# CDNI - URL Signing for HAS Content

Kent Leung (kleung@cisco.com)

# URL Signing

Content  access control:

- URL is appended with authorization parameters (e.g. client IP address, expiration time) and message digest (generated with key)
- URL is validated by surrogate (with key) to ensure the request has legitimate access to the content
- Symmetric key or asymmetric keys (private/ public key pair) ; key distribution is out of scope

Request routing methods:

- DNS request routing does not change the URL

  For symmetric key, CSP signs the URL with the same key used by Delivering CDN to validate URL. Key needs to be distributed from Authoritative CDN to the Delivering CDN. Exposing the key to the Delivering CDN that doesn't have relationship with the CSP may be problematic. This is a generic issue for URL Signing and not specific to HAS content.

  For asymmetric keys, CSP signs URL with its private key. Delivering CDN validates the URL with the public key which can be obtained by various methods.

- HTTP request routing changes the URL

  For symmetric key, CSP signs the original URL with the same key used by Authoritative CDN to validate URL. The Authoritative CDN (Upstream CDN) redirects the request to the Downstream CDN. The new URL is signed by the Upstream CDN with the same key used by the Downstream CDN to validate that URL. The key used by the Upstream CDN to validate the original URL is expected to be different than the key used to sign the new URL.

  For asymmetric keys, CSP signs the original URL with its private key. Authoritative CDN validates that URL with the CSP's public key. The Authoritative CDN (Upstream CDN) redirects the request to the Downstream CDN. The new URL is signed by the Upstream CDN with its private key. The Downstream CDN validates that URL with the Upstream CDN's public key.

# CDNI Interface Implications

Most of the CDNI Interfaces need to be enhanced for URL Signing:

- The CDNI Metadata interface should specify the content that is subject to URL signing and provide information for signing and validating an URL.
- The Downstream CDN should inform the Upstream CDN that it supports URL Signing in the asynchronous capabilities information advertisement as part of the Request Routing interface. This allows the CDN selection function in request routing to choose the Downstream CDN with URL signing capability when the CDNI metadata of the content requires this authorization method.
- The Logging interface provides information on the authorization method (e.g. URL Signing) and related authorization parameters used for content delivery.
- URL Signing has no impact on the Control interface.

# URL Signing for HAS Content

Options to consider:

1. Do nothing about HAS content
2. Flexible URL Signing for HAS content
3. Authorization Group ID for HAS content
4. Handle HAS content in CDN

# 1. Do Nothing about HAS Content

"Do Nothing" approach means that CSP can only perform URL Signing for the top level manifest file.

- URL Signing for the top level manifest file (provided by CSP)
- Top level manifest file contains chunk URLs or lower level manifest file URLs (i.e. no URL Signing for the embedded URLs)
- The lower level manifest files and chunks are delivered without content access authorization

**Effect on CDN Interfaces**:

- None

**Advantages**:

- Top level manifest file access is protected
- CDN does not need to be aware of HAS content
- CSP does not need to change the manifest files (i.e. embedded URLs remains the same)

**Drawbacks**:

- Lower level manifest files and chunks are ***not protected***, making this approach ***unqualified for content access authorization***

# 2. Flexible URL Signing for HAS Content

In addition to URL Signing for the top level manifest file, CSP performs flexible URL Signing for the lower level manifest files and chunk URLs.

- URL Signing for the top level manifest file (provided by CSP)
- Top level manifest file (dynamically generated by CSP) contains session-based chunk signed URLs or lower level manifest file signed URLs (i.e. URL Signing in the embedded URLs)
- Lower level manifest file (dynamically generated by CSP) contains session-based chunk signed URLs
- The chunk (segment/fragment) is delivered with content access authorization using flexible URL Signing which protects the invariant portion of the URL
- Segment URL (e.g. HLS) is individually signed for the invariant URL portion (Relative URL) or the entire URL (Absolute URL without Redirection) in the manifest file
- Fragment URL (e.g. Smooth Streaming) is signed for the invariant portion of the template URL in the manifest file
- The URL Signing expiration time for the chunk needs to be long enough to play video

**Effect on CDN Interfaces**:

- Requires the ability to exclude the variant portion of URL in the signing process (NOTE: Issue is specific to URL Signing support for HAS content and not CDNI?)

**Advantages**:

- Manifest file and chunks are protected
- CDN does not need to be aware of HAS content
- DNS-based request routing with asymmetric keys and HTTP-based request routing for Relative URL and Absolute URL without Redirection works

**Drawbacks**:

- CSP has to generate manifest files with session-based signed URLs and ***becomes involved*** in content access authorization for ***every HAS session***
- Manifest files are ***not cacheable***
- DNS-based request routing with symmetric key may be problematic due to need for transitive trust between CSP and Delivery CDN
- HTTP-based request routing for Absolute URL with Redirection does not work because the URL used Delivery CDN surrogate is unknown to the CSP

# 3. Authorization Group ID for HAS Content

CSP performs URL Signing for the top level manifest file. Based on the Authorization Group ID metadata, CDN validates the URL Signing or validates the HTTP cookie for request of content in the group.

