Abstract

RFC3261 [2] recommends that SIP requests be authenticated using HTTP Digest authentication as defined in RFC2617 [3], and many vendors have successfully added this function to their products. However implementation bugs are still being observed both at interoperability events and in the field.

This document provides worked examples of Digest authentication usage in SIP. It is intended to help new SIP implementers, and to provide a common set of test authentication examples against which an implementation may be checked.
Table of Contents

1. Introduction................................................... 3
2. Common Mistakes................................................ 3
   2.1 Quotation marks in Quality-of-protection (QOP) Description..3
   2.2 Absent QOP and qop=auth result in different digests........5
   2.3 Invalid validation of a nonce................................ 6
3. Worked Examples................................................ 7
   3.1 Algorithm and QOP not specified............................. 8
       3.1.1 Challenge........................................ 8
       3.1.2 Authenticated Request................................ 8
       3.1.3 Digest Generation Process.......................... 9
   3.2 auth and algorithm unspecified (MD5 assumed)..............11
       3.2.1 Challenge........................................ 11
       3.2.2 Authenticated Request............................. 11
       3.2.3 Digest Generation Process.......................... 12
   3.3 auth and MD5..............................................14
       3.3.1 Challenge........................................ 14
       3.3.2 Authenticated Request............................. 14
       3.3.3 Digest Generation Process.......................... 15
   3.4 auth and MD5-Sess.........................................17
       3.4.1 Challenge........................................ 17
       3.4.2 Authenticated Request............................. 17
       3.4.3 Digest Generation Process.......................... 18
   3.5 auth-int and MD5........................................ 20
       3.5.1 Challenge........................................ 20
       3.5.2 Authenticated Request............................. 20
       3.5.3 Digest Generation Process.......................... 21
   3.6 auth-int and MD5-Sess....................................24
       3.6.1 Challenge........................................ 24
       3.6.2 Authenticated Request............................. 24
       3.6.3 Digest Generation Process.......................... 25
4. Changes......................................................27
   4.1 Changes since draft-smith-sip-auth-digest-00...............27
   4.2 Changes since draft-smith-sipping-auth-digest-00.........27
Security Considerations..........................................28
IANA Considerations.............................................28
Normative References............................................28
Informative References..........................................28
Acknowledgments..................................................28
Authors’ Addresses...............................................29

Conventions used in this document
The key words "MUST", "MUST NOT" and "SHOULD NOT" in this document should be interpreted as described in RFC2119 [4].

This document uses the <hex> representations introduced in draft-ietf-sipping-torture-tests [5] to specify the input and output from cryptographic operations.
1. Introduction

This document performs two tasks. Firstly, it describes three common problems encountered with Digest authentication usage in SIP implementations. Secondly, it provides SIP implementers with a set of Digest authentication examples, complete with checkpoints, for the various stages of the Digest calculations. These examples can be used to test and prove implementations.

This document does not comment on the benefits or deficiencies of HTTP Digest authentication compared to other possible authentication schemes.

Similarly, HTTP Basic authentication is deprecated from SIP by RFC3261 [1] so is not covered by this document.

It is assumed that the reader is familiar with RFC3261 [1] and RFC2617 [3].

2. Common Mistakes

2.1 Quotation marks in Quality-of-protection (QOP) Description

In typical SIP usage, the Quality-of-Protection (QOP) indicates whether the authentication digest includes a hash of the body of the SIP message or just the name and password of the caller.

A challenging UAS may indicate which of these QOP options is supported by returning a list of permitted QOPs through the qop-options field of a challenge. The requesting UAC then chooses one of these QOPs and indicates which it has chosen through the message-qop field of the credentials. However, since the qop-options in a challenge consists of a comma separated list, the qop-value(s) associated with the qop-options MUST be surrounded by double-quotes. The message-qop in the resulting credentials consists of a single qop-value and MUST NOT be surrounded by double-quotes.

