Innovation, Sciences et Développement économique Canada
Office de la Propriété Intellectuelle du Canada

Innovation, Science and Economic Development Canada
Canadian Intellectual Property Office

CA 2916478 C 2020/07/28

(11)(21) 2 916 478

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) **Date de dépôt PCT/PCT Filing Date:** 2014/06/23
(87) **Date publication PCT/PCT Publication Date:** 2014/12/24
(45) **Date de délivrance/Issue Date:** 2020/07/28
(85) **Entrée phase nationale/National Entry:** 2015/12/21
(86) **N° demande PCT/PCT Application No.:** US 2014/043728
(87) **N° publication PCT/PCT Publication No.:** 2014/205450
(30) **Priorités/Priorities:** 2013/06/21 (US61/838,180); 2013/07/15 (US61/846,593); 2013/07/19 (US61/856,536); 2013/07/23 (US61/857,605); 2013/09/06 (US61/874,478); 2013/10/11 (US61/890,075); 2014/05/30 (US62/004,953); 2014/06/23 (US14/312,558); 2014/06/23 (US14/312,536)

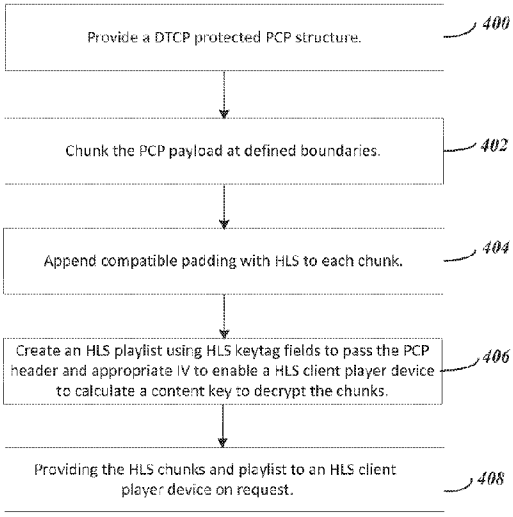
(51) **Cl.Int./Int.Cl.** ***H04N 21/845*** (2011.01), ***H04L 29/06*** (2006.01), ***H04N 21/2343*** (2011.01), ***H04N 21/266*** (2011.01), ***H04N 21/4363*** (2011.01), ***H04N 21/4402*** (2011.01), ***H04N 21/61*** (2011.01), ***H04N 21/647*** (2011.01), ***H04N 21/8355*** (2011.01)

(72) **Inventeurs/Inventors:**
MORONEY, PAUL, US;
MANGALORE, GEETHA, US;
FRANKS, WILLIAM P., US

(73) **Propriétaire/Owner:**
ARRIS ENTERPRISES LLC, US

(74) **Agent:** GOWLING WLG (CANADA) LLP

(54) **Titre : CONVERTISSEUR DTCP POUR HLS**
(54) **Title: DTCP CONVERTER FOR HLS**



(57) **Abrégé/Abstract:**
A method for DTCP to HLS conversion is provided that starts with a standard DTCP Protected Content Packet (PCP) structure. The PCP payload data is chunked at defined chunk boundaries. Each chunk is then appended with a pad to be compatible with HLS. An HLS playlist is then provided using the PCP header with identification of the chunks and a keytag. The chunk is encrypted with a DTCP key calculated by the DTCP standard using: (a) copy control bits; (b) a nonce, and (c) an exchange key ID. Relevant PCP header fields are provided in the keytag for the HLS playlist supporting the transaction that enables calculation of the DTCP content key to enable later decryption of the chunks. The system further provides a revised HLS GET for DLNA to enable trick play seek operations to be performed on the converted HLS.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 December 2014 (24.12.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/205450 A3

(51) International Patent Classification:
H04N 21/845 (2011.01) *H04N 21/4363* (2011.01)
H04N 21/2343 (2011.01) *H04N 21/4402* (2011.01)
H04N 21/266 (2011.01) *H04N 21/8355* (2011.01)
H04N 21/647 (2011.01) *H04L 29/06* (2006.01)
H04N 21/61 (2011.01)

(21) International Application Number:
PCT/US2014/043728

(22) International Filing Date:
23 June 2014 (23.06.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/838,180	21 June 2013 (21.06.2013)	US
61/846,593	15 July 2013 (15.07.2013)	US
61/856,536	19 July 2013 (19.07.2013)	US
61/857,605	23 July 2013 (23.07.2013)	US
61/874,478	6 September 2013 (06.09.2013)	US
61/890,075	11 October 2013 (11.10.2013)	US
62/004,953	30 May 2014 (30.05.2014)	US
14/312,536	23 June 2014 (23.06.2014)	US
14/312,558	23 June 2014 (23.06.2014)	US

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

(72) Inventors: **MORONEY, Paul**; 7357 Fay Avenue, La Jolla, California 92037 (US). **MANGALORE, Geetha**; 12279 Oak View Way, San Diego, California 92128 (US). **FRANKS, William P.**; 13295 Capstone Drive, San Diego, California 92130 (US).

(74) Agents: **WARD, Thomas A.** et al.; 3871 Lakefield Drive, Suwanee, Georgia 30024 (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, BN, 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, IR, IS, JP, KE, KG, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, 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.

[Continued on next page]

(54) Title: DTCP CONVERTER FOR HLS

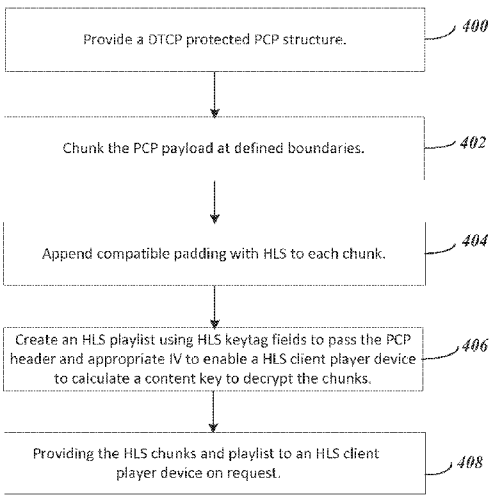


FIG. 4

(57) Abstract: A method for DTCP to HLS conversion is provided that starts with a standard DTCP Protected Content Packet (PCP) structure. The PCP payload data is chunked at defined chunk boundaries. Each chunk is then appended with a pad to be compatible with HLS. An HLS playlist is then provided using the PCP header with identification of the chunks and a keytag. The chunk is encrypted with a DTCP key calculated by the DTCP standard using: (a) copy control bits; (b) a nonce, and (c) an exchange key ID. Relevant PCP header fields are provided in the keytag for the HLS playlist supporting the transaction that enables calculation of the DTCP content key to enable later decryption of the chunks. The system further provides a revised HLS GET for DLNA to enable trick play seek operations to be performed on the converted HLS.

WO 2014/205450 A3

WO 2014/205450 A3

(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, 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, KM, 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))*

(88) Date of publication of the international search report:
19 February 2015

DTCP CONVERTER FOR HLS

[0001] TECHNICAL FIELD

[0002] The present invention relates to the field of digital video streaming. More particularly, the invention relates to securely streaming media content using both the HTTP Live Stream (HLS) standard and the Digital Transmission Content Protection (DTCP) over Internet Protocol standard.

