Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: October 1, 2012

S. Farrell
Trinity College Dublin
D. Kutscher
NEC
C. Dannewitz
University of Paderborn
B. Ohlman
A. Keranen
Ericsson
P. Hallam-Baker
Comodo Group Inc.

March 30, 2012

Naming things with hashes
draft-farrell-decade-ni-01

Abstract

This document defines a set of ways to identify a thing using the output from a hash function, specifying URI, URL and binary formats for these names. The various formats are designed to support, but not require, a strong link to the referenced object such that the referenced object may be authenticated to the same degree as the reference to it.

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 October 1, 2012.

Copyright Notice

Table of Contents

1. Introduction

URIs [RFC3986] are used in various protocols for identifying resources. In many deployments those URIs contain strings that are hash function outputs in order to ensure uniqueness in terms of mapping the URI to a specific resource, or to make URIs hard to guess for security reasons. However, there is no standard way to interpret those strings, and so today in general only the creator of the URI knows how to use the hash function output.

For example, protocols for accessing in-network storage servers (as defined in the IETF DECADE WG) need a way to identify the stored resources uniquely and in a location-independent way so that replicas on different servers can be accessed by the same name. Also, such applications may require verifying that a resource that has been obtained actually corresponds to the name that was used to request the resource, i.e., verifying the name-content binding.

Similarly, in the context of information-centric networking [ref.netinf-design] [ref.ccn] and elsewhere there is value in being able to compare a presented resource against the URI that was de-referenced in order to access that resource. If a cryptographically-strong comparison function can be used then this allows for many forms of in-network storage, without requiring as much trust in the infrastructure used to present the resource. The outputs of hash functions can be used in this manner, if presented in a standard way.

Additional applications might include creating references from web pages delivered over HTTP/TLS; DNS resource records signed using DNSSEC or Data values embedded in certificates, CRLs, OCSP tokens and other signed data objects.

Accordingly, the "ni" URI scheme allows for checking of the integrity of the URI/resource mapping, but it is OPTIONAL for implementations to do so when sending, receiving or processing "ni" URIs.

The URI scheme defined here allows for the use of a query-string, similar to how query-strings are used in HTTP URLs. A companion specification [niexts] describes specific values that can be used in such query strings in for various purposes. That document also specifies additional optional algorithms for truncated hashes and for hashing of dynamic objects.

In addition to the URI form we also define a ".well-known" URL equivalent as well as a binary format for use in protocols that require more compact forms of name and a human-speakable text form that could be used, e.g. for reading out (parts of) the name over a voice connection.

Not all uses of these names require use of the full hash output - truncated hashes can be safely used in some environments. For this reason, we define a new IANA registry for hash functions to be used with this specification.

[[Add a note somewhere that these are sort-of the same as magnet: links. http://en.wikipedia.org/wiki/Magnet_link]]

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Syntax definitions in this memo are specified according to ABNF [RFC4648].

[[Comments are included in double-square brackets, like this.]]

2. ni URI Format

In this section we provide an informal description of the ni URI syntax. An ni URI consists of the following components:

Scheme Name [Required] The scheme name is 'ni'.

Colon and Slashes [Required] The literal "://"

Authority [Optional] The optional authority component may assist applications in accessing the object named by an ni URI. Note that while the ni names with and without an authority differ syntactically, both names will almost always refer to the same object.

One slash [Required] The literal "/"

Digest Algorithm [Required] The name of the digest algorithm, as specified in the IANA registry titled "Data Structure for the Security Suitability of Cryptographic Algorithms registry 'Cryptographic Algorithms'" [RFC5698].

Separator [Required] The literal ";"

Digest Value [Required] The digest value encoded in the specified encoding. The digest value MAY be truncated at a 64 byte boundary.

Query Parameter separator [Optional] '?' The query parameter separator acts a separator between the digest value and the query parameters (if specified).

Query Parameters [Optional] A tag=value list of optional query parameters as are used with HTTP URLs.

