CDNI
Internet-Draft
Intended status: Standards Track
Expires: April 17, 2013

K. Leung
F. Le Faucheur
M. Caulfield
Cisco Systems
Oct 14, 2012

URI Signing for CDN Interconnection (CDNI)
draft-leung-cdni-uri-signing-01

Abstract

This document describes how the concept of URI signing supports the content access control requirements of CDNI and proposes a candidate URI signing scheme.

The proposed URI signing method specifies the information needed to be included in the URI and the algorithm used to authorize and to validate access request for the content referenced by the URI.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 17, 2013.

Table of Contents

1. Introduction

The overall problem space for CDN Interconnection is described in [RFC6707].

The CDNI Problem Statement [RFC6707], the CDN requirements document [I-D.ietf-cdni-requirements] and the CDNI Framework document[I-D.ietf-cdni-framework] discuss the need for the interconnected CDNs to be able to implement an access control mechanism that enforces the Content Service Provider (CSP) distribution policy.

Specifically, [I-D.ietf-cdni-framework] states:

"The CSP may also trust the CDN operator to perform actions such as ..., and to enforce per-request authorization performed by the CSP using techniques such as URI signing."

In particular, the following requirement is listed in [I-D.ietf-cdni-requirements]:

"META-17 [HIGH] The CDNI Metadata Distribution interface shall allow signaling of authorization checks and validation that are to be performed by the surrogate before delivery. For example, this could potentially include:

* need to validate URI signed information (e.g. Expiry time, Client IP address)."

This document proposes a URI signing scheme that allows Surrogates in interconnected CDNs to enforce a per-request authorization performed by the CSP. Splitting the role of performing per-request authorization by CSP and the role of validation of this authorization by the CDN allows any arbitrary distribution policy to be enforced across CDNs without the need of CDNs to have any awareness of the actual CSP distribution policy.

1.1. Terminology

This document uses the terminology defined in [RFC6707].

This document also uses the terminology of [RFC2104] including the following terms (reproduced here for convenience):

- MAC: message authentication code

- HMAC: hash-based message authentication code (HMAC)

- HMAC-SHA1: HMAC instantiation using SHA1 as the cryptographic hash function

- HMAC-MD5: HMAC instantiation using MD5 as the cryptographic hash function

In addition, the following terms are used throughout this document:

- URI Signature: message digest that is computed with an algorithm that uses the key, the Original URI and request attributes as inputs to the hash function. This digest is conveyed inside the Signed URI.

- Original URI: the URI before URI signing is applied.

- Signed URI: the URI containing the Original URI, the attributes and the URI Signature.

1.2. URI Signing Overview

URI Signing is an authorization method for content delivery. This is based on embedding the URI with information that can be validated to ensure the request has legitimate access to the content. There are two parts: 1) attributes that convey authorization restrictions (e.g. source IP address and time period), and 2) message digest that confirms the integrity and authenticity of the URI provided by the URI creator. The authorization attributes can be anything agreed upon between the entity that creates the URI and the entity that validates the URI. A key is used by the HMAC algorithm of the URI signing function to generate the message digest (i.e. sign the URI). A key is also used by the HMAC algorithm of the URI signature validating function to validate the message digest (i.e. URI signature). The two functions may or may not use the same key.

Two types of keys can be used for URI Signing: asymmetric keys and symmetric key. Asymmetric keys always have a key pair made up of a public key and private key. The private key and public key are used for signing and validating the URI, respectively. A symmetric key is used for both functions. Regardless of the type of key, the entity that validates the URI has to obtain the key. There are very different requirements for key distribution with asymmetric keys and with symmetric keys. Key distribution for symmetric keys requires confidentiality to prevent another party from getting access to the key, since it could then generate valid Signed URIs for unauthorized requests. Key distribution for asymmetric keys does not require confidentiality since public keys can typically be distributed openly (because they cannot be used for URI signing) and private keys are kept by the URI signing function.

URI Signing operates in the following way. After request authorisation, the CSP computed a Signed URO from the Original URI and provides the signed URI to the user out of band. The user request for the Signed URI is handled by the CDN which is responsible for validating the URI Signature before delivering the content.

2. Authorization Attributes in URI Signing

This section identifies the set of attributes that may be needed to enforce the CSP distribution policy. These attributes can therefore be covered by the URI Signature hash and can be embedded (by the signing function) in the as query component of the Signed URI (to enable subsequent signature validation by the signature validating function).

In order to provide flexibility in distribution policies to be enforced, the exact subset of attributes used for URI signature in a given request is a deployment decision. The defined keyword for each query string attribute is specified in parenthesis below.

- Version (VER) - An integer used for identifying the version of URI signing method with its set of capabilities.
- Expiry Time (ET) - Time in seconds when URI Signature expires since midnight 1/1/1970 UTC (i.e. UNIX epoch).
- Client IP (CIP) - IP address of the client, in a dotted decimal format.
- Key Owner (KO) - Identifier of the owner of the key used for URI signing, in an integer format.
- Key ID (KN) - A number that is used as an index, within the set of keys of a given Key Owner, to the key used for URI signing, in an integer format.
- Hash Function (HF) - A string used for identifying the hash function to compute the URI signature (e.g. "MD5", "SHA1").
- Algorithm (ALG) - An integer used for identifying the algorithm to compute the URI signature.
- Client ID (CID) - Identifier of the client such as IMSI, MSISDN, MEID, MAC address, etc.

The query string attributes are embedded within the Signed URI to be used for the content request in order to provide to the signature

validating function the information needed to enforce the distribution policy and to validate the URI Signature. Each of the attributes is further described below.

The Version attribute indicates which version of URI signing scheme is used (including which attributes and algorithms are supported. The present document specifies Version 0. More versions may be defined in the future.

The Expiry Time attribute ensures that the content authorization expires after a predetermined time. This limits the time window for content access and prevents replay of the request beyond the authorized time window.

The Client IP attribute is used to restrict content access to a particular End User, based on its client IP address for whom the content access was authorized.

The Key Owner and Key ID attributes are used to identify the key that is to be retrieved as input to the HMAC algorithm to compute the message digest for validating the signed URI.

The Hash function attribute indicates the HMAC hash function to be used for message digest computation.

The Algorithm indicates the specific algorithm for computation of the URI Signature. For example, this indicates whether the scheme component of the URI is to be covered by the signature computation or not.

The Client ID attribute is used to restrict content access to a particular user associated with this identifier. For example, it could be the information about the subscriber, device, or network access interface.

3. URI Signing and Validation

The keyword for embedding the actual URI Signature in the URI query string is "US".

The following steps are taken for signing a URI for the algorithms defined in this document. Note that some steps may be skipped if the attribute is not needed to enforce the distribution policy. The entire URI (i.e. scheme, authority, path, query, and fragment as defined in URI Generic Syntax [RFC3986]) is protected by the URI signature when the algorithm (i.e. "ALG") is set to 1. The scheme is removed from the URI when the algorithm is set to 2. This allows

the URI signature to be validated correctly in the case when a client performs a fallback to HTTP for a content referenced by an URI with RTSP scheme.

1. Check if the Original URI already contains a query string. If not, append a "?" character. If yes, append an "&" character.

2. Append the string "VER=0". This represents the version of URI Signing specified in this document.

3. Append the string "&ET=".

4. Get the current time in seconds since epoch (as an integer). Add the validity time in seconds as an integer.

5. Append this integer.

6. Append the string "&CIP=".

7. Append the client's IP address in dotted decimal format.

8. Append the string "&KO=".

9. Append the numeric value of the key owner corresponding to the key being used.

10. Append the string "&KN=".

11. Append the key ID number corresponding to the key being used.

12. Append the string "&HF=".

13. Append the string for the type of hash function.

14. Append the string "&ALG=".

15. Append the integer for the type of algorithm. If algorithm is "1", no additional logic needed by default. If algorithm is "2", remove the scheme part of the URI.

