RADIUS Extension for Digest Authentication
draft-ietf-radext-digest-auth-00.txt

Status of this Memo

This document is an Internet-Draft and is subject to all provisions of section 3 of RFC 3667. By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she become aware will be disclosed, in accordance with RFC 3668.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on June 8, 2005.

Copyright Notice

Copyright (C) The Internet Society (2004).

Abstract
Several protocols borrow the authentication mechanisms from the Hypertext Transfer Protocol, HTTP. This document specifies an extension to RADIUS that allows a RADIUS client in an HTTP-style server, upon reception of a request, retrieve and compute Digest authentication information from a RADIUS server. Additionally, a scenario describing the authentication of a user emitting an HTTP-style request is provided.

Table of Contents

1. Introduction ........................................... 4
   1.1 Terminology ....................................... 4
   1.2 Motivation ......................................... 4
   1.3 Overview ......................................... 4
     1.3.1 Scenario 1, RADIUS client chooses nonces ...... 6
     1.3.2 Scenario 2, RADIUS server chooses nonces .... 7
2. New RADIUS attributes ................................. 9
   2.1 Digest-Response attribute .......................... 9
   2.2 Digest-Realm attribute ............................. 9
   2.3 Digest-Nonce attribute ............................. 10
   2.4 Digest-Response-Auth attribute .................... 10
   2.5 Digest-Nextnonce attribute ....................... 11
   2.6 Digest-Method attribute ........................... 11
   2.7 Digest-URI attribute .............................. 11
   2.8 Digest-Qop attribute .............................. 12
   2.9 Digest-Algorithm attribute ....................... 12
   2.10 Digest-Entity-Body-Hash attribute ................ 12
   2.11 Digest-CNonce attribute ........................... 13
   2.12 Digest-Nonce-Count attribute ..................... 13
   2.13 Digest-Username attribute ....................... 14
   2.14 Digest-Opaque attribute ........................... 14
   2.15 Digest-Auth-Param attribute ..................... 14
   2.16 Digest-AKA-Auts attribute ....................... 15
   2.17 Digest-Domain attribute ........................... 15
   2.18 Digest-Stale attribute ........................... 16
   2.19 Digest-HA1 attribute .............................. 16
3. Detailed Description .................................... 18
   3.1 RADIUS Client Behavior ............................ 18
   3.2 RADIUS Server Behavior ............................ 20
4. Migration Path to Diameter ............................ 22
   4.1 RADIUS Client, Diameter Server ................... 22
   4.2 Diameter Client, RADIUS Server ................... 23
   4.3 Limitations ....................................... 23
5. Example .................................................. 24
6. IANA Considerations .................................... 27
7. Security Considerations ............................... 28
8. Change Log ............................................. 29
   8.1 Changes from draft-sterman-aaa-sip-04 ............. 29
8.2 Changes from draft-sterman-aaa-sip-03 .................. 29
8.3 Changes from draft-sterman-aaa-sip-02 .................. 29
8.4 Changes from draft-sterman-aaa-sip-01 .................. 29
9. References ............................................. 30
9.1 Normative References .................................... 30
9.2 Informative References ................................ 30
Authors’ Addresses ......................................... 32
A. Acknowledgments ....................................... 34
   Intellectual Property and Copyright Statements ........ 35
1. Introduction

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].

This document uses terminology from [RFC2617] and [RFC2865]

1.2 Motivation

Digest authentication is a simple authentication mechanism for HTTP and SIP. While it was not too successful in HTTP environments, it is the only SIP authentication mechanism that has been widely adopted. Due to the limitations and weaknesses of Digest authentication (see [RFC2617], section 4), additional PKI-based authentication and encryption mechanisms have been introduced into SIP: TLS [RFC2246] and S/MIME [RFC2633]. The majority of today’s SIP clients only supports HTTP digest.

Current RADIUS-based AAA infrastructures have been built and debugged over years. Some deficiencies of RADIUS have been mitigated with proprietary extensions. Operators are therefore reluctant to replace their RADIUS infrastructure in order to enable a single new authentication mechanism.

Given the complexity of the alternatives, simple clients will continue to support HTTP digest authentication only. Its interoperability with a back-end authentication protocol such as RADIUS is needed.

Operators that are about to replace their RADIUS-based AAA infrastructure are strongly recommended to use Diameter.

1.3 Overview

Figure 1 depicts the basic scenario that is relevant for this document. 'HTTP-style Client' and 'RADIUS Client' are entities using a protocol with support for HTTP Digest Authentication, like SIP or HTTP.
The approach taken here is to extend RADIUS to support Digest authentication by mimicking its native support for CHAP authentication. According to [RFC2865], the RADIUS server distinguishes between different authentication schemes by looking at the presence of an attribute specific for that scheme. For the three natively supported authentication schemes, these attributes are: User-Password for PAP (or any other clear-text password scheme), CHAP-Password for CHAP, and State + User-Password for challenge-response scheme. This document adds another attribute to be used in this role: Digest-Response. Also according to [RFC2865], "An Access-Request packet MUST contain either a User-Password or a CHAP-Password or a State. It MUST NOT contain both a User-Password and a CHAP-Password. If future extensions allow other kinds of authentication information to be conveyed, the attribute for that can be used instead of User-Password or CHAP-Password." The Digest-Response introduced here therefore can be used instead of User-Password or CHAP-Password.

The HTTP Authentication parameters found in the Proxy-Authorization or Authorization request header are mapped into newly defined RADIUS attributes. These new RADIUS attributes are defined in the document together with some other information required for calculating the correct digest response on the RADIUS server with exception of the password, which the RADIUS server is assumed to be able to retrieve from a data store given the username.

The nonces required by the digest algorithm are either generated by the RADIUS client or by the RADIUS server. A mix of nonce generation modes is not supported. If the RADIUS server generates nonces, all RADIUS clients MUST NOT try to generate nonces. If the RADIUS server does not generate nonces, all RADIUS clients MUST generate nonces locally. If at least one HTTP-style client requires AKA authentication [RFC3310], the RADIUS server MUST generate nonces and its RADIUS clients MUST NOT generate nonces locally.
1.3.1 Scenario 1, RADIUS client chooses nonces

The roles played by the entities in this scenario are as follows:

A: HTTP client / SIP UA

B: {HTTP server / HTTP proxy server / SIP proxy server / SIP UAS}
acting also as a RADIUS NAS (RADIUS client)

C: RADIUS server

The relevant order of messages sent in this scenario is as follows:

A sends B an HTTP/SIP request without authorization header (step 1).
B challenges A sending an HTTP/SIP "407 / 401 (Proxy) Authorization
required" response containing a locally generated nonce (step 2). A
sends B an HTTP/SIP request with authorization header (step 3). B
sends C a RADIUS Access-Request with attributes described in this
document (step 4). C responds to B with a RADIUS
Access-Accept/Access-Reject response (step 5). If credentials were
accepted B receives an Access-Accept response and the message sent
from A is considered authentic. If B receives an Access-Reject response, however, B then responds to A with a "407 / 401 (Proxy) Authorization required" response (step 6).

1.3.2 Scenario 2, RADIUS server chooses nonces

In most cases, the operation outlined in Section 1.3.1 is sufficient. It reduces the load on the RADIUS server to a minimum. However, when using AKA [RFC3310] the nonce is partially derived from a precomputed authentication vector. These authentication vectors are often stored centrally.

Figure 3 depicts a scenario, where the RADIUS server chooses nonces. It shows a generic case where entities A and B communicate in the front-end using protocols such as HTTP/SIP, while entities B and C communicate in the back-end using RADIUS.

---

The roles played by the entities in this scenario are as follows:
A: HTTP client / SIP UA

B: {HTTP server / HTTP proxy server / SIP proxy server / SIP UAS) acting also as a RADIUS NAS

C: RADIUS server

The relevant order of messages sent in this scenario is as follows:

A sends B an HTTP/SIP request without authorization header (step 1). B sends an Access-Request message with the newly defined Digest-Method and Digest-URI attributes but without a Digest-Nonce attribute to the RADIUS server, C (step 2). C chooses a nonce and responds with an Access-Challenge (step 3). This Access-Challenge contains Digest attributes, from which B takes values to construct an HTTP/SIP "(Proxy) Authorization required" response. The remaining steps are identical with scenario 1 (Section 1.3.1): B sends this response to A (step 4). A resends its request with its credentials (step 5). B sends an Access-Request to C (step 6). C checks the credentials and replies with Access-Accept or Access-Reject (step 7). Dependent on the C’s result, B processes A’s request or rejects it with a "(Proxy) Authorization required" response (step 8).
2. New RADIUS attributes

The term ‘HTTP-style’ denotes any protocol that uses HTTP-like headers and uses HTTP digest authentication as described in [RFC2617]. Examples are HTTP and SIP.

If not stated otherwise, the attributes have the following format:

```
  0                   1                   2
+---------------+---------------+----------+
|     Type      |  Length       | String...|
+---------------+---------------+----------+
```

2.1 Digest-Response attribute

If this attribute is present in an Access-Request message, the RADIUS server SHOULD view the Access-Request as a Digest one. When a RADIUS client receives a (Proxy-)Authorization header, it puts the request-digest value into a Digest-Response attribute.

Type
[IANA: use 102 if possible] for Digest-Response.

Length
>= 34

Text
This attribute MUST only be used in Access-Requests. It proves the user knows the password. The text field is usually 32 octets long and contains hexadecimal representation of 16 octet digest value as it was calculated by the authenticated client. The text field SHOULD be copied from request-digest of digest-response ([RFC2617]) without quotes.

2.2 Digest-Realm attribute

This attribute describes a protection space of the RADIUS server. See [RFC2617] 1.2 for details.

Type
[IANA: use 103 if possible] for Digest-Realm

Length >=3

Text

In Access-Requests, the RADIUS client takes the value of the realm directive (realm-value according to [RFC2617]) without quotes from the HTTP-style request it wants to authenticate. In Access-Challenge messages, the RADIUS server puts the expected realm value into this attribute. This attribute MUST only be used in Access-Request and Access-Challenge messages.

2.3 Digest-Nonce attribute

This attribute holds a random nonce to be used in the HTTP Digest calculation.