BACKGROUND

[0003] Cable system operators or other networks operators feed streaming media to a gateway device for distribution in a consumer's home. The gateway device can offer a singular means to access all forms of content-live, on-demand, online, over-the-top, or Digital Video Recorders (DVRs) within homes today. The gateway enables connection to the home network

CA 2916478 2017-06-09

devices, for example by connecting to a WiFi router or a Multimedia over Coax Alliance (MoCA) connection that provides IP over in-home coaxial cabling.

[0004] Consumers desire to use devices that comply with a common standards compliant approach to access streaming video from a home gateway, so that all their home devices will be able to receive streaming video content provided from the same home gateway. DTCP is a standard defined for a significant number of consumer devices. Apple's HLS is another standard often mandated by Apple to access content using its devices. DTCP and HLS are not compatible, but are close in some ways. It is desirable to use HLS player devices on DTCP compliant systems.

[0005] When an IP device in the home is a mobile client, like an iPad, it can travel and appear outside the home. The user outside the home may still desire to stream content from his home gateway's storage. To stream content from that gateway, cable box or DVR that are DTCP compatible using a WiFi router to a remote location, DTCP imposes certain requirements. Apple imposes other requirements based upon its HLS standard that are mandatory when remote connection occurs over a 3G or 4G network. Content provided from gateways, cable boxes and DVRs further encrypt their content or have other digital rights management schemes in place to prevent unauthorized copying or transfer of media content.

[0006] It is desirable for the DTCP standard to be implemented so that it is compatible with the HLS standard used by Apple® devices that run the iOS® operating system, such as the iPhone® and iPad®, allowing an HLS player to operate with a DTCP compatible system.

SUMMARY

[0007] Embodiments of the present invention provide a DTCP translator that starts with DTCP standard compliant video and converts the video for compatibility with the HLS standard. The embodiments further enable a trick play seek operation to be provided with the converted HLS.

[0008] In one embodiment, a method for DTCP to HLS conversion is provided that starts with a standard DTCP Protected Content Packet (PCP) structure. The PCP payload data is chunked at defined chunk boundaries. Each chunk is then appended with a pad encrypted with the same key and appropriate IV to be compatible with HLS. An HLS playlist is then provided using the PCP header with identification of the chunks and a keytag. The chunk is encrypted with a DTCP key calculated by the DTCP standard using: (a) copy control bits; (b) a nonce, and (c) an exchange key ID. Relevant PCP header fields are provided in the keytag for the HLS playlist, including the value of the copy control bits, the nonce and the exchange key ID, supporting the transaction that determines the exchange key and the subsequent calculation of the content key to enable encryption and decryption of the chunks.

[0009] Embodiments of the present invention also provide a process to enable trick play operation that is provided for HLS streaming video that has been converted by a system from DTCP. The system server for the trick play operation provides a modified SEEK operation when an HLS GET message is received from an HLS client player. For the process, a DLNA header is provided from the HLS client player by including it in the HLS GET message. The HLS client also provides a DLNA RANGE REQUEST that requests a range of chunks making up a video desired and a seek point from where a seek operation is needed. The HLS server recognizes the DLNA header of the HLS GET message and DLNA RANGE REQUEST and

obtains a range of chunks making up an extent of the recorded video using metadata fields. The server then generates a new HLS playlist with identification of the chunks and keytag corresponding to the seek operation. The server will provide chunks from the seek point and a rolling playlist to identify chunks and keytag from the seek point.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] Further details of the present invention are explained with the help of the attached drawings in which:

[0011] Fig. 1 provides a system overview, illustrating a connection of components that can stream video to an HLS device from a DTCP compliant system when using embodiments of the present invention.

[0012] Fig. 2 illustrates the structure of a PCP for DTCP that is repackaged to enable use with HLS.

[0013] Fig. 3 shows components of a system that use an HLS Adapter System according to embodiments of the present invention to stream video.

[0014] Fig. 4 provides a flowchart showing general operation to provide HLS adapted from DTCP according to embodiments of the present invention.

[0015] Fig. 5 provides a flowchart showing general operation to provide a trick play seek operation for HLS video that is adapted from DTCP.

DETAILED DESCRIPTION

System Overview

[0016] Fig. 1 provides a system overview, illustrating a connection of components that can stream video to an HLS device from a DTCP compliant system when using embodiments of the present invention. In Fig. 1, a cable network connection is shown provided to a gateway 100 of a cable customer's home 102. The cable network connection provided to the gateway 100 can be to a cable system operator or other streaming content provider such as a satellite system. The gateway 100 then provides content to DTCP compatible devices in a home network 104 in the consumer's home 102. The home network can include a router 106 that receives IP content from the gateway and distributes the content over a WiFi or cable connection to client devices 111-113. The router 106, although shown separately, can be part of the gateway 100.

[0017] The home network 104 can further offer an IP connection that extends outside the home using wireless or cable connections that extend to the cloud 114. The content from the cloud 114 can then be accessed by an HLS device 116 through a 3G or 4G network 118. Using methods of embodiments of the present invention, the DTCP content provided from the gateway 100 is converted to an HLS compatible format and transmitted as HLS so that the HLS player device 116 will be compatible to receive the content from the gateway 100.

HLS Adaptor Overview

[0018] Embodiments of the present invention operate based on a determination of what makes DTCP not HLS compatible, and provide a DTCP encapsulation scheme that keeps the maximum amount of DTCP as possible while adapting portions that are needed for HLS compatibility. HLS will be the output provided from the system to satisfy the Apple standard,

but the new DTCP encapsulation scheme will be used to alter DTCP compatible content to effectively form the "HLS adapter."

[0019] As a first requirement for HLS, the content must be HLS chunks. However, copy control and key definition can still come from DTCP. Looking at the Protected Content Packet (PCP) structure for DTCP, the content provided in the PCP payload can be broken into HLS chunks in real time. Besides the PCP payload, for HLS compatibility the standard PCP header information can be taken and made available to the HLS client device in the HLS key tag field, referred to herein as a new query field ?pcph = value. By providing the HLS key tag field, an advantage is that real DTCP keys can be used, exactly as DTCP would normally have. The encryption process can then be accomplished in a similar manner for HLS and DTCP, as the content encryption process for these standards differ only in the padding used.

PCP Structure

[0020] Fig. 2 illustrates the structure of a PCP for DTCP that is repackaged to enable use with HLS. In Fig. 2 the typical DTCP PCP 200 components are shown, including a header 202, encrypted data payload content 204 and padding 206. A typical DTCP PCP 200 structure encapsulates minutes of MPEG-2 transport format content 204 that is encrypted with AES-128 CBC, and includes the header 202 and padding 206. Standard HLS chunks are usually only seconds of content, rather than minutes, encrypted with the same AES-128 CBC, but with slightly different padding. The HLS adaptation of the DTCP according to embodiments of the present invention can, thus, be viewed as a repackaging of an encrypted DTCP PCP structure, encrypted under the identical key. The vast majority of the encrypted payload will be identical.