16. Append the string "&US=".

17. Store this as the message on which to compute the hash-based message authentication code (e.g. http://example.com/content.mov?VER=0&ET=1209422976&CIP=171.71.50.123&KO=1&KN=2&HF=1&ALG=1&US=).

18. For symmetric key, compute the message digest (i.e. URI signature) using the algorithm with key and message as inputs to the hash function. For asymmetric keys, after the message digest computation (as described previously only using the public key), use the public key again to encrypt the message digest.

19. Convert the message digest to its equivalent human readable hexidecimal value (e.g. f08b56f46075813e44b2d4888628a471).

20. Append this hexidecimal value to the previously created message. This is the complete Signed URI.

The following steps are taken for validating a Signed URI. Note that some steps are to be skipped if the corresponding attribute is not embedded in the Signed URI. The absence of a given attribute indicates enforcement of its purpose is not necessary in the distribution policy.

1. Check if the Signed URI contains a query string. If not, it is not a Signed URI. If the CDNI Metadata for the corresponding content indicate that access control is to be enforced via URI Signing, then the request is denied.

2. Extract the value from "US=" part of URI. This value is the URI signature.

3. Extract the values from "KO=" and "KN= part of URI. Use these values to locate the key value and also key type (i.e. asymmetric or symmetric)

4. Extract the value from "HF=" part of URI. The value is the type of hash function.

5. Extract the value from "ALG=" part of URI. The value is the type of algorithm.

6. Store URI excluding the part after "US=" as the message on which to compute the hash-based message authentication code.

7. If the extracted algorithm value is "1", keep message without change. If algorithm value is "2", remove the scheme part of the URI in the message.

8. Compute the message digest (i.e. URI signature) using the algorithm with key and message as inputs to the hash function (based on the extracted hash function value).

9. For symmetric key, compare this computed digest with the received URI Signature. For asymmetric keys, decrypt the URI Signature with the public key. Then compare the computed digest with the decrypted URI Signature. If the comparison is not a match, the request is denied. Otherwise, continue with next step. Note that the request is denied if any of the following validations failed.

10. Validate that the request came from the same IP address as indicated in the "CIP=".

11. Validate that the request arrived before expiration time as indicated in the "ET=" based on the current time.

4. Considerations for CDNI Interfaces

The CDNI Interfaces need enhancements to support URI Signing. A Downstream CDN that supports URI Signing needs to be able to advertise this capability to the Upstream CDN. The Upstream CDN selects a Downstream CDN based on such capability when the CSP requires access control to enforce its distribution policy via URI Signing. Also, the Upstream CDN need to be able to distribute via the CDNI Metadata interface the information necessary to allow the Downstream CDN to validate a Signed URI . Events that pertain to URI Signing (e.g. request denial or delivery after access authorization) need to be included in the logs communicated through the CDNI Logging interface.

4.1. CDNI Capabilities Advertisement

The Downstream CDN advertises its capability to support URI Signing via the CDNI Request Routing/Footprint & Capabilities Advertisement interface. The supported version of URI Signing needs to be included. TBD: to be taken into account by Footprint & Capabilities design team working on this area.

o URI Signing support and its version

4.2. CDNI Metadata Interface

The following CDNI metadata are specified for URI Signing. Note that the Key Owner and Key ID information are not needed if only one key is provided by CSP or Upstream CDN for the content or set of contents covered by the CDNI metadata. Also, the CDNI metadata for HMAC algorithm is not needed when the Algorithm attribute is embedded in the signed URI. TBD: CDNI Metadata Interface is work in progress.

- Content access control indication.
- Type of access control. Specifically, access to content is subject to URI Signing
- Key value along with its key index (i.e. Key Owner and Key ID) and type (asymmetric or symmetric) used for validating URI signature
- List of Downstream CDNs authorized for key distribution (i.e. trust relationship between CSP and CDNs) [Editor's Note: is this needed?]
- Algorithm for HMAC to be used for validation.

4.3. CDNI Logging Interface

The Downstream CDN reports that enforcement of the access control was applied to the request for content delivery. TBD: CDNI Logging interface is work in progress.

