A URN Namespace For Identifiers Based on Cryptographic Hashes
draft-thiemann-hash-urn-01.txt

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 4, 2004.

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

This document describes a URN namespace to identify immutable, typed resources using content-based unique identifiers. The naming scheme relies on an algorithm that computes identifiers from media types and cryptographic hashes without a central authority.

1. Conventions used in this document

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as defined in "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].
2. Introduction

A URN serves as a unique name for a resource [RFC1630]. Most URN namespaces involve a central authority to ensure uniqueness of assigned names. This approach has its merits but it requires organizational structures for processing requests for naming and for bookkeeping about used names. Thus, acquiring a URN becomes an involved task not to be undertaken on a day-to-day basis.

A URN namespace based on cryptographic hashes enables using and creating URNs on a day-to-day basis for storing and retrieving immutable resources. It relies on a decentralized, algorithmic assignment of identifiers by exploiting the uniqueness guarantees of (cryptographic) hashes. This document contains the assignment algorithm so that everyone can generate identifiers in this namespace.

The namespace provides identifiers for typed resources with application/octet-stream as a default type.

This namespace specification is for a formal namespace. The specification adheres to the guidelines given in "Uniform Resource Names (URN) Namespace Definition Mechanisms" [RFC3406].

3. Specification Template

Namespace ID:

"hash" requested.

Registration Information:

Registration Version Number: 1

Registration Date: 2003-09-??

Declared registrant of the namespace:

The CBUID Project
Institut fuer Informatik
Universitaet Freiburg
Georges-Koehler-Allee 079
D-79110 Freiburg
Germany

Contact:
Peter Thiemann
info@cbuid.org
Declaration of syntactic structure:

The Namespace Specific Strings (NSS) of all URNs assigned by the schema described in this document will conform to the syntax defined in section 2.2 of RFC2141 [RFC2141]. The formal syntax of the NSS is defined by the following normative ABNF [RFC2234] rules for <hash-nss>:

```
hash-nss    = [media-type] ":" [hash-scheme] ":" hash-value
hash-scheme = "md5" / "sha1" / "sha256" / "sha384" / "sha512"
hash-value  = 1*(ALPHA / DIGIT / ".")
```

The following are comments and restrictions not captured by the above grammar.

A <media-type> is any MIME media type [RFC2046] which is registered in the appropriate IANA registry [IANA-MT]. There is no default for the <media-type> specification. If omitted, then the media type is unspecified, thus leaving the application complete freedom to interpret the resource.

If the <hash-scheme> specification is omitted, then the length of the <hash-value> unambiguously selects one of "sha1", "sha256", "sha384", or "sha512" according to the following table.

<table>
<thead>
<tr>
<th>length of &lt;hash-value&gt;</th>
<th>implied &lt;hash-scheme&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>&quot;sha1&quot;</td>
</tr>
<tr>
<td>56</td>
<td>&quot;sha256&quot;</td>
</tr>
<tr>
<td>80</td>
<td>&quot;sha384&quot;</td>
</tr>
<tr>
<td>104</td>
<td>&quot;sha512&quot;</td>
</tr>
</tbody>
</table>

A <hash-value> is a non-empty sequence of characters encoding a sequence of bits which must be a valid hash for the specified hash-scheme. The encoding depends on the <hash-scheme>. If <hash-scheme> is "md5", then <hash-value> is the base16 encoding [RFC3548] of the 16 octets of the MD5 hash value of the resource (most significant octet first) so that the <hash-value> consists of 32 HEXDIG. If <hash-scheme> is "sha1", then <hash-value> is the base32 encoding [RFC3548] of the 20 octets of the SHA1 hash value of the resource (most significant octet first) so that the <hash-value> consists of 32 BASE32DIG. The other "sha" <hash-value>s are handled analogously according to the above table.

In any case, the <hash-value> MUST provide the correct number of bits for the chosen <hash-scheme>, 128 for "sha1", 256 for
"sha256", 384 for "sha384", and 512 for "sha512".

Examples:

urn:hash::md5:5307d294b6cc9854f2deed8c1628b72

urn:hash::sha1:LBPI666ED2QSWVD3VS05BG5R54TE22QL

urn:hash:::JRBFASJWGY3EKRSKJVF0VSEGMLFGETZVIJDKTURVGRKEKMRSGFA====

The implied <hash-scheme> for this identifier is "sha256" since the <hash-value> consists of 56 BASE32DIG and specifies 256 bits.

urn:hash:text/plain::LBPI666ED2QSWVD3VS05BG5R54TE22QL

The implied <hash-scheme> for this identifier is "sha1" since the <hash-value> consists of 32 BASE32DIG and specifies 160 bits.

urn:hash:message/rfc822:md5:5307d294b6cc9854f2deed8c1628b72

Relevant ancillary documentation:

None as yet.

Identifier uniqueness considerations:

Each identifier contains a cryptographic hash value for the referenced resource. The probability that two different resources have the same hash value depends on the hash function. For the MD5 hash where the hash value has 128 bits, it is conjectured [RFC1321] that the probability of a collision is in the order of 1/2^64 by reasoning with the birthday attack. For the sha1 hash where the hash value has 160 bits, the same attach yields a probability of 1/2^80 for a collision.

Identifier persistence considerations:

The binding between the identifier and the referenced resource is permanently established by the assignment algorithm that computes the identifier from the resource.