[0021] Fig. 2 further illustrates how the structure of a PCP for DTCP can be repackaged to enable use with HLS. In Fig 2, each HLS chunk, such as 221 and 231, is formed from the PCP payload 204 and must have an Initialization Vector (IV), such as 227 and 237, and end with padding, such as 225 and 235, defined by Apple as PKCS7. The PCP payload 204 is divided at 16 byte boundaries, such as 211 and 212, so that the HLS padding needed, including 225 and 235, will also involve adding 16 bytes, and so that subsequent PCP encrypted content can be incorporated directly into chunks as shown. Both the chunks 221 and 231 with their 16 byte boundaries 211 and 212 and the 16 byte padding 225 and 235 will be encrypted with the DTCP content key, and appropriate IV. The first HLS chunk uses IV1 and is calculated using information from the PCP header 202 and DTCP parameters including an exchange key. The 16 byte padding 225 of the first HLS chunk 221 added at the end is encrypted using the last 16 bytes 227 of encrypted data of the first HLS chunk. Each subsequent chunk will be encrypted with content key and initial IV used will be the last 16 bytes of encrypted data of previous block (same as last 16 bytes of PCP encrypted data prior to start of that chunk). For example, for the second chunk, the IV comes from the last 16 bytes of the first chunk 221 and is 227. With this choice of IV, it is not necessary to decrypt and reencrypt the second chunk 231. The same method is followed for all the subsequent chunks until the last chunk. Only when we reach the very end of the PCP is the last chunk handled slightly differently when it is not a multiple of 16 bytes. For the last chunk, the very last 16 bytes of the PCP must be decrypted, the padding discarded, PKCS padding added, and then those 16 bytes reencrypted.

HLS Adapter System

[0022] Fig. 3 shows components of a system that use an HLS Adapter System according to embodiments of the present invention to stream video. The system includes an HLS Client Device, such as an iPad or other Apple device 320 along with Cable Operator Systems 300 and 310. The Cable Operator Systems 300 and 310 function to stream video data with a home gateway as shown in Fig. 1. Although two cable operator systems 300 and 310 are shown, a single cable operator can house the components shown.

[0023] The Cable Operator System 300 provides DTCP standard communications and includes a DTCP Security Server 302 and a DTCP Content Server 304. The security server 302 provides for the Authentication and Key Exchange (AKE) standard transaction in DTCP and can communicate with external devices to deliver DTCP Exchange Keys. The content server 304 delivers streaming data in PCP format with a header, content and pad as described with respect to Fig. 2.

[0024] The Cable Operator System 310 provides for HLS communications that uses a "HLS Adapter" under embodiments of the present invention to adapt DTCP content. The Cable System 310 includes an HLS Playlist Server 312 and an HLS Content Server 314. The HLS content is delivered in HLS chunks to the HLS client device player 320 from content server 314. The chunks are created as described above with respect to Fig. 2. The HLS Playlist server 312 delivers an HLS playlist that includes a new "pcph" query field to include an exchange key ID taken from the standard PCP header 202 of Fig. 2. Note that although the content server 314 and playlist server 312 are shown as two separate devices, the components for the servers 312 and 314 can be combined to a single server device. The HLS client device player 320 takes the exchange key ID, and uses an internal application program(APP) or software development kit

(SDK) to communicate with the DTCP security server 302 to obtain the exchange key in a standard AKE transition for decryption of the HLS chunks using a content key calculated in a manner standard to DTCP.

[0025] The servers of the cable operator systems 300 and 302 and the client device 320 of Fig. 3 include at least one processor and at least one memory device to enable the processes of embodiments of the present invention. The memory devices will store software that when executed by the processor will cause the processor to perform the steps of the processes described.

PCP Header for Encryption

[0026] To provide for encryption and decryption, the PCP header information, including an exchange key ID, is obtained and appended to the usual HLS playlist keytag entry. The PCP header also provides copy control, nonce and CMI fields that are appended to the HLS playlist keytag entry to enable obtaining the exchange key. Of the 14 bytes of a typical DTCP PCP header, the last 4 length bytes are not needed, so the ?pcph query field used to append to the HLS playlist keytag entry would be only the first 10 bytes of the header, suitably base-64 encoded.

Proxy for Native Client HLS Player to Obtain Content Key

[0027] The content key can be provided to the client's native HLS player in several ways, which are client design choices. One way the content key can be provided is via a key proxy using 2-way SSL so that there is a secure binding to the native HLS player and its internal stored identity keys and certificate. For Fig. 3, the 2-way SSL connection can exist between the HLS

player 320 and the security server 302 that can serve as a proxy server, such that the content key can be securely delivered to the HLS player 320.

[0028] A second way in which the content key can be provided to the native HLS player is via Apple's "custom URL" approach. In this case, the player directs its key request to the custom URL handler "proxy" that matches the protocol present in the key tag URL.

[0029] In normal use, the native HLS Apple player 320 would present the query string to the client key proxy, along with the key request, so that the DTCP subsection could use that header in its computation of the DTCP content key. In other words, the HLS player 320 looks at the keytag and hands over the URL and query field from the key tag to a client SDK proxy 322. The SDK proxy 322 then uses the information to obtain the DTCP exchange key from the server 302 and then calculates the DTCP content key in part based upon the exchange key and the query field. The SDK proxy 322 then provides the content key internally to the HLS player.

Playlist Structure

[0030] The HLS playlist used with embodiments of the present invention would be the real-time rolling style playlist, with 3 HLS chunks defined. Thus real-time streaming can be supported under HLS with continuous groups of 3 HLS chunks provided, just as DLNA and DTCP can be used for streaming with a flow of PCPs. For the HLS adaptation of the present invention, each content chunk in the playlist would have a corresponding keytag, as the IV changes for each chunk as described above. Though, while rolling through a many minutes long PCP, only one actual ?pcph query field would be in use, as the chunks all come from the same PCP. Only when the end of one playlist was reached, and a second started, would there be two different ?pcph query fields, and two different keys, present and in use for a brief time.

[0031] The client HLS player 320 of Fig. 3 will issue a HTTP GET command to the HLS playlist server 312 to get the playlist. The HTTP GET has the following example HTTP header structure:

```
GET /directoryname/Movie.m3u8
Host: 192.168.1.100:7878
Accept: */*
User-Agent: ArrisSDKProxy/0.1 (iPad; en_us)
```

[0032] An example HLS playlist obtained by the HTTP GET command is shown below. The playlist, Movie m3u8, has a structure assuming approximately 2 second duration chunks, and shows the first 3 chunks of the content. The HLS playlist example is as follows:

```
#EXTM3U
#EXT-X-TARGETDURATION: 2
#EXT-X-MEDIA-SEQUENCE:0
#EXT-X-KEY:METHOD=AES-
128,URI="https://iprm.tv/key?pcph=AkRCAAKVyRMZZg==",IV=0xd7dd62914a57a
1a0d3f12a05a6885b1a
#EXTINF:2,
Movie_0.ts
#EXT-X-KEY:METHOD=AES-
128,URI="https://iprm.tv/key?pcph=AkRCAAKVyRMZZg==",IV=0x4f755cc6fe00c
7f69df46a9603f60aab
#EXTINF:2,
Movie_1.ts
#EXT-X-KEY:METHOD=AES-
128,URI="https://iprm.tv/key?pcph=AkRCAAKVyRMZZg==",IV=0x3c1f507b6806a
33c1e1a544e1eff1e86
#EXTINF:2,
Movie_2.ts
```

[0033] For the example above, the playlist attributes are described as follows. First, three different urls are provided with label "128, URI=https://..." one for each of the 3 chunks that contain the keytag ?pcph query strings. Note that this example is using an https proxy to accomplish delivery of the content key to the native client HLS player.