Type [IANA: use 104 if possible] for Digest-Nonce

Length >=3

Text

In Access-Requests, the RADIUS client takes the value of the nonce directive (nonce-value in [RFC2617]) without quotes from the HTTP-style request it wants to authenticate. If the Access-Request had a Digest-Method and a Digest-URI but no Digest-Nonce attribute and the RADIUS server is configured to choose nonces, it MUST put a Digest-Nonce attribute into its Access-Challenge message. This attribute MUST only be used in Access-Request and Access-Challenge messages.

2.4 Digest-Response-Auth attribute

Type [IANA: use 105 if possible] for Digest-Response-Auth.

Length >= 34

Text

This attribute MUST only be used in Access-Accept messages. This text proves the RADIUS server knows the password. The RADIUS server calculates a digest according to section 3.2.3 of [RFC2617] and copies the result into this attribute. The RADIUS client puts the attribute value without quotes into the rspauth directive of the Authentication-Info header. If the previously received Digest-Qop attribute was ‘auth-int’ (without quotes), the RADIUS server MUST send a Digest-HA1 attribute instead of Digest-Response-Auth.
2.5 Digest-Nextnonce attribute

This attribute holds a random nonce to be used in the HTTP Digest calculation.

Type
  [IANA: use 106 if possible] for Digest-Nextnonce

Length
  >=3

Text
  If the RADIUS server is configured to choose nonces it MAY put a Digest-Nextnonce attribute into an Access-Accept message. If this attribute is present, the RADIUS client MUST put the contents of this attribute into the nextnonce directive of an Authentication-Info header in its HTTP-style response. This attribute MUST only be used in Access-Accept messages.

2.6 Digest-Method attribute

This attribute holds the method string to be used in the HTTP Digest calculation.

Type
  [IANA: use 107 if possible] for Digest-Method

Length
  >=3

Text
  In Access-Requests, the RADIUS client takes the value of the request method from the HTTP-style request it wants to authenticate. This attribute MUST only be used in Access-Request messages.

2.7 Digest-URI attribute

This attribute holds the URI string to be used in the HTTP Digest calculation.

Type
  [IANA: use 108 if possible] for Digest-URI

Length
  >=3

Text
  If the HTTP-style request has an Authorization header, the RADIUS client puts the value of the "uri" directive in the (known as "digest-uri-value" in Section 3.2.2 of [RFC2617]) without quotes into this attribute. If there is no Authorization header, the RADIUS client takes the value of the request URI from the HTTP-style request it wants to
authenticate. The attribute MUST only be used in
Access-Request messages.

2.8 Digest-Qop attribute

This attribute holds the Quality of Protection parameter that
influences the HTTP Digest calculation.

Type
[IANA: use 109 if possible] for Digest-Qop
Length
>=3
Text
In Access-Requests, the RADIUS client takes the value of the
qop directive (qop-value as described in [RFC2617]) without the
quotes from the HTTP-style request it wants to authenticate.
If the directive contains more than one qop-value, the RADIUS
client puts each qop-value into a separate Digest-Qop
attribute. In Access-Challenge messages, the RADIUS server
SHOULD put the desired qop-value into this attribute. This
attribute MUST only be used in Access-Request and
Access-Challenge messages.

2.9 Digest-Algorithm attribute

This attribute holds the algorithm parameter that influences the HTTP
Digest calculation.

Type
[IANA: use 110 if possible] for Digest-Algorithm
Length
>=3
Text
In Access-Requests, the RADIUS client takes the value of the
algorithm directive (as described in [RFC2617], section 3.2.1)
without the quotes from the HTTP-style request it wants to authenticate.
In Access-Challenge messages, the RADIUS server
MAY put the desired algorithm into this attribute. This
attribute MUST only be used in Access-Request and
Access-Challenge messages.

2.10 Digest-Entity-Body-Hash attribute

When using the qop level ‘auth-int’, a hash of the message body’s
contents is required for digest calculation. Instead of sending the
complete body of the message, only its hash value is sent. This hash
value can be used directly in the digest calculation.
The Digest-Entity-Body-Hash attribute contains a hash of the entity body contained in the SIP message. This hash is required by certain authentication mechanisms, such as HTTP Digest with quality of protection set to "auth-int". RADIUS clients MUST use this attribute to transport the hash of the entity body when HTTP Digest is the authentication mechanism and the RADIUS server requires to verify the integrity of the entity body (e.g., qop parameter set to "auth-int"). Extensions to this document may define support for authentication mechanisms other than HTTP Digest. The clarifications described in Section 22.4 of [RFC2617] about the hash of empty entity bodies apply to the Digest-Entity-Body-Hash attribute. This attribute MUST only be sent in Access-Request packets.

2.11 Digest-CNonce attribute

This attribute holds the client nonce parameter that is used in the HTTP Digest calculation.

Type  [IANA: use 112 if possible] for Digest-CNonce
Length  >=3
Text

In Access-Requests, the RADIUS client takes the value of the cnonce directive (cnonce-value according to [RFC2617]) without quotes from the HTTP-style request it wants to authenticate. This attribute MUST only be used in Access-Request messages.