- URI signature validation events (e.g. invalid client IP address, expired signed URI, incorrect URI signature, successful validation)
- Delivery log with confirmation of access control enforcement (i.e Delivery CDN enforced URI Signing before content delivery)

5. URI Signing Operation

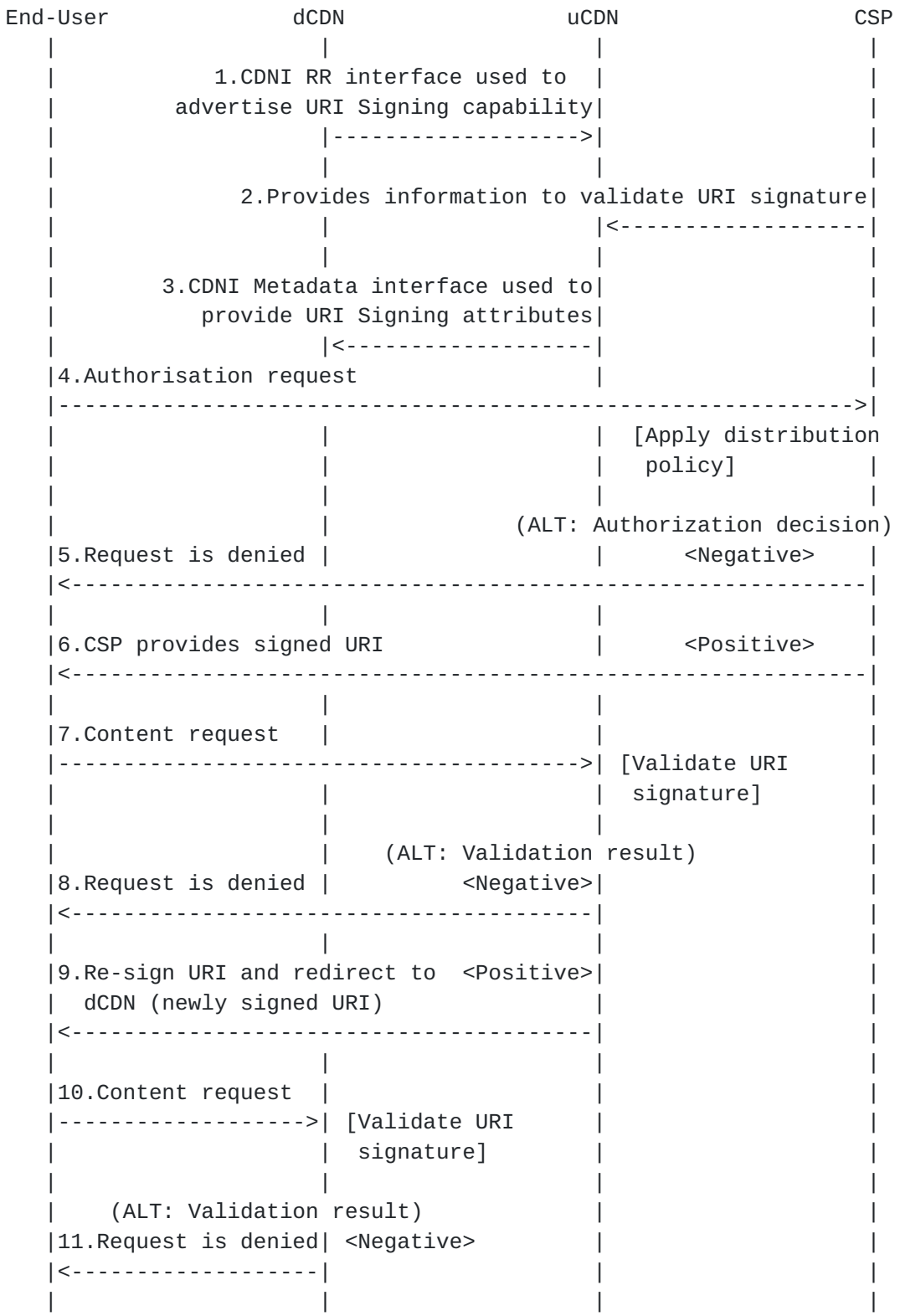
URI Signing supports both the HTTP-based and DNS-based request routing. HMAC [RFC2104] defines a hash-based message authentication code allowing two parties that share a symmetric key or asymmetric keys to establish the integrity and authenticity of a set of information (e.g. a message) through a cryptographic hash function.