[0034] For the keytag ?pcph query strings of this example, the 10 byte PCP header is base 64 encoded and provided in the HLS keytag "?pcph" query string. Thus 10 bytes become 16 as shown. The PCP header fields in this example [bytes 0 and 1] are for copy free content with redistribution control, and exchange key label 0x44, and baseline AES cipher. As required by DTCP, the first nonce field [header bytes 2-9] of any connection starts with the "PCP-UR" field, header bytes 2 and 3, and ends with a 48 bit field SN_C whose MSbit is a zero. The remaining 47 bits are random. Subsequent SN_C fields (from subsequent PCP packets) within the same content flow can increment as required. In this example, the playlist identifies the first 3 two-second chunks of what would have been the first PCP packet of a connection flow. The 3 keytags show 3 different IVs but the same PCP header, and thus indicate the same content key.

[0035] As normal to HLS, a new playlist is presented roughly each 2 second interval, dropping the oldest chunk and adding the newest. Once the content flow moved beyond the chunks in the first original PCP packet, on to what would have been the second PCP packet (and second PCP header), the playlist would show the last two chunks of the first PCP, and the first chunk of the second. For this playlist, a new ?pcph query field would be introduced, indicating a second key was in use for that newest chunk. This new query field would roll through the playlist and eventually be the only one present, until the third PCP was reached, and so on.

Overview Flowchart

[0036] Fig. 4 provides a flowchart showing the general process of operation to provide HLS adapted from DTCP according to embodiments of the present invention. In Fig. 4, the process begins in step 400 by providing a DTCP protected PCP structure. In the next step 402, the PCP payload is chunked at defined boundaries to provide the chunks required by HLS. Compatible

padding is then added to each chunk in step 404 that is compatible with HLS. In step 406 an HLS playlist is created using the PCP header to provide a keytag to enable an HLS client device to calculate a content key to decrypt the chunks. Finally, the HLS chunks and playlist are provided to the HLS client device on request.

PCP2 Support

[0037] PCP2 is a slightly different DTCP-IP packet format meant to allow the use of CMI descriptors in key derivation and rules processing. The ?pcph query field described above for the playlist needs no changes, as it already includes a packet type field that indicates whether the header format is PCP or PCP2. What is missing is the carriage of the CMI field. To account for the carriage of the CMI field, embodiments of the present invention add another query string field, called ?cmi=value. The ?cmi query field includes the equivalent of the "CMI field" of DTCP, which is a concatenation of all the CMI descriptors to be sent by the source. Again, in this case of the HLS keytag, this cmi query field must be base 64 encoded.

[0038] Thus in the example playlist, with CMI descriptor 0, 1 and 2 defined and carried, the DTCP CMI field would have 17 bytes, and a base 64 coded version would have 23 characters. An example of a change of the above playlist to include the CMI field is as follows:

```
#EXT-X-KEY:METHOD=AES-
128,URI="https://iprm.tv/key?pcph=AkRCAAKVyPMZZg&cmi=AAAAABAABAA
DKVyRMZZghFm=",IV=0xd7dd62914a57a1a0d3f12a05a6885b1a
```

In this revised playlist, the actual bytes shown for the CMI field do not correspond to a specific cmi, but are present to show format and length.

Equivalent HLS Formation

[0039] In the embodiment described above, an original DTCP PCP structure is initially constructed, and then a new equivalent chunk and playlist structure is created that complies with HLS. As an alternative, a PCP is not actually constructed per se. Instead, the PCP header fields are determined, and the exchange key, and thus the content key and the first IV according to the DTCP specification. The chunks are constructed and encrypted directly, as if a PCP existed. The resulting content bytes are identical to the case where an actual PCP was constructed and converted.

Alternative Quasi-PCP Structure Format

[0040] In the embodiment described originally above, an original DTCP PCP structure is initially used, and a new equivalent chunk and playlist structure is created that complies with HLS, and contains the vast majority of the same content bytes as were in the original PCP. As an alternative, a PCP is not actually constructed per se. Instead, the PCP header is determined, and thus the content key and the first IV according to the DTCP spec. Then the content is divided up into chunks, and the first chunk is encrypted with the first IV. However, after the first chunk, the second chunk, third chunk, etc are encrypted with any IV that is convenient, rather than one derived from an actual equivalent PCP (encrypted packet). The IV match what would have been determined from the actual PCP. Thus the encrypted HLS content bytes would not match the byes of an equivalent PCP structure, although the content key was identical.

[0041] This alternative quasi-PCP structure is a bit simpler than the strict DTCP standard compliant PCP structure use in the system described above. The quasi-PCP structure allows some of the preprocessing for HLS chunking to be removed. Also, this quasi-PCP structure

allows for removal of decryption and re-encryption preprocessing needed when creating HLS content from an existing PCP structure.

Trick Play Operations Using HLS Adapter

[0042] Normally for trick-play operations such as seek, rewind and fast forward, a large amount of video content needs to be available at one time to accomplish the procedure. With conventional HLS, only three chunks are typically available real time with 2 or 3 seconds of content per chunk. Thus, a trick-play operation to search the content of a video is not possible. For a gateway with DVR support, many minutes of content are available stored typically on a hard drive. As an update to the HLS adapter described above, and similar to the way in which DLNA and DTCP are used for trick modes, the rolling playlist style of HLS is used as described above, and the following adaptations are used to allow trick-play operations to be performed.

[0043] For a trick-play operation to search through completed recordings, it is important to provide a solution to the need for a SEEK anywhere in that recording, even though the rolling HLS playlist describes no such comprehensive list of chunks. According to embodiments of the present invention to accomplish a seek, the Digital Living Network Alliance (DLNA) standard seek operation structure is used, which is done in a DLNA header. If we are using an adapted HLS player with a SDK, such as 322 of Fig. 3, as part of the SDK the DLNA header can be added to the HLS GET command for the HLS playlist. To accomplish the DLNA seek, a small change must be supported in the HLS server so that it recognizes DLNA headers in an otherwise normal HLS GET.

[0044] On a SEEK request from the an HLS client device such as 320 of Fig. 3, the SDK such as 322 sends the HLS playlist request with the addition of the DLNA seek header (DLNA

RANGE REQUEST). Upon receipt of the request, the HLS Server such as 312, after recognizing the DLNA SEEK header, would identify and generate the new playlist and chunks corresponding to the SEEK limits. The SDK would likely already know the extent of any completely recorded asset, as part of DLNA or proprietary metadata fields for the content.

[0045] The native HLS player however lacks the ability to know anything about the SEEK operation, or seek point, as it will be processing only live rolling style playlists. This SEEK to time position X can occur anywhere between zero and the asset duration. This resulting seek will be supported as a channel change for the HLS player, as the HLS playlist will completely change to identify chunks after the new start time position X after the HLS GET command is processed at the server side, such as by HLS playlist server 312 of Fig. 3. When the application requests a SEEK operation be performed by the server, custom SDK code must issue its own playlist or content GET so as to convey the SEEK position. If a *custom* (ie, not native or built-in) HLS player is used with this application/SDK, then the DLNA header is added to an actual playlist or HLS GET content. After receiving the HLS GET command, the server side will generate the updated playlist to reflect the seek position and the HLS player will automatically start playback from this position.

[0046] For seek operations with live services, a rolling style HLS playlist must still be provided. Since live services are normally associated with a PAUSE buffer, also called a "live off disk" buffer, a way is needed to convey the available beginning and end of that buffer. DLNA has a method defined wherein a DLNA header can include an available range request. This is a slight extension to the above SEEK note above, and returns the available SEEK range in another DLNA header field.

[0047] Fig. 5 provides a flowchart showing general operation to provide a trick play seek operation for HLS video that is adapted from DTCP. In Fig. 5, the process begins in step 504 by the HLS client SDK providing a playlist request with a DLNA RANGE REQUEST header designating a desired seek point. In step 506, the server recognizes the DLNA header of the HLS GET message and DLNA RANGE REQUEST and obtains a range of chunks making up an extent of a desired recorded video. In step 508, the server generates a new HLS playlist with the identification of the chunks and keytag corresponding to the SEEK operation. Finally, in step 510, the server provides chunks from the seek point and a rolling playlist to identify chunks and the keytag.

[0048] Although the invention has been described in conjunction with specific embodiments, many alternatives, modifications and variations will be apparent to those skilled in the art. Accordingly, the invention described is intended to embrace all such alternatives, modifications and variations that fall within the spirit and broad scope of the appended claims.

What is claimed is:

1. A method for producing HLS streaming video on a system otherwise configured for a DTCP standard, the method comprising:

receiving a request for a DLNA SEEK operation for a requested new content;

wherein the request for the DLNA SEEK operation is provided upon receipt of a HLS GET message at a HLS server, and

wherein a DLNA SEEK header includes information for the DLNA SEEK operation and is provided from the HLS server in response to the HLS GET message from a client;

creating chunks for a DTCP Protected Content Packet (PCP) equivalent structure for a range of new content needed for the DLNA SEEK operation;

generating a new HLS playlist in response to the DLNA SEEK operation with identification of the chunks and a keytag;

providing a DTCP PCP header for the playlist that is configured to include an exchange key ID and copy control information;

deriving at least one IV to be inserted in each of the keytags;

starting each chunk with an IV, allowing placement of encrypted DTCP PCP content inside the chunk; and

adding padding at each chunk boundary.

2. The method of claim 1, wherein a change is supported so that an HLS server recognizes DLNA headers in an HLS GET command to accomplish the DLNA SEEK operation in DLNA.

3. The method of claim 2, wherein the change comprises:

obtaining an extent of the recorded asset or asset portion of an in progress recording using metadata fields, the in progress recording making up the requested new content for the DLNA SEEK operation; and

determining a current time seek point position x somewhere between the beginning and the end of the asset or asset portion.

4. The method of any one of claims 1 to 3, wherein the chunks of the playlist provide a range of content needed for the HLS get message.

5. The method of claim 4, wherein the HLS server processes a DLNA RANGE REQUEST command to obtain all the chunks needed for the HLS GET message sent by a client, and wherein the chunks for the playlist are obtained based on SEEK limits determined from the DLNA RANGE REQUEST command.

6. The method of claim 5, wherein the HLS server processes the HLS GET message received from a client to generate the new playlist and the chunks corresponding to the DLNA SEEK operation.

7. The method of claim 3,

wherein play by the server will begin at the seek point, and

wherein the server provides a rolling playlist to move forward.

8. An HLS server for producing HLS streaming video on a system otherwise configured for a DTCP standard, the server comprising:

at least one processor; and

a memory connected to the processor, the memory storing code that enables the processor to receive requests for video and provide video to an HLS client device, wherein the code configures the processor to perform the following steps:

receive a request for a DLNA SEEK operation for a requested new content;

wherein the request for the DLNA SEEK operation is provided upon receipt of a HLS GET message at a HLS server, and

wherein a DLNA SEEK header includes information for the DLNA SEEK operation and is provided from the HLS server in response to the HLS GET message from a client;

create chunks for a DTCP Protected Content Packet (PCP) equivalent structure for a range of new content needed for the DLNA SEEK operation;

generate a new HLS playlist in response to the DLNA SEEK operation with identification of the chunks and a keytag;

provide a DTCP PCP header for the playlist that is configured to include an exchange key ID and copy control information;

derive at least one IV to be inserted in each of the keytags;

starting each chunk with an IV, allowing placement of encrypted DTCP PCP content inside the chunk; and

add padding at each chunk boundary,

wherein the request for the DLNA SEEK operation is provided upon receipt of a HLS GET message at the HLS server from the HLS client device, and

wherein the DLNA header provided to the HLS server is included in the HLS GET message from a client.

9. The HLS server of claim 8 wherein the code further configures the processor to perform the following additional steps:

obtain an extent of the recorded asset or asset portion of an in progress recording using metadata fields, the in progress recording making up the requested new content for the DLNA SEEK operation; and

determine a current time seek point position x somewhere between the beginning and the end of the asset or asset portion.

10. The HLS server of claim 8 or 9, wherein the chunks of the playlist provide a range of content needed for the HLS get message.

11. The HLS server of claim 10, wherein the HLS server processes a DLNA RANGE REQUEST command to obtain all the chunks needed for the HLS GET message sent by a client, and wherein the chunks for the playlist are obtained based on SEEK limits determined from the DLNA RANGE REQUEST command.

12. The HLS server of claim 11, wherein the HLS server processes the HLS GET message received from a client to generate the new playlist and the chunks corresponding to the DLNA SEEK operation.