Examples of "double-quotes in credentials" and "missing double-quotes in challenges", both of which are invalid, have been observed.
The ABNF in RFC2617 [3] that specifies this syntax is reproduced here. The challenge contains a qop-options, and the qop-values are surrounded by double-quotes. Note that this is true even if only a single qop-value is specified.

```
challenge    =  "Digest" digest-challenge

digest-challenge  = 1#( realm | [ domain ] | nonce |
                       [ opaque ] |[ stale ] | [ algorithm ] |
                       [ qop-options ] | [auth-param] )

qop-options       = "qop" "=" <" 1#qop-value <">
qop-value         = "auth" | "auth-int" | token
```

The credentials contain a single message-qop value which is not surrounded by double quotes.

```
credentials      =  "Digest" digest-response

digest-response  = 1#( username | realm | nonce | digest-uri |
                     response | [ algorithm ] | [cnonce] |
                     [opaque] | [message-qop] |
                     [nonce-count] | [auth-param] )

message-qop      =  "qop" "=" qop-value
```

The following examples, taken from RFC3261 [1], display this subtle point. The WWW-Authenticate header is an example of a challenge, whilst the Authorization header is an example of credentials.

```
WWW-Authenticate: Digest
   realm="biloxi.com",
   qop="auth,auth-int",
   nonce="dcd98b7102dd2f0e8b11d0f600bf0c093",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"

Authorization: Digest username="bob",
    realm="biloxi.com",
    nonce="dcd98b7102dd2f0e8b11d0f600bf0c093",
    uri="sip:bob@biloxi.com",
    qop=auth,
    nc=000000001,
    cnonce="0a4f113b",
    response="6629fae49393a05397450978507c4ef1",
    opaque="5ccc069c403ebaf9f0171e9517f40e41"
```
2.2 Absent QOP and qop=auth result in different digests

Another common mistake arises from the following part of RFC2617 [3].

If the "qop" value is "auth" or "auth-int":

```
request-digest = <"" < KD ( H(A1), unq(nonce-value) :
    "" nc-value :
    "" unq(cnonce-value) :
    "" unq(qop-value) :
    "" H(A2) ) >
""
```

If the "qop" directive is not present (this construction is for compatibility with RFC 2069):

```
request-digest = <"" < KD ( H(A1), unq(nonce-value) ":" H(A2) ) >
""
```

Although the values for A1 and A2 are the same regardless of whether the "qop" directive is unspecified or auth, the Digest is different.
2.3 Invalid validation of a nonce

It has been observed that some implementations create, and then attempt to validate, nonces based on the contents of the request that has been challenged. This causes implementation problems where the partner does not maintain certain SIP header fields between the original and later, authenticated, request.

RFC 3261 does not specify a mechanism for creating a nonce for use in an authentication challenge. An implementation is free to generate a nonce in whichever way it sees fit and may choose to do so such that it may be validated and stale nonces detected to avoid replay attacks.

RFC3261 [1] section 8.1.3.5 describes how to act on receipt of a 4XX class response, including 401 and 407 challenges, and states the following:

In all of the above cases, the request is retried by creating a new request with the appropriate modifications. This new request constitutes a new transaction and SHOULD have the same value of the Call-ID, To, and From of the previous request, but the CSeq should contain a new sequence number that is one higher than the previous.

This has been interpreted by some implementations to mean that an authenticated request can be relied upon to always contain the same Call-ID, To and From as the previous request. However this assumption is wrong in the case of dialog creating requests.

If an initial dialog creating request fails, the dialog has ended. Reusing the same Call-ID and From tag appears to indicate that the dialog is expected to still exist. It does not and thus reuse of the same Call-ID and From tag should not be assumed.

Nonces SHOULD NOT be generated based on information such as the Call-ID and tags unless a dialog has successfully been established and the request is therefore in-dialog.
3. Worked Examples

This chapter gives worked examples of challenges and the resulting authorized requests using the various combinations of QOP and algorithm options defined in RFC2617 [3].

For each worked example, each phase of the generation of the request-dialog, the value assigned to the "response" parameter of the Authorization header, is presented. These examples, and the checkpoint values, may be used to test and prove new SIP implementations using Digest authentication.

