RADIUS Extension for Digest Authentication
draft-ietf-radext-digest-auth-04.txt

Status of this Memo

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 becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

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 March 3, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This document defines an extension to the RADIUS protocol to enable
support of Digest Authentication, for use with HTTP-style protocols like SIP and HTTP.

Table of Contents

1. Introduction ............................................. 4
   1.1. Terminology ........................................... 4
   1.2. Motivation ............................................ 4
   1.3. Overview .............................................. 5
       1.3.1. Scenario 1, RADIUS client chooses nonces ......... 6
       1.3.2. Scenario 2, RADIUS server chooses nonces ....... 7
2. Interoperability ......................................... 9
3. Detailed Description ..................................... 9
   3.1. RADIUS Client Behavior ............................... 9
   3.2. RADIUS Server Behavior .............................. 12
4. New RADIUS attributes .................................. 14
   4.1. Digest-Response attribute ............................ 14
   4.2. Digest-Realm attribute ............................... 14
   4.3. Digest-Nonce attribute ............................... 15
   4.4. Digest-Response-Auth attribute ....................... 15
   4.5. Digest-Nextnonce attribute .......................... 16
   4.6. Digest-Method attribute .............................. 16
   4.7. Digest-URI attribute ................................. 16
   4.8. Digest-Qop attribute ................................ 17
   4.9. Digest-Algorithm attribute ........................... 17
   4.10. Digest-Entity-Body-Hash attribute ................... 18
   4.11. Digest-CNonce attribute ............................. 18
   4.12. Digest-Nonce-Count attribute ......................... 19
   4.13. Digest-Username attribute ........................... 19
   4.14. Digest-Opaque attribute ............................. 20
   4.15. Digest-Auth-Param attribute ........................ 20
   4.16. Digest-AKA-Auts attribute ........................... 21
   4.17. Digest-Domain attribute ............................. 21
   4.18. Digest-Stale attribute .............................. 21
   4.19. Digest-HA1 attribute ................................. 22
   4.20. SIP-AOR .............................................. 22
5. Table of Attributes ..................................... 23
6. Example .................................................... 24
7. IANA Considerations ..................................... 28
8. Security Considerations ................................... 28
9. Acknowledgments ......................................... 30
10. References ............................................... 30
    10.1. Normative References ................................. 30
    10.2. Informative References .............................. 31
Appendix A. Change Log ..................................... 31
    A.1. Changes from draft-ietf-radext-digest-auth-03 .......... 31
    A.2. Changes from draft-ietf-radext-digest-auth-02 .......... 31
A.3. Changes from draft-ietf-radext-digest-auth-01 ........ 32
A.4. Changes from draft-ietf-radext-digest-auth-00 ........ 32
A.5. Changes from draft-sterman-aaa-sip-04 .............. 32
A.6. Changes from draft-sterman-aaa-sip-03 .............. 32
A.7. Changes from draft-sterman-aaa-sip-02 .............. 32
A.8. Changes from draft-sterman-aaa-sip-01 .............. 33
Authors’ Addresses ........................................... 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].

The use of normative requirement key words in this document shall
apply only to RADIUS Client and RADIUS Server implementations that
include the features described in this document. This document
creates no normative requirements for existing implementations.

HTTP-style protocol
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.

NAS
Network Access Server, the RADIUS client.

nonce
An unpredictable value used to prevent replay attacks. The
nonce generator may use cryptographic mechanisms to produce
nonces it can recognize without maintaining state.

protection space
The combination of realm and digest URI, the use of which is
authorized by the RADIUS server.

SIP UA
SIP User Agent, an Internet endpoint that uses the Session
Initiation Protocol.

SIP UAS
SIP User Agent Server, a logical entity that generates a
response to a SIP (Session Initiation Protocol) request.

