The DIGEST URI Scheme
draft-hallambaker-digesturi-00

Abstract

A URI scheme for referencing static data objects by means of a cryptographic digest mechanism is specified. The format is designed to resist content type substitution attacks and supports a choice of digest algorithms.

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 March 30, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Definitions
   1.1. Requirements Language
   1.2. Defined Terms
2. The DIGEST URI Type.
   2.1. The DIGEST URI TYPE
   2.2. Use in binary formats.
3. Object Digest Value Specifier
   3.1. Example: CA Certificate A
   3.2. Example: Text File
4. Security Considerations
   4.1. Integrity
   4.2. Confidentiality
   4.3. Weak Digest Algorithm
5. IANA Considerations
6. References
   6.1. Normative References
   6.2. Non Normative References
Appendix A. Example Certificates
   A.1. CA Certificate A
1. Definitions

1.1. Requirements Language

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].

1.2. Defined Terms

The following terms are used in this document:

Abstract Syntax Notation One (ASN.1)
A notation for describing abstract types and values, as specified in X.680 [X.680].

2. The DIGEST URI Type.

Provides a strong reference to a static data object.

Does not provide a means of resolution.

Allows an authenticated data source to provide an authenticated reference to a static data object.

Intended applications include creating references from

Web pages delivered over HTTP/TLS

DNS resource records signed using DNSSEC

Data values embedded in certificates, CRLs, OCSP tokens and other signed data objects.

2.1. The DIGEST URI TYPE

The DIGEST URI Type has the following format:

DIGEST:< Base64 (Object Digest Value Specifier) >

2.2. Use in binary formats.

The URI encoding of the Object Digest Value Specifier is compatible with ASCII encoding formats and MAY be used in any situation where a URI is specified.

In a binary format such as an ASN.1 signed object, a direct encoding of the data without the BASE64 encoding MAY be more convenient.
3. Object Digest Value Specifier

An Object Digest is an ASN.1 structure with three components:

- An ASN.1 Object Identifier specifying the object type of the referenced object
- An ASN.1 Object Identifier specifying the digest algorithm.
- Either:
  - An ASCII MIME Content type specifier.
  - An ASN.1 [X.690] encoded data field containing the digest value of the referenced object processed using the specified digest algorithm.

The ASN.1 structure is defined by the following schema:

```
DIGESTURI DEFINITIONS ::= 
BEGIN
ObjectDigestIdentifier ::= SEQUENCE {
  CHOICE {
    OIDtype        OBJECT IDENTIFIER,
    MIMEtype       IA5String
  }
  digestAlgorithm  OBJECT IDENTIFIER,
  digest           OCTET STRING
}
END
```

The Object Digest Identifier construction is designed to facilitate implementation in applications that already require ASN.1 handling mechanisms (i.e. most cryptographic applications) without causing an undue coding burden in cases where ASN.1 code is not already supported. Appendix C provides all the necessary information to create a fully compliant Object Digest Identifier implementation.

3.1. Example: CA Certificate A

The ODI of CA Certificate A (specified in Appendix B.1) is calculated as follows:

- ASN.1 Sequence tag: 3032
- ASN.1 OID id-at-cACertificate (2.5.4.37): 0603550425
- ASN.1 OID sha256 (2.16.840.1.101.3.4.2.1): 0609608648016503040201
- SHA-256 Digest Value: 042017cc980f6a84fb15e5da3f32afea62360f4ca29627feed68739a13062defe804

The DIGEST URI is DIGEST:MDIGA1UEjQYJYIZIAWUDBAIIBCAIXJgPaoT7FeXaPzKv6mI2D0yililf+7WhzmpMGLe/oBA==.

3.2. Example: Text File

The Digest URI of the text file "Hello World" is computed as follows:

- ASN.1 Sequence tag 3039
- ASN.1 IA5String 'text/plain' 160A746578742f706c61696e
- ASN.1 OID 'SHA-256' 0609608648016503040201
- SHA-256 Digest Value
4. Security Considerations

4.1. Integrity

No secret information is required to generate a DIGEST URI. Therefore a DIGEST 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 DIGEST URI value.

4.2. Confidentiality

Disclosure of a DIGEST 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.

4.3. Weak Digest Algorithm

[The digest algorithm MUST be strong]
[For most use cases collision resistance is a requirement]

5. IANA Considerations

[Assign the DIGEST URI type.]

6. References

6.1. Normative References


6.2. Non Normative References

[NIST-]
Appendix A. Example Certificates

The following certificates are used in the examples.