In order to remove any doubt regarding possible whitespace, the first example shows key values both using plain text and also the <hex> representation [5].

For each example, it may be assumed that a similar, but unauthorized, INVITE has been issued previously which the UAS has challenged.

Many SIP header fields that are not pertinent to the explanation of the digest calculation have been omitted for clarity.

Finally, it should be noted that in all cases, Bob’s password is:

zanzibar

or

<hex>7a616e7a69626172</hex>
3.1 Algorithm and QOP not specified

3.1.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest
   realm="biloxi.com",
   nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.1.2 Authenticated Request

INVITE sip:alice@atlanta.com.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
   realm="biloxi.com",
   nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
   uri="sip:bob@biloxi.com",
   nc=00000001,
   cnonce="0a4f113b",
   response="bf57e4e0d0bffe0fbaedce64d59aadd5e",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

(Alice’s SDP not shown)
3.1.3 Digest Generation Process

The challenge contains no QOP information. In order to be compliant with RFC2617 [3], implementations SHOULD provide a qop-options field. This field is only optional in RFC2617 [3] for back-compatibility with RFC2069 [6].

However implementations may choose to handle this challenge in order to provide maximum chance of interoperating against other implementations.

Given the absence of both algorithm and QOP, RFC2617 [3] defines $A_1$ as:

If the "algorithm" directive’s value is "MD5" or is unspecified, then

$A_1$ is:

$$A_1 = \text{unq(username-value)} :: \text{unq(realm-value)} :: \text{passwd}$$

where

$\text{passwd} = < \text{user’s password} >$

This value of $A_1$ is:

bob:biloxi.com:zanzibar

or

<hex>626f623a62696c6f78692e636f6d3a7a616e7a69626172</hex>

Using the definition of $H(A_1)$ given in RFC2617 [3], $H(A_1)$ has the following value:

12af60467a33e8518da5c68bbff12b11

From RFC2617 [3], $A_2$ is defined as:

If the "qop" directive’s value is "auth" or is unspecified, then $A_2$ is:

$$A_2 = \text{Method} :: \text{digest-uri-value}$$

The value of $A_2$ is

INVITE:sip:bob@biloxi.com
or

\[ \text{H(A2)} \text{ then has value:} \]

\[ 13a14a3eb5e2c24732a1a04fff543e92 \]

Finally, the Digest is given by the following:

If the "qop" directive is not present (this construction is for compatibility with RFC 2069):

\[
\text{request-digest} = \text{"} <" \text{ KD ( H(A1), unq(nonce-value) ":" H(A2) ) } \text{"} >
\]

\[ \text{H(A1)} \text{ and H(A2)} \text{ have been calculated above. The nonce-value is provided by the server in the challenge.} \]

From the definition of KD given in RFC2617 [3] it follows that the request-digest has value

\[
<" \text{ H(12af60467a33e8518da5c68bbff12b11:} \\
\text{dcd98b7102dd2f0e8b11d0f600bfb0c093:} \\
\text{13a14a3eb5e2c24732a1a04fff543e92) } \text{"} >
\]

or

\[
<" \text{ H(<hex>313261663630343637613333653835313864613563363862626666} \\
\text{31326231313a646364393862373130326464326630653862313164} \\
\text{30663630362666230633039333a313361313461336523653263} \\
\text{3234373332613161303466666353433653932</hex>) } \text{"} >
\]

This results in a final value of:

\[ \text{Digest} = "bf57e4e0d0bffc0fbaedce64d59add5e" \]
3.2 auth and algorithm unspecified (MD5 assumed)

3.2.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest realm="biloxi.com",
qop="auth,auth-int",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.2.2 Authenticated Request

INVITE sip:alice@atlanta.com.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
realm="biloxi.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
uri="sip:bob@biloxi.com",
qop=auth,
nc=00000001,
cnonce="0a4f113b",
response="89eb0059246c02b2f6ee02c7961d5ea3",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

(Alice’s SDP not shown)
3.2.3 Digest Generation Process

Firstly, it should be noted that the challenge contained two qop-options, "auth" and "auth-int", and the UAC chose to use a message-qop of "auth". Secondly, no algorithm is specified in the challenge so the default value of MD5 is used.

