                  RADIUS Extension for Digest Authentication

Status of This Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

   This document defines an extension to the Remote Authentication
   Dial-In User Service (RADIUS) protocol to enable support of Digest
   Authentication, for use with HTTP-style protocols like the Session
   Initiation Protocol (SIP) and HTTP.

Table of Contents

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 the Session Initiation
     Protocol (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
     HTTP-style protocols differ in their definition of the
     protection space. For HTTP, it is defined as the combination
     of realm and canonical root URL of the requested resource for
     which the use is authorized by the RADIUS server. In the case
     of SIP, the realm string alone defines the protection space.

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 for use with SIP [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 Transport Layer Security (TLS)
[RFC4346] and Secure MIME (S/MIME) [RFC3851]. However, Digest
Authentication support is mandatory in SIP implementations, and
Digest Authentication is the preferred way for a SIP UA to
authenticate itself to a proxy server. Digest Authentication is used
in other protocols as well.

To simplify the provisioning of users, there is a need to support
this authentication mechanism within Authentication, Authorization,
and Accounting (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 Authentication and Key Agreement (AKA)
[RFC3310] is also supported. A companion document [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.

```
                        HTTP/SIP..
          +------------+   (1)      +------------+
          |            |--------->|            |
          | HTTP-style |   (2)      | HTTP-style |
          | client     |<---------|  server    |
          |            |   (3)      |            |
          |            |--------->|            |
          |            |   (4)      |            |
          |            |<---------|            |
          +-----------+          +-----------+
```
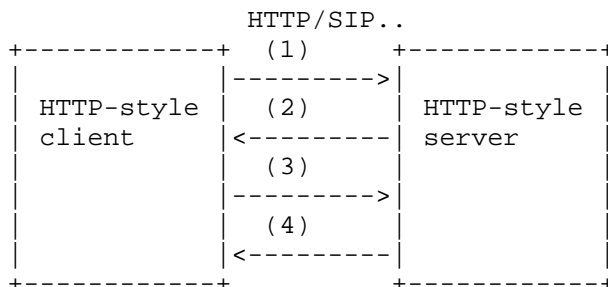
Figure 1: Digest operation without RADIUS

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 from 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 cannot use the User-Password attribute, since it
does not receive a password from the HTTP-style client. The
CHAP-Challenge and CHAP-Password attributes described in [RFC1994]
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.

The nonces required by the digest algorithm are generated by the
RADIUS server. Generating them in the RADIUS client would save a
round-trip, but introduce security and operational issues. Some
digest algorithms -- e.g., AKA [RFC3310] -- would not work.

Figure 2 depicts a scenario in which the HTTP-style server defers
authentication to a RADIUS server. Entities A and B communicate
using HTTP or SIP, while entities B and C communicate using RADIUS.

```
                  HTTP/SIP              RADIUS

       +-----+     (1)     +-----+               +-----+
       |     |=========>|     |     (2)     |     |     |
       |     |          |     |---------->|     |     |
       |     |          |     |     (3)     |     |     |
       |     |    (4)    |     |<----------|     |     |
       |     |<=========|     |          |     |     |
       |     |    (5)    |     |          |     |     |
       |     |=========>|     |          |     |     |
       |  A  |          |  B  |     (6)     |  C  |     |
       |     |          |     |---------->|     |     |
       |     |          |     |     (7)     |     |     |
       |     |          |     |<----------|     |     |
       |     |    (8)    |     |          |     |     |
       |     |<=========|     |          |     |     |
       +-----+          +-----+               +-----+

       ====> HTTP/SIP
       ----> RADIUS
```
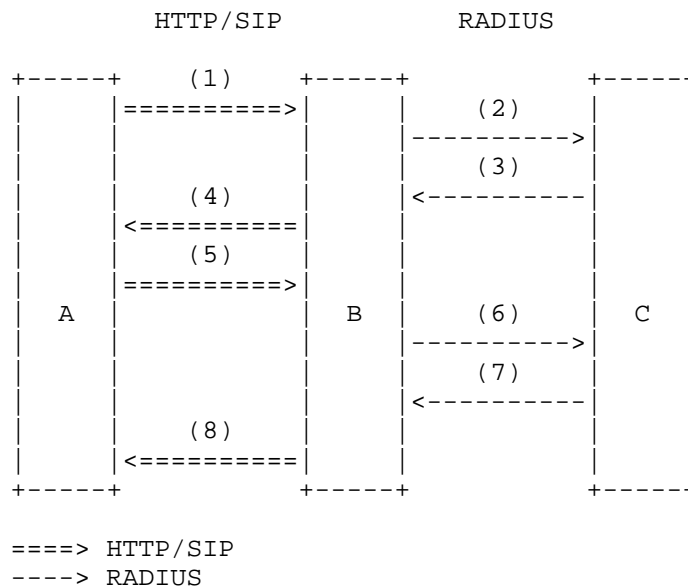
                  Figure 2: HTTP Digest over RADIUS