2.12 Digest-Nonce-Count attribute

This attribute holds the nonce count parameter that is used to detect replay attacks.

Type  [IANA: use 113 if possible] for Digest-Nonce-Count
Length  10
Text
In Access-Requests, the RADIUS client takes the value of the nc directive (nc-value according to [RFC2617]) without quotes from the HTTP-style request it wants to authenticate. The attribute MUST only be used in Access-Request messages.

2.13 Digest-Username attribute

This attribute holds the user name parameter that is used in the HTTP digest calculation.

Type

[IANA: use 114 if possible] for Digest-Username

Length

>= 3

Text

In Access-Requests, the RADIUS client takes the value of the username directive (username-value according to [RFC2617]) without quotes from the HTTP-style request it wants to authenticate. The RADIUS server SHOULD NOT use this value for password finding, but only for digest calculation purpose. In order to find the user record containing the password, the RADIUS server SHOULD use the value of the ([RFC2865]-)User-Name attribute. This attribute MUST only be used in Access-Request packets.

2.14 Digest-Opaque attribute

This attribute holds the opaque parameter that is passed to the HTTP-style client. The HTTP-style client will pass this value back to the server (ie the RADIUS client) without modification.

Type

[IANA: use 115 if possible] for Digest-Opaque

Length

>=3

Text

This attribute is only used when the RADIUS server chooses nonces. In Access-Requests, the RADIUS client takes the value of the opaque directive (opaque-value according to [RFC2617]) without quotes from the HTTP-style request it wants to authenticate and puts it into this attribute. In Access-Challenge messages, the RADIUS server MAY include this attribute. This attribute MUST only be used in Access-Request and Access-Challenge messages.

2.15 Digest-Auth-Param attribute

This attribute is a placeholder for future extensions.
The Digest-Auth-Param is the mechanism whereby the RADIUS client and RADIUS server can exchange possible extension parameters contained in Digest headers that are not understood by the RADIUS client and for which there are no corresponding stand-alone attributes. Unlike the previously listed Digest-* attributes, the Digest-Auth-Param contains not only the value, but also the parameter name, since the parameter name is unknown to the RADIUS client. If the Digest header contains several unknown parameters, then the RADIUS implementation MUST repeat this attribute and each instance MUST contain one different unknown Digest parameter/value combination. This attribute corresponds to the "auth-param" parameter defined in section 3.2.1 of [RFC2617]. The text consists of the whole parameter, including its name and the equal ('='') sign and quotes. This attribute MAY be used in any type of RADIUS messages.

2.16 Digest-AKA-Auts attribute

This attribute holds the auts parameter that is used in the Digest AKA ([RFC3310]) calculation.

In Access-Requests, the RADIUS client takes the value of the auts directive (auts-param according to section 3.4 of [RFC3310]) without quotes from the HTTP-style request it wants to authenticate. It is only used if the algorithm of the digest-response denotes a version of AKA digest [RFC3310]. This attribute MUST only be used in Access-Request messages.

2.17 Digest-Domain attribute

When a RADIUS client has asked for a nonce, the RADIUS server MAY send one or more Digest-Domain attributes in its Access-Challenge message. The RADIUS client puts them into the quoted, space-separated list of URIs of the ‘domain’ directive of a WWW-Authenticate header. The URIs in the list define the protection space (see [RFC2617], section 3.2.1).
Type
[IANA: use 118 if possible] for Digest-Domain
Length
3
Text
This attribute consists of a single URI, that defines a
protection space. RADIUS servers MAY send one or more
attributes of this type in Access-Challenge messages. This
attribute MUST only be used in Access-Challenge messages.

2.18 Digest-Stale attribute
The RADIUS server uses this attribute to tell whether it has accepted
a nonce.

Type
[IANA: use 119 if possible] for Digest-Stale
Length
3
Text
The attribute has either the value ‘true’ or ‘false’ (both
values without quotes). If the nonce presented by the RADIUS
client was stale, the value is ‘true’ and is ‘false’ otherwise.
The RADIUS client puts the content of this attribute into a
‘stale’ directive of the WWW-Authenticate header in the
HTTP-style response to the request it wants to authenticate.
The attribute MUST only be used in Access-Challenge messages
and only if the RADIUS server chooses nonces.

2.19 Digest-HA1 attribute
This attribute is used to allow the generation of an
Authentication-Info header, even if the HTTP-style response’s body is
required for the calculation of the rspauth value.

Type
[IANA: use 120 if possible] for Digest-HA1
Length
>= 34
Text
This attribute contains the hexadecimal representation of H(A1)
as described in [RFC2617], section 3.1.3, 3.2.1 and 3.2.2.2.
It SHOULD be used in Access-Accept messages if the required
quality of protection (‘qop’) is ‘auth-int’.
This attribute MUST NOT be sent if the qop parameter was not
specified or has a value of ‘auth’ (in this case, use
Digest-Response-Auth instead).
The Digest-HA1 attribute MUST only be sent by the RADIUS server or processed by the RADIUS client if at least one of the following conditions is true:

+ The Digest-Algorithm attribute’s value is ‘MD5-sess’ or ‘AKA\text{-}v1\text{-}MD5\text{-}sess’.
+ The authenticity and integrity of the Access-Accept message is secured by cryptographic or equivalently secure means.

