IIJ's DRM Initiatives

3.1 Introduction

DRM stands for digital rights management and refers to technologies used to control copyrights by, for instance, imposing restrictions on the use and duplication of digital content. DRM applies to static content such as text and images as well as digital content in general, including music and games. In this article, I will discuss DRM as it relates to video delivery from the perspective of the IIJ Media Sphere Service, a video delivery platform that I work on at IIJ.

3.2 Overview of DRM

The digitization of music, video, and other content progressed rapidly in the 90s, and as the Internet became widespread in the latter half of that decade, the way such content was distributed changed dramatically. Local video rental stores turned into DVD rental stores, and these days we have online delivery services that let us easily watch what we want, when we want, wherever we want.

While this digitization of content brought considerable lifestyle benefits, it also made it simple to copy content without any degradation in quality, such that it was easy to envision piracy taking place. The distribution of illegal content is detrimental not only to the content rights holders but also to the companies involved in production, sales, delivery, and the various other aspects of the content business. Such behavior, if allowed to run rampant, threatens to send the industry itself into decline and, ultimately, discourage the production of engaging content. The end result of this is that we the consumers would no longer be able to enjoy such content.

It was against this backdrop that DRM technology was conceived as a means of protecting not only the content itself but also the development and advancement of the industry as a whole. The hope is that DRM will ensure the appropriate use of content and impose controls over acts such as the copying of that content.

It is crucial to note that DRM technology is not perfect. By its nature, it imposes restrictions on the content consumer's playback environment. There are inevitably cases in which even well-intentioned end users are unable to enjoy content depending on their playback environment. And several cases of DRM being misused have also been reported in the past.

Yet when it comes to Internet-based content delivery, DRM technology is now widespread and stable, and it is likely to grow in importance going forward.

3.3 The Evolution of DRM Services at IIJ

You may or may not be aware of this, but the DRM technology that is used to protect content is not all that new. While writing this article, I asked a long-time IIJ employee about this. He told me that he had already begun to perceive a need for DRM in the late 90s and had, accordingly, been gathering information at events such as the exhibitions run by Streaming Media[*1], a news media outlet that deals with online video delivery.

The first real service IIJ released was DRM for Flash Video in 2008[*2]. IIJ later added support for PlayReady, and in 2015 released a service that used the open-standard Marlin DRM system[*3].

The IIJ Media Sphere Service currently supports Apple's FairPlay Streaming, Google's Widevine, and Microsoft's PlayReady DRM systems. By supporting these three DRM systems, we believe we are at present able to cover a fairly comprehensive range of video playback environments.

*1 Streaming Media (https://www.streamingmedia.com/).

*2 IIJ, "IIJ Adds DRM Functionality to its Flash Video Delivery Solution" (https://www.iij.ad.jp/news/pressrelease/2008/pdf/FlashDRM.pdf, in Japanese).

*3 IIJ, "IIJ Begins Offering IIJ DRM Service/ExpressPlay®" (https://www.iij.ad.jp/en/news/pressrelease/2015/0126.html).

By developing DRM functionality and providing it as a service within IIJ Media Sphere, IIJ has now made it easy for users to take advantage of DRM content protection without worrying about video player environments or content packaging. Not using an external DRM provider also makes it easier for IIJ to produce cost estimates. In terms of service sustainability as well, we believe that developing the system in-house and operating it within our own facilities will make it possible to deliver even greater reliability. This is because maintaining the underlying platform is key to guaranteeing an operating environment over the long term, and the same goes for the programs, including the source code. Going forward, we hope to continue to provide support and information about devices and create all sorts of value-added by implementing reporting, analysis, and other features.

Several employees at IIJ, including myself, have passed the Widevine certification program, enabling IIJ to become a Certified Widevine Implementation Partner.

3.4 DRM Features

The basic concept behind current DRM is to protect content by encrypting it and only allow it to be used in appropriate playback environments. DRM, as the name suggests, allows for a variety of specific rights management features to be used. These features are used by content providers and distributors in the form of policies that take the nature of their businesses into account.

