SIP Core
Internet-Draft
Updates: 3261 (if approved)
Intended status: Standards Track
Expires: November 10, 2019

R. Shekh-Yusef
Avaya

May 9, 2019

The Session Initiation Protocol (SIP) Digest Authentication Scheme draft-ietf-sipcore-digest-scheme-02

Abstract

This document updates the Digest Access Authentication scheme used by the Session Initiation Protocol (SIP) to add support for secure digest algorithms to replace the broken MD5 algorithm.

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 https://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 November 10, 2019.

Table of Contents

1. Introduction

The SIP protocol [RFC3261] uses the same mechanism used by the HTTP protocol for authenticating users, which is a simple challenge-response authentication mechanism that allows a server to challenge a client request and allows a client to provide authentication information in response to that challenge.

The SIP protocol uses the Digest Authentication scheme that is used with the HTTP authentication mechanism, which by default uses MD5 as the default algorithm.

The HTTP Digest Access Authentication [RFC7616] document defines the Digest Authentication scheme and defines a few algorithms that could be used with the Digest Authentication scheme, and establishes a registry for these algorithms to allow for additional algorithms to be added in the future.

This document updates the Digest Access Authentication scheme used by SIP to support the list of digest algorithms defined in the "Hash Algorithms for HTTP Digest Authentication" registry defined by [RFC7616].

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].

2. The Updated SIP Digest Authentication Scheme

This section describes the modifications to the operation of the Digest mechanism as specified in [RFC3261] in order to support the SHA- 256 and SHA-512/256 algorithms as described in [RFC7616], and also to require support for the "qop" option."

2.1. Hash Algorithms

The Digest scheme has an 'algorithm' parameter that specifies the algorithm to be used to compute the digest of the response. The IANA registry named "HTTP Digest Hash Algorithms" specifies the algorithms that correspond to 'algorithm' values, and specifies a priority for each algorithm.

[RFC3261] specifies only one algorithm, MD5, which is used by default. This document extends [RFC3261] to allow use of any registered algorithm.

The priority of the algorithm defines its usage preference. UAs SHOULD prefer algorithms with higher priorities.

Note that [RFC7616] defines a -sess variant for each algorithm; the -sess variants are not used with SIP.

2.2. Representation of Digest Values

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, SHA-256 and SHA-512/256 by 64 hexadecimal characters.

2.3. The Authenticate Response Header Field

When a UAS receives a request from a UAC, and an acceptable Authorization header field is not sent, the UAS can challenge the originator to provide credentials by rejecting the request with a 401/407 status code with the WWW-Authenticate/Proxy-Authenticate header field. The UAS MAY include multiple WWW-Authenticate/Proxy-Authenticate headers to allow the UAS to utilize the best available algorithm supported by the client.

If the UAS challenges with multiple WWW-Authenticate/Proxy-Authenticate headers with the same realm, then each one of these headers MUST use a different digest algorithm. The UAS MUST add these headers to the response in the order that it would prefer to see them used, starting with the most preferred algorithm at the top, followed by the less preferred algorithms.

2.4. The Authorization Request Header Field

When the UAC receives a response with multiple header fields with the same realm it SHOULD use the topmost header field that it supports, unless a local policy dictates otherwise. The client MUST ignore any challenge it does not understand.

When the UAC receives a 401 response with multiple WWW-Authenticate header fields with different realms it SHOULD retry and include an Authorization header field containing credentials that match the topmost header field of any one of the realms.

If the UAC cannot respond to any of the challenges in the response, then it should abandon attempts to send the request; e.g., if the UAC does not have credentials for any of the realms.

2.5. Forking

Section 22.3 of [RFC3261] discusses the operation of the proxy-to-user authentication, which describes the operation of the proxy when it forks a request. This section introduces some clarification to that operation.

If a request is forked, various proxy servers and/or UAs may wish to challenge the UAC. In this case, the forking proxy server is responsible for aggregating these challenges into a single response. Each WWW-Authenticate and Proxy-Authenticate value received in responses to the forked request MUST be placed into the single response that is sent by the forking proxy to the UA.

When the forking proxy places multiple WWW-Authenticate and Proxy-Authenticate header fields from one received response into the single response it MUST maintain the order of these header fields. The ordering of the header field values from the various proxies is not significant.

