Rapport du projet d'Internet Nouvelle Génération
dans le cadre du Master 2 Cybersécurité et e-Santé

Mise en œuvre de la diffusion du contenu protégé DRM avec DASH

Année universitaire 2021 – 2022

Présenté par

GUARDIA Quentin BONNET Ludivine ISSELNANE Hacene

{prénom}.{nom}@etu.u-paris.fr

Sous la direction de MEHAOUA Ahmed et NGUYEN Antoine

Table des matières

Introduction

Le streaming, ou diffusion en continu, prend une importance croissante sur internet. C'est une méthode qui permet à un client de lire une vidéo directement sur un serveur, sans avoir à la télécharger préalablement. Parmi les méthodes de diffusion existantes, on peut notamment citer HTTP Live Streaming (HLS) et Dynamic Adaptive Streaming over HTTP, aussi appelée MPEG-DASH ou DASH.

Cette dernière est la seule solution de streaming certifiée ISO et a été adoptée comme standard international. Pour décrire brièvement son fonctionnement, DASH coupe les fichiers vidéos en segments courts, qui fournissent usuellement des parties de vidéos qui durent entre deux et quatre secondes. Ces segments sont en général transmis grâce au protocole HTTP. DASH bénéficie d'un fichier manifeste contenant les métadonnées du flux, ainsi que des liens vers les segments.

Une œuvre vidéo peut être soumise à des protections numériques. On parle de Gestion de Droits Numériques, ou Digital Rights Management (DRM) en anglais. Les technologies DRM visent à contrôler l'utilisation, la modification et la diffusion d'œuvres protégées. Elles agissent directement sur le support numérique physique, ou bien sur le support de transmission, comme MPEG-DASH. Pour protéger une œuvre, DRM couple le chiffrement de l'œuvre à un accès conditionnel. Par exemple, l'accès à une œuvre peut être autorisé uniquement s'il y a une preuve d'achat ou de souscription.

On peut alors poser la problématique suivante : comment MPEG-DASH peut-il diffuser du contenu protégé avec DRM ?

Pour répondre à cela, nous étudierons d'abord le fonctionnement de diffusion DASH, puis les différentes protections DRM. Enfin, nous nous pencherons sur la diffusion de contenu DRM avec MPEG-DASH, avec des exemples pratiques.

Diffusion MPEG DASH

DASH est une technologie de streaming à débit binaire adaptatif. Comme expliqué précédemment, elle divise un fichier multimédia en segments, qui sont délivrés à un client via HTTP. Le fichier manifeste XML Media Presentation Description (MPD) décrit les informations du segment : URL, timing, caractéristiques du média comme la résolution vidéo et les débits binaires, et cætera. Les segments peuvent contenir n'importe quelle donnée multimédia. À savoir que le fichier manifeste fournit des guides et formats à utiliser avec deux types de conteneurs : le format de fichier média de base ISO (par exemple, le format de fichier MP4) et le flux de transport MPEG-2.

DASH fonctionne avec plusieurs codecs audio et vidéo. Chaque fichier multimédia a généralement plusieurs représentations possibles, c’est-à-dire des versions à différentes résolutions ou débits binaires. La sélection de cette version peut être effectuée en fonction des conditions du réseau, des capacités du dispositif et des préférences de l'utilisateur. Ce qui permet un streaming à débit binaire adaptatif et une équité en matière de qualité d'expérience (QoE). Autrement dit, la qualité des multimédias est adaptée au maximum de sorte à éviter les coupures. Cependant il n’y a pas de normalisation quant aux algorithmes adaptatifs, ce qui laisse une certaine liberté aux fournisseurs.

MPEG-DASH s’est beaucoup popularisé durant la dernière décennie. La technologie est nativement installée sur Android via le lecteur ExoPlayer, sur les Smart TV Samsung, LG et Sony dès 2012, Philips NetTV, Panasonic Viera et Chromecast, entre autres. YouTube ainsi que Netflix prennent déjà en charge MPEG-DASH, et différents lecteurs MPEG-DASH sont disponibles.