Perhaps the most familiar such feature is video playback permissions, which make it possible to allow content playback only when your desired conditions are met. Another type of limitation that may also be familiar to you are those imposed on the number of devices that can simultaneously play back content.

You can also control content quality. For example, you can allow only audio playback, or manage the playback environment options—SD quality (480p), HD quality (1080p), and UHD quality (4k and above).

The same goes for HDCP. You may have seen the term HDCP in recent years when purchasing a device like a TV or smartphone, or in content delivery service literature. HDCP is a DRM component technology. It stands for High-Bandwidth Digital Content Protection and is a type of encryption technology developed by Intel in 2000 to protect copyright by preventing unauthorized copying. HDCP is used to encrypt digital interfaces like HDMI when transmitting video. If HDCP is not supported on both the video output device and the input device, content may not be playable or image quality may be limited. A simple use case, for example, is when you want to impose restrictions on output to analog devices. The current widespread version, HDCP 2.2, was released over a decade ago, so there is probably not too much to worry about in terms of getting it to work, but if you are trying to transmit video over something like HDMI and are having problems playing content, we recommend that you check for HDCP support on all of your devices.

Finally, let's look at device security levels. Widevine, for instance, specifies three security levels for devices: L1, L2, and L3. L1 is Widevine's most secure device level, and applies to devices that can decrypt and play videos within Trusted Execution Environment (TEE) hardware. L3 devices are those that do not have a TEE and thus perform decryption and video playback in software only. Based on these levels, it is possible to control playback so that only L1 devices are allowed to play back high-definition content, while L3 devices can only play back in low quality.

For Android smartphones, you can use an app called DRM Info to check the device's security level, so give it a try if you're interested.

3.5 How DRM Works

Below, I explain how DRM works, dealing first with the encryption process and then the decryption process.

3.5.1 Content Encryption

As I said, the basic concept behind DRM is to encrypt content. So, in general, during the video packaging process, the packaging system interacts with the DRM provider's key server when performing encryption. But there is a whole range of DRM systems out there, and thus a whole range of methods are in use. So a standard called CPIX (Content Protection Information Exchange), developed by the DASH Industry Forum (DASH-IF), has become wide-spread in the industry as a way of preventing packagers from having to deal with all these DRM systems separately. CPIX was originally developed for MPEG-DASH but now also supports HLS. The advantages of using CPIX are as follows.

1. Interoperability
   CPIX enables interoperability between different DRM systems. This allows content providers and delivery platforms to use multiple DRM providers, making it easy to deliver content across a range of devices and platforms.
2. Simplified workflow
   By adhering to CPIX, content providers and delivery platforms can use a common key and encryption format across multiple DRM systems. This makes key management and encryption procedures much simpler, resulting in more efficient workflows.
3. Enhanced security
   CPIX also offers security benefits. The use of a common encryption format and key mapping makes it easier to apply a consistent security policy to prevent content breaches and unauthorized copying.
4. Open standard
   CPIX is an open standard and has been adopted industry-wide. This helps prevent vendor lock-in[*4].