5.1. HTTP Redirection

For HTTP-based request routing, HMAC is applied to a set of information that is unique to a given end user content request using key information that is specific to a pair of adjacent CDNI hops (e.g. between the CSP and the Authoritative CDN, between the Authoritative CDN and a Downstream CDN). This allows a CDNI hop to ascertain the authenticity of a given request received from a previous CDNI hop.

The URI signing scheme described below is based on the following

steps (assuming HTTP redirection, iterative request routing and a CDN path with two CDNs). Note that Authoritative CDN and Upstream CDN are used exchangeably.



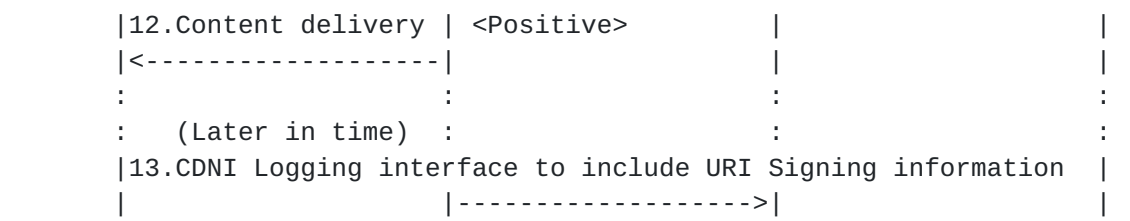


Figure 1: HTTP-based Request Routing with URI Signing

1. Using the CDNI Request Routing/Footprint & Capabilities Advertisement interface, the Downstream CDN advertises its capabilities including URI Signing support to the Authoritative CDN.

2. CSP provides to the Authoritative CDN the information needed to validate URI signatures from that CSP. For example, this information may include a hashing function, algorithm, and a key value.

3. Using the CDNI Metadata interface, the Authoritative CDN communicates to a Downstream CDN the information needed to validate URI signatures from the Authoritative CDN for the given CSP. For example, this information may include a hashing algorithm and private key corresponding to the trust relationship between the Authoritative CDN and the Downstream CDN.

4. On receipt of a given authorisation request on the CSP portal, the CSP makes a specific authorization decision for this unique request based on its arbitrary distribution policy.

5. If the authorization decision is negative, the CSP rejects the request.

6. If the authorization decision is positive, the CSP computes a Signed URI that is based on unique parameters of that request and conveys it to the end user as the URI to use to request the content.

7. On receipt of the corresponding content request, the authoritative CDN validates the URI Signature in the URI using the information provided by the CSP.

8. If the validation is negative, the authoritative CDN rejects the request

9. If the validation is positive, the authoritative CDN computes a Signed URI that is based on unique parameters of that request

and provides to the end user as the URI to use to further request the content from the Downstream CDN

10. On receipt of the corresponding content request, the Downstream CDN validates the URI Signature in the Signed URI using the information provided by the Authoritative CDN in the CDNI Metadata

11. If the validation is negative, the Downstream CDN rejects the request and sends an error code (e.g. 403) in the HTTP response.