Protection DRM

Les outils de gestion des droits numériques, plus communément « digital rights management » (DRM) en anglais, sont un ensemble de technologies de contrôle d'accès permettant de restreindre l'utilisation de matériel propriétaire et d'œuvres protégées par des droits d'auteur. Les technologies DRM tentent de contrôler l'utilisation, la modification et la distribution d'œuvres protégées par le droit d'auteur, comme des contenus multimédias, des logiciels, des DVD, des Blu-rays ou même certains systèmes au sein des appareils.

Ainsi, DRM permet de soumettre des contenus à un accès conditionnel. Ce qui peut par exemple restreindre l'accès à un support en fonction de la zone géographique ou du matériel de lecture. Il est également possible de restreindre des fonctionnalités plus précises, comme la copie du contenu ou l'accélération du contenu dans le but d'éviter des publicités. En réalité, beaucoup de cas de figure peuvent justifier une restriction.

Dans le monde entier, de nombreuses lois récentes pénalisent le contournement des DRM, la création et la distribution d'outils utilisés pour ce contournement et la communication sur ce contournement. Ces lois font partie du Digital Millennium Copyright Act aux États-Unis et ou de lois issues de l'EUCD dans l'Union européenne. Par exemple en France, on peut citer la DADVSI.

Diffusion de contenu DRM avec MPEG DASH

Exemple théorique

Nous avons présenté les technologies MPEG-DASH et DRM. Pour mieux nous approprier le sujet, nous allons à présent diffuser du contenu MPEG-DASH, qui sera contrôlé par DRM.

Pour cela, nous allons segmenter puis chiffrer une vidéo MP4. Le chiffrement se fait par une clé associée à une licence. La vidéo doit alors être alors diffusée par un serveur web sécurisé. Nous utiliserons le localhost. On fournit au lecteur l'accès à la clé de chiffrement via un fichier manifeste MPD et l'accès à la licence. Sans ces accès, la vidéo ne peut être lue.

D'un point de vue plus technique, la vidéo est chiffrée et conditionnée sur un serveur de diffusion, qui partage la clé de chiffrement et sa clé d'identification (KeyID) avec le serveur de la licence. Le client demande les segments de la vidéo en fonction du fichier manifeste, au réseau de diffusion de contenu (CDN). Il envoie pour cela la KeyID dont il bénéficie grâce au fichier manifeste également. Le CDN demande en conséquent la clé de chiffrement et la licence. Le client fait alors la requête de ces deux éléments au serveur de la licence. Il a pour cela l'URL du serveur de la licence dans le script de la page HTML. Le serveur de la licence lui envoie en réponse les informations nécessaires. Le client les retransmet au CDN, qui peut dès lors diffuser le contenu chiffré.

