control bit is set if payload is encrypted;
- If there is an adaptation field, it is left in the clear;
- Only payloads longer than 15 bytes are encrypted using the 128-bit AES ECB mode with a 128-bit key, block by block;
- Null packet and those packets whose payload is less than 16 bytes long are left in clear; and
- If there is a residual block, it is also left in clear.

[0082] The following highlights key exchange elements of HLS playlists as discussed in this disclosure in sample playlist files and their corresponding modifications on the streamer and client device. This sample shows m3u8 playlist file as the client device tunes to channel 701. In each of the following examples there is a media sequence (541), a target duration (2), an encryption method, a URI and a list of content segments (#EXTINF:2). The URI references a LocalServerFQDN. An FQDN refers to a Fully Qualified Domain Name

1. **Streamer to Client Device with AES-128-ECB-MP2TS**

#EXTM3U

#EXT-X-MEDIA-SEQUENCE:541

#EXT-X-TARGETDURATION:2

#EXT-X-ALLOW-CACHE:NO

#EXT-X-KEY:METHOD=AES-128-ECB-MP2TS,URI="iprm://LocalServerFQDN **/content/701/channelkey.txt"**

#EXTINF:2,

http://LocalServerFQDN /content/701_00541.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00542.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00543.ts

2. **Client SDK to Player with AES-128-ECB-MP2TS**

#EXTM3U

#EXT-X-MEDIA-SEQUENCE:541

#EXT-X-TARGETDURATION:2

#EXT-X-ALLOW-CACHE:NO

#EXT-X-KEY:METHOD=NONE

#EXTINF:2,

http://LocalServerFQDN /content/701_00541.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00542.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00543.ts

3. **Streamer to Client Device with AES-128**

#EXTM3U

#EXT-X-MEDIA-SEQUENCE:541

#EXT-X-TARGETDURATION:2

#EXT-X-ALLOW-CACHE:NO

#EXT-X-KEY:METHOD=AES-128,URI="iprm://LocalServerFQDN **/content/701/channelkey.txt"**

#EXTINF:2,

http://LocalServerFQDN /content/701_00541.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00542.ts

#EXTINF:2,

http://LocalServerFQDN /content/701_00543.ts

4. **Client SDK to Player with AES-128**

#EXTM3U

#EXT-X-MEDIA-SEQUENCE:541

#EXT-X-TARGETDURATION:2

#EXT-X-ALLOW-CACHE:NO

#EXT-X-KEY:METHOD=AES-128,URI="https://localhost/content/701/channelkey.txt"

#EXTINF:2,

http:// LocalServerFQDN /content/701_00541.ts

#EXTINF:2,

http:// LocalServerFQDN /content/701_00542.ts

#EXTINF:2,

http:// LocalServerFQDN /content/701_00543.ts

[0083] FIG. 9 and FIG 10 illustrate an example server device 900 and end-user device 1000. Server device 900 may be implemented as streamer 115, 215. Device 900 comprises a processor (CPU) 905, a memory 910, e.g., random access memory (RAM) and/or read only memory (ROM), and various input/output devices 915, (e.g., storage devices, including but not limited to, a tape drive, a floppy drive, a hard disk drive or a compact disk drive, a receiver, a transmitter, and other devices commonly required in multimedia, e.g., content delivery, encoder, decoder, system components, Universal Serial Bus (USB) mass storage, network attached storage, storage device on a network cloud).

[0084] End-user device 1000 may be implemented as client device 135, 235. Device 1000 comprises a processor (CPU) 1005, a memory 1010, e.g., random access memory (RAM) and/or read only memory (ROM), and various input/output devices 1015, (e.g., storage devices, including but not limited to, a tape drive, a floppy drive, a hard disk drive or a compact disk drive, a receiver, a transmitter, and other devices commonly required in multimedia, e.g., content delivery, encoder, decoder, system

components, Universal Serial Bus (USB) mass storage, network attached storage, storage device on a network cloud).

[0085] The processes described above, including but not limited to those presented in connection with FIGs.1-8, may be implemented in general, multi-purpose or single purpose processors. Such a processor, e.g. processor 905, 1005, will execute instructions, either at the assembly, compiled or machine-level, to perform that process. Those instructions can be written by one of ordinary skill in the art following the description of presented above and stored or transmitted on a computer readable medium, e.g., a non-transitory computer-readable medium. The instructions may also be created using source code or any other known computer-aided design tool. A computer readable medium may be any medium capable of carrying those instructions and include a CD-ROM, DVD, magnetic or other optical disc, tape, silicon memory (e.g., removable, non-removable, volatile or non-volatile), packetized or non-packetized wireline or wireless transmission signals.