The entities have the following roles:

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. 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).
Depending on C's result, B processes A's request or rejects it with a
"(Proxy) Authorization required" response (step 8).

2. Detailed Description

2.1. RADIUS Client Behavior

The attributes described in this document are sent in cleartext.
Therefore, were a RADIUS client to accept secure 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 RADIUS exchange.

2.1.1. Credential Selection

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.

2.1.2. Constructing an Access-Request

If a matching (Proxy-)Authorization header is present and contains
HTTP Digest information, the RADIUS client checks the 'nonce'
parameter.

If the RADIUS client recognizes the nonce, 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,
and opaque directives into the respective Digest-Realm, Digest-Nonce,
Digest-URI, Digest-Qop, Digest-Algorithm, Digest-CNonce,
Digest-Nonce-Count, Digest-Username, and Digest-Opaque attributes.
The RADIUS client puts the request method into the Digest-Method
attribute.

Due to syntactic requirements, HTTP-style protocols have to escape
with backslash all quote and backslash characters in contents of HTTP
Digest directives. When translating directives into RADIUS
attributes, the RADIUS client only removes the surrounding quotes
where present. See Section 3 for an example.

If the Quality of Protection (qop) directive's value is 'auth-int',
the RADIUS client calculates H(entity-body) as described in
[RFC2617], Section 3.2.1, and puts the result in a
Digest-Entity-Body-Hash 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.

2.1.3. Constructing an Authentication-Info Header

After having received an Access-Accept from the RADIUS server, the
RADIUS client constructs an Authentication-Info header:

o 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 as if 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).

   o 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 in 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'.

         + IP Security (IPsec) is configured to protect traffic between
            the RADIUS client and RADIUS server 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.

   o 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.

   When the RADIUS server provides a Digest-Nextnonce attribute in the
   Access-Accept packet, the RADIUS client puts the contents of this
   attribute into a 'nextnonce' directive. Now it can send an
   HTTP-style response.

2.1.4. Failed Authentication

   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-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
surrounding quotes), the RADIUS client sends an error response (401
or 407) containing a WWW-/Proxy-Authenticate header with the
directive 'stale' and the digest directives derived from the Digest-*
attributes.

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].

2.1.5. Obtaining Nonces

The RADIUS client has two ways to obtain nonces: 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 client constructs a (Proxy-)Authenticate header using the
received Digest-Nonce and Digest-Realm attributes to fill the nonce
and realm directives. The RADIUS server can send Digest-Qop,
Digest-Algorithm, Digest-Domain, and Digest-Opaque attributes in the
Access-Challenge carrying the nonce. If these attributes are
present, the client MUST use them.

2.2. 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
and one or more Digest-Qop, and MAY add Digest-Domain or
Digest-Opaque attributes to the Access-Challenge packet.

2.2.1. General Attribute Checks

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, and 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.

   The RADIUS server removes '\' characters that escape quote and '\'
   characters from the text values it has received in the Digest-*
   attributes.

   If the mandatory attributes are present, the 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.

   The RADIUS server determines the age of the nonce in Digest-Nonce by
   using an embedded time-stamp or by looking it up in a local table.
   The RADIUS server MUST check the integrity of the nonce if it embeds
   the time-stamp in the nonce. Section 2.2.2 describes how the server
   handles old nonces.