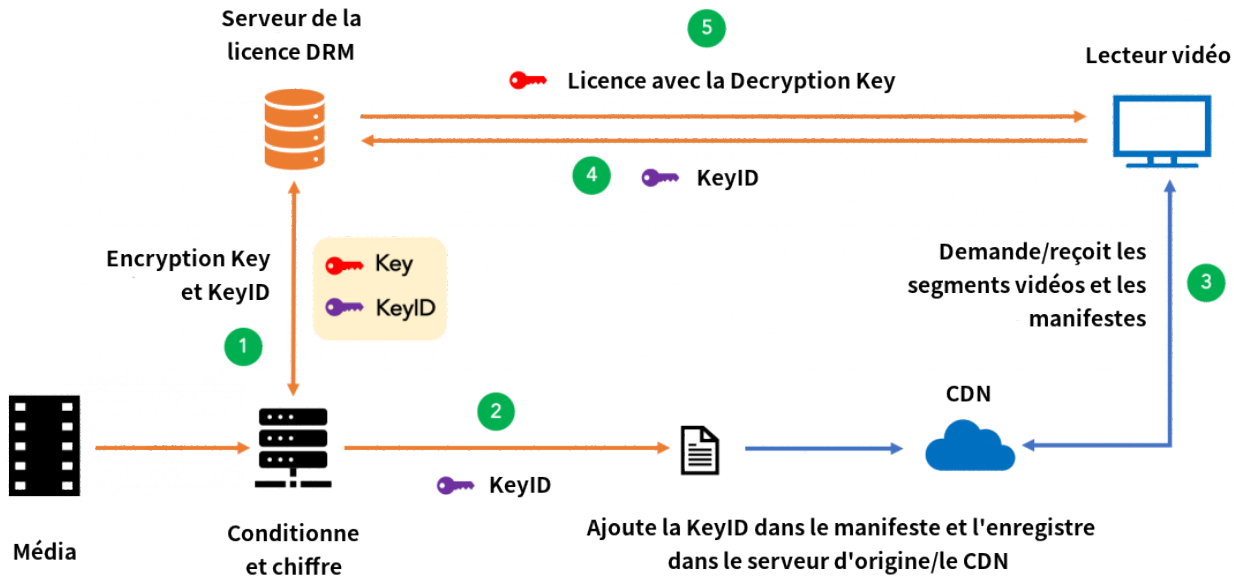

Figure 1: Obtention de la vidéo protégée par une licence

Donc le lecteur décode au fur et à mesure les segments qu'il reçoit, avec la clé de chiffrement. Cela permet à l'utilisateur de bénéficier de la vidéo en streaming grâce à la possession de la licence.

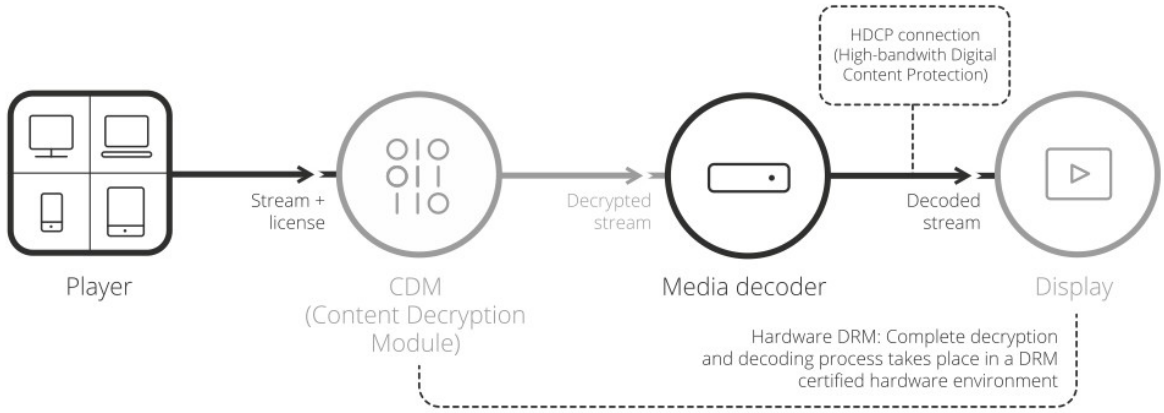

Figure 2: Déchiffrement et lecture de la vidéo par le lecteur, source : www.unified-streaming.com

Exemple pratique

Licence et clés

Voici plus concrètement comment nous allons procéder à la diffusion protégée. La première étape consiste à obtenir l'URL d'une licence, la clé de chiffrement associée et la KeyID. Normalement, ce genre de service est proposé de manière payante. Par chance, nous avons trouvé un serveur proposant ces services. Il s'agit d'un serveur Widevine, proposé par Google. Ainsi, nous avons ces informations :

- l'URL vers la licence : https://drm-widevine-licensing.axtest.net/AcquireLicense
- La clé de chiffrement : 166634c675823c235a4a9446fad52e4d
- Et la KeyID : 9eb4050de44b4802932e27d75083e266

Vidéo mp4

Les étapes qui suivent ont été testées sur Linux. Pour la deuxième étape, on télécharge une vidéo MP4, dans le nouveau dossier ~/videos

```
mkdir -p ~/videos
```

```
cd ~/videos
```

```
wget http://shengbin-static.stor.sinaapp.com/bmw.mp4 -O in.mp4
```

Outil de fragmentation et de chiffrement

On télécharge ensuite l'outil Bento4 à partir de Github. Il donne la possibilité de fragmenter et chiffrer des vidéos, autrement dit de la conditionner. Voici la suite de commande à exécuter :

```
git clone https://github.com/axiomatic-systems/Bento4.git
```

```
cd Bento4
```

```
mkdir cmakebuild
```

```
cd cmakebuild
```

```
cmake -DCMAKE_BUILD_TYPE=Release ..
```

```
make
```