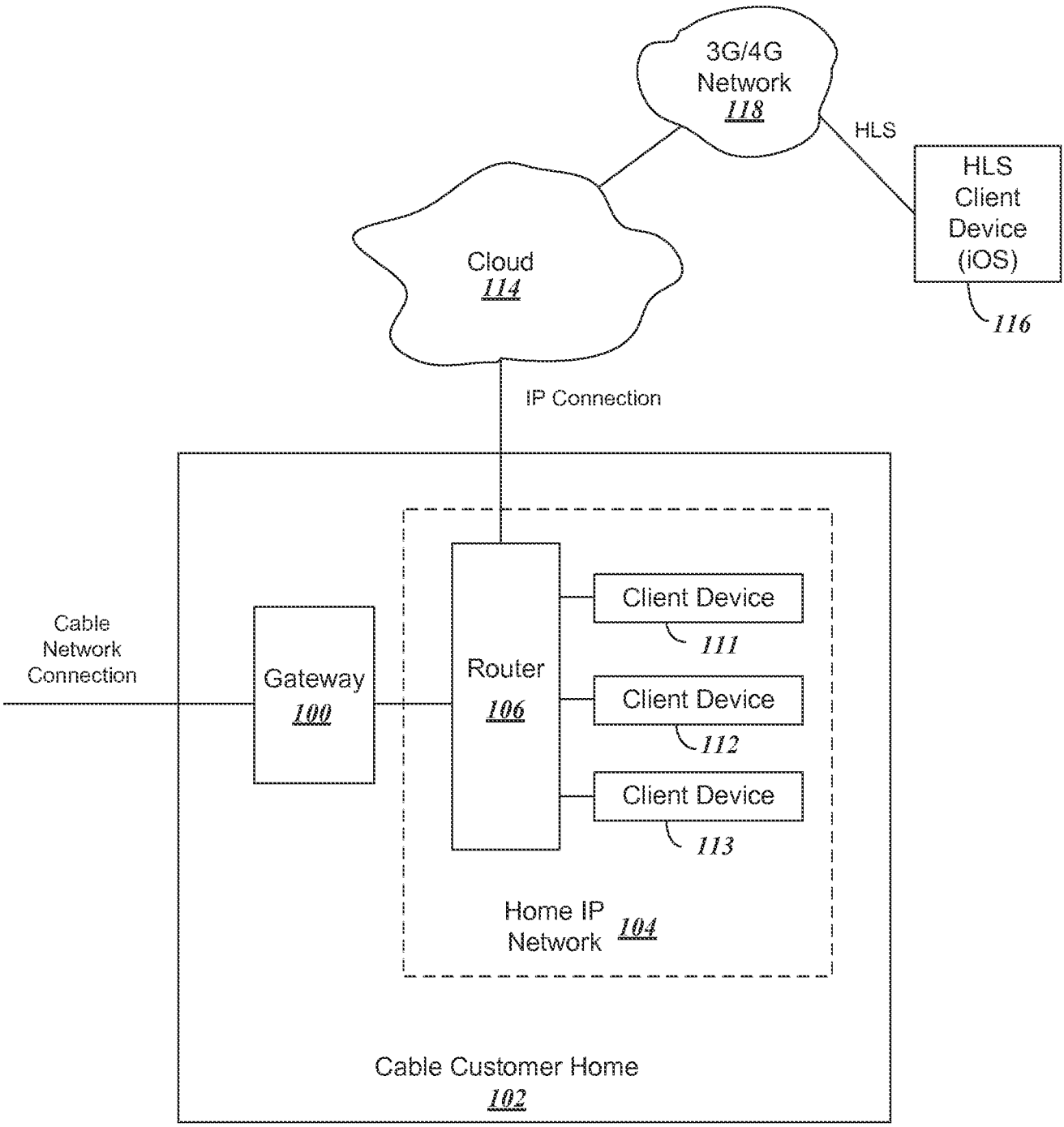

3G/4G
Network
118
HLS
HLS
Client
Device
(iOS)
116
Cloud
114
IP Connection
Cable
Network
Connection
Gateway
100
Router
106
Client Device
111
Client Device
112
Client Device
113
Home IP
Network
104
Cable Customer Home
102

FIG. 1

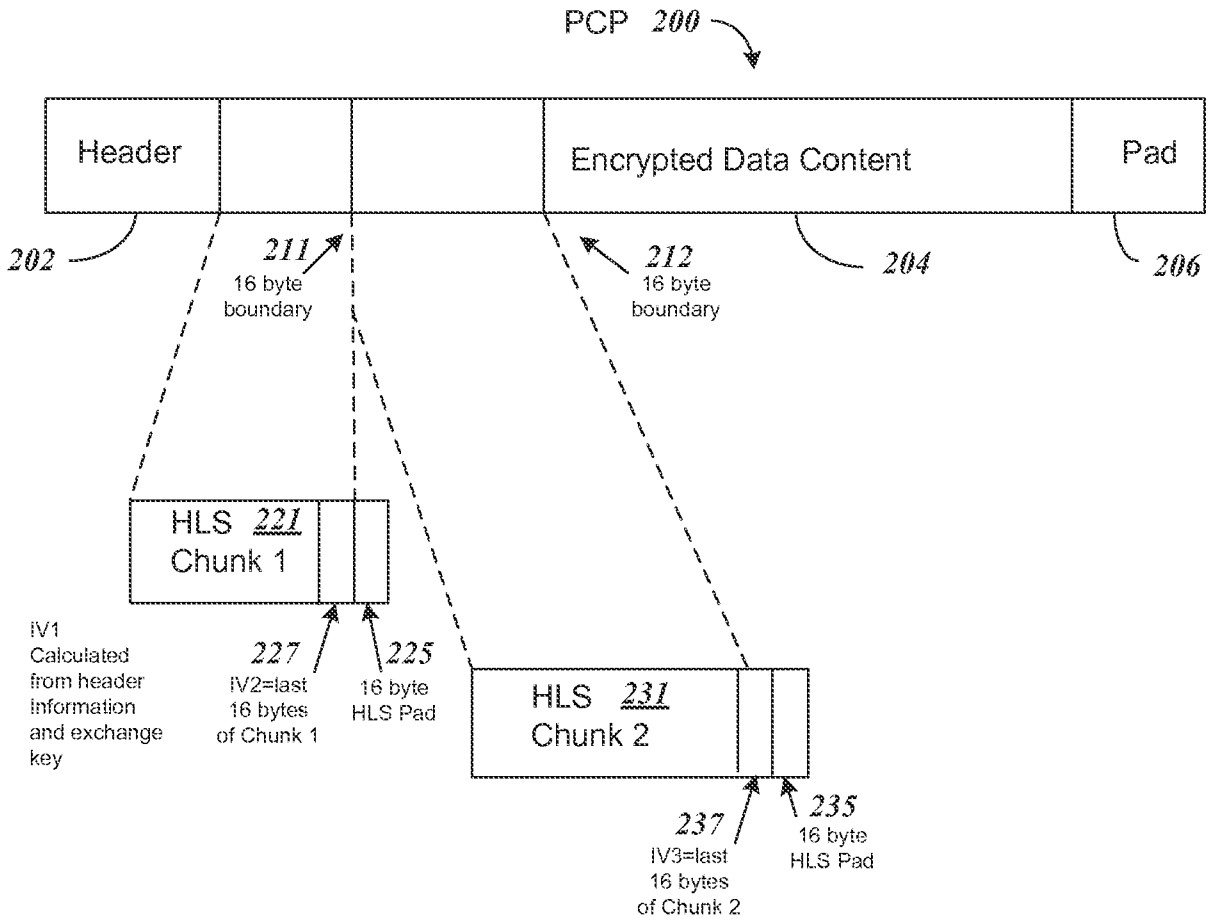

PCP 200
Header
Encrypted Data Content
Pad
202
211
16 byte boundary
212
16 byte boundary
204
206
HLS 221
Chunk 1
IV1 Calculated from header information and exchange key
227
IV2=last 16 bytes of Chunk 1
225
16 byte HLS Pad
HLS 231
Chunk 2
237
IV3=last 16 bytes of Chunk 2
235
16 byte HLS Pad