This attribute MUST only be used in Access-Accept messages.
3. Detailed Description

3.1 RADIUS Client Behavior

If a RADIUS client has no encrypted or otherwise secured connection to its RADIUS server, it MUST NOT accept secured connections (like https or sips) from its HTTP-style clients (or else the HTTP-style clients would have a false sense of security).

On reception of an HTTP-style request message, the RADIUS client looks for a (Proxy-)Authorization header where the realm directive matches its locally configured realm value. If such a header is present and contains HTTP digest information, the RADIUS client checks the 'nonce' parameter. If the RADIUS client generates nonces but did not issue the received nonce, it responds with a 401 (Unauthorized) or 407 (Proxy Authentication Required) to the HTTP-style client. In this error response, the RADIUS client sends a new nonce.

If the RADIUS client recognizes the nonce or does not generate nonces, it takes the header directives and puts them into a RADIUS Access-Request message. It puts the 'response' directive into a Digest-Response attribute and the realm / nonce / digest-uri / qop / algorithm / cnonce / nc / username / opaque directives into the respective Digest-Realm / Digest-Nonce / Digest-URI / Digest-Qop / Digest-Algorithm / Digest-CNonce / Digest-Nonce-Count / Digest-Username / Digest-Opaque attributes. The request method is put into the Digest-Method attribute. Now, the RADIUS client sends the Access-Request message to the RADIUS server.

The RADIUS server processes the message and responds with an Access-Accept or an Access-Reject message.

The RADIUS client constructs an Authentication-Info header:
- If the Access-Accept message contains a Digest-Response-Auth attribute, the RADIUS client checks the Digest-Qop attribute:
  * If the Digest-Qop attribute's value is 'auth' or not specified, the RADIUS client puts the Digest-Response-Auth attribute’s content into the Authentication-Info header’s 'rspauth' directive of the HTTP-style response.
  * If the Digest-Qop attribute's value is 'auth-int', the RADIUS client ignores the Access-Accept message and behaves like it had received an Access-Reject message (Digest-Response-Auth can’t be correct as the RADIUS server does not know the contents of the HTTP-style response’s body).
- If the Access-Accept message contains a Digest-HA1 attribute, the RADIUS client checks the 'qop' and 'algorithm' directives in the Authorization header of the HTTP-style request it wants to
authorize:
* If the ‘qop’ directive is missing or its value is ‘auth’, the RADIUS client ignores the Digest-HA1 attribute. It does not include an Authentication-Info header into its HTTP-style response.
* If the ‘qop’ directive’s value is ‘auth-int’ and at least one of the following conditions is true, the RADIUS client calculates the contents of the HTTP-style response’s ‘rspauth’ directive:
  + The algorithm directive’s value is ‘MD5-sess’ or ‘AKAv1-MD5-sess’.
  + The Access-Accept message was secured by cryptographic or equivalently secure means.
It creates the HTTP-style response message and calculates the hash of this message’s body. It uses the result and the Digest-URI attribute’s value of the corresponding Access-Request message to perform the H(A2) calculation. It takes the Digest-Nonce, Digest-Nonce-Count, Digest-CNonce and Digest-Qop values of the corresponding Access-Request and the Digest-HA1 attribute’s value to finish the computation of the ‘rspauth’ value.
* If the Access-Accept message contains neither a Digest-Response-Auth nor a Digest-HA1 attribute, the RADIUS client will not create an Authentication-Info header for its HTTP-style response.

The RADIUS server MAY have added a Digest-Nextnonce attribute into an Access-Accept message. If the RADIUS client discovers this, it puts the contents of this attribute into a ‘nextnonce’ directive. Now it can send an HTTP-style response.

If the RADIUS client did receive an HTTP-style request without a (Proxy-)Authorization header matching its locally configured realm value, it obtains a new nonce and sends an error response (401 or 407) containing a (Proxy-)Authenticate header.

If the RADIUS client receives an Access-Reject or no response from the RADIUS server, it sends an error response to the HTTP-style request it has received.

The RADIUS client has three ways to obtain nonces: it generates them locally, it has received one in a Digest-Nextnonce attribute of a previously received Access-Accept message, or it asks the RADIUS server for one. To do the latter, it sends an Access-Request containing a Digest-Method and a Digest-URI attribute but without a Digest-Nonce attribute. The RADIUS server chooses a nonce and responds with an Access-Challenge containing a Digest-Nonce attribute.
The RADIUS server can send Digest-Qop, Digest-Algorithm, Digest-Realm, Digest-Domain and Digest-Opaque attributes in the Access-Challenge carrying the nonce. If these attributes are present, the client MUST use them.

If the RADIUS client receives an Access-Challenge message in response to an Access-Request containing a Digest-Nonce attribute, the RADIUS server did not accept the nonce. If a Digest-Stale attribute is present in the Access-Challenge and has a value of ‘true’ (without quotes), the RADIUS client sends an error (401 or 407) response containing WWW-/Proxy-Authenticate header with the directive ‘stale’ and the digest directives derived from the Digest-* attributes.