The persistence of an identifier for some resource A might be compromised by coming up with a different resource B with the same identifier. However, this corresponds to solving the "second preimage problem" for either the MD5 algorithm or an algorithm of the SHA family. This problem turns out to be much
harder than just producing a collision. In fact, the handbook of applied cryptography [HAC] estimates that computing a second preimage takes on the order of $2^{128}$ steps for MD5 and $2^{160}$ steps for SHA1.

Process of identifier assignment:

Assignment is completely open, following the algorithm below.

The inputs of the algorithm are
- the name <hash-scheme> of a hash function
- a media type for <media-type>
- a resource (a sequence of octets)

The algorithm applies the hash function to the resource, converts the resulting bit sequence into a valid <hash-value> according to the <hash-scheme>, and constructs the URN by concatenating the <media-type>, the <hash-scheme>, and the <hash-value> using the syntax described above. Algorithms for computing the hash functions mentioned in this document are defined in the following references:

- md5  [RFC1321]
- sha1  [RFC3174]
- sha256 [FIPS180-2]
- sha384 [FIPS180-2]
- sha512 [FIPS180-2]

The conversion of a <hash-value> to a string in base16 encoding proceeds as follows. The bits in the <hash-value> are converted from most significant to least significant bit, four bits at a time to their ASCII presentation. Each sequence of four bits is represented by its hexadecimal digit from "0123456789abcdef". That is, binary 0000 gets represented by the character '0', 0001, by '1', and so on up to the representation of 1111 as 'f'.

The conversion of a <hash-value> to a string in base32 encoding proceeds as follows. The bits in the <hash-value> are converted from most significant to least significant bit, five bits at a time to their ASCII presentation. Each sequence of five bits is represented by its base32 digit from "abcdefghijklmnopqrstuvwxyz234567" as defined in [RFC3548]. That is, binary 00000 gets represented by the character 'a', 00001, by 'b', and so on up to the representation of 11111 as '7'. A value that does not consist of a number of bits which is divisible by five is padded with zero bits to the next multiple of five. The length of a base32 encoded bit string is always
divisible by eight. Padding of an incomplete 8 character group
is done using the character '='.

Process of identifier resolution:

Not specified.

Rules for Lexical Equivalence:

Lexical equivalence is identity after normalization. An
identifier in the cbuid URN namespace is normalized by
converting all characters to lower case.

Conformance with URN Syntax:

There are no additional characters reserved.

Validation mechanism:

Each identifier in the namespace MUST conform with the syntax
specified above.

Scope:

The namespace is global and public.

4. IANA Considerations

This document includes a URN namespace registration that is to be
entered into the IANA registry for URN NIDs.

5. Namespace Considerations

Many URN namespaces are assigned to organizations and rely on a
centralized registry to achieve uniqueness and persistency. In
contrast, the hash namespace is not tied to any organization.
Assignment of identifiers can be performed and verified individually,
while uniqueness is still preserved (with a probability close to 1).

The hard coding of the hashing schemes into the namespace definition
is intentional. This is because a valid identifier should be able to
act as a proxy for the the named resource. That way, metainformation
of descriptive or authoritative nature (such as endorsements,
signatures, etc) can be attached to the identifier and need not be
bundled with the actual resource. Such a proxy functionality is only
guaranteed as long as the underlying hashing scheme is not
compromised, that is, as long as no collisions are found.
The encoding of the hash value is also hard coded into the definition. We have chosen not to make the encoding an additional parameter of the URN scheme for two reasons:

1. it would make identifier normalization non-trivial;
2. each hashing scheme has a standard encoding, which should be reflected in the identifier.

One problem is the phasing out of compromised hash schemes. For instance, many believe that MD5 is "not sufficiently secure" on the grounds that it only provides 128 bit hashes and that colliding inputs have been constructed. However, the only known approach for solving the second preimage problem, which appears to be more relevant for the application as an identifier, is brute force search through on the order of $2^{128}$ inputs.

If a procedure for computing a second preimage in significantly fewer operations is ever published, then resolvers SHOULD refuse to resolve the compromised hash scheme. This is in line with the semantics of URNs which need to identify a resource uniquely but the resource need not be available forever (cf. the discussion in BCP 66 [RFC3406]).

6. Community Considerations

Similar URNs are in use in peer-to-peer file transfer systems. Most of them do not include a mediatype, although this practice can provide extra guarantees. For example, a provider of metainformation can state that mediatype of the resource has been verified by including the mediatype in the published URN. For many formats, the mediatype provides an additional self-verifiable attribute.

Some URI schemes in common use may be easily derived from the hash scheme.

1. The sha1 scheme
   
   urn:sha1:<sha1-hash-value>

   is equivalent to

   urn:hash::sha1:<sha1-hash>

   and even to

   urn:hash::::<sha1-hash>

2. Another proposed scheme is based on the data URL
7. Security Considerations

The use of the namespace per se does have security implications. However, it should be kept in mind that the uniqueness guarantee given by cryptographic hashes is only probabilistic and that no known procedure (save bitwise comparison) can provide a 100% guarantee of the identity of the hashed resource.

Normative References


Informational References


Contributors

Stephanie Kollenz
Matthias Neubauer

Author's Address

Peter Thiemann
Institut fuer Informatik
Universitaet Freiburg
Georges-Koehler-Allee 079
D-79110 Freiburg
Germany

Phone: +49 761 203 8051
EMail: thiemann@acm.org
URL: http://www.informatik.uni-freiburg.de/~thiemann

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice