Models for adaptive-streaming-aware CDNI

draft-brandenburg-cdni-has-03

IETF 84

Vancouver

July 31, 2012

Ray van Brandenburg

Francois Le Faucheur

Kent Leung

Why this draft?

- Although CDNI should be content-agnostic, HAS content poses some unique challenges
  - Very large number of (possibly distributed) files
  - Session-less nature makes session-based logging difficult and chunk-based logging bulky
  - Manifest file interferes with Request Routing
  - Etc…

- This draft…
  - Introduces terminology
  - Discusses some of the problem areas when combing HAS and CDNI
  - Introduces different options for level of HAS awareness in CDNI
  - Provides recommendation to WG on level of HAS awareness to support in deliverables of current CDNI charter

Necessities vs. Optimizations

- Support for adaptive streaming in CDNI is a requirement
- First version of CDNI specs (i.e. CDNI deliverables as per current charter) should be focused on **'making HAS work'**
  - Only those features that are absolutely necessary in order to allow for delivery of HAS in a scalable manner
- Later iterations (after re-chartering of WG) can include further optimizations
  - Some of those are described in the draft

File Management of HAS content

- Option 1.1: No HAS awareness
  - 'Do Nothing'-approach
  - dCDN is unaware of relationship between chunks, forced to store chunks as individual files.
- Option 1.2: Allow single file storage of fragmented content
  - Full 'HAS-awareness'
  - CDNI Metadata Interface signals type of HAS, name of manifest, etc.
  - Allows dCDN to store fragmented content as single file
- Option 1.3: Access correlation hint
  - Add 'Access Correlation Hint' to CDNI Metadata of all chunks belonging to same content collection
  - Can be used by dCDN to know which files are likely to be requested after each other in small time window
- **Recommendation:**
  - In initial version of CDNI Interfaces go for Option 1.1
  - Option 1.2 can be considered for re-chartering after initial solution is completed

Content Acquisition of HAS content

- Option 2.1: No HAS awareness
  - 'Do Nothing'-approach
  - dCDN is unaware of relationship between chunks, forced to acquire chunks as individual files
  - Increased overhead
- Option 2.2: Allow single file acquisition of fragmented content
  - Full 'HAS-awareness'
  - CDNI Metadata Interface signals type of HAS, name of manifest, etc.
  - Allows dCDN to acquire fragmented content as single file
- **Recommendation:**
  - In initial version of CDNI Interfaces go for Option 2.1
  - Option 2.1 can be considered for re-chartering after initial solution is completed

Dealing with manifest files and Request Routing

- Option 3.1: No HAS awareness
  - 'Do Nothing'-approach
  - Absolute URLs with Redirection can cause very significant overhead (one full CDNI redirection process for every chunk)
  - Relative URLs work, except in cases where Path Absolute Relative URLs are used
  - Absolute URLs without redirection not supported
- Option 3.2: Manifest File rewriting by uCDN
  - Allow uCDN to rewrite manifest file (e.g. change URLs to point to dCDN Request Router)
  - Does not require changes to CDNI Interfaces. Uses existing CDNI RR Interface for obtaining location of dCDN RR (or surrogate)
  - Transparent to dCDN (no HAS awareness required)
  - Can be optional feature (not mandatory for uCDNs)
- Option 3.3: Two-step Manifest File rewriting
  - Also allow dCDN to rewrite manifest file
  - Requires full 'HAS-awareness' on behalf of dCDN
  - Requires changes to CDNI interfaces

Dealing with manifest files and Request Routing - 2

- Option 3.1: No HAS awareness
  - 'Do Nothing'-approach
  - (…)
- Option 3.2: Manifest File rewriting by uCDN
  - Allow uCDN to rewrite manifest file (e.g. change URLs to point to dCDN Request Router)
  - (…)
- Option 3.3: Two-step Manifest File rewriting
  - Also allow dCDN to rewrite manifest file
  - (…)
- **Recommendation:**
  - Mandatory support for Option 3.1
  - Allow Option 3.2 for uCDN that support this
  - Do not support Option 3.3, but mark as candidate for possible re-chartering in the future

Logging of HAS content

- Option 4.1: Do Nothing Specific for HAS
  - Per-chunk logging (chunks are individual content items)
- Option 4.2: Content Collection Identifier (CCID) Approach
  - Per-chunk logging (chunks are individual content items)
  - CCID is distributed through Metadata Interface
  - CCID is included by dCDN in log entries, facilitates log summarization at later stage
- Option 4.3: Log Transport Compressions
  - Log files are compressed before transportation across Logging Interface
- Option 4.4: Full HAS Awareness (per-session logs)
  - dCDN is fully HAS aware and creates per-session logs
- **Recommendation:**
  - Mandatory support for Option 4.1 (per-chunk logging)
  - Allow Option 4.2 as an optional feature for CDNs that support this
  - Optionally, common compression (e.g. gzip) can be applied to log files