3.2 RADIUS Server Behavior

If the RADIUS server receives an Access-Request message with a Digest-Method and a Digest-URI attribute but without a Digest-Nonce attribute, it chooses a nonce. It puts the nonce into a Digest-Nonce attribute and sends it in an Access-Challenge message to the RADIUS client. The RADIUS server MUST add Digest-Algorithm, Digest-Realm, SHOULD add one or more Digest-Qop and MAY add Digest-Domain, Digest-Opaque attributes to the Access-Challenge message. If the server cannot choose a nonce, it replies with an Access-Reject message.

If the RADIUS server receives an Access-Request message containing a Digest-Response attribute, it looks for the following attributes: Digest-Realm, Digest-Nonce, Digest-Method, Digest-URI, Digest-Qop, Digest-Algorithm, Digest-Username. Depending on the content of Digest-Algorithm and Digest-Qop, it looks for Digest-Entity-Body-Hash, Digest-CNonce and Digest-AKA-Auts, too. See [RFC2617] and [RFC3310] for details. If it has issued a Digest-Opaque attribute along with the nonce, the Access-Request MUST have a matching Digest-Opaque attribute.

If mandatory attributes are missing, it MUST respond with an Access-Reject message. If the attributes are present, the RADIUS server calculates the digest response as described in [RFC2617]. To look up the password, the RADIUS server uses the RADIUS User-Name attribute. All other values are taken from the Digest attributes described in this document. If the calculated digest response equals the string received in the Digest-Response attribute, the authentication was successful. If not, the RADIUS server responds with an Access-Reject.

If the authentication was successful, the RADIUS server adds an attribute to the Access-Accept message which can be used by the RADIUS client to construct an Authentication-Info header:
o If the Digest-Qop attribute’s value is ‘auth’ or unspecified, the RADIUS server SHOULD put a Digest-Response-Auth attribute into the Access-Accept message.

o If the Digest-Qop attribute’s value is ‘auth-int’ and at least one of the following conditions is true, the RADIUS server SHOULD put a Digest-HA1 attribute into the Access-Accept message:
  * The Digest-Algorithm attribute’s value is ‘MD5-sess’ or ‘AKAv1-MD5-sess’.
  * The authenticity and integrity of the Access-Accept message is secured by cryptographic or equivalently secure means.

In all other cases, Digest-Response-Auth or Digest-HA1 MUST NOT be sent.

RADIUS servers issuing nonces MAY construct a Digest-Nextnonce attribute and add it to the Access-Accept message. This is useful to limit the lifetime of a nonce and to save a round-trip in future requests (see nextnonce discussion in [RFC2617], section 3.2.3).

If the RADIUS server does not accept the nonce received in an Access-Request message but authentication was successful, the RADIUS server MUST send an Access-Challenge message containing a Digest-Stale attribute set to ‘true’ (without quotes). The RADIUS server MUST add Digest-Nonce, Digest-Algorithm, Digest-Realm, SHOULD add one or more Digest-Qop and MAY add Digest-Domain, Digest-Opaque attributes to the Access-Challenge message.
4. Migration Path to Diameter

The attributes specified in this document correspond to some AVPs defined in [I-D.ietf-aaa-diameter-sip-app].

4.1 RADIUS Client, Diameter Server

If an Access-Request message contains a Digest-Nonce attribute, the gateway maps all Digest-* attributes to a Diameter SIP-Authorization AVP. If the Access-Request message contains a Digest-Method and a Digest-URI attribute but no Digest-Nonce attribute, the gateway maps the RADIUS attributes to Diameter according to Table 1. The gateway constructs a MAR message and sends it to the Diameter server.

```
+---------------+------------+
| RADIUS        | Diameter   |
+---------------+------------+
| Digest-Method | SIP-Method |
| Digest-URI    | SIP-AOR    |
+---------------+------------+
```

Table 1

The Diameter Server responds with a MAA message. This message contains a Result-Code AVP set to the value DIAMETER_MULTI_ROUND_AUTH and challenge parameters in a SIP-Authenticate AVP. The gateway translates the AVPs of SIP-Authenticate AVP and puts the resulting RADIUS attributes into an Access-Challenge message. It sends the Access-Challenge message to the RADIUS client.

The gateway maps an Access-Request message containing a Digest-Response attribute to an MAR message with a Diameter SIP-Authorization AVP. All RADIUS attributes of the Access-Request message are mapped to the corresponding Diameter AVPs. The gateway sends the MAR message to the Diameter server.

If the authentication was successful, the Diameter server replies with an MAA containing a SIP-Authentication-Info and a Digest-Response AVP. The gateway converts these AVPs to the corresponding RADIUS attributes and constructs a RADIUS message. If the Result-Code AVP is Diameter_SUCCESS, an Access-Accept is sent. In all other cases, an Access-Reject is sent.

