230413.EME

Created: 2023-04-15 11:17:28

Questions

- Abstract & Status of This Document
  - The below text is “Abstract” and “Status of This Document” chapters of https://www.w3.org/TR/encrypted-media (2023/4/13)
- 1 - Introduction
  - The below text is “1. Introduction” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)
  - What is CDN? (2023/4/13)
  - What is Media Stack? (2023/4/13)
  - What is MediaKeySession? (2023/4/13)
  - Which components does MediaKeySession contain? (2023/4/13)
  - Does Media Stack contain many MediaKeySession? (2023/4/13)
  - Where do frames exist in Encrypted Media Extensions? (2023/4/13)
- 2 - Definitions
  - The below text is “2. Definitions” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)
  - The below text is the second part of “2. Definitions” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)

Q & A

Abstract & Status of This Document

Question: The below text is “Abstract” and “Status of This Document” chapters of https://www.w3.org/TR/encrypted-media (2023/4/13)

Answer: (Hide)

---

1 - Introduction

Question: The below text is “1. Introduction” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)

Answer: (Hide)

Question: What is CDN? (2023/4/13)

Answer:

A Content Delivery Network (CDN) is a system of distributed servers that deliver web content, such as images, videos, stylesheets, and scripts, to users based on their geographical location. The primary goal of a CDN is to improve the performance and availability of web content by reducing the latency experienced by users when they access resources on the internet.

When a user requests a resource, the CDN directs the request to the nearest server (also known as an edge server) in its network, minimizing the distance the data needs to travel and thus reducing latency. CDNs are commonly used by websites with a global audience to ensure faster loading times and a better user experience.

In addition to reducing latency, CDNs can also improve reliability and security by distributing the load across multiple servers and providing protection against DDoS attacks and other web threats.

Question: What is Media Stack? (2023/4/13)

Answer:

A Media Stack refers to the collection of software components, protocols, and technologies used to process, transmit, and display multimedia content, such as audio and video, on various devices. The media stack is responsible for handling tasks such as encoding, decoding, streaming, buffering, and rendering media content in a way that ensures a smooth and seamless user experience.

A typical media stack consists of several layers, including:

1. Application Layer: This layer contains the user interface and interacts with the user, enabling them to access and control the media content.
2. Media Framework Layer: This layer manages the playback of media content and provides APIs for application developers to build media-related features.
3. Codec Layer: This layer handles the encoding and decoding of media content. Codecs (encoder-decoder) are used to compress and decompress audio and video data to enable efficient transmission and storage.
4. Transport Layer: This layer is responsible for transmitting media content over a network. It includes protocols like Real-Time Messaging Protocol (RTMP), Dynamic Adaptive Streaming over HTTP (DASH), and HTTP Live Streaming (HLS) for streaming media content over the internet.
5. Operating System and Hardware Layer: This layer includes the operating system, device drivers, and hardware components that interact with the media stack. It ensures proper integration of media playback with the device's hardware, such as audio and video output systems.

These layers work together to provide a seamless media playback experience for users, regardless of the device or network conditions. The media stack must be optimized for performance, scalability, and compatibility to handle various media formats, codecs, and streaming protocols.

Question: What is MediaKeySession? (2023/4/13)

Answer:

`MediaKeySession` is an interface in the Encrypted Media Extensions (EME) API, which is a part of the HTML5 standard. The EME API enables web applications to play encrypted media content by providing a means for key exchange and license management between the web application, the Content Decryption Module (CDM), and a license server.

A `MediaKeySession` object represents a single session between the web application and the CDM. Each session is associated with a unique set of keys or licenses needed to decrypt the protected media content. The `MediaKeySession` interface provides methods for generating and exchanging messages between the web application and the CDM, allowing the CDM to request and receive decryption keys from a license server.

The typical steps for using `MediaKeySession` are:

1. Create a `MediaKeys` object, which represents the CDM, and associate it with an HTML5 `HTMLMediaElement` (e.g., a `<video>` element).
2. Create a `MediaKeySession` object by calling the `createSession()` method on the `MediaKeys` object.
3. Generate a license request by calling the `generateRequest()` method on the `MediaKeySession` object with the required Initialization Data (obtained from the encrypted media content).
4. Send the license request to the license server.
5. The license server responds with the required decryption keys or licenses.
6. Update the `MediaKeySession` with the decryption keys or licenses received from the license server using the `update()` method.