12. If the validation is positive, the Downstream CDN serves the request and delivers the content.

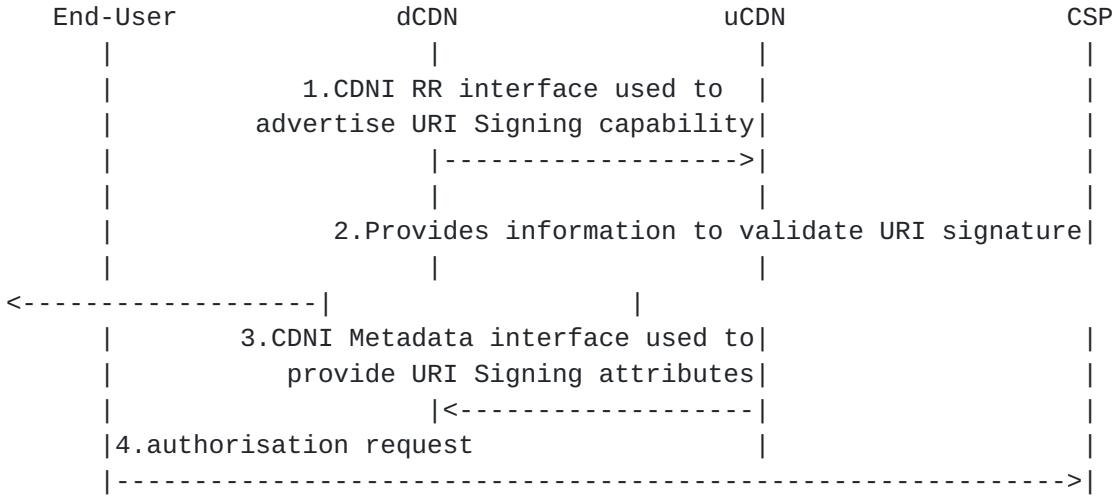
13. At a later time, Downstream CDN reports logging events that includes URI signing information.

With HTTP-based request routing, URI Signing matches well the general chain of trust model of CDNI both with symmetric key and asymmetric keys because the key information only need to be specific to a pair of adjacent CDNI hops.

5.2. DNS Redirection

For DNS-based request routing, HMAC is applied to a set of information that is unique to a given end user content request using a secret key shared between CSP and the Delivery CDN. The Delivery CDN needs to obtain the key information to validate the Signed URL, which is computed by the CSP based on its distribution policy.

The URI signing scheme described below is based on the following steps (assuming iterative DNS request routing and a CDN path with two CDNs). Note that Authoritative CDN and Upstream CDN are used exchangeably.