[0086] While the foregoing is directed to embodiments of the present disclosure, other and further embodiments may be devised without departing from the basic scope thereof, and the scope thereof is determined by the claims that follow.

CLAIMS

What is claimed is:

1. A method for providing live streaming content using Digital Rights Management (DRM) based key management via a server device, comprising:

encrypting live streaming content using a key established by DRM-based key management;

segmenting the encrypted content;

creating a uniform resource identifier (URI) of a DRM-based live streaming content key, the URI indicating usage of a DRM key management protocol;

sending the URI to a client device in a playlist; and

sending the encrypted content to the client device.

2. The method of claim 1, wherein the playlist is sent to an application of the client device.

3. The method of claim 2, wherein when Advanced Encryption Standard 128 bit Cypher Block Chaining (AES-128 CBC) encryption is used, the encrypted content is sent to a media player of the client device.

4. The method of claim 2, wherein when Advanced Encryption Standard 128 bit Electronic Code Book (AES-128 ECB) MPEG-2 Transport Stream (MP2TS) encryption is used, the encrypted content is sent to the application of the client device.

5. A method of claim 2, wherein the server device is also comprising

receiving and processing a channel change request from a client device;

deriving a new DRM-based live streaming content key;

transcoding and encrypting live streaming content using the DRM-based live streaming content key;

6. The method of claim 5, wherein receiving and processing a channel change request from the client device further comprises:

monitoring content copy control information (CCI) to listen for changes; and

obtaining the new CCI data;

generating a new key based in part on the change in CCI;

transcoding and encrypting live streaming content using the new DRM-based live streaming content key;

7. The method of claim 5, wherein deriving the DRM-based live streaming content key further comprises:

creating a rights data file and storing a subkey used to derive the DRM-based live streaming content key.

8. The method of claim 5, wherein the URI provides a reference to the DRM-based live streaming content key.

9. The method of claim 7, comprising:

determining a change in a detected parameter while currently tuned to a channel and using the DRM-based live streaming content key;

deriving a new DRM-based live streaming content key in response to the detected parameter;

transcoding and encrypting live streaming content using the new DRM-based live streaming content key;

segmenting the encrypted content; and

creating a playlist, the playlist comprising at least a URI of the new DRM-based live streaming content key, an encryption method, and a list of a plurality of the encrypted segments, the URI indicating usage of a DRM key management protocol.

10. The method of claim 9, wherein the detected parameter comprises changes in copy control information.

11. The method of claim 9, wherein the detected parameter comprises an amount of time the previous key has been used.

12. The method of claim 9, wherein the detected parameter comprises a program boundary.

13. A method for providing live streaming using Digital Rights Management (DRM) based key management via a client device, comprising:

sending a request for a playlist from a media player of the client device; and

receiving, at an application, the playlist from a server, the playlist comprising at least a Uniform Resource Identifier (URI) of a DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments, the URI indicating usage of a DRM key management protocol; and

using the DRM key management protocol to establish the DRM-based live streaming content key.

14. The method of claim 13, wherein the DRM-based live streaming content key is sent from the application of the client device to a media player of the client device using Secure Socket Layer (SSL) encryption over Secure HyperText Transfer Protocol (HTTPS).

15. The method of claim 14, wherein the plurality of encrypted segments is decrypted, decoded, and presented by the media player.

16. The method of claim 13, wherein the application decrypts the plurality of encrypted segments using the DRM-based live streaming content key to produce un-encrypted content segments.

17. The method of claim 16, wherein the un-encrypted content segments are sent from the application to a media player of the client device.

18. A method of claim 13, further comprising

sending a tune channel command from a client application of the client device to a server, the tune channel command including a channel identifier;

receiving a playlist URI from the server, after establishing the DRM-based live streaming content key, storing a temporary Rights Data File that includes a subkey used to derive the content decryption key; and

parsing the playlist to determine an applied encryption type.

19. The method of claim 18, further comprising:

decrypting the plurality of encrypted content segments;

generating a new playlist for the media player with an encryption method set to none and having no key URI; and

presenting the new playlist to the media player.

20. The method of claim 18, further comprising:

changing a format of the DRM-based live streaming content key to match a media player of the client device;

generating a new playlist for the media player that includes the applied encryption type with a key uniform resource locator (URI) pointed to a local Secure HyperText Transfer Protocol (HTTPS) server embedded within the client application; and

presenting the new playlist to the media player

21. The method of claim 13, further comprising:

in response to a server changing the content encryption key, receiving a playlist from the server, the playlist comprising at least a Uniform Resource Identifier (URI) of a new DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments, the URI indicating usage of a DRM key management protocol;

performing a key exchange protocol with the server under the indicated DRM key management system to establish the new key;

storing a temporary Rights Data File that includes a subkey used to derive the new decryption key; and

parsing the playlist to determine an applied encryption type.