2.6. HTTP Modifications

This section describes the modifications and clarifications required to apply the HTTP Digest authentication scheme to SIP. The SIP scheme usage is similar to that for HTTP. For completeness, the bullets specified below are mostly copied from section 22.4 of [RFC3261]; the only semantic changes are specified in bullets 7 and 8 below.

SIP clients and servers MUST NOT accept or request Basic authentication.

The rules for Digest authentication follow those defined in HTTP, with "HTTP/1.1" replaced by "SIP/2.0" in addition to the following differences:

1. The URI included in the challenge has the following BNF:

   URI = Request-URI ; as defined in [RFC3261], Section 25

2. The 'uri' parameter of the Authorization header field MUST be enclosed in quotation marks.

3. The BNF for digest-uri-value is:

   digest-uri-value = Request-URI

4. The example procedure for choosing a nonce based on Etag does not work for SIP.

5. The text in [RFC7234] regarding cache operation does not apply to SIP.

6. [RFC7616] requires that a server check that the URI in the request line and the URI included in the Authorization header field point to the same resource. In a SIP context, these two URIs may refer to different users, due to forwarding at some proxy. Therefore, in SIP, a server MAY check that the Request-URI in the Authorization header field value corresponds to a user for whom the server is willing to accept forwarded or direct requests, but it is not necessarily a failure if the two fields are not equivalent.

7. As a clarification to the calculation of the A2 value for message integrity assurance in the Digest authentication scheme, implementers should assume, when the entity-body is empty (that is, when SIP messages have no body) that the hash of the entity-body resolves to the hash of an empty string:

```
H(entity-body) = <algorithm>("")
```

For example, when the chosen algorithm is SHA-256, then:

```
H(entity-body) = SHA-256("") =
"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
```

8. Servers MUST be able to properly handle "qop" parameter received in an authorization header field, and clients MUST be able to properly handle "qop" parameter received in WWW-Authenticate and Proxy-Authenticate header fields. Servers MUST always send a "qop" parameter in WWW-Authenticate and Proxy-Authenticate header field values, and clients MUST send the "qop" parameter in any resulting authorization header field.

The usage of the Authentication-Info header field continue to be allowed, since it provides integrity checks over the bodies and provides mutual authentication.

2.7. Augmented BNF for the SIP Protocol

This document updates the Augmented BNF for the SIP Protocol as follows.

It extends the request-digest as follows to allow for different digest sizes:

```
request-digest = LDQUOT *LHEX RDQUOT
```

The number of hex digits must be specified by the specification of the algorithm used.

It extends the algorithm parameter as follows to allow for SHA2 algorithms to be used:

```
algorithm = "algorithm" EQUAL ( "MD5" / "SHA-512-256" / "SHA-256"
/ token )
```

3. Security Considerations

This specification adds new secure algorithms to be used to with the Digest mechanism to authenticate users, but leaves the broken MD5 algorithm for backward compatibility.

This opens the system to the potential of a downgrade attack by man-in-the-middle. The most effective way of dealing with this type of attack is to either validate the client and challenge it accordingly, or remove the support for backward compatibility by not supporting MD5.

See section 5 of [RFC7616] for a detailed security discussion of the Digest scheme.

4. IANA Considerations

[RFC7616] defines an IANA registry named "Hash Algorithms for HTTP Digest Authentication" to simplify the introduction of new algorithms in the future. This document will use the algorithms defined in that registry.

5. Acknowledgments

The author would like to thank the following individuals for their careful reviews, comments, and suggestions: Paul Kyzivat, Olle Johansson, Dale Worley, Michael Procter, Inaki Baz Castillo, Tolga Asveren, Christer Holmberg, and Brian Rosen.

6. Normative References

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

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, H., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC7234] Fielding, R., Nottingham, M., and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, June 2014.

[RFC7616] Shekh-Yusef, R., Ahrens, D., and S. Bremer, "HTTP Digest Access Authentication", RFC 7616, September 2015.

Author's Address

Rifaat Shekh-Yusef
Avaya
425 Legget Dr.
Ottawa, Ontario
Canada

Phone: +1-613-595-9106
EMail: rifaat.ietf@gmail.com