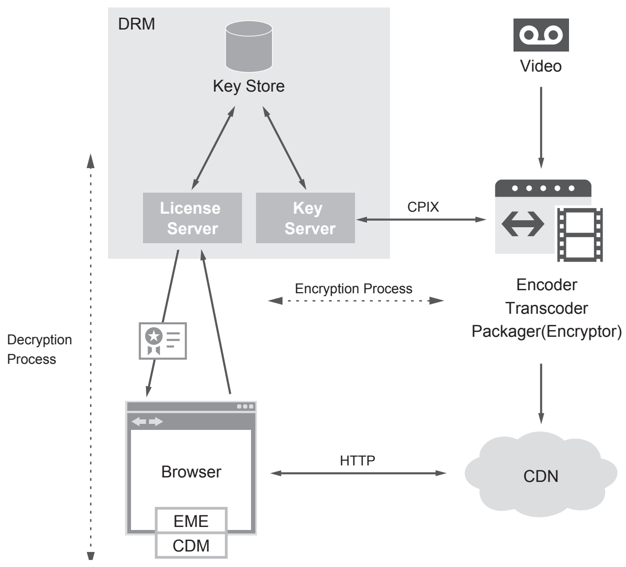


Figure 1: The Encryption and Decryption Processes

*4 DASH Industry Forum, DASH-IF Implementation Guidelines: Content Protection Information Exchange Format (https://dashif.org/docs/CPIX2.2/Cpix.pdf).

Let’s look at an example.

```
  '
    <cpix:CPIX id="cpixsample" xmlns:cpix="urn:dashif:org:cpix"
xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc">
      <cpix:ContentKeyList>
        <cpix:ContentKey kid="f269e534-c4f1-4721-9d62-26dc7ed241bd"
explicitIV="8mnlNMTx...">
        <cpix:Data>
          <pskc:Secret>
           <pskc:PlainValue>vfMB2...</pskc:PlainValue>
          </pskc:Secret>
        </cpix:Data>
      </cpix:ContentKey>
      </cpix:ContentKeyList>
      <cpix:DRMSystemList>
        <cpix:DRMSystem kid="f269e534-c4f1-4721-9d62-26dc7ed241bd"
systemId="edef8ba9-79d6-4ace-a3c8-27dcd51d21ed">
          <cpix:PSSH>AAAAP3Bzc...</cpix:PSSH>
        </cpix:DRMSystem>
        <cpix:DRMSystem kid="f269e534-c4f1-4721-9d62-26dc7ed241bd"
systemId="9a04f079-9840-4286-ab92-e65be0885f95">
          <cpix:PSSH>AAADBnBzc...</cpix:PSSH>

<cpix:ContentProtectionData>PG1zcHI6cHJvIHhtbG...</cpix:ContentProt
ectionData>
        </cpix:DRMSystem>
        <cpix:DRMSystem kid="f269e534-c4f1-4721-9d62-26dc7ed241bd"
systemId="94ce86fb-07ff-4f43-adb8-93d2fa968ca2">
          <cpix:URIExtXKey>c2tkOi...</cpix:URIExtXKey>
        </cpix:DRMSystem>
      </cpix:DRMSystemList>
    </cpix:CPIX>
  `
```

CPIX is written in XML. I will now go over a number of key points using the example above.

The ContentKey element represents the information required for content encryption. This is based on an extends RFC 6030 Portable Symmetric Key Container (PSKC)[*5].

The DRMSystem elements represent information specific to each DRM system. The systemId strings are IDs that identify each DRM system and are determined by DASH-IF[*6].

Intheexampleabove,edef8ba9-79d6-4ace-a3c8-27dcd51d21ed referstoWidevine,9a04f079-9840-4286-ab92-e65be0885f95 refers to Microsoft PlayReady, and 94ce86fb-07ff-4f43-adb8-93d2fa968ca2 refers to Apple FairPlay.

The PSSH element represents data used in the MP4 PSSH (Protection System Specific Header) box, a type of video container. This is a type of MP4 box used to store information related to digital content encryption and DRM systems. The PSSH box contains the encryption key, the encryption method, and other information about the DRM system.

The URIExtXKey element, as you can probably tell from the fact that the relevant DRM system is used by Apple

*5 RFC Editor, RFC 6030 Portable Symmetric Key Container (PSKC), October 2010 (https://www.rfc-editor.org/info/rfc6030).
*6 DASH Industry Forum, Content Protection (https://dashif.org/identifiers/content_protection/).

FairPlay, is an item that affects the EXT-X-KEY used in HLS playlists.