A.1. CA Certificate A

CA Certificate A is a self signed certificate signed with a 2048 bit RSA key:

```
-----BEGIN CERTIFICATE-----
MIIADTCAMgYGB Advisory 1 BAM TCM 4 YwDwYK qe EwHcNMTA xMTExMTgx MjAz WhcNMjAx MTA4
MTgxMjAzWjYoMREwDwYDVQKewHYB211IEiUyZETMMEGA1UEAxMKRXhhBxBsZSBDBQC
ARSwCwC1YhvcNAsAEBAADIBAwQgEoIAALBALHvos3yEe8u96Ese2rPATXAPbYVG6BM2GTLK COS6MZa9C6ZPf1eZ/TZ/EG1KBwRJ1LIVxWdKwJZ7GBY7+fMDZ2
7zkx66U4C+Cam98Nrjdugxox18hbscIBxhChERgamp0U/fh8HoWv8ExAYLZ1xGm
bM7mVknvolN9Eeg1EmYN+qHexGJp1WFMr4NKvHAATET6B1a95CPmOOGw9p9vQic
SJC6CDAhKaa7725+MqN9D57H085Rlih52XZMmJVFicxPb2K7CVWht4t5G6856f6
SxXvWDRyc61Um0xZBdDVIoU8pAXESTdM0WvXKl8ZUaBoZ7/YnSSTse1iW65C
AwEAe8A9M9dsw5yDyVR90PABQAQQEBAQDaEAMBA1u6EDwEBAQQFMAQFQEw6AYDVR0g
BBEwDzANz8rBgE8AYKUTYUAATABLaSkqkhkJ6w9QBAUDggEABGcN6xaQqxyJl95Y5e
Ps+XEYDK1iwVmwNnRfbdUzUqWpaQPcJcScHzeMe91CUZiGDN3JYwWwhiUrQAAKmgrq
Zga4F9y0pCFHABPHXKgeM1h7MrA7Mw9Khudo4FHhrb2Z00n4DDb1itBFNas5t/Cb
6McCrwpB5h1jA537u+G6vF7mage859WvIjn1y59NcfCQ6YJ6A5VknrV695VnL
JSFpN9vSSNRh6ejQpU9tcbqPtAEP3Mju7d1x8B3Q4f9e1kYxxc/dbJ3K1Q8F4nc
Jc4NBz1akvF+QbcbzEgkhMDk1vxJvD8Jt6+4FZrsRe16bsGY4DQhPv650Ivbr3uw
329dd+g=
-----END CERTIFICATE-----
```

In binary form, the certificate data is:

```
0000 30 82 03 01 30 82 01 eb a9 03 02 01 02 01 01 00
0010 30 6b 06 0b 01 2a 08 46 86 86 f7 0d 01 01 05 30 28 31
0020 11 30 0f 06 09 2a 86 48 86 f7 0d 01 01 03 82 01 1f 30 0b 06 03 55 04 0a 0a 13 08 41 63 6d 65 20 49
0030 6e 31 13 11 06 03 55 04 03 13 0a 01 45 78 61 6d 70 6c 65 20 43 41 30 82 01 00 0e 30 0f 06 03 55 1d 0f 01 01 04 04 03 02 00 04 30 03 01 01 01 30 18 06 03 55 1d 20
0050 30 6b 06 0b 01 2a 08 46 86 86 f7 0d 01 01 05 30 28 31
0060 11 30 0f 06 09 2a 86 48 86 f7 0d 01 01 03 82 01 1f 30 0b 06 03 55 04 0a 0a 13 08 41 63 6d 65 20 49
0070 6e 31 13 11 06 03 55 04 03 13 0a 01 45 78 61 6d 70 6c 65 20 43 41 30 82 01 00 0e 30 0f 06 03 55 1d 0f 01 01 04 04 03 02 00 04 30 03 01 01 01 30 18 06 03 55 1d 20
0080 6b 06 0b 01 2a 08 46 86 86 f7 0d 01 01 05 30 28 31
0090 11 30 0f 06 09 2a 86 48 86 f7 0d 01 01 03 82 01 1f 30 0b 06 03 55 04 0a 0a 13 08 41 63 6d 65 20 49
00a0 6e 31 13 11 06 03 55 04 03 13 0a 01 45 78 61 6d 70 6c 65 20 43 41 30 82 01 00 0e 30 0f 06 03 55 1d 0f 01 01 04 04 03 02 00 04 30 03 01 01 01 30 18 06 03 55 1d 20
00b0 11 30 0f 06 09 2a 86 48 86 f7 0d 01 01 03 82 01 1f 30 0b 06 03 55 04 0a 0a 13 08 41 63 6d 65 20 49
00c0 6e 31 13 11 06 03 55 04 03 13 0a 01 45 78 61 6d 70 6c 65 20 43 41 30 82 01 00 0e 30 0f 06 03 55 1d 0f 01 01 04 04 03 02 00 04 30 03 01 01 01 30 18 06 03 55 1d 20
00d0 11 30 0f 06 09 2a 86 48 86 f7 0d 01 01 03 82 01 1f 30 0b 06 03 55 04 0a 0a 13 08 41 63 6d 65 20 49
00e0 6e 31 13 11 06 03 55 04 03 13 0a 01 45 78 61 6d 70 6c 65 20 43 41 30 82 01 00 0e 30 0f 06 03 55 1d 0f 01 01 04 04 03 02 00 04 30 03 01 01 01 30 18 06 03 55 1d 20
```
The SHA-256 digest of the certificate data is:

```
17cc980f6a84fb15e5da3f32afea62360f4ca29627feed68739a13062defe804
```
Use of the SHA-1 digest algorithm is not recommended due to concerns for the security of the algorithm.

**B.4. DER Data Encoding Prefixes**

The rules of ASN.1 encoding state that every data value is preceded by a data type identifier and a length identifier. In the case of an Object Digest Identifier the data type identifier is always OCTET STRING (04) and the length for all currently defined digest algorithms will be less than 128 bytes (1024 bits) and thus use the single byte encoding form in which bit 7 is set to 0 and the lower 7 bits specify the length.

The length prefixes for commonly used digest lengths in hexadecimal notation are thus:

- 160 bits: 04 14
- 224 bits: 04 1C
- 256 bits: 04 20
- 384 bits: 04 30
- 512 bits: 04 40

**Authors’ Addresses**

Phillip Hallam-Baker  
Comodo Group Inc.  
Email: philliph@comodo.com

Rob Stradling  
Comodo CA Ltd.  
Email: rob.stradling@comodo.com