On crée les alias des exécutables qui nous intéressent, soit mp4dash qui chiffre les vidéos et mp4fragment qui les segmente. On édite le fichier ~./bashrc à cette fin.

```
nano ~/.bashrc
```

Plus précisément, on y concatène les lignes suivantes:

alias mp4dash=$HOME/dash-drm/Bento4/Source/Python/wrappers/mp4dash
alias mp4fragment=$HOME/dash-drm/Bento4/cmakebuild/mp4fragment

Si cela ne marche pas, il faut remplacer $HOME par le chemin absolu. Il est ensuite nécessaire de recharger le fichier bashrc de la sorte :

```
source ~/.bashrc
```

Fragmentation et chiffrement

Dans le dossier de la vidéo in.mp4, on peut segmenter cette dernière avec la commande suivante :

```
mp4fragment in.mp4 in-fragment.mp4
```

Puis, l'on chiffre le fichier obtenu avec la clé widevine que l'on possède.

```
mp4dash --widevine-header provider:widevine_test --encryption-key
9eb4050de44b4802932e27d75083e266:166634c675823c235a4a9446fad52e4d in-
fragment.mp4
```

On retrouve le contenu chiffré et le fichier manifeste dans le dossier videos/output. La vidéo est prête à être exploitée pour la diffusion

Configuration du lecteur

On prépare le lecteur. Pour cela, on créé le fichier index.html sans changer de dossier :

```
<!doctype html>
<html>
<head>
<title>DASH</title>
<script src="https://cdn.dashjs.org/latest/dash.all.min.js"></script>
<script>
    function init() {
        var protData = {
            "com.widevine.alpha": {
                "serverURL":
"https://drm-widevine-licensing.axtest.net/AcquireLicense",
                "httpRequestHeaders": {
                    "X-AxDRM-Message":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ2ZXJzaW9uIjoxLCJjb21fa2V5X2lkIjoiYjMzNj
RlYjUtNTFmNi00YWUzLThjOTgtMzNjZWQ1ZTMxYzc4IiwibWVzc2FnZSI6eyJ0eXBlIjoiZW50aXRsZW
1lbnRfbWVzc2FnZSIsImZpcnN0X3BsYXlfZXhwaXJhdGlvbiI6NjAsInBsYXlyZWFkeSI6eyJyZWFsX3
RpbWVfZXhwaXJhdGlvbiI6dHJ1ZX0sImtleXMiOlt7ImlkIjoiOWViNDA1MGQtZTQ0Yi00ODAyLTkzMm
UtMjdkNzUwODNlMjY2IiwiZW5jcnlwdGVkX2tleSI6ImxLM09qSExZVzI0Y3Iya3RSNzRmbnc9PSJ9XX
19.FAbIiPxX8BHi9RwfzD7Yn-wugU19ghrkBFKsaCPrZmU"
                },
                priority: 0
            }
        };
        var video,
            player,
            url = "videos/output/stream.mpd";

        video = document.querySelector("video");
        player = dashjs.MediaPlayer().create();
        player.initialize(video, url, true);
        player.setProtectionData(protData);
    }

    function check() {
        if (location.protocol === 'http:' && location.hostname !== 'localhost')
{
            var out = 'This page has been loaded under http. This might result
in the EME APIs not being available to the player and any DRM-protected content
will fail to play. ' +
                'If you wish to test manifest URLs that require EME support,
then <a href=\'https:' +
window.location.href.substring(window.location.protocol.length) + '\'>reload
this page under https</a>.'
            var div = document.getElementById('http-warning');
            div.innerHTML = out;
            div.style.display = ''
        }
    }
</script>
</head>
<body onload="init()">

        <div><video data-dashjs-player autoplay controls></video></div>
</body>
</html>
```

Plusieurs choses importantes sont à noter. D'abord, on inclut la bibliothèque dash.js pour avoir un lecteur MPEG-DASH dans le body. Ensuite, l'ouverture de la page lance une fonction définissant notamment l'URL de la licence et le chemin vers le fichier manifeste. Ainsi, le code ci-dessus, avec les autres paramètres de configuration, est suffisant pour diffuser la vidéo chiffrée. Il ne manque plus qu'à mettre en place le serveur.

