HTTPAuth Working Group
Internet-Draft
Updates: 2617 (if approved)
Intended Status: Standards Track
Expires: March 6, 2014

R. Shekh-Yusef
D. Ahrens
Avaya

September 2, 2013

HTTP Digest Update
draft-ietf-httpauth-digest-update-05

Abstract

This documents specifies extensions to the HTTP Digest Authentication mechanism to add support for new digest algorithms to the HTTP Digest Access Authentication scheme.

This document also defines an extension to the HTTP Digest Authentication mechanism to allow the server to indicate its support for the UTF-8 character encoding scheme.

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction

This document specifies extensions to the HTTP Digest Access Authentication scheme by adding support for SHA2-256 [FIPS 180-3] and SHA2-512/256 [FIPS 180-3] hash algorithms. RFC2617 specifies the MD5 algorithm as the default hash algorithm used in the digest access authentication scheme. Since RFC2617 was first proposed, the MD5 algorithm has been broken. In 2008 the US-CERT issued a note that MD5 "should be considered cryptographically broken and unsuitable for further use" [CERT-VU].

RFC2617 does not define how to treat Unicode characters [UNICODE] outside the ASCII range [RFC20] with the "Digest" scheme. This document also defines an extension to the HTTP Digest Authentication mechanism to allow the server to indicate its support for the UTF-8 character encoding scheme.

1.1 Terminology

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 RFC2119 [RFC2119].

2 Syntax Convention

In the interest of clarity and readability, the extended parameters or the headers and parameters in the examples in this document might be broken into multiple lines. Any line that is indented in this document is a continuation of the preceding line.

3 Digest Access Authentication Scheme

The Digest Access Authentication scheme is based on a simple challenge-response paradigm. The Digest scheme challenges using a nonce value. A valid response contains a checksum of the username, the password, the given nonce value and the requested URI. In this way the password is never sent in the clear.

3.1 Introduction

3.1.1 Representation of digest values

An optional header allows the server to specify the algorithm used to create the checksum or digest. By default the SHA2-256 algorithm is used, with SHA2-512/256 being used as a backup algorithm. To maintain backwards compatibility, the MD5 algorithm is still supported but not recommended.

The size of the digest depends on the algorithm used. The bits in the digest are converted from the most significant to the least significant bit, four bits at a time to the ASCII representation as follows. Each four bits is represented by its familiar hexadecimal notation from the characters 0123456789abcdef, that is binary 0000 is represented by the character '0', 0001 by '1' and so on up to the representation of 1111 as 'f'. If the MD5 algorithm is used to calculate the digest, then the digest will be represented as 32 hexadecimal characters, SHA2-256 and SHA2-512/256 by 64 hexadecimal characters.

3.1.2 Limitations

The Digest authentication scheme suffers from many known limitations as specified in RFC2617, section 3.1.4. The update in this document does not address those limitations.

HTTP Digest authentication, when used with human-memorable passwords, is vulnerable to dictionary attacks. Such attacks are much easier than cryptographic attacks on any widely used algorithm, including those that are no longer considered secure. In other words, algorithm agility does not make this usage any more secure.

As a result, Digest authentication SHOULD be used only with passwords that have a reasonable amount of entropy, e.g. 128-bit or more. Such passwords typically cannot be memorized by humans but can be used for automated web services.

It is recommended that Digest authentication be used over a secure channel like HTTPS.

3.2 Specification of Digest Headers

The modifications to the formats of the WWW-Authenticate Header line and the Authorization header line are specified below.

3.2.1 The WWW-Authenticate Response Header

If a server receives a request for an access protected object, and an acceptable Authorization header is not sent, the server responds with a "401 Unauthorized" status code, and a WWW-Authenticate header. The server MAY include multiple WWW-Authenticate headers to allow the server to utilize the best available algorithm supported by the client.

The algorithm directive is extended as follows:

```
algorithm = "algorithm" "=" (
                  "MD5" | "MD5-sess" |
                  "SHA2-256" | "SHA2-256-sess" |
                  "SHA2-512-256" | "SHA2-512-256-sess" |
                  token)
```

Algorithm

A string indicating a pair of algorithms used to produce the digest and a checksum. If the algorithm parameter is not present it is assumed to be "MD5" to maintain backwards compatibility with existing implementations. If the algorithm is not understood, the challenge should be ignored and a different challenge used if there is more than one.

The string obtained by applying the digest algorithm to the data "data" with "secret" will be denoted KD(secret, data) and the string obtained by applying the checksum algorithm to the data "data" will be denoted H(data). The notation unq(x) means the value of the quoted string X without the surrounding quotes.

```
For the  "MD5 and "MD5-sess" algorithms
    H(data) = MD5(data)

For the "SHA2-256" and "SHA2-256-sess" algorithms
    H(data) = SHA2-256(data)

For the "SHA2-512-256" and "SHA2-512-256-sess" algorithms
    H(data) = SHA2-512-256(data)
```

and

```
    KD(secret, data) = H(concat(secret, ":", data))
```

i.e the digest is the hash of the secret concatenated with a colon concatenated with the data. The " -sess" algorithm is intended to allow efficient 3rd party authentication servers; for the difference in usage see the description in section RFC2617, Section 3.2.2.2.