It is OPTIONAL for implementations to check the integrity of the URI/resource mapping when sending, receiving or processing "ni" URIs.

When verifying whether two NI URIs refer to same object, an implementation MUST only consider the Digest Algorithm identifier and the Digest Value, i.e., it MUST NOT consider the authority field or any parameters.

The digest value MUST be encoded using base64url [RFC4648] encoding.

The query segment of an URI is NOT hierarchical. Thus escape encoding of slash '/' characters is NOT required. Since application code often attempts to enforce such encoding, decoders MUST recognize the use of URI escape encoding. Section 3.4 of [RFC3986] states that "The characters slash ("/") and question mark ("?") may represent data within the query component."

Consequently no special escaping mechanism is required for the query parameter portion of ni URIs. URI escaping is however frequently imposed automatically by scripting environments. Thus to ensure interoperability, implementations SHOULD NOT generate URIs that employ URI character escaping, and implementations MUST accept any URIs that employ URI character escaping. [[That might need to be more specific.]]

The Named Information URI has the following syntax:

```
niname ="ni://" [ authority ] "/" alg ";" val [ "?" query ]
alg = 1*CHAR
val = 1*CHAR
```

Figure 1: ni Name syntax

The "authority" and "query" types are as in the URI specification. [RFC3986]

Implementations MUST support the sha-256 algorithm as specified in [RFC4055].

Implementations MAY support other algorithms specified in the Data Structure for the Security Suitability of Cryptographic Algorithms

registry 'Cryptographic Algorithms' [RFC5698].

Note that additional algorithms are specified in the companion document to this one [niexts] that implementations can choose to support if they wish. Those algorithms use a different IANA registry defined in that document.

The "val" field MUST contain the output of applying the hash function ("alg") to its defined input, which defaults to the object bytes that are expected to be returned when the URI is de-referenced.

3. URL Format

We define a bidirectional mapping between the ni URI scheme and a subset of the the HTTP scheme that makes use of the .well-known URI [RFC5785] by defining an "ni" suffix (see Section 10).

The HTTP(s) mapping MAY be used in any context where legacy clients without support for ni identifiers is required without loss of interoperability or functionality. A legacy client interprets the ni identifier as an ordinary HTTP(s) URL while a ni aware client can determine the corresponding ni form of the URI and apply ni processing.

Implementations SHOULD support this mapping, in both directions. [[Not sure if we really want 2119 language for the mapping, nor if we need to specify both directions, so this is kind of a placeholder.]]

For an ni name of the form "ni://n-authority/alg;val?query-string" the corresponding HTTP URL produced by this algorithm is "http://h-authority/.well-known/ni/alg/val?query-string". If the ni name has a specified authority then the h-authority MUST have the same value. If the ni name has no authority specified (i.e. the n-authority string is empty), a h-authority value MAY be derived from the application context. For example, if the mapping is being done in the context of a web page then the origin [websec-origin] for that web site can be used. Of course, there are in general no guarantees that the object named by the ni name will be available at the corresponding HTTP URL. But in the case that any data is returned, the retreiver can determine if it is the correct content.

If an application is presented with a HTTP URL with "/.well-known/ni/" as the start of its pathname component, then the reverse mapping to an ni name either including or excluding the authority might produce an ni name that is meaningful depending on the application.

In all of the above the application MAY use the "https" URI scheme if security considerations warrant use of TLS.

[[Might want to add a URL-fragment thing for other HTTP URLs too.]]

4. Binary Format

When a more space-efficient version of the identifier is needed, the identifier can be presented in binary format. The binary format identifier consists of two parts: header and the hash. The header defines how the identifier has been created and the hash part contains a (possibly truncated) result of a one-way hash over the a constant value, identifier header, and the public key. The binary presentation of the identifier is shown in Figure 2.