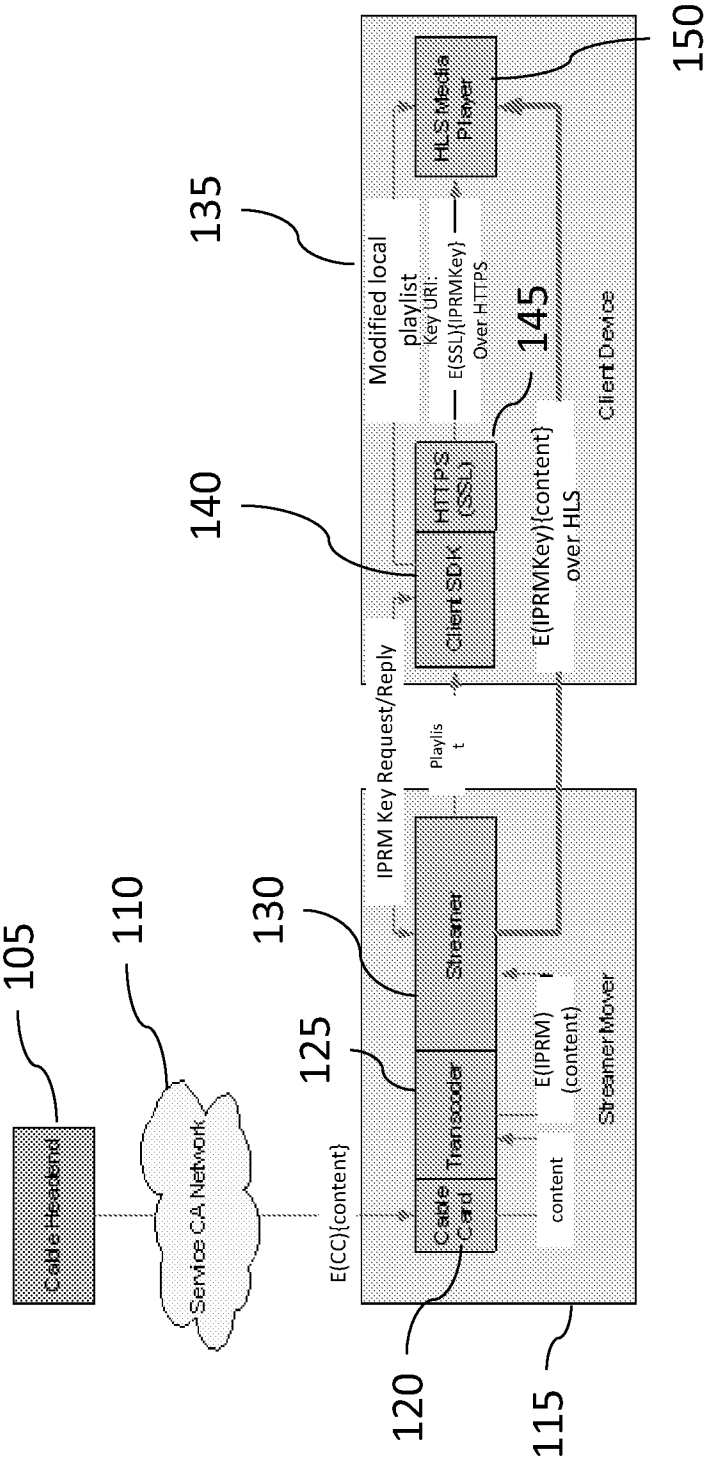

105
110
130
125
135
140
145
150
120
115
Cable Headend
Service CA Network
E(CC){content}
Cable Card
Transcoder
Streamer
IPRM Key Request/Reply
Playlist
E(IPRM){content}
content
Streamer Mover
Modified local playlist
Key URI:
E{SSL}{IPRMKey}
Over HTTPS
HTTPS (SSL)
Client SDK
E(IPRMKey){content}
over HLS
Client Device
HLS Media Player