2.2.2. Authentication

   If the Access-Request message has passed the checks described above,
   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

   o is authorized to access the protection space and

   o 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
a different user.

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 the response values match, but the RADIUS server considers the
nonce in the Digest-Nonce attribute as too old, it sends an
Access-Challenge packet containing a new nonce and a Digest-Stale
attribute with a value of 'true' (without surrounding quotes).

If the response values don't match, the RADIUS server responds with
an Access-Reject.

2.2.3. Constructing the Reply

If the authentication was successful, the RADIUS server adds an
attribute to the Access-Accept packet that 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 packet.

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 packet:

   * The Digest-Algorithm attribute's value is 'MD5-sess' or
      'AKAv1-MD5-sess'.

   * IPsec is configured to protect traffic between the RADIUS
      client and RADIUS server with IPsec (see Section 8).

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

RADIUS servers 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 surrounding quotes). The RADIUS
     server MUST add Message-Authenticator (see [RFC3579]), Digest-Nonce,
     Digest-Realm, SHOULD add Digest-Algorithm and one or more Digest-Qop
     and MAY add Digest-Domain, Digest-Opaque attributes to the
     Access-Challenge packet.

3. New RADIUS Attributes

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

      0   1   2
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
     +-+
     | Type | Length | Text ...
     +-+

     Quote and backslash characters in Digest-* attributes representing
     HTTP-style directives with a quoted-string syntax are escaped. The
     surrounding quotes are removed. They are syntactical delimiters that
     are redundant in RADIUS. For example, the directive

     realm="the \"example\" value"

     is represented as follows:

     +-+
     | Digest-Realm | 23 | the \"example\" value |
     +-+

3.1. 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
          103 for Digest-Response.
     Length
          >= 3

Text

    When using HTTP Digest, the text field is 32 octets long and
    contains a hexadecimal representation of a 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 surrounding quotes.

3.2. Digest-Realm Attribute

Description

    This attribute describes a protection space component of the
    RADIUS server. HTTP-style protocols differ in their definition
    of the protection space. See [RFC2617], Section 1.2, for
    details. It MUST only be used in Access-Request and
    Access-Challenge packets.

Type

    104 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
    surrounding quotes from the HTTP-style request it wants to
    authenticate. In Access-Challenge packets, the RADIUS server
    puts the expected realm value into this attribute.

3.3. 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, the RADIUS server
    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

    105 for Digest-Nonce

Length

    >=3

Text

    In Access-Requests, the RADIUS client takes the value of the
    nonce directive (nonce-value in [RFC2617]) without surrounding
    quotes from the HTTP-style request it wants to authenticate.
    In Access-Challenge packets, the attribute contains the nonce
    selected by the RADIUS server.

3.4. 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 surrounding 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 surrounding
      quotes into the rspauth directive of the Authentication-Info
      header.
   Type
      106 for Digest-Response-Auth.
   Length
      >= 3
   Text
      The RADIUS server calculates a digest according to section
      3.2.3 of [RFC2617] and copies the result into this attribute.
      Digest algorithms other than the one defined in [RFC2617] MAY
      define digest lengths other than 32.

3.5. Digest-Nextnonce Attribute

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

   Description

      The RADIUS server 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
      107 for Digest-Nextnonce
   Length
      >=3
   Text
      It is recommended that this text be base64 or hexadecimal data.

3.6. 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
>   108 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.

3.7. 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
>   109 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 found in
>   the HTTP-style request Authorization header (known as
>   "digest-uri-value" in section 3.2.2 of [RFC2617]) without
>   surrounding 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.

3.8. 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
>   110 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]) 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.

3.9. Digest-Algorithm Attribute

   Description
        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
        111 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)
        from the HTTP-style request it wants to authenticate. In
        Access-Challenge packets, the RADIUS server SHOULD put the
        desired algorithm into this attribute.

3.10. 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 [RFC3261] 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
        112 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 that the integrity of the entity body (e.g., qop
parameter set to "auth-int") be verified. Extensions to this
document may define support for authentication mechanisms other
than HTTP Digest.