Given these values of algorithm and QOP, RFC2617 [3] defines A1 as:

If the "algorithm" directive’s value is "MD5" or is unspecified, then

A1 is:

\[
A1 = \text{unq(username-value)} : \text{unq(realm-value)} : \text{passwd}
\]

where

\[
\text{passwd} = \langle \text{user’s password} \rangle
\]

This value of A1 is:

bob:biloxi.com:zanzibar

Using the definition of H(A1) given in RFC2617 [3], H(A1) has the following value:

12af60467a33e8518da5c68bbff12b11

From RFC2617 [3], A2 is defined as follows:

If the "qop" directive’s value is "auth" or is unspecified, then

A2 is:

\[
A2 = \text{Method } : \text{digest-uri-value}
\]

Thus the value of A2 is

INVITE:sip:bob@biloxi.com

H(A2) then has value:

13a14a3eb5e2c24732a1a04fff543e92
Finally, the Digest is given by the following:

If the "qop" value is "auth" or "auth-int":

\[
\text{request-digest} = \langle \rangle < \text{KD} (H(A1), \text{unq(nonce-value)})
\]
\[
\quad \quad = \langle \rangle : \text{nc-value}
\]
\[
\quad \quad = \langle \rangle : \text{unq(cnonce-value)}
\]
\[
\quad \quad = \langle \rangle : \text{unq(qop-value)}
\]
\[
\quad \quad = \langle \rangle : H(A2)
\]

\text{H(A1)} and \text{H(A2)} have been calculated above. The nonce-value is provided by the server in the challenge. The nonce-count (nc-value) and cnonce-value are generated by the client as per RFC2617 [3] and the qop-value is chosen from the qop-options proposed by the server.

From the definition of KD given in RFC2617 [3] it follows that the request-digest has value

\[
\langle \rangle \text{H(12af60467a33e8518da5c68bbff12b11:}
\]
\[
dcd98b7102dd2f0e8b11d0f600bfb0c093:
\]
\[
00000001:
\]
\[
0a4f113b:
\]
\[
\text{auth:}
\]
\[
13a14a3eb5e2c24732a1a04fff543e92) \langle \rangle
\]

This results in a final value of:

\text{Digest} = "89eb0059246c02b2f6ee02c7961d5ea3"
3.3 auth and MD5

3.3.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest
realm="biloxi.com",
qop="auth,auth-int",
algorithm=MD5,
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.3.2 Authenticated Request

INVITE sip:alice@atlanta.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
realm="biloxi.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
uri="sip:bob@biloxi.com",
qop=auth,
algorithm=MD5,
nc=00000001,
cnonce="0a4f113b",
response="89eb0059246c02b2f6ee02c7961d5ea3",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

(Alice’s SDP not shown)
3.3.3 Digest Generation Process

No choice is allowed in the selection of an algorithm. The challenge indicates one, and only one algorithm and the challenged party must accept this choice or fail to present an authenticated request.

The presence of MD5 as an algorithm is exactly equivalent to no algorithm being specified.

Given these values of algorithm and QOP, RFC2617 [3] defines A1 as:

If the "algorithm" directive’s value is "MD5" or is unspecified, then

A1 is:

\[ A1 = \text{unq(username-value)} : \text{unq(realm-value)} : \text{passwd} \]

where

\[ \text{passwd} = < \text{user’s password} > \]

This value of A1 is:

bob:biloxi.com:zanzibar

Using the definition of H(A1) given in RFC2617 [3], H(A1) has the following value:

12af60467a33e8518da5c68bbff12b11

From RFC2617 [3], A2 is defined as follows:

If the "qop" directive’s value is "auth" or is unspecified, then

A2 is:

\[ A2 = \text{Method} : \text{digest-uri-value} \]

Thus the value of A2 is

INVITE:sip:bob@biloxi.com

H(A2) then has value:

13a14a3eb5e2c24732a1a04f6f543e92
Finally, the Digest is given by the following:

If the "qop" value is "auth" or "auth-int":

\[
\text{request-digest} = \langle\rangle < KD \left( H(A1), \text{unq(nonce-value)} \right.
\]
\[
\quad \langle\rangle : \text{nc-value} \n\]
\[
\quad \langle\rangle : \text{unq(cnonce-value)} \n\]
\[
\quad \langle\rangle : \text{unq(qop-value)} \n\]
\[
\quad \langle\rangle : H(A2) \rangle
\]

\(H(A1)\) and \(H(A2)\) have been calculated above. The nonce-value is provided by the server in the challenge. The nonce-count (nc-value) and cnonce-value are generated by the client as per RFC2617 [3] and the qop-value is chosen from the qop-options proposed by the server.

From the definition of KD given in RFC2617 [3] it follows that the request-digest has value:

\[
\langle\rangle \ H(12af60467a33e8518da5c68bbff12b11:
\quad \text{dcd9b7102dd2f0e8b11d0f600bfb0c093:}
\quad 00000001:
\quad 0a4f113b:
\quad \text{auth:}
\quad 13a14a3eb5e2c4732a1a04fff543e92) \langle\rangle
\]

This results in a final value of:

Digest = "89eb0059246c02b2f6ee02c7961d5ea3"
3.4 auth and MD5-Sess

3.4.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest
    realm="biloxi.com",
    qop="auth,auth-int",
    algorithm=MD5-sess,
    nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
    opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.4.2 Authenticated Request

INVITE sip:alice@atlanta.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
    realm="biloxi.com",
    nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
    uri="sip:bob@biloxi.com",
    qop=auth,
    algorithm=MD5-sess,
    nc=00000001,
    cnonce="0a4f113b",
    response="e4e4ea61d186d07a92c9e1f6919902e9",
    opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

(Alice’s SDP not shown)
3.4.3 Digest Generation Process

Algorithm MD5-sess can be used to implement a SIP server that never knows the actual password for a given user. The SIP server is required to request the password, through an undefined mechanism, from a 3rd party authentication server, which returns a digest of the password which can then be used to authenticate the user.

Given these values of algorithm and QOP, RFC2617 [3] defines A1 as:

\[
A1 = H( \text{unq(username-value)} \text{"":} \text{unq(realm-value)} \\
\text{"":} \text{passwd} ) \\
\text{"":} \text{unq(nonce-value)} \text{"":} \text{unq(cnonce-value)}
\]

where

\[
\text{passwd} = < \text{user’s password} >
\]

Now the MD5 hash of username-value, realm-value and passwd would normally be returned by the authentication server. This has the following value.

\[
H(\text{bob:biloxi.com:zanzibar})
\]

or

\[
12af60467a33e8518da5c68bbff12b11
\]

So the value of A1 is:

\[
12af60467a33e8518da5c68bbff12b11: \\
dcd98b7102dd2f0e8b11d0f600bfb0c093: \\
0a4f113b
\]

Using the definition of H(A1) given in RFC2617 [3], H(A1) has the following value:

\[
4f36886771c77832be5c5a8de5a7ec82
\]

From RFC2617 [3], A2 is defined as follows:

If the "qop" directive’s value is "auth" or is unspecified, then A2 is:

\[
A2 = \text{Method } ":" \text{ digest-uri-value}
\]
Thus the value of A2 is

INVITE:sip:bob@biloxi.com

H(A2) then has value:

13a14a3eb5e2c24732a1a04fff543e92

Finally, the Digest is given by the following:

If the "qop" value is "auth" or "auth-int":

request-digest = "" < KD ( H(A1), unq(nonce-value) 
                 
                 "": nc-value 
                 
                 "": unq(cnonce-value) 
                 
                 "": unq(qop-value) 
                 
                 "": H(A2) 
                 ) "">

H(A1) and H(A2) have been calculated above. The nonce-value is provided by the server in the challenge. The nonce-count (nc-value) and cnonce-value are generated by the client as per RFC2617 [3] and the qop-value is chosen from the qop-options proposed by the server.