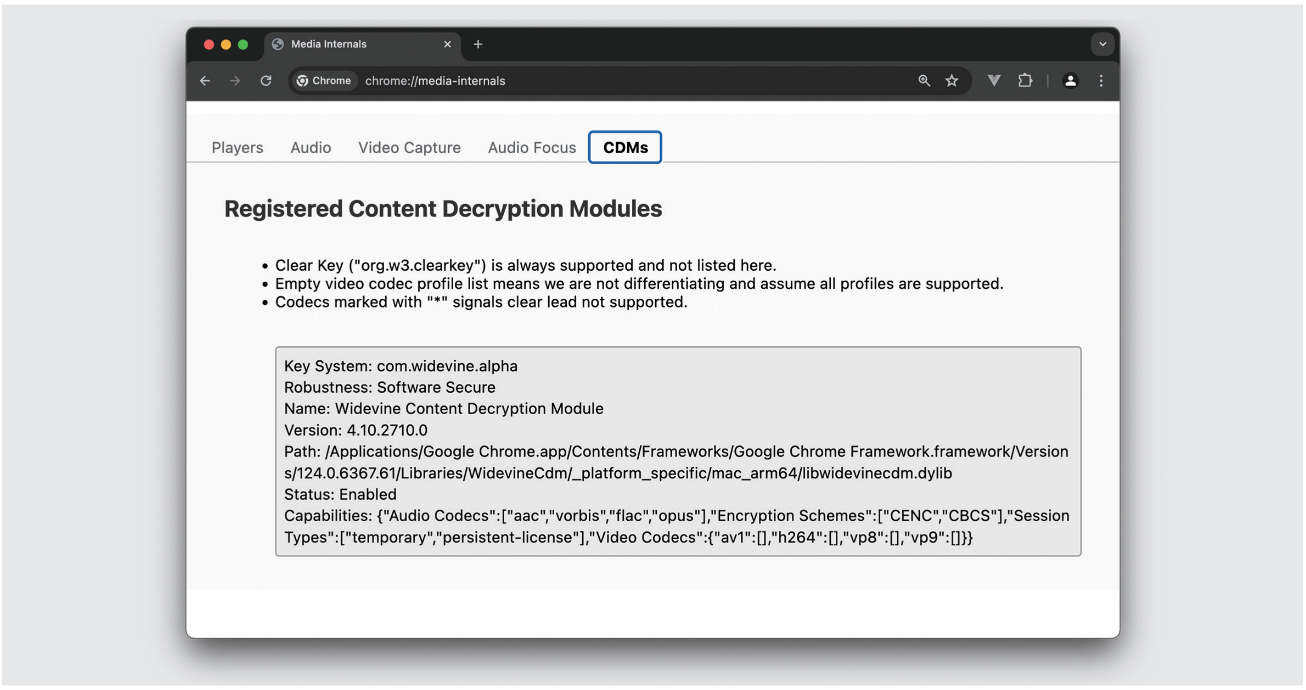
This information is used during the content packaging process to encrypt the content.

The SPEKE (Secure Packager and Encoder Key Exchange) standard, which extends CPIX and is available on AWS, is also very widespread. This is also an open specification, so anyone can figure out how it works[*7].

3.5.2 Content Decryption

So at this point, our content has been encrypted. We are probably seldom aware of the existence of DRM when casually watching videos in a browser or the like. Yet, some complex processing needs to take place for us to play encrypted content. This section gives a simple explanation of what happens.

A W3C specification called EME (Encrypted Media Extensions), which provides a communication channel



Registered Content Decryption Modules

- Clear Key ("org.w3.clearkey") is always supported and not listed here.
- Empty video codec profile list means we are not differentiating and assume all profiles are supported.
- Codecs marked with "*" signals clear lead not supported.

```
Key System: com.widevine.alpha
Robustness: Software Secure
Name: Widevine Content Decryption Module
Version: 4.10.2710.0
Path: /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/Version
s/124.0.6367.61/Libraries/WidevineCdm/_platform_specific/mac_arm64/libwidevinecdm.dylib
Status: Enabled
Capabilities: {"Audio Codecs":["aac","vorbis","flac","opus"],"Encryption Schemes":["CENC","CBCS"],"Session
Types":["temporary","persistent-license"],"Video Codecs":{"av1":[],"h264":[],"vp8":[],"vp9":[]}}
```

Figure 2: Chrome's media-internals

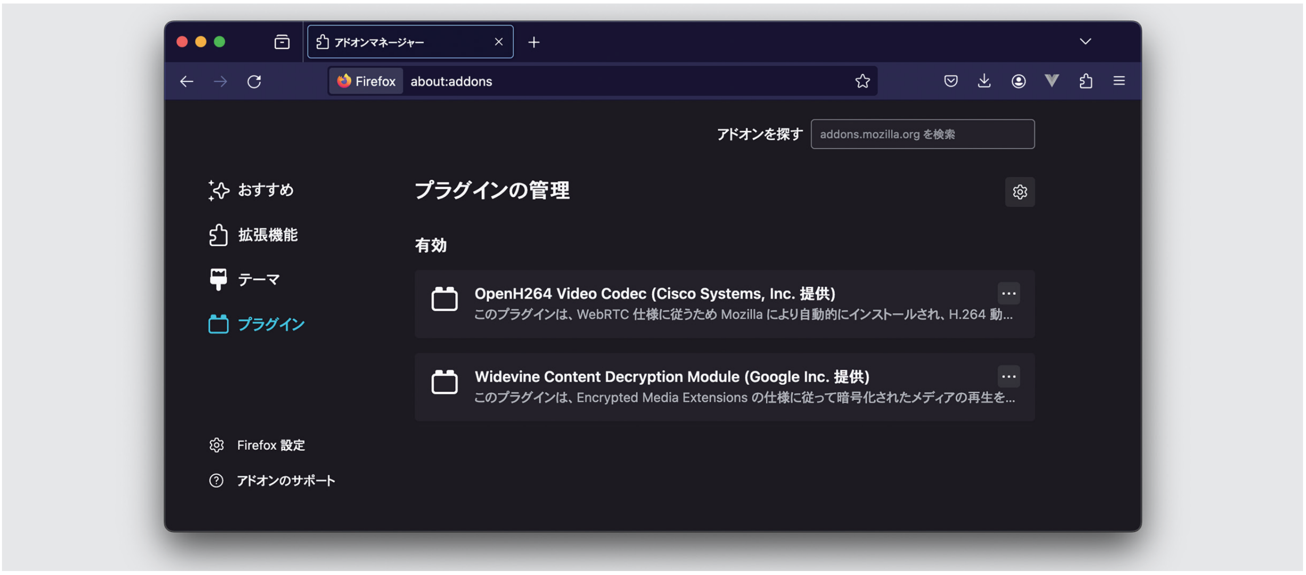
*7 AWS, SPEKE API specification (https://docs.aws.amazon.com/speke/latest/documentation/speke-api-specification.html).

between web browsers and DRM software, and the CDMs (Content Decryption Modules) provided by DRM vendors in accord with that specification, are key to decrypting the content.

CDMs are closed source, and Widevine, for example, provides a CDM in the form of a plugin for Google Chrome as well as for Firefox (Figures 2 and 3). Apple FairPlay's CDM is only available for Apple products, such as Safari, so it cannot be used on Google Chrome and other browsers.

When the EME specification was being developed, CDMs were the subject of strong opposition from supporters of an open web, but in the end, the specification was formulated in such a way that the CDMs would only be responsible for confidentiality and integrity for the purposes of decryption, while the video players and other such applications would be in charge of the data communications. If EME and CDM had not been formulated in this way, DRM providers would not have been able to take a consistent approach, and content delivery services may all have had to use their own specific applications.

In Google Chrome, you can view detailed information on the CDMs by typing "chrome://media-internals/" into the address bar, so take a look if you're interested.



OpenH264 Video Codec (Cisco Systems, Inc. 提供)
このプラグインは、WebRTC 仕様に従うため Mozilla により自動的にインストールされ、H.264 動...

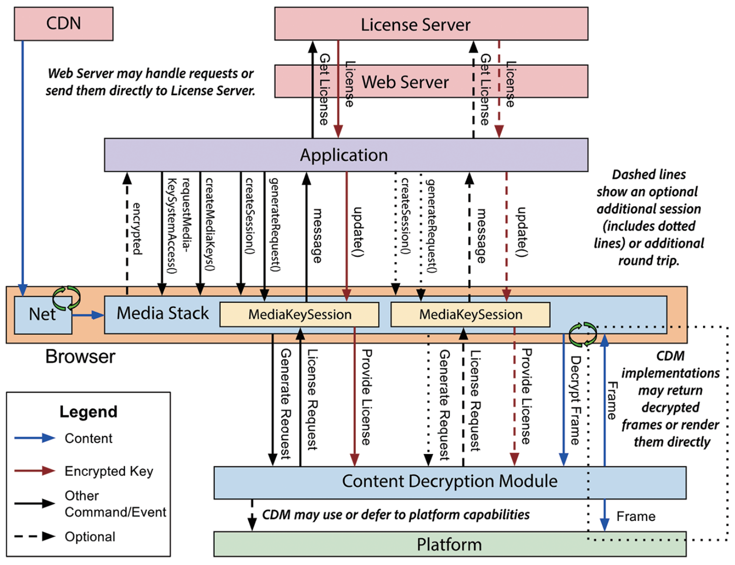
Widevine Content Decryption Module (Google Inc. 提供)
このプラグインは、Encrypted Media Extensions の仕様に従って暗号化されたメディアの再生を...

Figure 3: Firefox plugins

Figure 4 shows the decryption flow when using EME and CDM. Video players are only able to play content after communicating with a CDM using EME. This involves the following steps.

1. The browser fires an encrypted event, which tells the video player that the content is encrypted, and the video player calls the requestMediaKeySystemAccess() method to use a specific CDM.
2. The createMediaKeys() method initializes the keys.
3. The createSession() method creates a session to control key expiration time.
4. The generateRequest() method tells the CDM to generate a license request. The license is the most important piece of data, containing the decryption key and information needed for using DRM features.
5. The CDM generates and returns a license request. The video player learns about the license request via a message event.
6. The video player sends the license request to the license server.
7. The video player receives the license returned by the license server.
8. The returned license is passed to the CDM using the update() method.

The CDM assesses the returned license based on the DRM policy and decrypts the content, and the video is thus played.



CDN
License Server
Web Server may handle requests or send them directly to License Server.
Web Server
Get License
License
Application
encrypted
requestMedia-KeySystemAccess()
createMediaKeys()
createSession()
generateRequest()
message
update()
Dashed lines show an optional additional session (includes dotted lines) or additional round trip.
Net
Media Stack
MediaKeySession
MediaKeySession
Browser
Generate Request
License Request
Provide License
Decrypt Frame
Frame
CDM implementations may return decrypted frames or render them directly
Content Decryption Module
CDM may use or defer to platform capabilities
Platform
Legend
Content
Encrypted Key
Other Command/Event
Optional

Source: W3C, Encrypted Media Extensions (https://www.w3.org/TR/encrypted-media/).

Figure 4: Decryption Flow with EME and CDM

3.6 Conclusion

In this article, I have discussed DRM as it relates to video delivery, with a look at the development of the IIJ Media Sphere Service. These days, no small number of people enjoy video content via content delivery services. I have endeavored to explain how DRM technology is actually used behind the scenes in this space and roughly how DRM works. While I have been somewhat constrained by space limitations, I thank you for reading, and I hope you found this informative.

Mitsuo Kuroishi
Deputy Manager, Delivery Development Section, Content Delivery Services Department, Network Division, IIJ
Mr. Kuroishi has worked on the development of a range of services since joining IIJ in 2002. His favorite saying is “Code speaks louder than words,” and he looks forward to Emacs updates.