FIG. 1

100

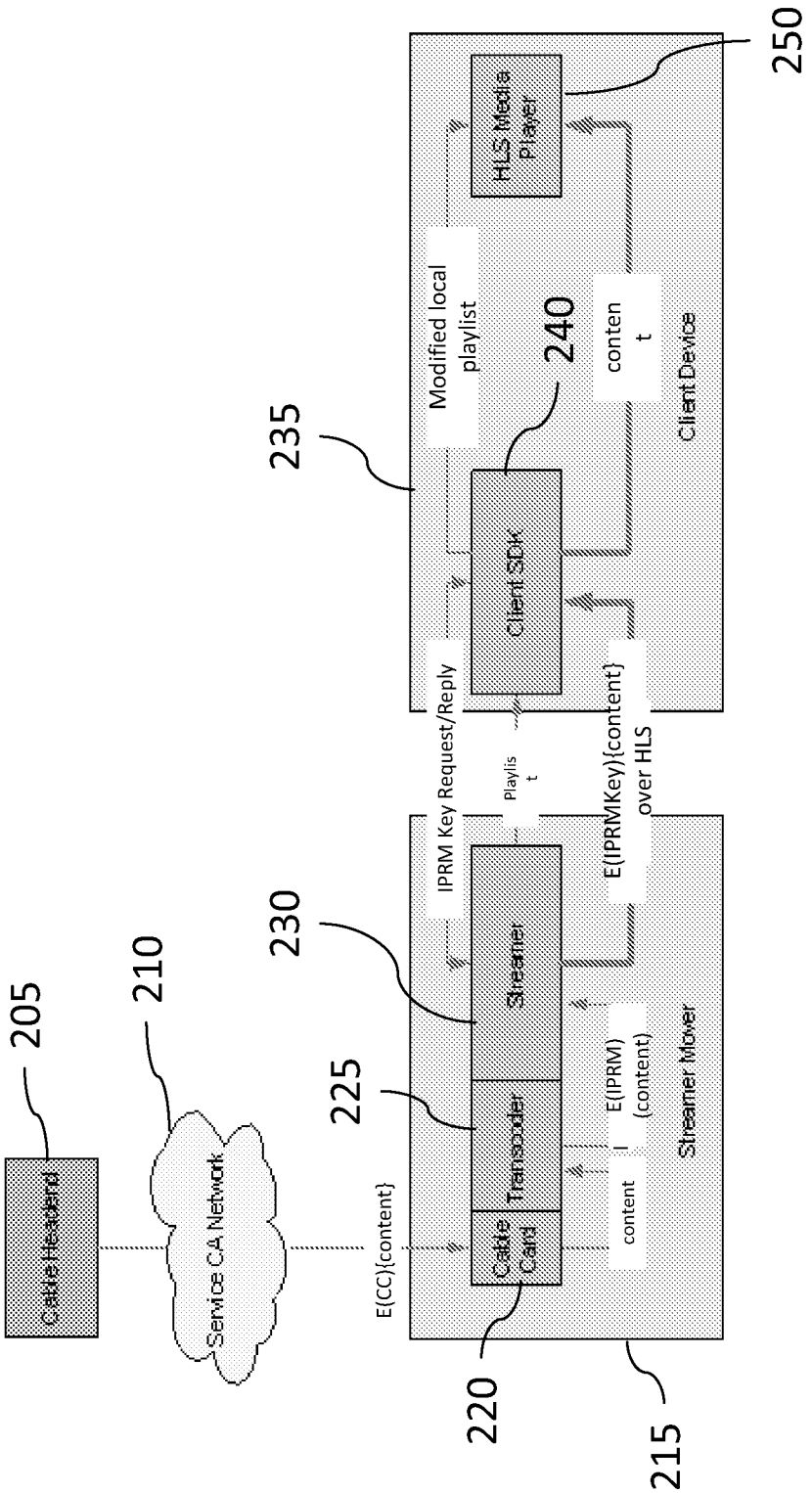

205
Cable Headend
210
Service CA Network
230
225
E(CC){content}
Cable Card
Transcoder
Streamer
IPRM Key Request/Reply
Playlist
E(IPRM){content}
over HLS
E(IPRM)
(content)
content
Streamer Mover
220
215
235
Modified local playlist
240
Client SDK
content
Client Device
HLS Media Player
250
200