Serveur de diffusion

Nous allons créer un serveur web local avec python. Il faut noter que la technologie DRM nécessite une connexion sécurisée HTTPS et non HTTP. Dans ce but, nous signons notre propre certificat :

```
openssl req -new -x509 -keyout localhost.pem -out localhost.pem -days 365 -nodes
```

Puis, en se basant sur ce certificat, nous allons tirer profit du code suivant, pour lancer le serveur, en l'enregistrant sous « serveur.py » dans le même dossier.

```
import http.server, ssl
server_address = ('localhost', 4443)
httpd = http.server.HTTPServer(server_address,
http.server.SimpleHTTPRequestHandler)
httpd.socket = ssl.wrap_socket(httpd.socket,
      server_side=True,
      certfile='localhost.pem',
      ssl_version=ssl.PROTOCOL_TLS)
httpd.serve_forever()
```

On lance alors le serveur avec cette commande :

```
python3 serveur;py
```

Résultat

On peut observer le résultat avec l'URL https://0.0.0.0:4443 .

Le navigateur confirme la protection DRM, et la diffusion MPEG-DASH fonctionne correctement.

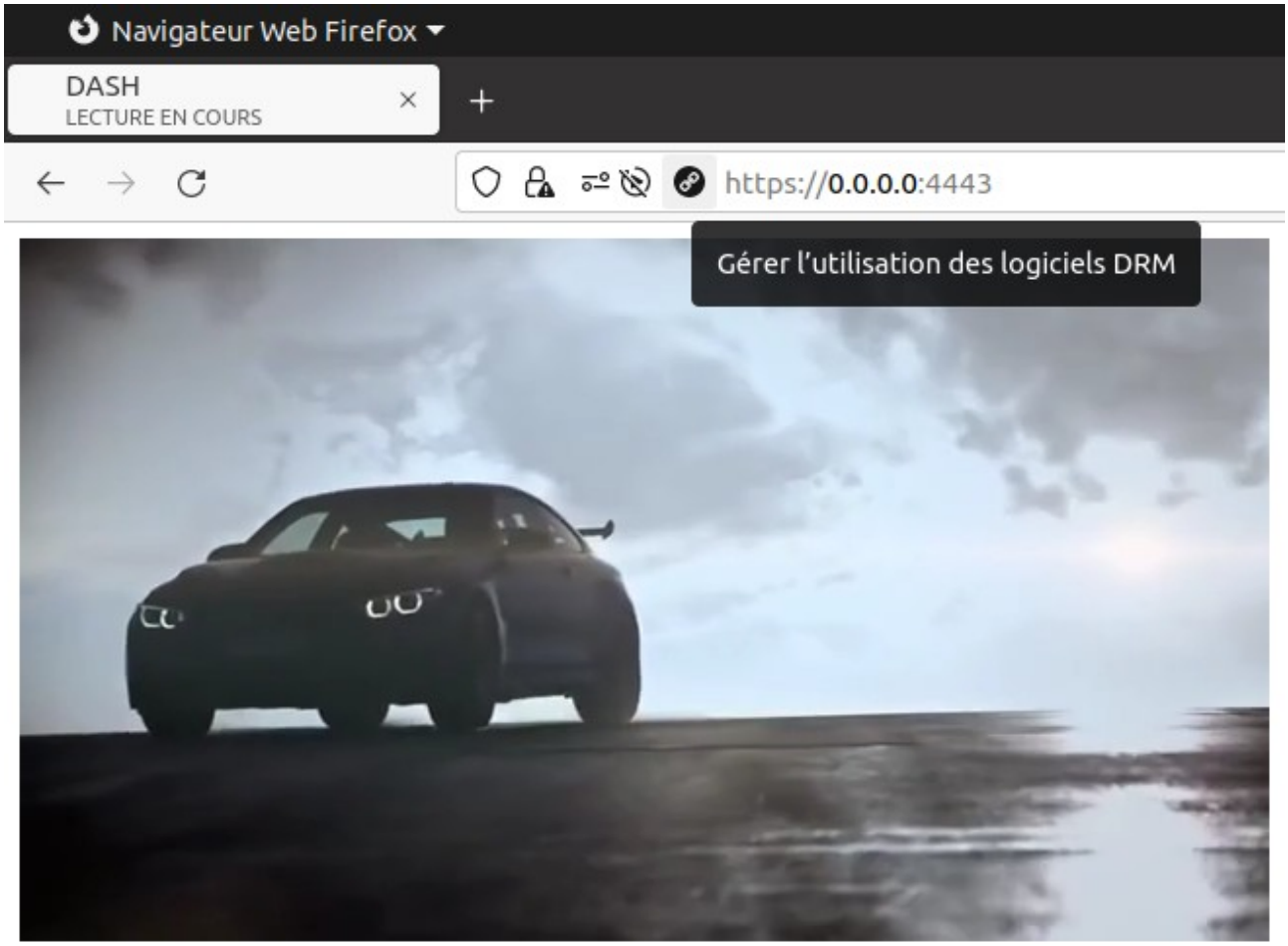

Figure 3: Capture d'écran du navigateur avec la diffusion soumise à des DRM

On peut observer les requêtes GET des segments chiffrés à partir du terminal qui nous a servi à lancer le serveur web :

```
quentin@quentin-msi:~/dash-drm$ python3 serveur.py
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/stream.mpd HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/audio/en/mp4a.40.2/init.mp4 HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/video/avc1/init.mp4 HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/video/avc1/seg-1.m4s HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/audio/en/mp4a.40.2/seg-1.m4s HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/audio/en/mp4a.40.2/seg-2.m4s HTTP/1.1" 304 -
127.0.0.1 - - [01/Mar/2022 10:13:29] "GET /videos/output/video/avc1/seg-2.m4s HTTP/1.1" 304 -
```

Figure 4: Journaux des requêtes sur le serveur web local

Conclusion

Pour conclure, comme tout contenu, le streaming MPEG-DASH est susceptible d'être soumis à des restrictions légales. Les DRM offrent alors des moyens techniques pour restreindre le contenu aux entités autorisées.