From the definition of KD given in RFC2617 [3] it follows that the request-digest has value

<" H(4f36886771c77832be5c5a8de5a7ec82: 
   dcd98b7102dd2f0e8b11d0f600bfb0c093: 
   00000001: 
   0a4f113b: 
   auth: 
   13a14a3eb5e2c24732a1a04fff543e92) "">

This results in a final value of:

Digest = "e4e4ea61d186d07a92c9e1f6919902e9"
3.5 auth-int and MD5

The use of qop value auth-int indicates that integrity protection is required. This means that the calculated digest includes a hash of the body of the authenticated message, which allows a change to the body of the message to be detected.

This may lead to possible problems traversing Network Address Translators (NATs), Back-to-back UAs (B2BUAs) and Application Level Gateways (ALGs), any of which may modify the body in order to permit the SIP request to traverse some form of network boundary. In this case, the NAT/B2BUA/ALG must also act as an endpoint and police and possibly modify the authentication header.

3.5.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest realm="biloxi.com",
   qop="auth,auth-int",
   algorithm=MD5,
   nonce="dcd98b7102dd2f0e8b11d0f600bf0c093",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.5.2 Authenticated Request

INVITE sip:alice@atlanta.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
   realm="biloxi.com",
   nonce="dcd98b7102dd2f0e8b11d0f600bf0c093",
   uri="sip:bob@biloxi.com",
   qop=auth-int,
   algorithm=MD5,
   nc=00000001,
   cnonce="0a4f113b",
   response="bdbeebeb2da6adb6bca02599c2239e192"
Digest Authentication Examples for SIP       June 2005

opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

v=0
o=bob 2890844526 2890844526 IN IP4 media.biloxi.com
s=-
c=IN IP4 media.biloxi.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtppmap:0 PCMU/8000
m=video 51372 RTP/AVP 31
a=rtppmap:31 H261/90000
m=video 53000 RTP/AVP 32
a=rtppmap:32 MPV/90000

3.5.3 Digest Generation Process

No choice is allowed in the selection of an algorithm. The challenge indicates one, and only one algorithm (and thus no double-quotes are present) and the challenged party must accept this choice or fail to present an authenticated request.

The presence of MD5 as an algorithm is exactly equivalent to no algorithm being specified.

Given these values of algorithm and QOP, RFC2617 [3] defines A1 as:

If the "algorithm" directive’s value is "MD5" or is unspecified, then

A1 is:

A1       = unq(username-value) ":" unq(realm-value) ":" passwd

where

passwd   = < user’s password >

This value of A1 is:

bob:biloxi.com:zanzibar

Using the definition of H(A1) given in RFC2617 [3], H(A1) has the following value:

12af60467a33e8518da5c68bbff12b11

From RFC2617 [3], A2 is defined as follows:
If the "qop" value is "auth-int", then A2 is:

\[
A2 = \text{Method} \backslash \text{:} \text{digest-uri-value} \backslash \text{:} \text{H(entity-body)}
\]

Thus the value of A2 is

\[
\text{INVITE:} \text{sip:} \text{bob@biloxi.com:H(entity-body)}
\]

entity-body consists of every byte comprising the body after the blank line that separates the SIP headers from the SIP body, up to the limit imposed by Content-Length or the end of the UDP datagram if no Content-Length field is present. Note that white-space is relevant in this case. Be careful of the presence, or absence, carriage return an/or linefeeds at the end of the data!

Thus \text{H(entity-body)} is given by

\[
\text{H(v=0 o=bob 2890844526 2890844526 IN IP4 media.biloxi.com s=- c=IN IP4 media.biloxi.com t=0 0 m=audio 49170 RTP/AVP 0 a=rtpmap:0 PCMU/8000 m=video 51372 RTP/AVP 31 a=rtpmap:31 H261/90000 m=video 53000 RTP/AVP 32 a=rtpmap:32 MPV/90000 )}
\]