FIG. 2

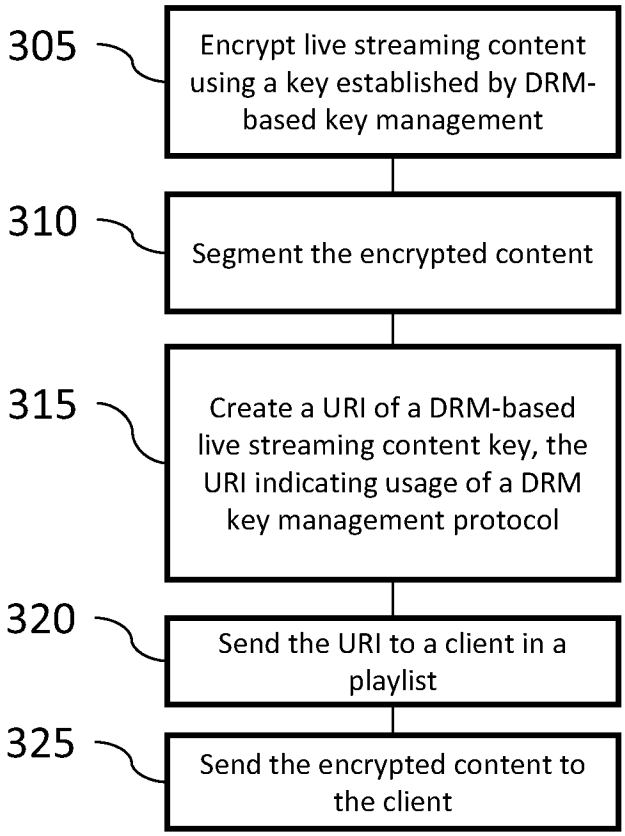

305
Encrypt live streaming content using a key established by DRM-based key management
310
Segment the encrypted content
315
Create a URI of a DRM-based live streaming content key, the URI indicating usage of a DRM key management protocol
320
Send the URI to a client in a playlist
325
Send the encrypted content to the client
300

FIG. 3

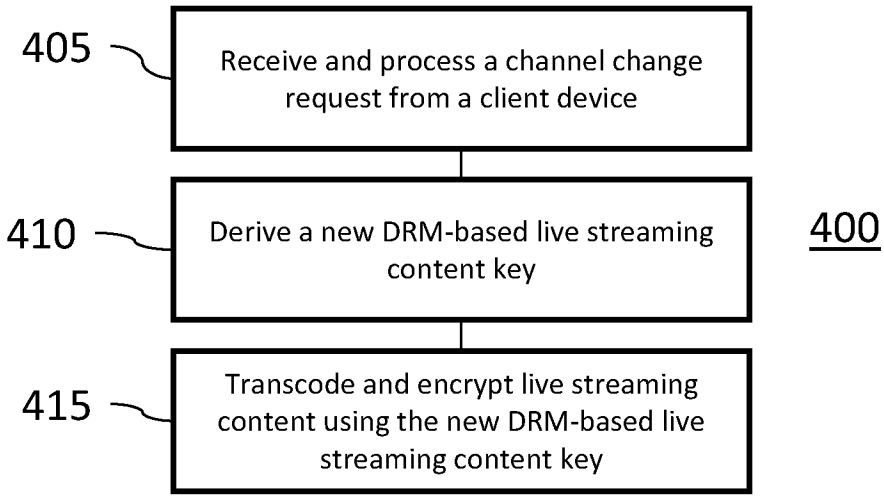

405
Receive and process a channel change request from a client device
410
Derive a new DRM-based live streaming content key
400
415
Transcode and encrypt live streaming content using the new DRM-based live streaming content key