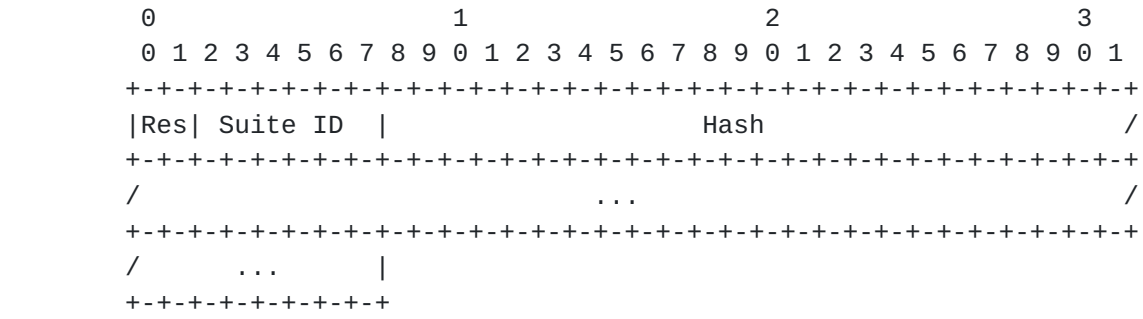

Figure 2: Identifier Format

The Res field is reserved for future use and MUST be set to zero if only Suite IDs defined in this document are used.

The hash algorithm and truncation length is given by the Suite ID. For maintaining efficient encoding for the binary presentation, only few hash algorithms and truncation lengths are supported. The following initial Suite IDs are defined:

ID	Hash algorithm	Truncation	Text
1	SHA-256	120 bits	sha-256-120
2	SHA-3	120 bits	sha-3-120
32	Reserved		

Figure 3: Suite Identifiers

The Suite ID value 32 is reserved for compatibility with ORCHIDs [RFC4843]. The hash algorithm matching to the Suite ID MUST be used for generating the hash. The hash is truncated by taking the 120

last (rightmost) bits of the hash, resulting in total length of 128 bits for the whole identifier. Future suite IDs MAY define different truncating rules and hence different length identifiers.

5. Human-readable Format

Sometimes the identifier may need to be used in a format that is easy for humans to read and possibly communicate, for example, over the phone. For this purpose, the following encoding is RECOMMENDED.

The hash is encoded using base32 encoding [RFC4648] and lower-case alphabets.

The hash is preceded by the the hash algorithm identifier (TBD)

After the hash, there MAY be a checksum; calculated as defined in Section 5.1.

The identifier, hash, and checksum fields are delimited by colon (:) character.

ABNF TBD

5.1. Checksum

When the identifier is communicated using a non-reliable channel, it should include a checksum for detecting errors in the communication. The human readable format checksum MUST be calculated as a crc32 over the parts preceding the checksum, i.e., the algorithm identifier, the delimiter, and the hash value. The result of crc32 is encoded like the hash: with base32 encoding and lower-case alphabets.

6. Public Key Identifiers

When the identifier is calculated from a cryptographic public key, the hash is calculated over constant value (TBD) and the public key in a X.509 SubjectPublicKeyInfo structure (Section 4.1 of [RFC5280]).

[[Need to check why that constant was wanted. Doesn't seem useful and would break DANE compatibility.]]

7. Examples

[[Note: check examples and make sure they're correct sometime.]]

The following digest URI specifies a reference to the text "Hello World !" using the SHA-2 algorithm with 256 bit output and no authority field:

ni:///sha-256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

And the same example shown with an authority would be:

ni://example.com/sha-256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

The following HTTP URL represents a mapping from the previous ni name based on the algorithm outlined above.

http://example.com/.well-known/ni/sha-256/
B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

8. Security Considerations

No secret information is required to generate or verify an ni URI. Therefore an ni URI only provides a proof of integrity for the referenced object and the proof of integrity provided is only as good as the proof of integrity for the ni URI. In other words, the digest value can provide name-data integrity binding the ni name value to the object bytes returned when the ni name is de-referenced using some protocol.

Disclosure of an ni URI value does not necessarily entail disclosure of the referenced object but may enable an attacker to determine the contents of the referenced object by reference to a search engine or other data repository or, for highly formatted object with little variation, by simply guessing the value and checking if the digest value matches.