or

\[
\text{H(<hex>763D300D0A6F3D626F622032323839303834334352362032383930383433435 32362049E20495034206D656469612E62696C6F78692E636F6D0A73DD 2D0A63D494E20495034206D656469612E62696C6F78692E636F6D0A743D020300D0A61D3D617564696E6F20343331373020525402F4156502030 0D0A61D3D724706D61703A020543D552F383030300D0A61D3D76696E6F2006F2035313337220525402F4156502033310D0A61D3D724706D61703A33 3120483236312F39303030300D0A63D76696E6F2006F203533310D0A61D3D724706D61703A3332204D50562F39303030 300D0A</hex>)}
\]

\[
\text{H(entity-body)} \text{ has the value}
\]

\[
c1ed018b8ec4a3b170c09f2f5b564e48
\]

\[
\text{H(A2)} \text{ has the value:}
\]
Finally, the Digest is given by the following:

If the "qop" value is "auth" or "auth-int":

request-digest = " " < KD ( H(A1), unq(nonce-value) 
                    " :" nc-value  
                    " :" unq(cnonce-value)  
                    " :" unq(qop-value)  
                    " :" H(A2) ) > ""

H(A1) and H(A2) have been calculated above. The nonce-value is
provided by the server in the challenge. The nonce-count (nc-value)
and cnonce-value are generated by the client as per RFC2617 [3] and
the qop-value is chosen from the qop-options proposed by the server.

From the definition of KD given in RFC2617 [3] it follows that the
request-digest has value

" " H(12af60467a33e8518da5c68bbff12b11:
     dcd98b7102dd2f0e8b11d0f600bf0c093:
     00000001:
     0a4f113b:
     auth-int:
     3e8ec46a56447dbb073e1171b1be0683) ""

This results in a final value of:

Digest = "bdbeebb2da6adb6bca02599c2239e192"
3.6 auth-int and MD5-Sess

3.6.1 Challenge

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKfjsduen
From: Bob <sip:bob@biloxi.com>;tag=9483056782
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
WWW-Authenticate: Digest
   realm="biloxi.com",
   qop="auth,auth-int",
   algorithm=MD5-sess,
   nonce="dcd98b7102dd2f0e8b11d0f600bfbb0c093",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Length: 0

3.6.2 Authenticated Request

INVITE sip:alice@atlanta.com.com SIP/2.0
Via: SIP/2.0/UDP s19.biloxi.com;branch=z9hG4bKnashds8
Max-Forwards: 70
From: Bob <sip:bob@biloxi.com>
To: Alice <sip:alice@atlanta.com>;tag=8493745023
Call-ID: ab734d9e6b793b
CSeq: 83952 INVITE
Contact: <sip:bob@s19.biloxi.com>
Authorization: Digest username="bob",
   realm="biloxi.com",
   nonce="dcd98b7102dd2f0e8b11d0f600bfbb0c093",
   url="sip:bob@biloxi.com",
   qop=auth-int,
   algorithm=MD5-sess,
   nc=00000001,
   cnonce="04f113b",
   response="91984da2d8663716e91554859c22ca70",
   opaque="5ccc069c403ebaf9f0171e9517f40e41"
Content-Type: application/sdp
Content-Length: 142

v=0
o=bob 2890844526 2890844526 IN IP4 media.biloxi.com
s=-
c=IN IP4 media.biloxi.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 51372 RTP/AVP 31
3.6.3 Digest Generation Process

Note that no choice is allowed in the selection of an algorithm. The challenge indicates one, and only one algorithm (and thus no double-quotes are present) and the challenged party must accept this choice if it fail to present an authenticated request.

Also note that the presence of MD5 as an algorithm is exactly equivalent to no algorithm being specified.