FIG. 4

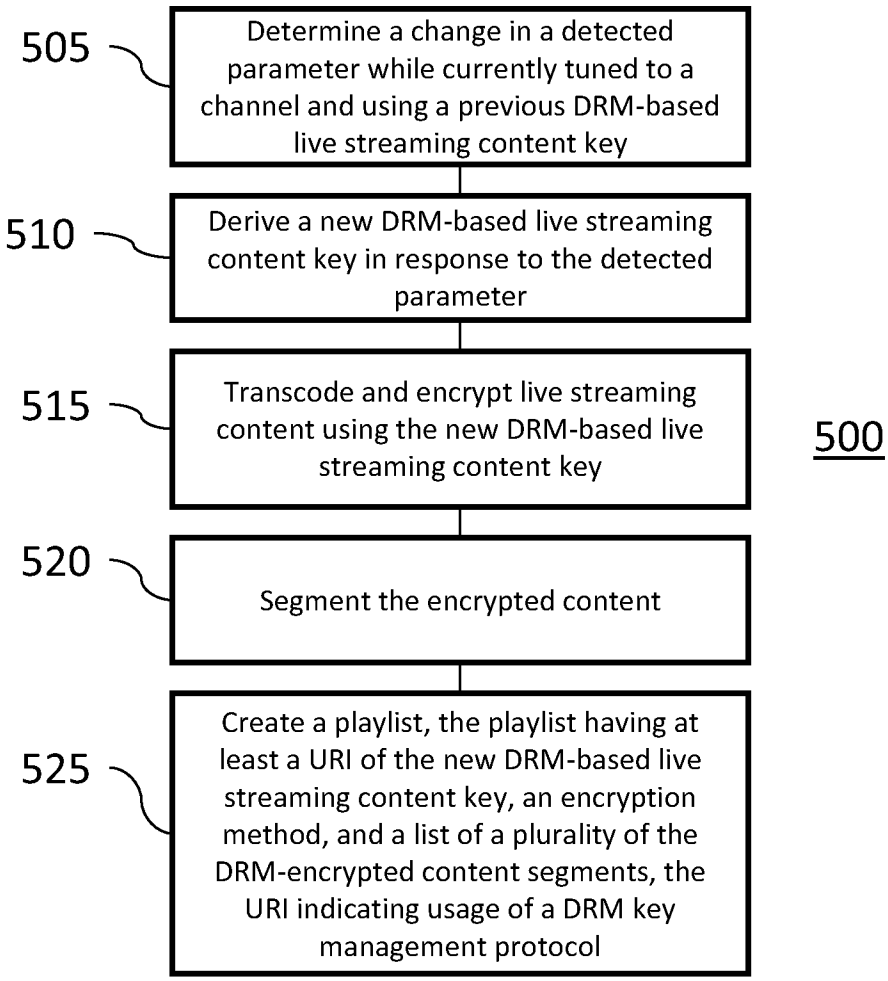

505
Determine a change in a detected parameter while currently tuned to a channel and using a previous DRM-based live streaming content key
510
Derive a new DRM-based live streaming content key in response to the detected parameter
515
Transcode and encrypt live streaming content using the new DRM-based live streaming content key
500
520
Segment the encrypted content
525
Create a playlist, the playlist having at least a URI of the new DRM-based live streaming content key, an encryption method, and a list of a plurality of the DRM-encrypted content segments, the URI indicating usage of a DRM key management protocol

FIG. 5

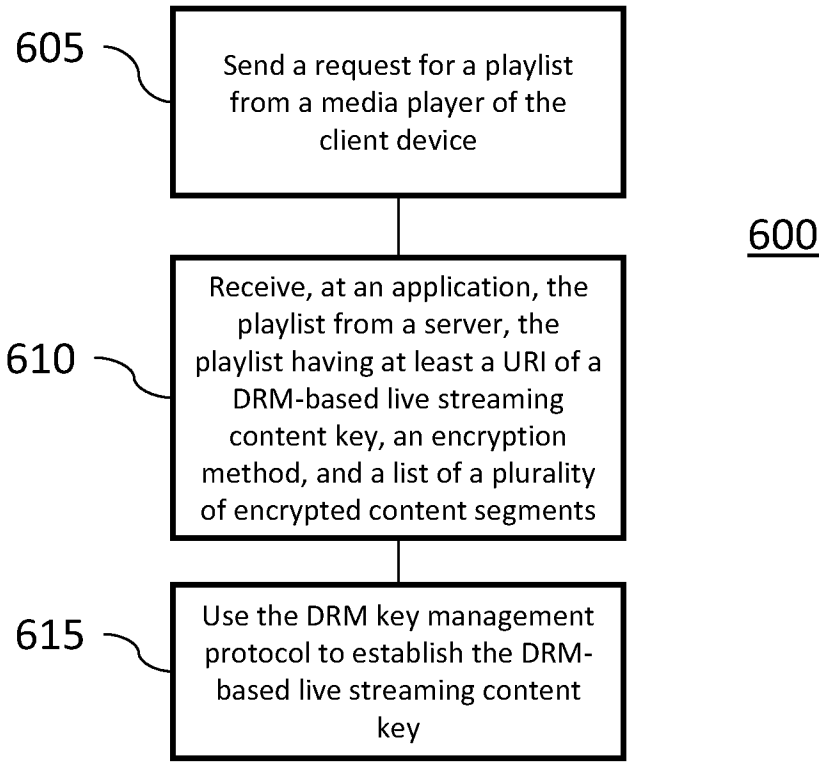

605
Send a request for a playlist from a media player of the client device
600
610
Receive, at an application, the playlist from a server, the playlist having at least a URI of a DRM-based live streaming content key, an encryption method, and a list of a plurality of encrypted content segments
615
Use the DRM key management protocol to establish the DRM-based live streaming content key