FIG. 2

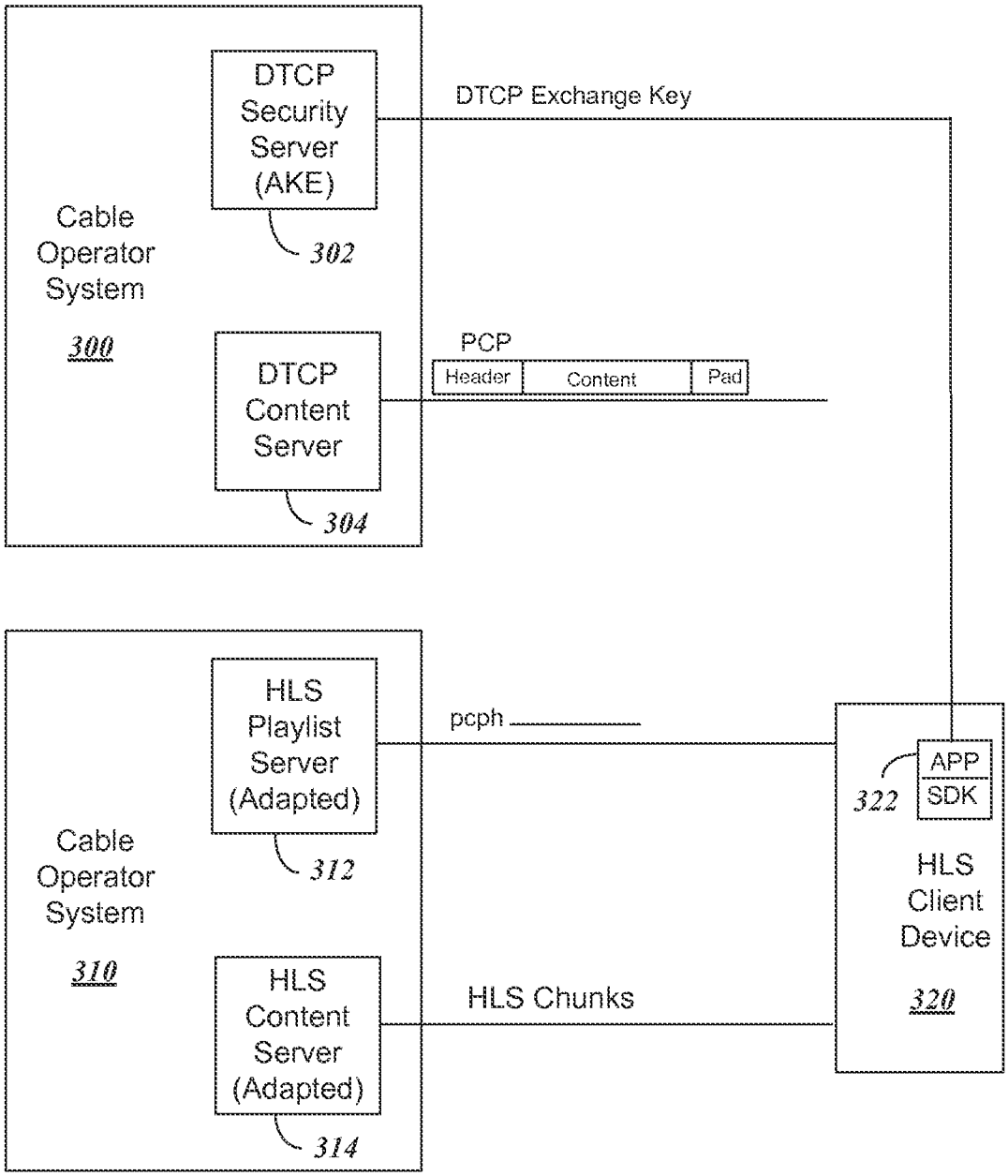

DTCP Security Server (AKE)
302
Cable Operator System
300
DTCP Exchange Key
PCP
Header
Content
Pad
DTCP Content Server
304
HLS Playlist Server (Adapted)
312
pcph
APP
SDK
322
HLS Client Device
320
Cable Operator System
310
HLS Content Server (Adapted)
314
HLS Chunks

FIG. 3

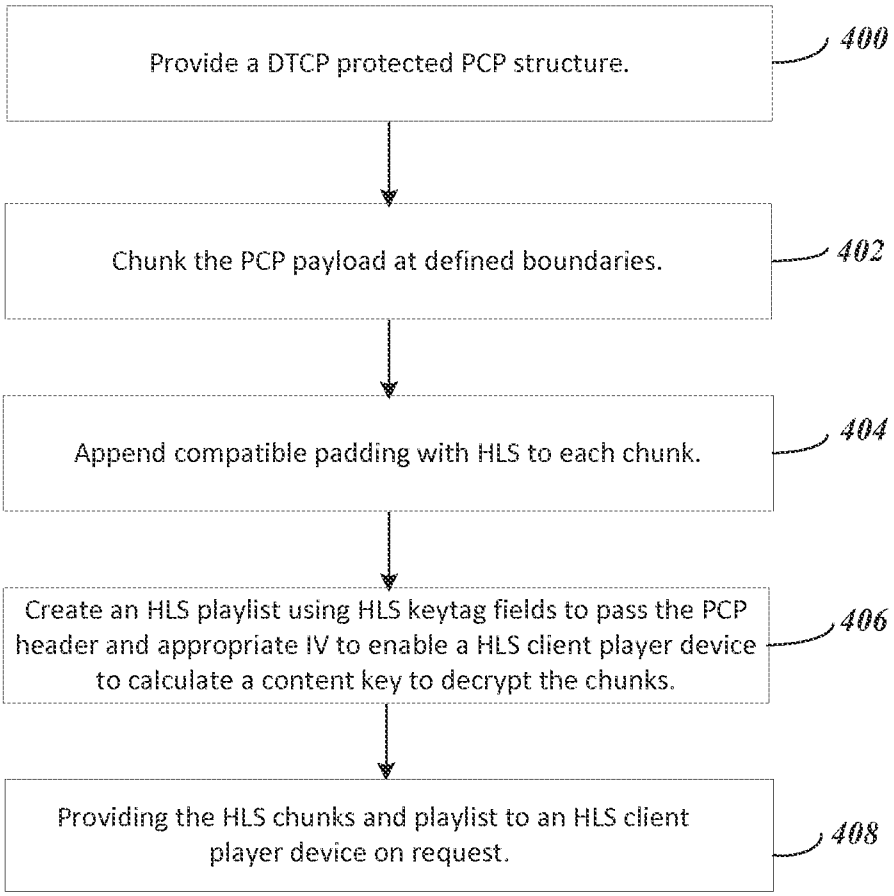

Provide a DTCP protected PCP structure.
400
Chunk the PCP payload at defined boundaries.
402
Append compatible padding with HLS to each chunk.
404
Create an HLS playlist using HLS keytag fields to pass the PCP header and appropriate IV to enable a HLS client player device to calculate a content key to decrypt the chunks.
406
Providing the HLS chunks and playlist to an HLS client player device on request.
408