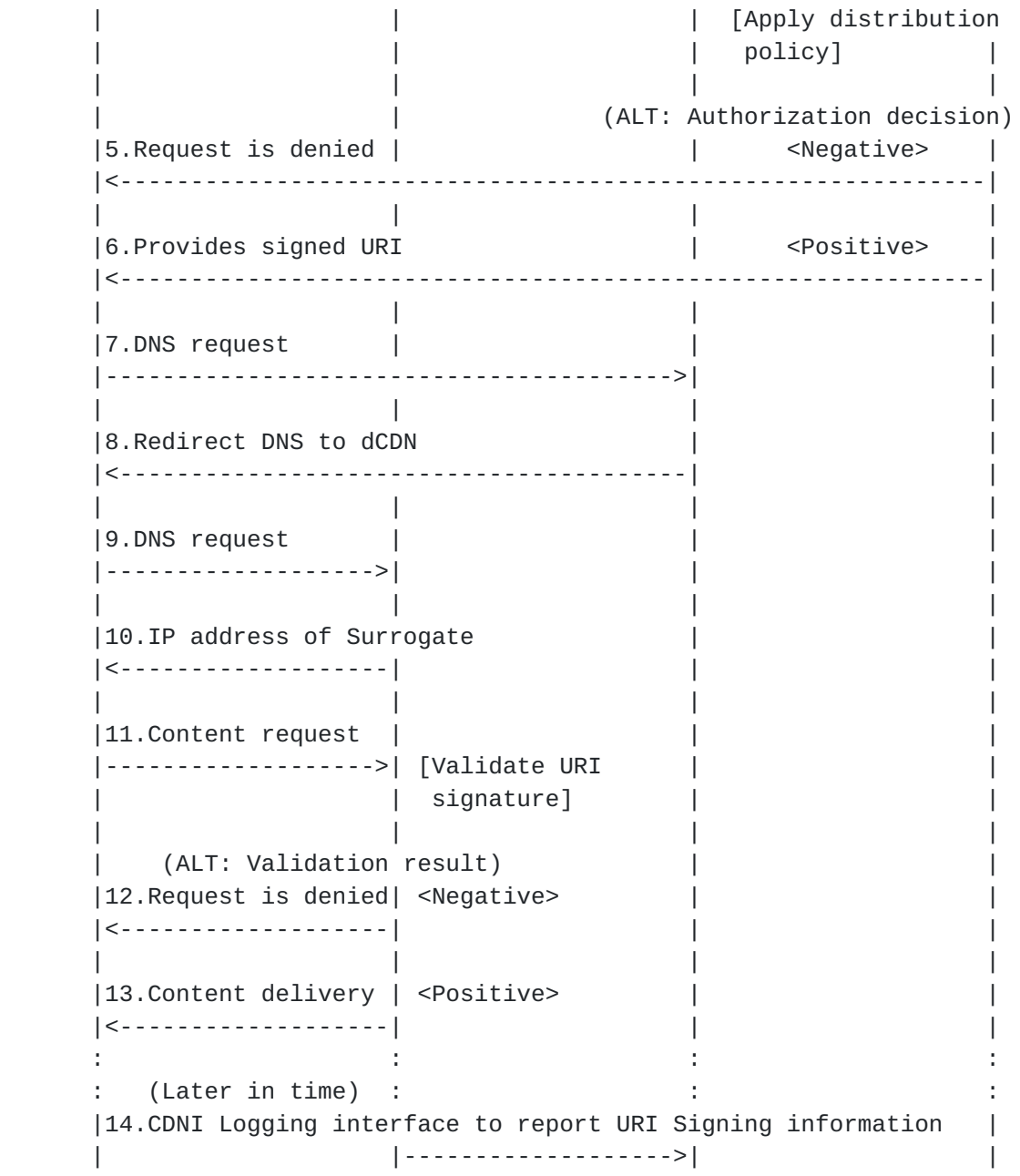


Figure 2: DNS-based Request Routing with URI Signing

1. Using the CDNI Request Routing interface, the Downstream CDN advertises its capabilities including URI Signing support to the Authoritative CDN.

2. CSP provides to the Authoritative CDN the information needed to validate cryptographic signatures from that CSP. For example, this information may include a hash function, algorithm, and a key.

3. Using the CDNI Metadata interface, the Authoritative CDN communicates to a Downstream CDN the information needed to validate cryptographic signatures from the CSP (i.e. private key between CSP and participating CDNs). This requires a relationship between CSP and Downstream CDN. The CDNI metadata specifies CDNs with trust relationships according to the CSP. The set of Downstream CDNs is limited by this criteria.

4. On receipt of a given authorisation request on the CSP portal, the CSP makes a specific authorization decision for this unique request based on its arbitrary distribution policy.

5. If the authorization decision is negative, the CSP rejects the request

6. If the authorization decision is positive, the CSP computes a cryptographic signature that is based on unique parameters of that request and includes it in the URI provided to the end user to request the content.

7. End user sends DNS request to the authoritative CDN.

8. On receipt of the DNS request, the authoritative CDN redirects the request to the Downstream CDN.

9. End user sends DNS request to the Downstream CDN.

10. On receipt of the DNS request, the Downstream CDN responds with IP address of one of its Surrogates.

11. On receipt of the corresponding content request, the Downstream CDN validates the cryptographic signature in the URI using the information provided by the Authoritative CDN in the CDNI Metadata

12. If the validation is negative, the Downstream CDN rejects the request and sends an error code (e.g. 403) in the HTTP response.

13. If the validation is positive, the Downstream CDN serves the request and delivers the content.

14. At a later time, Downstream CDN reports logging events that includes URI signing information.

With DNS-based request routing, URI Signing matches well the general chain of trust model of CDNI when used with asymmetric keys because the only key information that need to be distributed across multiple CDNI hops including non-adjacent hops is the public key, that is

generally not confidential.