FIG. 6

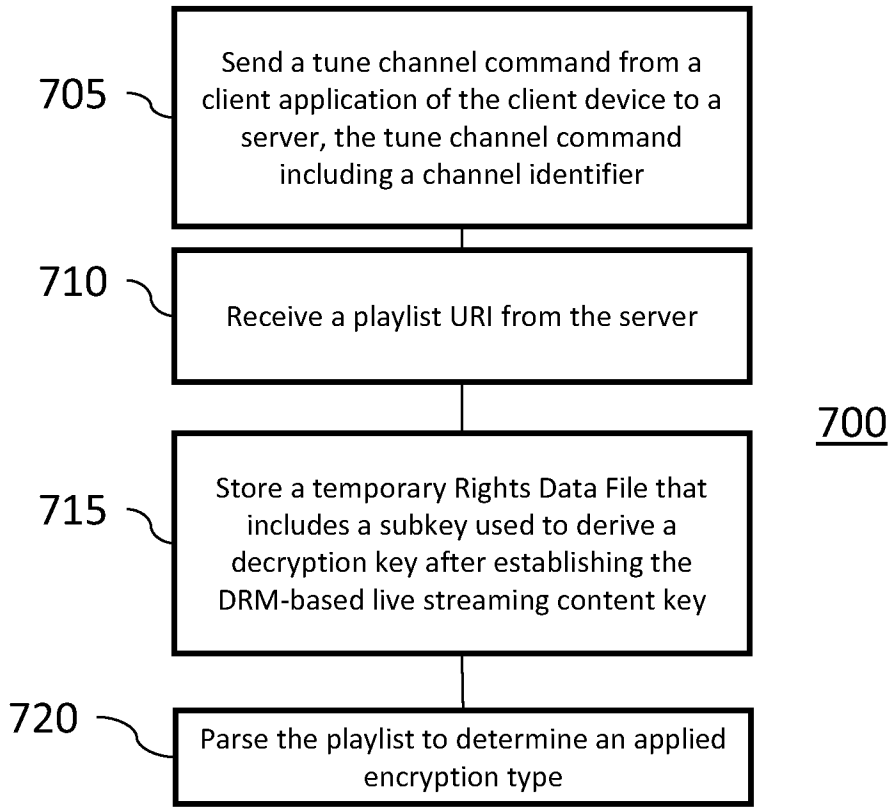

705
Send a tune channel command from a client application of the client device to a server, the tune channel command including a channel identifier
710
Receive a playlist URI from the server
700
715
Store a temporary Rights Data File that includes a subkey used to derive a decryption key after establishing the DRM-based live streaming content key
720
Parse the playlist to determine an applied encryption type

FIG. 7

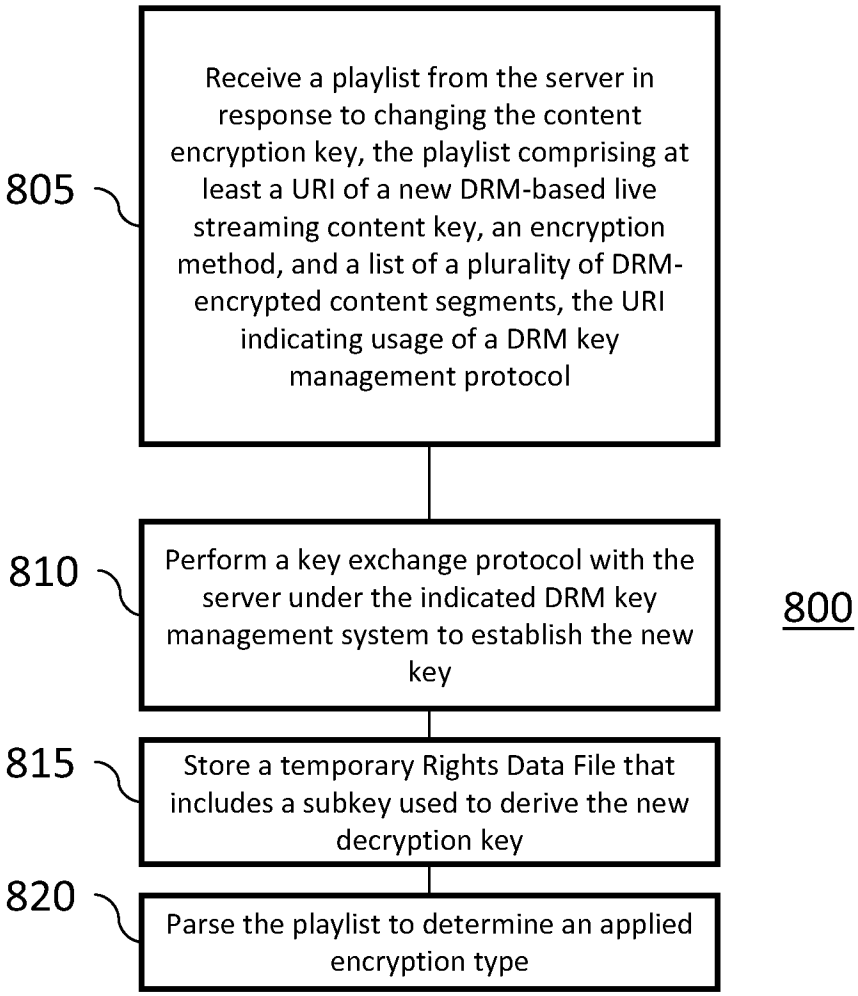

Receive a playlist from the server in response to changing the content encryption key, the playlist comprising at least a URI of a new DRM-based live streaming content key, an encryption method, and a list of a plurality of DRM-encrypted content segments, the URI indicating usage of a DRM key management protocol
805
Perform a key exchange protocol with the server under the indicated DRM key management system to establish the new key
810
800
Store a temporary Rights Data File that includes a subkey used to derive the new decryption key
815
Parse the playlist to determine an applied encryption type
820