Given these values of algorithm and QOP, RFC2617 [3] defines A1 as:

\[
\text{If the "algorithm" directive’s value is "MD5-sess" then}
\]

\[
\begin{align*}
\text{A1} &= H( \text{unq(username-value)} \ ":\ " \text{unq(realm-value)} \\
& \quad \ ":\ " \text{passwd} ) \\
& \quad \ ":\ " \text{unq(nonce-value)} \ ":\ " \text{unq(cnonce-value)}
\end{align*}
\]

where

\[
\text{passwd} = < \text{user’s password} >
\]

Now the MD5 hash of username-value, realm-value and passwd would normally be returned by the authentication server. This has the following value.

\[
H(\text{bob:biloxi.com:zanzibar})
\]

or

\[
12af60467a33e8518da5c68bbff12b11
\]

So the value of A1 is:

\[
12af60467a33e8518da5c68bbff12b11:
\text{dcd98b7102dd2f0e8b11d0f600bf0b0c093:}
0a4f113b
\]

Using the definition of \(H(A1)\) given in RFC2617 [3], \(H(A1)\) has the following value:

\[
4f36886771c77832be5c5a8de5a7ec82
\]

From RFC2617 [3], A2 is defined as follows:
If the "qop" value is "auth-int", then A2 is:

\[
A2 = \text{Method "": digest-uri-value "": } H(\text{entity-body})
\]

Thus the value of A2 is

\[
\text{INVITE:sip:bob@biloxi.com:H(entity-body)}
\]

Thus \( H(\text{entity-body}) \) is given by

\[
H(\text{entity-body}) = H(v=0 \text{o=bob 2890844526 2890844526 IN IP4 media.biloxi.com} \text{s=-} \text{c=IN IP4 media.biloxi.com} \text{t=0 0} \text{m=audio 49170 RTP/AVP 0} \text{a=rtpmap:0 PCMU/8000} \text{m=video 51372 RTP/AVP 31} \text{a=rtpmap:31 H261/90000} \text{m=video 53000 RTP/AVP 32} \text{a=rtpmap:32 MPV/90000})
\]

\( H(\text{entity-body}) \) has the value

\[
c1ed018b8ec4a3b170c0921f5b564e48
\]

Finally, the Digest is given by the following:

If the "qop" value is "auth" or "auth-int":

\[
\text{request-digest } = \langle"\rangle < KD ( H(A1), \text{ unq(nonce-value)} \text{"": nc-value} \text{"": unq(cnonce-value)} \text{"": unq(qop-value)} \text{"": } H(A2) ) \rangle <"\rangle
\]

\( H(A1) \) and \( H(A2) \) have been calculated above. The nonce-value is provided by the server in the challenge. The nonce-count (nc-value) and cnonce-value are generated by the client as per RFC2617 [3] and the qop-value is chosen from the qop-options proposed by the server.
From the definition of KD given in RFC2617 [3] it follows that the request-digest has value

<"> H(4f36886771c77832be5c5a8de5a7ec82:
dcd98b7102dd2f0e8b11d0f600bf0c093:
00000001:
0a4f113b:
auth-int:
3e8ec46a56447dbb073e1171b1be0683) <">

This results in a final value of:

Digest = "91984da2d8663716e91554859c22ca70"

4. Changes

*** [Note to RFC editor. Please remove this entire section when this draft is published as an RFC.] ***

4.1 Changes since draft-smith-sip-auth-digest-00

Moved to Sipping WG and renamed draft-smith-sipping-auth-digest-00.

Corrections to SDP subject field and resultant digest values.

Updated boiler place in accordance with RFC 3667 and RFC 3668.

4.2 Changes since draft-smith-sipping-auth-digest-00

Corrected calculated digests for sections 3.1 and 3.2.

Updated boiler place in accordance with RFC 3978 and RFC 3979.
Security Considerations

This document introduces no new protocol elements and therefore has no effect on the security of SIP.

IANA Considerations

This document has no actions for IANA.

Normative References

1 Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.


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


Informative References

None.

Acknowledgments

Paul and Ian would like to thank their colleagues at Data Connection for their input during the production of this document, and WG members for useful feedback.
Authors’ Addresses

Paul D. Smith  
Data Connection Ltd  
100 Church Street  
Enfield  
Middlesex  
EN2 6BQ  
United Kingdom  
Email: paul.d.smith@dataconnection.com

Ian Clarkson  
Data Connection Ltd  
100 Church Street  
Enfield  
Middlesex  
EN2 6BQ  
United Kingdom  
Email: ian.clarkson@dataconnection.com

Disclaimer of validity

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat 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 implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.