FIG. 4

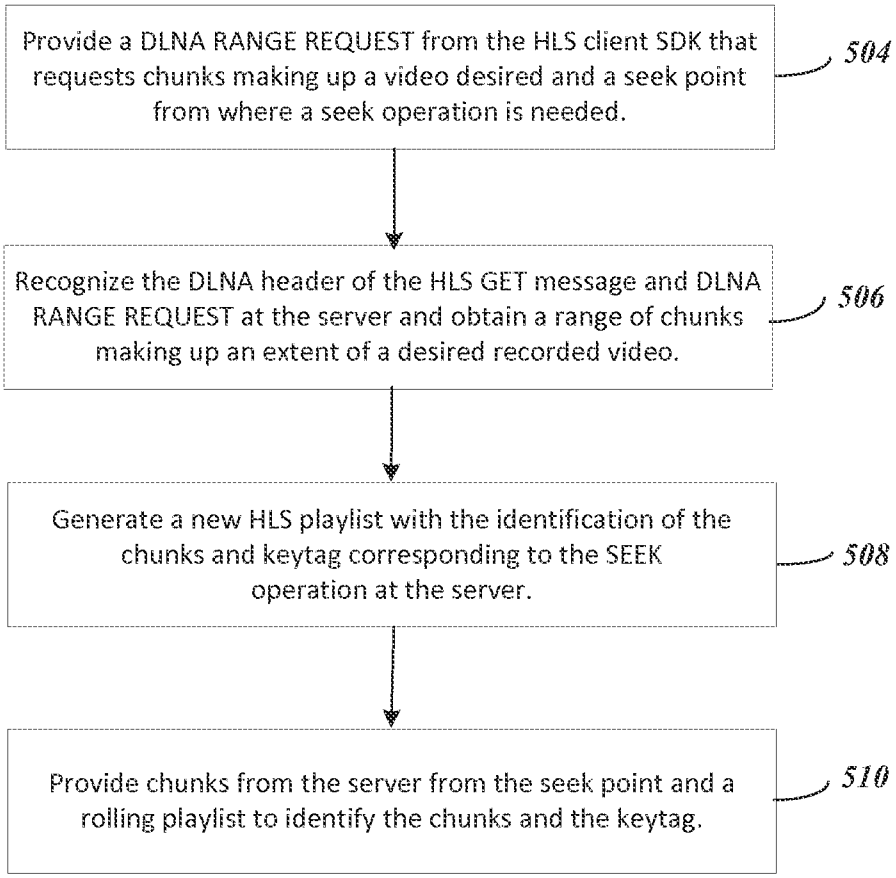
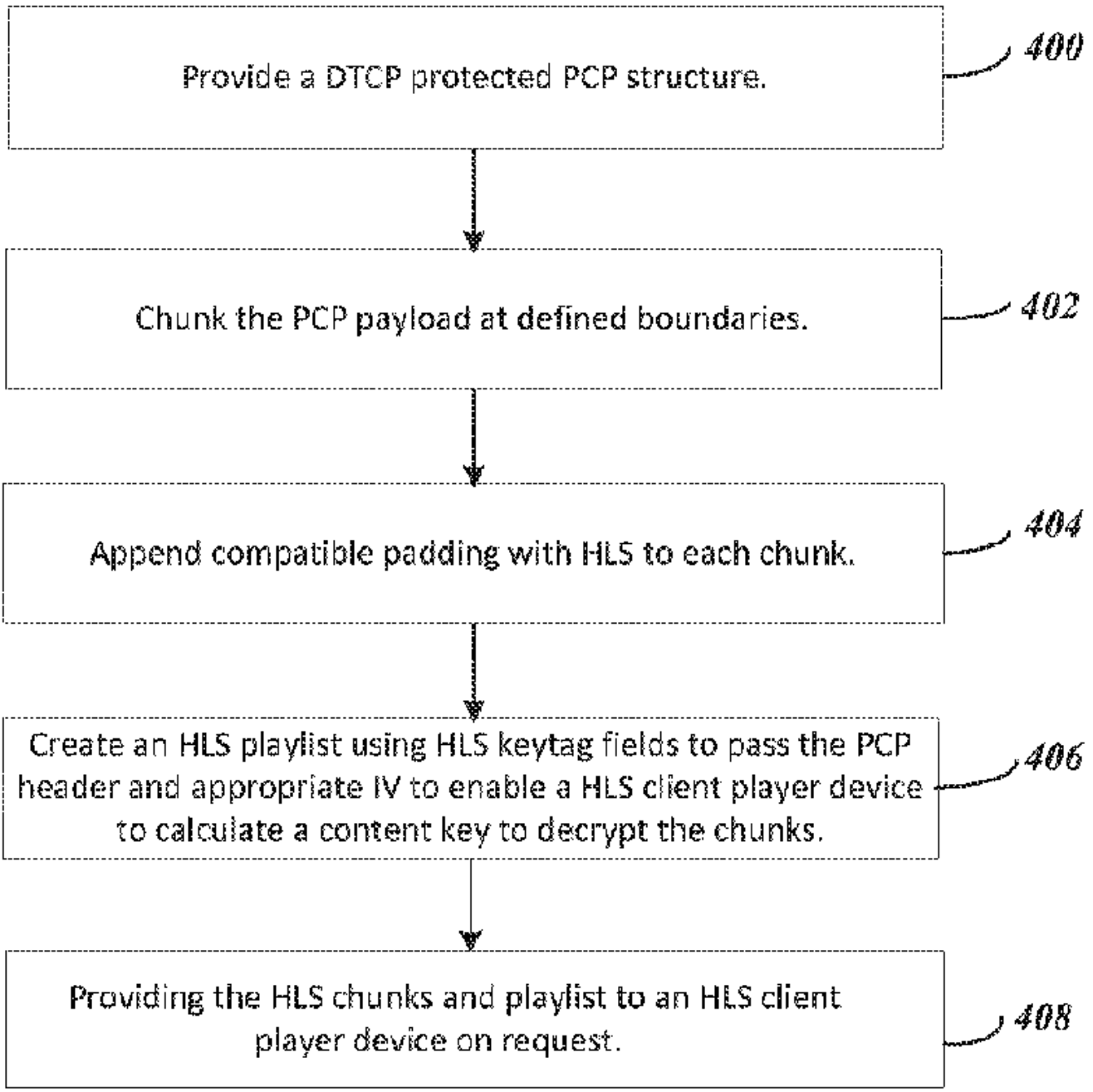

Provide a DLNA RANGE REQUEST from the HLS client SDK that requests chunks making up a video desired and a seek point from where a seek operation is needed.
504
Recognize the DLNA header of the HLS GET message and DLNA RANGE REQUEST at the server and obtain a range of chunks making up an extent of a desired recorded video.
506
Generate a new HLS playlist with the identification of the chunks and keytag corresponding to the SEEK operation at the server.
508
Provide chunks from the server from the seek point and a rolling playlist to identify the chunks and the keytag.
510

FIG. 5


Provide a DTCP protected PCP structure.
400
Chunk the PCP payload at defined boundaries.
402
Append compatible padding with HLS to each chunk.
404
Create an HLS playlist using HLS keytag fields to pass the PCP header and appropriate IV to enable a HLS client player device to calculate a content key to decrypt the chunks.
406
Providing the HLS chunks and playlist to an HLS client player device on request.
408