With DNS-based request routing, URI Signing does match well the general chain of trust model of CDNI when used with symmetric keys because the symmetric key information needs to be distributed across multiple CDNI hops including non-adjacent hops. This raises a security concern for applicability of URI Signing with Symmetric keys in case of DNS-based inter-CDN request routing.

6. HTTP Adaptive Bit Rate

TBD - HTTP ABR calls for specific support by URI Signing ("flexible URI signing") as discussed in [I-D.brandenburg-cdni-has]. This will be added in a future version of this document.

7. IANA Considerations

This document requests IANA to create a new registry for CDNI URI Signing. The following query string attribute names (a.k.a. keywords) are assigned for the authorization attributes used in CDNI URI Signing. There is no intention to claim any query string attribute for URI beyond the CDNI URI Signing context. That means the entities that sign the URI or validate the URI signature comply to the keywords specified in the query string for the URI Signing function only when URI Signing is used and only in the context of CDNI.

- US (URI signature>
- VER (Version)
- ET (Expiry time)
- CIP (Client IP address)
- KO (Key owner)
- KN (Key ID)
- HF (Hash Function)
- ALG (Algorithm)
- CID (Client ID)

This document requests IANA to create a registry for each of the

defined query string attribute and assign the folowing values for the authorization attribute:

VER: 0 (Base)

HF: "MD5", "SHA1", "SHA256"

ALG: 1 (Full URI), 2 (URI without scheme)

CID: "MAC:<value>", "IMSI:<value>", "MSISDN:<value>", "MEID:<value>", "NAI:<value>" (TBD)

8. Security Considerations

A symmetric key needs to be shared by the entity thats produces the URI signature and the entity that validates the URI signature. In the case of DNS-based request routing, CSP that signed the URI may not have a relationship with the Downstream CDN that validates the signed URI. In this case, the Upstream CDN shall select only the Downstream CDN with a relationship with CSP. Otherwise, asymmetric keys should be used for DNS-based request routing. The Downstream CDN only needs to use the CSP's public key to validate the signed URI. Asymmetric keys method does not require a trust relationship between the two entities participating in URI Signing (i.e. signing function and signature validating function).

For HTTP-based request routing, the two entities participating in URI Signing are always the adjacent Upstream CDN and Downstream CDN because of the hop by hop nature of the redirection. Therefore, either symmetric key or asymmetric keys can be used because the adjacent Upstream CDN and Downstream CDN have a relationship.

The following security threats are identified (TBD):

- Client IP address spoofing
- Illegitimate client behind a NAT
- Replay of request

9. Acknowledgements

TBD

10. References

10.1. Normative References

[I-D.ietf-cdni-framework]
Peterson, L. and B. Davie, "Framework for CDN Interconnection", draft-ietf-cdni-framework-01 (work in progress), July 2012.

[I-D.ietf-cdni-requirements]
Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", draft-ietf-cdni-requirements-03 (work in progress), June 2012.

[I-D.ietf-cdni-use-cases]
Bertrand, G., Emile, S., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", draft-ietf-cdni-use-cases-10 (work in progress), August 2012.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", RFC 6707, September 2012.

10.2. Informative References

[I-D.brandenburg-cdni-has]
Brandenburg, R., Deventer, O., Faucheur, F., and K. Leung, "Models for adaptive-streaming-aware CDN Interconnection", draft-brandenburg-cdni-has-03 (work in progress), July 2012.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

Authors' Addresses

Kent Leung
Cisco Systems
3625 Cisco Way
San Jose 95134
USA

Phone: +1 408 526 5030
Email: kleung@cisco.com

Francois Le Faucheur
Cisco Systems
Greenside, 400 Avenue de Roumanille
Sophia Antipolis 06410
France

Phone: +33 4 97 23 26 19
Email: flefauch@cisco.com

Matt Caulfield
Cisco Systems
1414 Massachusetts Avenue
Boxborough, MA 01719
USA

Phone: +1 978 936 9307
Email: mcaulfie@cisco.com