3.11. 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
        113 for Digest-CNonce
   Length
        >=3
   Text
        This attribute includes the value of the cnonce-value [RFC2617]
        without surrounding quotes, taken from the HTTP-style request.

3.12. 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
        114 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 surrounding
        quotes from the HTTP-style request it wants to authenticate.

3.13. 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
        115 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 surrounding quotes from the HTTP-style request it wants
        to authenticate.

3.14. 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 MUST only be used in
        Access-Request and Access-Challenge packets.
    Type
        116 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
        surrounding 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.

3.15. 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
       117 for Digest-Auth-Param
Length
       >=3
Text
       The text consists of the whole parameter, including its name
       and the equal sign ('=') and quotes.

3.16. 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
       118 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 surrounding quotes from the HTTP-style
       request it wants to authenticate.

3.17. 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. Together with Digest-Realm, the
       URIs in the list define the protection space (see [RFC2617],
       section 3.2.1) for some HTTP-style protocols. This attribute
       MUST only be used in Access-Challenge packets.
   Type
       119 for Digest-Domain
   Length
       3
   Text
       This attribute consists of a single URI that defines a
       protection space component.

3.18. 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.
   Type
        120 for Digest-Stale
   Length
        3
   Text
        The attribute has either the value 'true' or 'false' (both
        values without surrounding quotes).

3.19. 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'.

        + IPsec is configured to protect traffic between RADIUS client
           and RADIUS server with IPsec (see Section 8).

        This attribute MUST only be used in Access-Accept packets.
   Type
        121 for Digest-HA1
   Length
        >= 3

     Text
          This attribute contains the hexadecimal representation of H(A1)
          as described in [RFC2617], sections 3.1.3, 3.2.1, and 3.2.2.2.

3.20. SIP-AOR Attribute

   Description
          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, for example, 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
          122 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.

4. Diameter Compatibility

   This document defines support for Digest Authentication in RADIUS. A
   companion document "Diameter Session Initiation Protocol (SIP)
   Application" [SIP-APP] defines support for Digest Authentication in
   Diameter, and addresses compatibility issues between RADIUS and
   Diameter.

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.

Req	Accept	Reject	Challenge	#	Attribute
1	0	0	0	1	User-Name
1	1	1	1	80	Message-Authenticator
0-1	0	0	0	103	Digest-Response
0-1	0	0	1	104	Digest-Realm
0-1	0	0	1	105	Digest-Nonce
0	0-1	0	0	106	Digest-Response-Auth
					(see Note 1, 2)
0	0-1	0	0	107	Digest-Nextnonce
0-1	0	0	0	108	Digest-Method
0-1	0	0	0	109	Digest-URI
0-1	0	0	0+	110	Digest-Qop
0-1	0	0	0-1	111	Digest-Algorithm (see
					Note 3)
0-1	0	0	0	112	Digest-Entity-Body-Hash
0-1	0	0	0	113	Digest-CNonce
0-1	0	0	0	114	Digest-Nonce-Count
0-1	0	0	0	115	Digest-Username
0-1	0	0	0-1	116	Digest-Opaque
0+	0+	0	0+	117	Digest-Auth-Param
0-1	0	0	0	118	Digest-AKA-Auts
0	0	0	0+	119	Digest-Domain
0	0	0	0-1	120	Digest-Stale
0	0-1	0	0	121	Digest-HA1 (see Note 1,
					2)
0-1	0	0	0	122	SIP-AOR

                                Table 1

    [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-Algorithm is missing, 'MD5' is assumed.