If the Diameter found the nonce to be stale, it will respond with a new challenge in a SIP-Authenticate AVP of an MAA message. The gateway handles this MAA like the first MAA message containing challenge parameters, as described in above.
4.2 Diameter Client, RADIUS Server

The Diameter client sends a Diameter MAR to the gateway. If the MAR message does not contain SIP-Auth-Data-Item AVPs, the gateway constructs an Access-Request message and maps the SIP-AOR and SIP-Method AVPs to RADIUS attributes according to Table 1. The gateway sends the Access-Request message to the RADIUS server which will respond with an Access-Challenge. The gateway creates a MAA message with a Result-Code AVP set to DIAMETER_MULTI_ROUND_AUTH and maps the Digest-* attributes to Diameter AVPs in a SIP-Authenticate AVP. The gateway sends the resulting MAA to the Diameter client, which will respond with a new MAR.

The gateway checks the SIP-Auth-Data-Item AVPs of this MAR for an AVP where the Digest-Realm AVP matches the locally configured realm value. It takes the AVPs from this SIP-Auth-Data-Item AVP, converts them into the corresponding RADIUS attributes and constructs a RADIUS Access-Request message. The gateway sends the Access-Request message to the RADIUS server. If the RADIUS server responds with an Access-Accept message, the gateway converts the RADIUS attributes to Diameter AVPs, constructs a MAR with a Result-Code AVP set to DIAMETER_SUCCESS and sends this message to the Diameter client. If the RADIUS server responds with an Access-Reject message, the gateway converts the RADIUS attributes to Diameter AVPs, constructs a MAR with a Result-Code AVP set to DIAMETER_ERROR_IDENTITIES_DONT_MATCH and sends this message to the Diameter client.

4.3 Limitations

This document covers not all functionality found in [I-D.ietf-aaa-diameter-sip-app].

- There is no equivalent to Diameter’s UAR/UAA, SAR/SAA, LIR/LIA, RTR/RTA and PPR/PPA messages
- The operational mode where the Diameter server sends the expected digest response to the client is not supported.

The operational mode where the RADIUS client chooses nonces is not supported in [I-D.ietf-aaa-diameter-sip-app].
5. Example

This is an example sniffed from the traffic between a softphone (A), a Proxy Server (B) and example.com RADIUS server (C). The communication between the Proxy Server and a SIP PSTN gateway is omitted for brevity. The SIP messages are not shown completely.

A→B

INVITE sip:97226491335@10.0.69.38 SIP/2.0

B→A

SIP/2.0 100 Trying

B→A

SIP/2.0 407 Proxy Authentication Required
Proxy-Authenticate: Digest realm="example.com",
nonce="3badala0", algorithm="md5"
Content-Length: 0

A→B

ACK sip:97226491335@10.0.69.38 SIP/2.0

A→B

INVITE sip:97226491335@10.0.69.38 SIP/2.0
Proxy-Authorization: Digest algorithm="md5",
nonce="3badala0",
opaque="", realm="example.com"
response="f3ce87e6984557cd0feecc26f3c5e97a4"
uri="sip:97226491335@10.0.69.38", username="12345678"

B→C

Code = 1 (Access-Request)
Attributes:
NAS-IP-Address = a 0 45 26 (10.0.69.38)
NAS-Port-Type = 5 (Virtual)
User-Name = "12345678"
Digest-Response = "f3ce87e6984557cd0fecc26f3c5e97a4"
Digest-Realm = "example.com"
Digest-Nonce = "3badala0"
Digest-Method = "INVITE"
Digest-URI = "sip:97226491335@10.0.69.38"
Digest-Algorithm = "md5"
Digest-Username = "12345678"

C->B
Code = 2 (Access-Accept)
Attributes:
Digest-Response-Auth = "6303c41b0e2c3e524e413c4e8c0e954"

B->A
SIP/2.0 180 Ringing

B->A
SIP/2.0 200 OK

A->B
ACK sip:97226491335@10.0.69.38:5060 SIP/2.0

A second example shows the traffic between a web browser (A), web server (B) and a RADIUS server (C).

A->B
GET /index.html HTTP/1.1

B->A
HTTP/1.1 407 Authentication Required
WWW-Authenticate: Digest realm="example.com",
domain="/index.html",
nonce="a3086ac8", algorithm="md5"
Content-Length: 0

A->B

GET /index.html HTTP/1.1
Authorization: Digest algorithm="md5", nonce="a3086ac8",
,opaque="",realm="example.com"
, response="f052b68058b2987aba493857ae1ab002"
, uri="/index.html", username="12345678"

B->C

Code = 1 (Access-Request)
Attributes:
NAS-IP-Address = a 0 45 26 (10.0.69.38)
NAS-Port-Type = 5 (Virtual)
User-Name = "12345678"
Digest-Response = "f052b68058b2987aba493857ae1ab002"
Digest-Realm = "example.com"
Digest-Nonce = "a3086ac8"
Digest-Method = "GET"
Digest-URI = "/index.html"
Digest-Algorithm = "md5"
Digest-Username = "12345678"

C->B

Code = 2 (Access-Accept)
Attributes:
Digest-Response-Auth =
"e644aa513effbf61caff67103ff6433c"

B->A

HTTP/1.1 200 OK
...
<html>
...
6. IANA Considerations

This document serves as IANA registration request for a number of values from the RADIUS attribute type number space:

<table>
<thead>
<tr>
<th>placeholder</th>
<th>value assigned by IANA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digest-Response</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Realm</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Nonce</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Noncnonce</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Response-Auth</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Method</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-URI</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Qop</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Algorithm</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Entity-Body-Hash</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-CNnonce</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Noncense-Count</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Username</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Opaque</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Auth-Param</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-AKA-Auts</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Domain</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Stale</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-HA1</td>
<td>TBD</td>
</tr>
</tbody>
</table>

Table 2
7. Security Considerations

The RADIUS extensions described in this document make RADIUS a transport protocol for the data that is required to perform a digest calculation. It adds the vulnerabilities of HTTP Digest (see [RFC2617], section 4) to those of RADIUS (see [RFC2865], section 8 or here [1]).