1.2. Motivation

The HTTP Digest Authentication mechanism, defined in [RFC2617], was
subsequently adapted to use with SIP in [RFC2543] (obsoleted by
[RFC3261]). Due to the limitations and weaknesses of Digest
Authentication (see [RFC2617], section 4), additional authentication
and encryption mechanisms are defined in SIP [RFC3261], including TLS
[RFC2246] and S/MIME [RFC2633]. However, Digest Authentication has
been widely implemented within SIP clients and to support those
clients there is a need for support of Digest Authentication within
AAA protocols such as RADIUS [RFC2865] and Diameter [RFC3588].

This document defines an extension to the RADIUS protocol to enable
support of Digest Authentication, for use with SIP, HTTP, and other
HTTP-style protocols using this authentication method. Support for
Digest mechanisms such as AKA [RFC3310] is also supported. A companion document [I-D.ietf-aaa-diameter-sip-app] defines support for Digest Authentication within Diameter.

1.3. Overview

HTTP digest is a challenge-response protocol used to authenticate a client’s request to access some resource on a server. Figure 1 shows a single HTTP digest transaction.

Figure 1: digest operation without RADIUS

If the client sends a request without any credentials (1), the server will reply with an error response (2) containing a nonce. The client creates a cryptographic digest from parts of the request, from the nonce it received from the server, and a shared secret. The client re-transmits the request (3) to the server, but now includes the digest within the packet. The server does the same digest calculation as the client and compares the result with the digest it received in (3). If the digest values are identical, the server grants access to the resource and sends a positive response to the client (4). If the digest values differ, the server sends a negative response to the client (4).

Instead of maintaining a local user database, the server could use RADIUS to access a centralized user database. However, RADIUS [RFC2865] does not include support for HTTP digest authentication. The RADIUS client can not use the User-Password attribute, since it does not receive a password from the HTTP-style client. The CHAP-Challenge and CHAP-Password attributes are also not suitable since
the CHAP algorithm is not compatible with HTTP digest.

This document defines new attributes that enable the RADIUS server to perform the digest calculation defined in [RFC2617], providing support for Digest Authentication as a native authentication mechanism within RADIUS.

This document defines new attributes that enable the RADIUS server to perform the digest calculation defined in [RFC2617].

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. This specification assumes that both the RADIUS client and server are appropriately configured to generate the nonces in either the RADIUS client or the RADIUS server, but not in both at the same time. Implementations, though, do not have the means to verify this behavior.

1.3.1. Scenario 1, RADIUS client chooses nonces

```

---+   +----+    (1)    +----+           +----+
    |   |<==========|   |           |   |
    A   |    (2)    |   |           |   |
    |   |<==========|   |           |   |
    |   |    (3)    |   |           |   |
    |   |<==========|   |           |   |
    +   |           |   |    (4)    |   |
         |           |   |----------|   |
         |           |   |    (5)    |   |
         |           |   |<----------|   |
         |           |   |    (6)    |   |
         +<========+   +--------+   +-----+

=>>> HTTP/SIP
----> RADIUS
```

Figure 2: 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

While the usage scenario described in Section 1.3.1 minimizes the load on the RADIUS server, alternatives are required in some situations. When using AKA [RFC3310] the nonce is partially derived from a precomputed authentication vector, which is often stored centrally.

Figure 3 depicts a scenario in which the RADIUS server chooses nonces. In this case entities A and B communicate using HTTP or SIP, while entities B and C communicate 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 following messages are sent in this scenario:

A sends B an HTTP/SIP request without an authorization header (step 1). B sends an Access-Request packet 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. Interoperability

An implementation supporting this extension MUST include a Digest-Response attribute within an Access-Request packet where Digest Authentication is desired. An Access-Request MUST NOT contain both a Digest-Response attribute and another authentication attribute, such as User-Password, CHAP-Password, or EAP-Message.

RADIUS clients and servers MUST support both nonce generation modes. As there is no automatic capability exchange, the operator MUST make sure that the RADIUS client software uses the correct nonce generation mode when accessing a specific RADIUS server:

- If the RADIUS server generates nonces, its RADIUS clients MUST NOT try to generate nonces.
- If the RADIUS server does not generate nonces, its 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.

RADIUS implementations MUST offer respective configuration options.

3. Detailed Description

3.1. RADIUS Client Behavior

The attributes described in this document are sent in cleartext. Therefore were a RADIUS client to accept secured connections (https or sips) from HTTP-style clients, this could result in information intentionally protected by HTTP-style clients being sent in the clear during the RADIUS exchange.

If the packets between RADIUS client and RADIUS server are not protected with IPsec, the RADIUS client MUST NOT accept secured connections (like https or sips) from its HTTP-style clients, so that HTTP-style clients are not provided with a false sense of security.

On reception of an HTTP-style request message, the RADIUS client checks whether it is authorized to authenticate the request. Where an HTTP-style request traverses several proxies and each of the proxies requests to authenticate the HTTP-style client, the request at the HTTP-style server may contain multiple credential sets.

The RADIUS client can use the ‘realm’ directive in HTTP to determine which credentials are applicable. Where none of the realms are of
interest, the RADIUS client MUST behave as though no relevant credentials were sent. In all situations the RADIUS client MUST send zero or exactly one credential to the RADIUS server. The RADIUS client MUST choose the credential of the (Proxy-)Authorization header if the realm directive matches its locally configured realm.

If such a (Proxy-)Authorization 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 packet. 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. The RADIUS client adds a Message-Authenticator attribute, defined in [RFC3579] and sends the Access-Request packet to the RADIUS server.

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

The RADIUS client constructs an Authentication-Info header:
- If the Access-Accept packet 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 packet and behaves like it had received an Access-Reject packet (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 packet 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 packets between RADIUS client and RADIUS server are protected with IPsec (see Section 8).

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 packet 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 packet 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 packet. 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 from the RADIUS server, it sends an error response to the HTTP-style request it has received. If the RADIUS client does not receive a response, it retransmits or fails over to another RADIUS server as described in [RFC2865].

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 packet, 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. It adds a Message-Authenticator (see [RFC3579]) attribute to the Access-Request packet. 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 packet 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 packet 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 packet to the RADIUS client. The RADIUS server MUST add Digest-Realm, Message-Authenticator (see [RFC3579]), SHOULD add Digest-Algorithm, one or more Digest-Qop and MAY add Digest-Domain, Digest-Opaque attributes to the Access-Challenge packet. If the server cannot choose a nonce, it replies with an Access-Reject packet.

If the RADIUS server receives an Access-Request packet 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 the Digest-Algorithm attribute is missing, 'MD5' is assumed. If the RADIUS server 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 packet. 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. The RADIUS server MUST check if the user identified by the User-Name attribute
- is authorized to access the protection space defined by the Digest-URI and Digest-Realm attributes,
- is authorized to use the URI included in the SIP-AOR attribute, if this attribute is present.

If any of those checks fails, the RADIUS server MUST send an Access-Reject.

Correlation between User-Name and SIP-AOR AVP values is required just
to avoid that any user can register or misuse a SIP-AOR allocated to another user.

A RADIUS server MUST check if the RADIUS client is authorized to serve users of the realm mentioned in the Digest-Realm attribute. If the RADIUS client is not authorized, the RADIUS server MUST send an Access-Reject. The RADIUS server SHOULD log the event so as to notify the operator, and MAY take additional action such as sending an Access-Reject in response to all future requests from this client, until this behavior is reset by management action.

All values required for the digest calculation are taken from the Digest attributes described in this document. If the calculated digest response equals the value 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 packet which can be used by the RADIUS client to construct an Authentication-Info header:
- 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 packet
- 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 packet:
  - The Digest-Algorithm attribute’s value is ‘MD5-sess’ or ‘AKAv1-MD5-sess’.
  - The packets between RADIUS client and RADIUS server are protected with IPsec (see Section 8).

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 packet. 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). The RADIUS server adds a Message-Authenticator attribute (see [RFC3579]) and sends the Access-Accept packet to the RADIUS client.

If the RADIUS server does not accept the nonce received in an Access-Request packet but authentication was successful, the RADIUS server MUST send an Access-Challenge packet containing a Digest-Stale attribute set to ‘true’ (without quotes). The RADIUS server MUST add Message-Authenticator (see [RFC3579]), Digest-Nonce, Digest-Realm, SHOULD add Digest-Algorithm, one or more Digest-Qop and MAY add Digest-Domain, Digest-Opaque attributes to the Access-Challenge packet.
4. New RADIUS attributes

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

```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

4.1. Digest-Response attribute

**Description**

If this attribute is present in an Access-Request message, a RADIUS server implementing this specification MUST treat the Access-Request as a request for Digest Authentication. When a RADIUS client receives a (Proxy-)Authorization header, it puts the request-digest value into a Digest-Response attribute. This attribute (which enables the user to prove possession of the password) MUST only be used in Access-Requests.

**Type**

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

**Length**

>= 3

**Text**

When using HTTP digest, the text field is 32 octets long and contains a hexadecimal representation of 16 octet digest value as it was calculated by the authenticated client. Other digest algorithms MAY define different digest lengths. The text field MUST be copied from request-digest of digest-response ([RFC2617]) without quotes.

4.2. Digest-Realm attribute

**Description**

This attribute describes a protection space of the RADIUS server. See [RFC2617] 1.2 for details. It MUST only be used in Access-Request and Access-Challenge packets.

**Type**
4.3. Digest-Nonce attribute

Description
This attribute holds a nonce to be used in the HTTP Digest calculation. 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 packet. This attribute MUST only be used in Access-Request and Access-Challenge packets.

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. In Access-Challenge packets, the attribute contains the nonce selected by the RADIUS server.

4.4. Digest-Response-Auth attribute

Description
This attribute enables the RADIUS server to prove possession of the password. If the previously received Digest-Qop attribute was ‘auth-int’ (without quotes), the RADIUS server MUST send a Digest-HA1 attribute instead of a Digest-Response-Auth attribute. The Digest-Response-Auth attribute MUST only be used in Access-Accept packets. The RADIUS client puts the attribute value without quotes into the rspauth directive of the Authentication-Info header.

Type
The RADIUS server calculates a digest according to section 3.2.3 of [RFC2617] and copies the result into this attribute. Other digest algorithms than the one defined in [RFC2617] MAY define digest lengths other than 32.

4.5. Digest-Nextnonce attribute

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

Description
If the RADIUS server is configured to choose nonces it MAY put a Digest-Nextnonce attribute into an Access-Accept packet. 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 packets.

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

Length
>=3

Text
It is recommended that this text be base64 or hexadecimal data.

4.6. Digest-Method attribute

Description
This attribute holds the method value to be used in the HTTP Digest calculation. This attribute MUST only be used in Access-Request packets.

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.

4.7. Digest-URI attribute
Description
This attribute is used to transport the contents of the digest-uri directive or the URI of the HTTP-style request. It MUST only be used in Access-Request packets.

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.

4.8. Digest-Qop attribute

Description
This attribute holds the Quality of Protection parameter that influences the HTTP Digest calculation. This attribute MUST only be used in Access-Request and Access-Challenge packets. A RADIUS client SHOULD insert one of the Digest-Qop attributes it has received in a previous Access-Challenge packet. RADIUS servers SHOULD insert at least one Digest-Qop attribute in an Access-Challenge packet. Digest-Qop is optional in order to preserve backward compatibility with a minimal implementation of [RFC2069].

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. In Access-Challenge packets, the RADIUS server puts a desired qop-value into this attribute. If the RADIUS server supports more than one "quality of protection" value, it puts each qop-value into a separate Digest-Qop attribute.

4.9. Digest-Algorithm attribute
This attribute holds the algorithm parameter that influences the HTTP Digest calculation. It MUST only be used in Access-Request and Access-Challenge packets. If this attribute is missing, "MD5" is assumed.

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 packets, the RADIUS server SHOULD put the desired algorithm into this attribute.

4.10. Digest-Entity-Body-Hash attribute

Description

When using the qop level ‘auth-int’, a hash of the HTTP-style 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 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.

Type

[IANA: use 111 if possible] for Digest-Entity-Body-Hash

Length

>=3

Text

The attribute holds the hexadecimal representation of H(entity-body). 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.

4.11. Digest-CNonce attribute
Description
This attribute holds the client nonce parameter that is used in
the HTTP Digest calculation. It MUST only be used in Access-
Request packets.

Type
[IANA: use 112 if possible] for Digest-CNonce

Length
>=3

Text
This attribute includes the value of the cnonce-value [RFC2617]
without quotes, taken from the HTTP-style request.

4.12. Digest-Nonce-Count attribute

Description
This attribute includes the nonce count parameter that is used
to detect replay attacks. The attribute MUST only be used in
Access-Request packets.

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.

4.13. Digest-Username attribute

Description
This attribute holds the user name used in the HTTP digest
calculation. The RADIUS server MUST use this attribute only
for the purposes of calculating the digest. In order to
determine the appropriate user credentials, the RADIUS server
MUST use the User-Name (1) attribute, and MUST NOT use the
Digest-Username attribute. This attribute MUST only be used in
Access-Request packets.

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.
4.14. Digest-Opaque attribute

Description
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 (i.e., the RADIUS client) without modification. This attribute is only used when the RADIUS server chooses nonces and MUST only be used in Access-Request and Access-Challenge packets.

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

Length
>=3

Text
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 packets, the RADIUS server MAY include this attribute.

4.15. Digest-Auth-Param attribute

Description
This attribute is a placeholder for future extensions and corresponds to the "auth-param" parameter defined in section 3.2.1 of [RFC2617]. The Digest-Auth-Param is the mechanism whereby the RADIUS client and RADIUS server can exchange auth-param extension parameters contained within 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 MUST ONLY be used in Access-Request, Access-Challenge, or Access-Accept packets.

Type  
[IANA: use 116 if possible] for Digest-Auth-Param

Length
>=3

Text
The text consists of the whole parameter, including its name and the equal ("=") sign and quotes.
4.16. Digest-AKA-Auts attribute

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

Type
[IANA: use 117 if possible] for Digest-AKA-Auts

Length
>=3

Text
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.

4.17. Digest-Domain attribute

Description
When a RADIUS client has asked for a nonce, the RADIUS server MAY send one or more Digest-Domain attributes in its Access-Challenge packet. 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). RADIUS servers MAY send one or more attributes of this type in Access-Challenge packets. This attribute MUST only be used in Access-Challenge packets.

Type
[IANA: use 118 if possible] for Digest-Domain

Length
3

Text
This attribute consists of a single URI, that defines a protection space.

4.18. Digest-Stale attribute

Description
This attribute is sent by a RADIUS server in order to notify the RADIUS client whether it has accepted a nonce. 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 packets and only if the RADIUS server chooses nonces.

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).

4.19. Digest-HA1 attribute

Description
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. It SHOULD be used in Access-Accept packets 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 'AKAv1-MD5-sess'.
+ The packets between RADIUS client and RADIUS server are protected with IPsec (see Section 8).

This attribute MUST only be used in Access-Accept packets.

Type
[IANA: use 120 if possible] for Digest-HA1

Length
>= 3

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.

4.20. SIP-AOR

Type
This attribute is used for the authorization of SIP messages. The SIP-AOR attribute identifies the URI the use of which must be authenticated and authorized. The RADIUS server uses this attribute to authorize the processing of the SIP request. The SIP-AOR can be derived from, e.g., the To header field in a SIP REGISTER request (user under registration), or the From header field in other SIP requests. However, the exact mapping of this attribute to SIP can change due to new developments in the
protocol. This attribute MUST only be used when the RADIUS client wants to authorize SIP users and MUST only be used in Access-Request packets.

Type

[IANA:use 121 if possible] for SIP-AOR

Length

>=3

Text

The syntax of this attribute corresponds either to a SIP URI (with the format defined in [RFC3261] or a TEL URI (with the format defined in [RFC3966]). The SIP-AOR attribute holds the complete URI, including parameters and other parts. It is up to the RADIUS server what components of the URI are regarded in the authorization decision.

5. Table of Attributes

The following table provides a guide to which attributes may be found in which kinds of packets, and in what quantity.
### Table 1

<table>
<thead>
<tr>
<th>Req</th>
<th>Accept</th>
<th>Reject</th>
<th>Challenge</th>
<th>#</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>User-Name</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>TBD</td>
<td>Message-Authenticator</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Response</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>TBD</td>
<td>Digest-Realm</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>TBD</td>
<td>Digest-Nonce</td>
</tr>
<tr>
<td>0</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Response-Auth</td>
</tr>
<tr>
<td>0</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-NextNonce</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Method</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-URI</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>1+</td>
<td>TBD</td>
<td>Digest-Qop</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>TBD</td>
<td>Digest-Algorithm (see Note 3)</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Entity-Body-Hash</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-CNonce</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Nonce-Count</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-Username</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>TBD</td>
<td>Digest-Opaque</td>
</tr>
<tr>
<td>0+</td>
<td>0+</td>
<td>0</td>
<td>0+</td>
<td>TBD</td>
<td>Digest-Auth-Param</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-AKA-Auts</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0+</td>
<td>TBD</td>
<td>Digest-Domain</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0-1</td>
<td>TBD</td>
<td>Digest-Stale</td>
</tr>
<tr>
<td>0</td>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>Digest-HA1 (see Note 1, 2)</td>
</tr>
<tr>
<td>0-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>TBD</td>
<td>SIP-AOR</td>
</tr>
</tbody>
</table>

[Note 1] Digest-HA1 MUST be used instead of Digest-Response-Auth if Digest-Qop is ‘auth-int’.
[Note 2] Digest-Response-Auth MUST be used instead of Digest-HA1 if Digest-Qop is ‘auth’.
[Note 3] If Digest-Algorith is missing, ‘MD5’ is assumed

### 6. 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@example.com SIP/2.0
   From: <sip:12345678@example.com>
   To: <sip:97226491335@example.com>

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@example.com SIP/2.0

A->B

INVITE sip:97226491335@example.com SIP/2.0
   Proxy-Authorization: Digest algorithm="md5",nonce="3badala0"
   ,opaque="",realm="example.com"
   ,response="f3ce87e6984557cd0fecc26f3c5e97a4"
   ,uri="sip:97226491335@10.0.69.38",username="12345678"
   From: <sip:12345678@example.com>
   To: <sip:97226491335@example.com>

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@example.com"
Digest-Algorithm = "md5"
Digest-Username = "12345678"
SIP-AOR = "sip:12345678@example.com"

C->B

Code = 2 (Access-Accept)
Attributes:
    Digest-Response-Auth = "6303c41b0e2c3e524e413cafe8cce954"

B->A

SIP/2.0 180 Ringing

B->A

SIP/2.0 200 OK

A->B

ACK sip:97226491335@example.com 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 = "e644aa513effbfelcaff67103ff6433c"

B->A

HTTP/1.1 200 OK
...
<html>
...
7. 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-Nextnonce</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-Algorithim</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-Entity-Body-Hash</td>
<td>TBD</td>
</tr>
<tr>
<td>Digest-CNonce</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>
<tr>
<td>SIP-AOR</td>
<td>TBD</td>
</tr>
</tbody>
</table>

Table 2

8. Security Considerations

The RADIUS extensions described in this document enable RADIUS to transport the data that required to perform a digest calculation. As a result, RADIUS inherits the vulnerabilities of HTTP Digest (see [RFC2617], section 4) in addition to RADIUS security vulnerabilities described in [RFC2865] Section 8 and [RFC3579] Section 4.

An attacker compromising a RADIUS client or proxy can carry out man-in-the-middle attacks even if the paths between A, B and B, C (Figure 2) have been secured with TLS or IPsec.

The RADIUS server MUST check the Digest-Realm attribute it has received from a client. If the RADIUS client is not authorized to serve HTTP-style clients of that realm, it might be compromised.
RADIUS clients implementing the extension described in this document may authenticate HTTP-style requests received over the Internet. As compared with use of RADIUS to authenticate link layer network access, an attacker may find it easier to cover their tracks in such a scenario.

An attacker can attempt a denial of service attack on one or more RADIUS servers by sending a large number of HTTP-style requests. 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 provided by the RADIUS client, as described in [RFC2617], section 3.2.1.

RADIUS servers SHOULD 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, causing the RADIUS client to use a weaker authentication scheme than intended.

The Message-Authenticator attribute, described in [RFC3579] section 3.2 MUST be included in Access-Request, Access-Challenge, Access-Reject and Access-Accept messages that contain attributes described in this specification.

The Digest-HA1 attribute contains no random components if the algorithm is ‘MD5’ or ‘AKAv1-MD5’. This makes offline dictionary attacks easier and enables 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. The RADIUS transaction can be regarded as one leg on the path between the HTTP-style client and the HTTP-style server. To prevent RADIUS from representing the weak link, a RADIUS client receiving an HTTP-style request via TLS or IPsec MUST use an equally secure connection to the RADIUS server. There are two ways to achieve this:

- the RADIUS client may reject HTTP-style requests received over TLS or IPsec
- the RADIUS client require that traffic be sent and received over IPsec.

RADIUS over IPsec, if used, MUST conform to the requirements described in [RFC3579] section 4.2.
9. 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 Arkko, Avi Lior and Jun Wang.

10. References

10.1. Normative References


10.2. Informative References

[I-D.ietf-aaa-diameter-sip-app]


Appendix A. Change Log

RFC editor: please remove this section prior to RFC publication.

A.1. Changes from draft-ietf-radext-digest-auth-03

- new 'Interoperability' section, requiring support for both nonce generation modes.
- removed Diameter migration path section (again)
- reference to server behavior in Security Considerations section
- fixed text/table mismatch regarding Digest-Domain attributes

A.2. Changes from draft-ietf-radext-digest-auth-02

- added Diameter migration path section (again)
- various typos
A.3. Changes from draft-ietf-radext-digest-auth-01

- removed Diameter migration path section
- included Digest-URI and Digest-Realm in the authorization decision, not just in the digest calculation
- RADIUS server must check if a RADIUS client is authorized to serve the realm mentioned in Digest-Realm
- moved 'Detailed Description' sections in front of 'New RADIUS attributes' section
- replaced 'IPsec or otherwise secured connection' with IPsec
- changed MAY to SHOULD for Digest-Algorithm in Access-Challenge
- changed type of Digest-Entity-Body-Hash to text (all other H([..) result attributes are hex and text, too)
- new abstract
- Terminology section changed
- 'Changes' section as appendix

A.4. Changes from draft-ietf-radext-digest-auth-00

- SIP-AOR attribute added
- clarified use of Digest-Qop
- attribute overview table added

A.5. 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

A.6. 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

A.7. Changes from draft-sterman-aaa-sip-02

- Relaxed restrictions for Digest-Domain, Digest-Realm, Digest-Opaque, Digest-Qop and Digest-Algorithm
A.8. 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.
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

Wolfgang Beck
Deutsche Telekom AG
Am Kavalleriesand 3
Darmstadt 64295
Germany

Email: beckw@t-systems.com
Intellectual Property Statement

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.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.