- URL Signing for the top level manifest file (provided by CSP)
- Top level manifest file contains chunk URLs or lower level manifest file URLs (i.e. no URL Signing for the embedded URLs)
- The lower level manifest files and chunks are delivered with content access authorization using HTTP cookie that contains state associated with authorization of the top level manifest file
- Authorization Group ID metadata is used to associate the related content (i.e. manifest files and chunks). It also specifies content (e.g. regexp method) that needs to be validated by either URL Signing or HTTP cookie.
- Duration of the chunk access may be included in the URL Signing of the top level manifest file and set in the cookie; Duration may be in the metadata instead

**Effect on CDN Interfaces**:

- CDNI Metadata Interface - Authorization Group ID metadata identifies the content that is subject to validation of URL Signing or validation of HTTP cookie associated with the URL Signing
- CDNI Logging Interface – Report the authorization method used to validate the request for content delivery

**Advantages**:

- Manifest file and chunks are protected
- CDN does not need to be aware of HAS content
- CSP does not need to change the manifest files

**Drawbacks**:

- Authorization Group ID metadata is required (i.e. CDNI Metadata Interface enhancement)
- Using HTTP cookie requires enabling this function and the cookie security implications apply
- Logic needed for handling surrogate switchover

# 4. Handle HAS Content in CDN

CDN is aware of HAS content and uses URL Signing and HTTP cookie for content access authorization

- URL Signing for the top level manifest file (provided by CSP)
- Top level manifest file contains chunk URLs or lower level manifest file URLs (i.e. no URL Signing for the embedded URLs)
- The lower level manifest files and chunks are delivered with content access control using HTTP cookie that contains session state associated with authorization of the top level manifest file
- CDN is aware of the HAS content type and handles segment/fragment by associating such content with the content access authorization provided with URL Signing
- Duration of the chunk access may be included in the URL Signing of the top level manifest file and set in the cookie

**Effect on CDN Interfaces**:

- CDNI Metadata Interface – New metadata identifies the content that is subject to validation of URL Signing and information in the cookie for the type of HAS content
- Request Routing interface – Downstream CDN should inform the Upstream CDN that it supports URL Signing for known HAS content types in the asynchronous capabilities information advertisement. This allows the CDN selection function in request routing to choose the appropriate Downstream CDN when the CDNI metadata identifies the content.
- CDNI Logging Interface – Report the authorization method used to validate the request for content delivery

**Advantages**:

- Manifest file and chunks are protected
- CSP does not need to change the manifest files

**Drawbacks**:

- CDN needs to be ***aware of HAS content***
- Use of HTTP cookie (for HAS session state) requires enabling this function and the cookie security implications apply
- Logic needed for handling surrogate switchover

# Conclusions

**Summary**:

"Do nothing about HAS content" (#1) approach requires no change to CSP or CDN but is undesirable because of the lack of protection for the content.

"Flexible URL Signing for HAS content" (#2) approach requires flexible URL Signing support in CDN and depends on CSP to be involved in every HAS session.

"Authorization Group ID for HAS content" (#3) approach requires new CDNI metadata to associate the URL Signing with HTTP cookie to validate a request for content in the logical group.

"Handle HAS content in CDN" (#4) requires CDN to be aware of HAS content and impacts multiple CDNI Interfaces.

**Recommendations**:

Debatable between #2 vs. #3. Is the requirement for CSP to generate manifest files for every HAS session a practical approach? Is non-cacheable manifest file a major issue or minor nuisance? What are the other key factors to decide between the two choices?

(FUTURE) Option #4 has some advantages that should be considered for future support (e.g. CDN that is aware of HAS content can manage the content more efficiently at a broader context. Content distribution, storage, delivery, deletion, access authorization, etc. can all benefit.)

| | Do Nothing with HAS Content | Handle HAS Content in CSP | Group ID Metadata for HAS Content | Handle HAS Content in CDN |
|---|---|---|---|---|
| **Top level manifest file (TLMF) protection** | Y | Y | Y | Y |
| **Lower level manifest file (LLMF) and chunk protection** | N | Y | Y | Y |
| **No manifest file change per HAS session by CSP** | Y | N | Y | Y |
| **No URL rewrite in manifest file by CDN** | Y | N (HTTP)<br>Y ( DNS) | Y | Y |
| **No HTTP cookie used to track HAS session** | Y | Y | N | N |
| **No state maintained for surrogate switchover** | Y | Y | N | N |
| **No group ID metadata used** | Y | Y | N | Y |
| **No HAS awareness** | Y | Y | Y | N |
| **Authorization time window** | Y (TLMF with URL Signing)<br>N (LLMF & chunk) | Y (TLMF/ LLMF/ chunk with URL Signing) | Y (TLMF with URL Signing & LLMF/ chunk with cookie) | Y (TLMF with URL Signing & LLMF/ chunk with cookie) |
| **Live streaming supported** | N (Lower level manifest file & chunk are not protected) | Y (manifest file change per HAS session) | Y | Y |
| **Surrogate never received top level manifest file** | N | Y | Y | Y |
| **Manifest file is hosted separately from chunk** | N | Y | N | N |