URL Signing of HAS Content - 1

- Option 5.1: Do nothing Specific for HAS
  - CSP can only perform URL Signing for the top level manifest file. URLs embedded in the manifest file do not change.
  - Lack of protection for lower level manifest files and chunks
- Option 5.2: Flexible URL Signing (CSP & uCDN)
  - CSP/uCDN performs flexible URL Signing (which protects the invariant portion of the URL) for the lower level manifest files and chunk URLs.
  - Manifest files and chunks are protected
  - uCDN/dCDN & dCDN do not need to be aware of HAS content
  - DNS-based request routing with asymmetric keys and HTTP-based request routing for Relative URL works
  - CSP & uCDN 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 & uCDN and Delivery CDN
  - HTTP-based request routing for Absolute URL with Redirection does not work because the URL used by Delivery CDN surrogate is unknown to the CSP & uCDN

URL Signing of HAS Content - 2

- Option 5.3: Authorization Group ID and HTTP Cookie
  - 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.
  - 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
  - Authorization Group ID metadata is required (i.e. CDNI Metadata Interface enhancement)
  - Requires the use of HTTP cookie which may be considered to be not desirable or even feasible
  - Manifest file has to be delivered by surrogate
- Option 5.4: HAS-awareness in CDN (HTTP Cookie and Manifest)
  - CDN is aware of HAS content and uses URL Signing and HTTP cookie/manifest file for content access authorization
  - Manifest file and chunks are protected
  - CSP does not need to change the manifest files
  - Requires full HAS awareness on part of uCDN and dCDN
  - Requires CDNI Interfaces extensions
  - Requires the use of HTTP cookie which may be considered to be not desirable or even feasible OR requires CDN to generate or rewrite the manifest file
  - Manifest file has to be delivered by surrogate

URL Signing of HAS Content - 3

	Do Nothing	Flexible URL Signing (CSP / uCDN)	Authorization Group ID Metadata and HTTP Cookie	HAS-aware CDN (Cookie / Manifest)
[+] Top level manifest file (TLMF) protection	+	+ / +	+	+ / +
[+] Lower level manifest file (LLMF) and chunk protection	-	+ / +	+	+ / +
[-] CSP creates manifest file for each HAS session	+	- / +	+	+ / +
[-] uCDN creates manifest file for each HAS session	+	+ / -	+	+ / +
[-] Use of HTTP cookie	+	+ / +	-	- / +
[-] State maintained for surrogate switchover	+	+ / +	-	- / +
[-] CDNI metadata is required	-	+ / +	-	- / -
[-] Requires HAS awareness	+	+ / +	+	- / -
[+] Manifest files are hosted separately from dCDN	-	+ / +	-	- / -
[+] Surrogate delivers chunk without receiving the top level manifest file	+	+ / +	+*	+* / -

[+] denotes positive property in the option
[-] denotes negative property in the option
* when cookie has state created by other surrogate that received the top level manifest file

URL Signing of HAS Content - 4

- **Recommendation:**
  - Process of elimination. “Do nothing” (#1) is not acceptable because of the lack of protection for the bulk of HAS content. Solution should not require the use of HTTP cookie, remove “Authorization Group ID” (#3). Also, it should not require HAS awareness in the CDN, remove “HAS-awareness in CDN” (#4).
  - Flexible URL Signing (#3) is recommended because the approach protects all the content, does not require Downstream CDN to be aware of HAS, does not impact CDNI interfaces, supports all different types of devices, and supports the common cases of request routing for HAS content (i.e. DNS-based request routing with asymmetric keys and HTTP-based request routing for Relative URL). The requirement for CSP/Upstream CDN to manipulate the manifest file is not considered to be a significant obstacle as long as Downstream CDN remains unaware of HAS.

Other

- Content Purge
  - Draft does net yet include recommendation for Content Purge, but describes two options
  - Option 6.1: Do Nothing. Content Purge per chunk
  - Option 6.2: Include Content Collection ID for purging of all files related to CCID
  - Recommendation: Option 6.1 mandatory, Option 6.2 optional.

- Capabilities
  - Include method for dCDNs to advertise whether they support optional HAS-specific features

Summary

- Recommendation 1: No HAS-specific File Management. dCDN is not explicitly made aware of relationship between chunks; each chunk is considered an independent content item
- Recommendation 2: No HAS-specific Content Acquisition. Content Acquisition on a per-chunk basis
- Recommendation 3: Mandatory requirement to support per-chunk request routing. uCDNs can optionally rewrite manifest files to point directly to dCDN and reduce request routing overhead
- Recommendation 4: Mandatory requirement to support per-chunk logging. Optional feature to include Content Collection Identifier in log entries.
- Recommendation 5: Mandatory requirement to support Flexible URL Signing

Next Steps

- Include summary in CDNI Framework document
- Align CDNI Requirements
- Standardize HAS-specific elements in relevant drafts
  - HAS-specific elements for logging in Logging Interface
  - Content Purge in Control Interface
  - Etc.
- Clean up document
  - Formalize recommendations
- Where will we store this HAS analysis?
  - move current document towards Informational RFC?
  - Move to Framework as an Appendix?