If an attacker gets access to a RADIUS client or RADIUS proxy, it can perform man-in-the-middle attacks even if the connections between A, B and B, C (Figure 2) have been secured with TLS or IPSec.

SIP or HTTP requests occur much more frequently than dial-in requests. RADIUS servers implementing this specification must meet that additional performance requirements. An attacker could try to overload the RADIUS infrastructure by excessively sending SIP or HTTP requests. This kind of attack was more difficult when RADIUS was just used for dial-in authentication: the attacker could be identified by the DSL / Cable interface or with some help of the PSTN provider.

To make simple denial of service attacks more difficult, the nonce issuer (RADIUS client or server) MUST check if it has generated the nonce received from an HTTP-style client. This SHOULD be done statelessly. For example, a nonce could consist of a cryptographically random part and some kind of signature of the RADIUS client, as described in [RFC2617], section 3.2.1.

RADIUS servers MAY include Digest-Qop and Digest-Algorithm attributes in Access-Challenge messages. A man in the middle can modify or remove those attributes in a bidding down attack. In this case, the RADIUS client would use a weaker authentication scheme than intended. **RFC 3579** [RFC3579], section 3.2 describes a Message-Authenticator attribute which MUST be used to improve the integrity protection of RADIUS messages.

The Digest-HA1 attribute contains no random components if the algorithm is 'MD5' or 'AKAv1-MD5'. This makes offline dictionary attacks easier and can be used for replay attacks.

HTTP-style clients can use TLS with server side certificates together with HTTP-Digest authentication. Instead of TLS, IPSec can be used, too. TLS or IPSec secure the connection while Digest Authentication authenticates the user. If a RADIUS client accepts such connections, it MUST have an equally secure connection to the RADIUS server.
8. Change Log

8.1 Changes from draft-sterman-aaa-sip-04

- clarified usage of Digest-HA1
- clarified usage of Digest-Stale (is sent in an Access-Challenge now)
- clarified allowed attribute usage for message types
- changed attribute type to ‘Text’ where the corresponding Diameter AVPs have a UTF8String
- added Diameter client - RADIUS server handling

8.2 Changes from draft-sterman-aaa-sip-03

- addressed ‘auth-int’ issue
- New Digest-Nextnonce attribute
- revised abstract, motivational section and examples
- Access-Challenge instead of ‘Access-Accept carrying a Digest-Nonce attribute’
- shortened SIP messages in example, removed real-world addresses and product names

8.3 Changes from draft-sterman-aaa-sip-02

- Relaxed restrictions for Digest-Domain, Digest-Realm, Digest-Opaque, Digest-Qop and Digest-Algorithm
- Additional security considerations for Digest-Domain, Digest-Qop and Digest-Algorithm usage in Access-Accept messages

8.4 Changes from draft-sterman-aaa-sip-01

- Replaced Sub-attributes with flat attributes
- aligned naming with [I-D.ietf-aaa-diameter-sip-app]
- Added how a server must treat unknown attributes.
- Added a section ‘Migration path to Diameter’
- Added an optional attribute for support of the digest scheme described in informational [RFC3310].
- Added a mode of operation where the RADIUS server chooses the nonce. This was required for AKA [RFC3310], but can be useful for ordinary Digest authentication when the qop directive is not used. This required the addition of several attributes.
9. References

9.1 Normative References


9.2 Informative References


[RFC3579] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible

URIs


Authors’ Addresses

Baruch Sterman
Kayote Networks
P.O. Box 1373
Efrat 90435
Israel

EMail: baruch@kayote.com

Daniel Sadolevsky
SecureOL, Inc.
Jerusalem Technology Park
P.O. Box 16120
Jerusalem 91160
Israel

EMail: dscreat@dscreat.com

David Schwartz
Kayote Networks
P.O. Box 1373
Efrat 90435
Israel

EMail: david@kayote.com

David Williams
Cisco Systems
7025 Kit Creek Road
P.O. Box 14987
Research Triangle Park NC 27709
USA

EMail: dwilli@cisco.com
Internet-Draft    RADIUS Extension for Digest Authentication  December 2004

Wolfgang Beck
Deutsche Telekom AG
Am Kavalleriesand 3
Darmstadt  64295
Germany

EMail: beckw@t-systems.com
Appendix A. Acknowledgments

We would like to acknowledge Kevin Mcdermott (Cisco Systems) for providing comments and experimental implementation.

Many thanks to all reviewers, especially to Miguel Garcia, Jari Arrko, Avi Lior and Jun Wang.