Once the `MediaKeySession` has been updated with the appropriate decryption keys or licenses, the media content can be decrypted by the CDM and played back by the `HTMLMediaElement`. The `MediaKeySession` also provides methods for managing key expiration and handling session close events.

Question: Which components does MediaKeySession contain? (2023/4/13)

Answer:

The `MediaKeySession` interface in the Encrypted Media Extensions (EME) API does not "contain" components in a hierarchical sense, but it does provide several properties and methods that allow web applications to interact with Content Decryption Modules (CDMs) for handling encrypted media content. Here are the key properties and methods associated with `MediaKeySession`:

1. Properties: - `sessionId`: A read-only property that represents a unique identifier for the session. It is generated by the CDM and can be used to reference the session for subsequent operations.
2. Methods: - `generateRequest(initDataType, initData)`: Generates a license request based on the provided Initialization Data (`initData`) and its type (`initDataType`). This method returns a Promise that resolves once the request is created, allowing the application to send the request to the appropriate license server. - `load(sessionId)`: Loads data for a specified `sessionId` into a new session. This method is useful for loading previously persisted session data, like when resuming playback after a page refresh. It returns a Promise that resolves with a boolean indicating if the data was successfully loaded or not. - `update(response)`: Updates the session with the license or key data received from the license server. The `response` parameter contains the data. This method returns a Promise that resolves once the session has been updated with the new data. - `close()`: Closes the session and releases any resources associated with it, such as decryption keys. This method returns a Promise that resolves when the session is closed. - `remove()`: Removes any license or key data associated with the session, making the keys no longer usable for decryption. This method returns a Promise that resolves when the data has been removed.
3. Event Handlers: - `onkeystatuseschange`: An event handler that gets called when key status changes occur, such as when a key expires or is released. - `onmessage`: An event handler that gets called when the CDM sends a message to the web application, usually related to license requests or errors.

These properties, methods, and event handlers work together to enable web applications to manage encrypted media sessions, request licenses, update keys, and handle related events.

Question: Does Media Stack contain many MediaKeySession? (2023/4/13)

Answer:

The term "Media Stack" refers to the collection of components, technologies, and interfaces that are involved in media processing and playback in a web browser. This includes components like media codecs, streaming protocols, and APIs like HTMLMediaElement, Media Source Extensions (MSE), and Encrypted Media Extensions (EME).

In the context of EME and encrypted media playback, a Media Stack may interact with multiple MediaKeySessions. Each MediaKeySession represents a unique session for managing encrypted media keys, licenses, and associated events.

During the playback of encrypted media, an application can create and manage multiple MediaKeySessions for different purposes, such as handling different pieces of content, managing multiple keys or licenses, or dealing with different encryption schemes. So, in this sense, the Media Stack can indeed work with multiple MediaKeySessions.

However, it is essential to understand that the Media Stack does not "contain" MediaKeySessions in a hierarchical sense, but rather interacts with them as part of the overall media processing and playback workflow.

Question: Where do frames exist in Encrypted Media Extensions? (2023/4/13)

Answer:

In the context of Encrypted Media Extensions (EME), frames refer to the individual video or audio samples that are part of the media stream. EME does not directly handle frames but focuses on the encryption and decryption aspect of media content.

Frames exist within the encrypted media container, such as MPEG-DASH or HLS. These media containers have segments, which contain encrypted frames. The media container's role is to package the video and audio frames along with the necessary metadata required for playback, including encryption-related information like initialization data.

When the media player requests a segment, it processes the encrypted frames and passes the encryption-related information to the EME. The EME then handles the necessary key management, license acquisition, and decryption of content using the

Content Decryption Module (CDM). Once the frames are decrypted, they are passed back to the media player for decoding and rendering.

In summary, frames exist within the encrypted media container, and EME manages the process of decryption but does not directly handle the frames themselves. The media player, together with EME and the underlying Content Decryption Module (CDM), work cohesively to handle, decrypt, and render the frames for playback.

---

2 - Definitions

Question: The below text is “2. Definitions” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)

Answer: (Hide)

Question: The below text is the second part of “2. Definitions” chapter of https://www.w3.org/TR/encrypted-media (2023/4/13)

Answer: (Hide)