FIG. 8

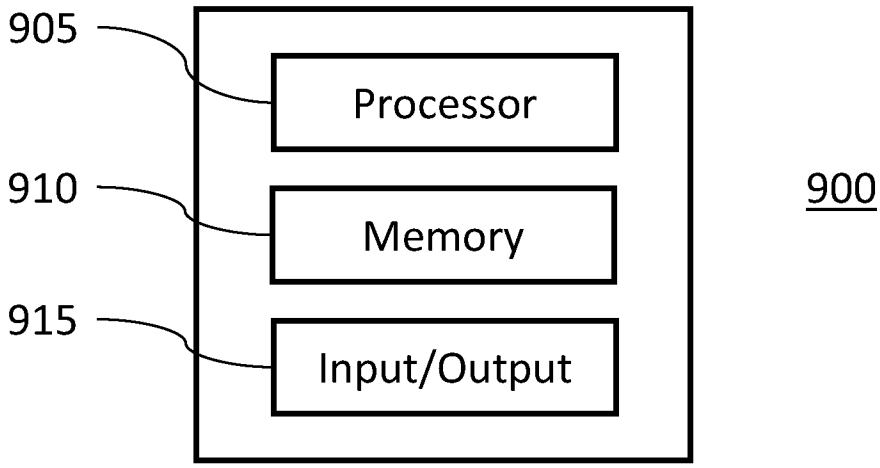

905
Processor
910
Memory
915
Input/Output
900

FIG. 9

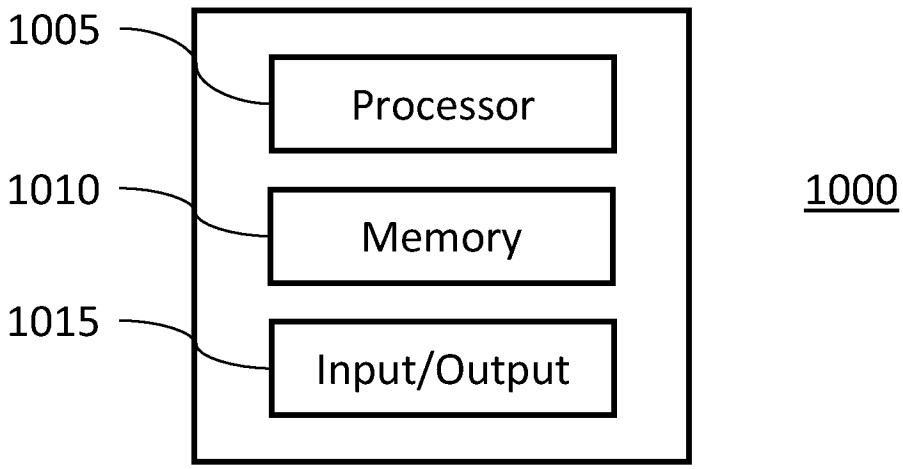

1005
Processor
1010
Memory
1015
Input/Output
1000

FIG. 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2012/030469

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2010/169303 A1 (BIDERMAN DAVID [US] ET AL) 1 July 2010 (2010-07-01) paragraphs [0013] - [0018], [0037] - [0041], [0044], [0047], [0050], [0094], [0095]	1-21
X	PANTOS R ET AL: "HTTP Live Streaming; draft-pantos-http-live-streaming-05.txt", HTTP LIVE STREAMING; DRAFT-PANTOS-HTTP-LIVE-STREAMING-05.TXT, INTERNET ENGINEERING TASK FORCE, IETF; STANDARDWORKINGDRAFT, INTERNET SOCIETY (ISOC) 4, RUE DES FALAISES CH- 1205 GENEVA, SWITZERLAND, no. 5, 19 November 2010 (2010-11-19), pages 1-22, XP015072670, [retrieved on 2010-11-19] paragraphs [0002], [0004], [0005], [6.2.3], [6.2.4]	1-21

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
31 July 2012	06/08/2012

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Veen, Gerardus

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2012/030469

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2010169303 A1	01-07-2010	EP 2475149 A2	11-07-2012
		US 2010169303 A1	01-07-2010
		US 2010169453 A1	01-07-2010
		US 2010169458 A1	01-07-2010
		US 2010169459 A1	01-07-2010
		US 2012110141 A1	03-05-2012