3.2.2 The Authorization Request Header

The client is expected to retry the request, passing an Authorization Request Header line. The Authorization Request Header line is modified as follows:

```
request-digest       = <"> digest-size LHEX <">
digest-size       = "32" | "64"
```

The values of the opaque and algorithm fields must match those supplied in the WWW-Authenticate response header for the entity being requested.

response
    A string of hex digits as defined in RFC2617 which proves that the user knows a password.

3.3 Digest Operation

The modifications specified in this document does not introduce any change to the digest operation specified in RFC2617.

3.4 Security Protocol Operation

When a server receives a request to access a resource, the server might challenge the client by responding with "401 Unauthorized" status code, and include one or more WWW-Authenticate headers. If the server challenges with multiple Digest headers, then each one of these headers MUST use a different digest algorithm. The server MUST add these Digest headers to the response in order of preference, starting with the most preferred header, followed by the less preferred headers.

This specification defines the following preference list, starting with the most preferred algorithm:

* SHA2-256 as the default algorithm.
* SHA2-512/256 as a backup algorithm.
* MD5 for backward compatibility.

A future version of this document might add SHA3 [SHA3] as a backup algorithm, once its definition has been finalized and published.

When the client receives the response it SHOULD use the topmost header that it supports, unless a local policy dictates otherwise. The client should ignore any challenge it does not understand.

NOTE: There is some concern around the support for the SHA2-512/256 algorithm in the common implementation of SHA2.

3.5 Example

The following example is borrowed from RFC2617 and assumes that an access protected document is being requested from the server via a GET request. The URI of the document is http://www.nowhere.org/dir/index.html". Both client and server know that the username for this document is "Mufasa" and the password is "Circle of Life" (with one space between each of the three words).

The first time the client requests the document, no Authorization header is sent, so the server responds with:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest
        realm = "testrealm@host.com",
        qop="auth, auth-int",
        algorithm="SHA2-256",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
WWW-Authenticate: Digest
        realm="testrealm@host.com",
        qop="auth, auth-int",
        algorithm="MD5",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40ef41"
```

The client may prompt the user for their username and password, after which it will respond with a new request, including the following Authorization header if the client chooses MD5 digest:

```
        Authorization:Digest username="Mufasa",
                realm="testrealm@host.com",
                nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
                uri="/dir/index.html",
                qop="auth",
                algorithm="MD5",
                nc=00000001,
                cnonce="0a4f113b",
                response="6629fae49393a05397450978507c4ef1",
                opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

If the client chooses to use the SHA2-256 algorithm for calculating the response, the client responds with a new request including the following Authorization header:

```
        Authorization:Digest username="Mufasa",
                realm="testrealm@host.com",
                nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
                uri="/dir/index.html",
                qop="auth",
                algorithm="SHA2-256",
                nc=00000001,
                cnonce="0a4f113b",
                response="5abdd07184ba512a22c53f41470e5eea7dcaa3a93
                                a59b630c13dfe0a5dc6e38b",
                opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

4 Internationalization

The "Digest" mechanism allows for new parameters to be defined and used with Authenticate and Authorization requests. This document defines a new optional "charset" auth-param that could be used by the server to indicate the encoding scheme it supports.

In challenges, servers MAY use the "charset" authentication parameter (case-insensitive) to express the character encoding they expect the user agent to use.

The only allowed value is "UTF-8", to be matched case-insensitively, indicating that the server expects the UTF-8 character encoding to be used ([RFC3629]).

5 Security Considerations

This specification updates the Digest Access Authentication scheme specified in RFC2617 to add support for the SHA2-256 and SHA2-512/256 hash algorithms. Support for these additional hash algorithms does not alter the security properties of the Digest Access Authentication scheme.

6 Acknowledgments

The authors would like to thank Geoff Baskwill and Eric Cooper for their careful review and comments on the pre published version of this document.

The authors would also like to thank Stephen Farrell, Yoav Nir, Phillip Hallam-Baker, Manu Sporny, Paul Hoffman, Julian Reschke, and Sean Turner for their careful review and comments on and off the mailing list.

Special thanks to Yaron Sheffer for his thorough review, comments on and off the list, and for the text he provided for the limitation section.

7 References

7.1 Normative References

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

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC6365] Hoffman, P., Klensin, J., "Terminology Used in Internationalization in the IETF", BCP: 166, RFC 6365, September 2011.

[UNICODE] The Unicode Consortium, "The Unicode Standard, Version 6.0". <http://www.unicode.org/versions/Unicode6.0.0/>.

[RFC20] Cerf, V., "ASCII format for Network Interchange", RFC 20, October 1969.

7.2 Informative References

[FIPS180-3] National Institute of Standards and Technology (NIST), FIPS Publication 180-3: Digital Signature Standard, June 2009.

[CERT-VU] Vulnerability Note VU#836068, "MD5 vulnerable to collision attacks", December 2008.

[SHA3] National Institute of Standards and Technology (NIST), "CRYPTOGRAPHIC HASH AND SHA-3 STANDARD DEVELOPMENT". http://csrc.nist.gov/groups/ST/hash/index.html

8 Authors' Addresses

Rifaat Shekh-Yusef
Avaya
250 Sydney Street
Belleville, Ontario
Canada

Phone: +1-613-967-5267
Email: rifaat.ietf@gmail.com

David Ahrens
Avaya

EMail: ahrensdc@gmail.com