Ainsi, nous avons mis en application des restrictions. Grâce à la technologie Widevine de Google, nous avons obtenu une licence et une clé de chiffrement. Avec l'outil Bento4 et cette clé, nous avons segmenté puis chiffré la vidéo pour qu'elle puisse être diffusée par un lecteur MPEG-DASH. Parmi les fichiers de sortie, il y a le fichier manifeste qui permet d'assurer le streaming.

Pour observer le résultat, il a fallu créer un serveur web sécurisé. Sur la page du lecteur, étaient liés l'URL de la licence et le chemin vers le fichier manifeste, contenant la clé. Nous avons constaté que la vidéo était correctement diffusée, malgré le chiffrement, grâce à l'accès Widevine.

Si notre but était de devenir un diffuseur de contenu, il aurait été nécessaire d'avoir les droits adéquats pour diffuser les vidéos, ainsi qu'une bande passant adaptée au trafic important.

Webographie

« LES PRINCIPAUX FORMATS DE FLUX VIDEO LIVE DASH ET HLS » par Jérémy Bernard le 19 juillet 2017, https://blog.eleven-labs.com/fr/video-live-dash-hls/, consulté le 05/12/2021.

« Qu'est-ce que le MPEG-DASH ? | Comparatif entre les protocoles HLS et DASH » par Cloudfare, https://www.cloudflare.com/fr-fr/learning/video/what-is-mpeg-dash/, consulté le 05/12/2021.

« What is a video CDN » par Cloudfare, https://www.cloudflare.com/fr-fr/learning/video/what-is-video-cdn/, consulté le 05/12/2021

« HLS vs DASH » par Vidbeo le 30 juin 2016, https://www.vidbeo.com/blog/hls-vs-dash/, consulté le 12/12/2021.

« Diffusion en flux adaptatif dynamique sur HTTP » par Wikipédia, https://fr.wikipedia.org/wiki/Diffusion_en_flux_adaptatif_dynamique_sur_HTTP, consulté le 12/12/2021.

« How to create Widevine DRM protected content » par Bitmovin le 14 novembre 2019, https://bitmovin.com/docs/encoding/tutorials/how-to-create-widevine-drm-protected-content, consulté le 20/12/2021.

« MPEG DASH Adaptive Streaming » par Bento4, https://www.bento4.com/developers/dash/, consulté le 05/01/2022.

« MPEG DASH ENCRYPTION AND DRM » par Bento4, https://www.bento4.com/developers/dash/encryption_and_drm/, consulté le 05/01/2022.