SIP Core
Internet-Draft
Updates: 3261 (if approved)
Intended status: Standards Track
Expires: May 1, 2020

R. Shekh-Yusef
Avaya

October 29, 2019

The Session Initiation Protocol (SIP) Digest Authentication Scheme draft-ietf-sipcore-digest-scheme-12

Abstract

This document updates RFC 3261 by updating the Digest Access Authentication scheme used by the Session Initiation Protocol (SIP) to add support for more secure digest algorithms, e.g., SHA-256 and SHA-512-256, 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 May 1, 2020.

Table of Contents

1. Introduction

The Session Initiation Protocol [RFC3261] uses the same mechanism that the Hypertext Transfer Protocol (HTTP) uses for authenticating users. This mechanism is called Digest Access Authentication, and it is a simple challenge-response mechanism that allows a server to challenge a client request and allows a client to provide authentication information in response to that challenge. The version of Digest Access Authentication that [RFC3261] references is specified in [RFC2617].

The default hash algorithm for Digest Access Authentication is MD5. However, it has been demonstrated that the MD5 algorithm is not collision resistant, and is now considered a bad choice for a hash function [RFC6151].

The HTTP Digest Access Authentication [RFC7616] document obsoletes [RFC2617] and adds stronger algorithms that can be used with the Digest Authentication scheme, and establishes a registry for these algorithms, known as the "Hash Algorithms for HTTP Digest Authentication" registry, so that algorithms can be added in the future.

This document updates the Digest Access Authentication scheme used by SIP to support the algorithms listed 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", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. SIP Digest Authentication Scheme Updates

This section describes the modifications to the operation of the Digest mechanism as specified in [RFC3261] in order to support the algorithms defined in the "Hash Algorithms for HTTP Digest Authentication" registry described in [RFC7616].

It replaces the reference used in [RFC3261] for Digest Access Authentication, substituting [RFC7616] for the obsolete [RFC2617], and describes the modifications to the usage of the Digest mechanism in [RFC3261] resulting from that reference update. It adds support for the SHA-256 and SHA-512/256 algorithms. It adds required support for the "qop" parameter. It provides additional User Agent Client (UAC) and User Agent Server (UAS) procedures regarding usage of multiple SIP Authorization, WWW-Authenticate and Proxy-Authenticate header fields, including in which order to insert and process them. It provides guidance regarding forking. Finally, it updates the SIP BNF as required by the updates.

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 the "Hash Algorithms for HTTP Digest Authentication" specifies the algorithms that correspond to 'algorithm' values.

[RFC3261] specifies only one algorithm, MD5, which is used by default. This document extends [RFC3261] to allow use of any algorithm listed in the "Hash Algorithms for HTTP Digest Authentication" registry.

A UAS prioritizes which algorithm to use based on the ordering of the challenge header fields in the response it is preparing. That process is specified in section 2.3 and parallels the process used in HTTP specified by [RFC7616].

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 SHA-256 or SHA-512/256 algorithm is used to calculate the digest, then the digest will be represented as 64 hexadecimal characters.

2.3. UAS Behavior

When a UAS receives a request from a UAC, and an acceptable Authorization header field is not received, 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 respectively. The UAS MAY add multiple WWW-Authenticate/Proxy-Authenticate header fields to allow the UAS to utilize the best available algorithm supported by the client.

If the UAS challenges with multiple WWW-Authenticate/Proxy-Authenticate header fields with the same realm, then each one of these header fields MUST use a different digest algorithm. The UAS MUST add these header fields 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. The UAS cannot assume that the client will use the algorithm specified at the topmost header field.

2.4. UAC Behavior

When the UAC receives a response with multiple WWW-Authenticate/ Proxy-Authenticate 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 add an Authorization header field containing credentials that match the topmost header field of any one of the realms, unless a local policy dictates otherwise.

If the UAC cannot respond to any of the challenges in the response, then it SHOULD abandon attempts to send the request unless a local policy dictates otherwise. For example, if the UAC does not have credentials or has stale credentials for any of the realms, the UAC will abandon the request.

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 clarifies 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 UAC.

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

2.6. HTTP Digest Authentication Scheme 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 1, 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" [RFC7616] replaced by "SIP/2.0" in addition to the following differences:

1. The URI included in the challenge has the following BNF [RFC5234]:

   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 UAS MUST check if the Request-URI in the Authorization/Proxy-Authorization header field value corresponds to a user for whom the UAS is willing to accept forwarded or direct requests, but MAY still accept it 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. A UAS MUST be able to properly handle "qop" parameter received in an Authorization/Proxy-Authorization header field, and a UAC MUST be able to properly handle "qop" parameter received in WWW-Authenticate and Proxy-Authenticate header fields. However, for backward compatibility reasons, the "qop" parameter is optional for RFC3261-based clients and servers to receive. If the "qop" parameter is not specified, then the default value is "auth".

A UAS MUST always send a "qop" parameter in WWW-Authenticate and Proxy-Authenticate header field values, and a UAC MUST send the "qop" parameter in any resulting authorization header field.

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

2.7. Augmented BNF for SIP

This document updates the Augmented BNF [RFC5234] for SIP 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 is implied by the length of the value of the algorithm used.

It extends the algorithm parameter as follows to allow for any algorithm in the registry to be used:

algorithm = "algorithm" EQUAL (("MD5" / "SHA-512-256" / "SHA-256")[-sess]) / token)

Each one of these algorithms might have a "-sess" variant, e.g., MD5-sess, SHA-256-sess, etc, as defined in [RFC7616]

3. Security Considerations

This specification adds new secure algorithms to be used with the Digest mechanism to authenticate users. The broken MD5 algorithm

remains only for backward compatibility with [RFC2617] but its use is NOT RECOMMENDED.

This opens the system to the potential of a downgrade attack by an on-path attacker. 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 specifies that algorithms defined in that registry may be used in SIP digest authentication.

This document has no actions for IANA.

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, Brian Rosen, Jean Mahoney, Adam Roach, Barry Leiba, Roni Even, Benjamin Kaduk, and Alissa Cooper.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

6.2. Informative References

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

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <https://www.rfc-editor.org/info/rfc6151>.

Author's Address

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

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