The integrity of the referenced content would be compromised if a weak digest were used.

If a truncated digest is used, certain security properties MAY be affected. In general a digest algorithm is designed to produce sufficient bits to prevent a 'birthday attack' collision occurring. To ensure that the difficulty of discovering two pieces of content that result in the same digest with a work factor O(2^x) by brute force requires a digest length of 2x. Many security applications only require protection against a 2nd pre-image attack which only requires a digest length of x to achieve the same work factor.

[[Don't reduce too much, and don't rely on a digest that has been truncated as being the strength of the original digest alg.]]

9. Acknowledgements

This work has been supported by the EU FP7 project SAIL. The authors would like to thank SAIL participants to our naming discussions, especially Jean-Francois Peltier, for their input.

The authors would also like to thank Bob Moskowitz, Tero Kivinen, Zach Shelby, Carsten Bormann, David McGrew, Eric Rescorla, and Tobias Heer for their comments and input to the document.

[[Mention folk on the WebSec list who contributed to the discussions]]

10. IANA Considerations

IANA is requested to create a new registry for (TBD) Suite IDs. Initial values are given in Section 4, future assignments are to be made through IETF Review or IESG Approval [RFC5226].

10.1. Assignment of Network Information (ni) URI Scheme

The procedures for registration of a URI scheme are specified in RFC 4395 [RFC4395]. The following is the proposed assignment template.

URI scheme name: ni

Status: Permanent

URI scheme syntax. See Section 2

URI scheme semantics. See Section 2

Encoding considerations. See Section 2

Applications/protocols that use this URI scheme name: General applicability with initial use cases provided by WEBSEC and DECADE

Interoperability considerations: TBS

Security considerations: See Section 8

Contact: TBD

Author/Change controller: IETF

References: As specified in this document

10.2. Assignment of Well Known URI prefix ni

The procedures for registration of a Well Known URI entry are specified in RFC 5785 [RFC5785]. The following is the proposed assignment template.

URI suffix: ni

Change controller: IETF

Specification document(s): This document

Related information: None

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

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

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[RFC4395] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC4843] Nikander, P., Laganier, J., and F. Dupont, "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID)", RFC 4843, April 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S.,

Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5698] Kunz, T., Okunick, S., and U. Pordesch, "Data Structure for the Security Suitability of Cryptographic Algorithms (DSSC)", RFC 5698, November 2009.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, April 2010.

11.2. Informative References

[niexts] Hallam-Baker, P., Stradling, R., Farrell, S., Kutscher, C., and B. Ohlman, "The Network Information (ni) URI Scheme: Parameters", draft-hallambaker-decade-ni-params-00 (work in progress), October 2011.

[ref.ccn] Jacobsen, K, D, F, H, and L, "Networking Named Content", CoNEXT 2009 , December 2009.

[ref.netinf-design] Ahlgren, D'Ambrosio, Dannewitz, Marchisio, Marsh, Ohlman, Pentikousis, Rembarz, Strandberg, and Vercellone, "Design Considerations for a Network of Information", Re-Arch 2008 Workshop , December 2008.

[websec-origin] Barth, A., "The Web Origin Concept", draft-ietf-websec-origin-06 (work in progress), October 2011.

Authors' Addresses

Stephen Farrell
Trinity College Dublin
Dublin, 2
Ireland

Phone: +353-1-896-2354
Email: stephen.farrell@cs.tcd.ie

Dirk Kutscher
NEC
Kurfuersten-Anlage 36
Heidelberg,
Germany

Phone:
Email: kutscher@neclab.eu

Christian Dannewitz
University of Paderborn
Paderborn
Germany

Email: cdannewitz@upb.de

Borje Ohlman
Ericsson
Stockholm S-16480
Sweden

Email: Borje.Ohlman@ericsson.com

Ari Keranen
Ericsson
Jorvas 02420
Finland

Email: ari.keranen@ericsson.com

Phillip Hallam-Baker
Comodo Group Inc.

Email: philliph@comodo.com