6. Examples

   This is an example selected from the traffic between a softphone (A),
   a Proxy Server (B), and an example.com RADIUS server (C). The
   communication between the Proxy Server and a SIP Public Switched
   Telephone Network (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->C

      Code = 1 (Access-Request)
      Attributes:
      NAS-IP-Address = c0 0 2 26 (192.0.2.38)
      NAS-Port-Type = 5 (Virtual)
      User-Name = 12345678
      Digest-Method = INVITE
      Digest-URI = sip:97226491335@example.com
      Message-Authenticator =
       08 af 7e 01 b6 8d 74 c3 a4 3c 33 e1 56 2a 80 43

   C->B

      Code = 11 (Access-Challenge)
      Attributes:
      Digest-Nonce = 3bada1a0
      Digest-Realm = example.com
      Digest-Qop = auth
      Digest-Algorithm = MD5
      Message-Authenticator =
       f8 01 26 9f 70 5e ef 5d 24 ac f5 ca fb 27 da 40

   B->A

      SIP/2.0 407 Proxy Authentication Required
      Proxy-Authenticate: Digest realm="example.com"
          ,nonce="3bada1a0",qop=auth,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="3bada1a0"
        ,realm="example.com"
        ,response="f3ce87e6984557cd0fecc26f3c5e97a4"
        ,uri="sip:97226491335@example.com",username="12345678"
        ,qop=auth,algorithm=MD5
   From: <sip:12345678@example.com>
   To: <sip:97226491335@example.com>

B->C

   Code = 1 (Access-Request)
   Attributes:
   NAS-IP-Address = c0 0 2 26 (192.0.2.38)
   NAS-Port-Type = 5 (Virtual)
   User-Name = 12345678
   Digest-Response = f3ce87e6984557cd0fecc26f3c5e97a4
   Digest-Realm = example.com
   Digest-Nonce = 3bada1a0
   Digest-Method = INVITE
   Digest-URI = sip:97226491335@example.com
   Digest-Qop = auth
   Digest-Algorithm = md5
   Digest-Username =  12345678
   SIP-AOR =  sip:12345678@example.com
   Message-Authenticator =
        ff 67 f4 13 8e b8 59 32 22 f9 37 0f 32 f8 e0 ff

C->B

   Code = 2 (Access-Accept)
   Attributes:
   Digest-Response-Auth =
                  6303c41b0e2c3e524e413cafe8cce954
   Message-Authenticator =
        75 8d 44 49 66 1f 7b 47 9d 10 d0 2d 4a 2e aa f1

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->C

      Code = 1 (Access-Request)
      Attributes:
      NAS-IP-Address = c0 0 2 26 (192.0.2.38)
      NAS-Port-Type = 5 (Virtual)
      Digest-Method = GET
      Digest-URI = /index.html
      Message-Authenticator =
       34 a6 26 46 f3 81 f9 b4 97 c0 dd 9d 11 8f ca c7

   C->B

      Code = 11 (Access-Challenge)
      Attributes:
      Digest-Nonce = a3086ac8
      Digest-Realm = example.com
      Digest-Qop = auth
      Digest-Algorithm = MD5
      Message-Authenticator =
       f8 01 26 9f 70 5e ef 5d 24 ac f5 ca fb 27 da 40

   B->A

      HTTP/1.1 401 Authentication Required
      WWW-Authenticate: Digest realm="example.com",
          nonce="a3086ac8",qop=auth,algorithm=MD5
      Content-Length: 0
```

```
A->B

   GET /index.html HTTP/1.1
   Authorization: Digest algorithm=MD5,nonce="a3086ac8"
        ,realm="example.com"
        ,response="f052b68058b2987aba493857ae1ab002"
        ,uri="/index.html",username="12345678"
        ,qop=auth,algorithm=MD5

B->C

   Code = 1 (Access-Request)
   Attributes:
   NAS-IP-Address = c0 0 2 26 (192.0.2.38)
   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-Username =  12345678
   Digest-Qop = auth
   Digest-Algorithm = MD5
   Message-Authenticator =
       06 e1 65 23 57 94 e6 de 87 5a e8 ce a2 7d 43 6b

C->B

   Code = 2 (Access-Accept)
   Attributes:
   Digest-Response-Auth =
       e644aa513effbfe1caff67103ff6433c
   Message-Authenticator =
       7a 66 73 a3 52 44 dd ca 90 e2 f6 10 61 2d 81 d7

B->A

   HTTP/1.1 200 OK
   ...

   <html>
   ...
```

7. IANA Considerations

   This document serves as an IANA registration request for a number of
   values from the RADIUS attribute type number space. The IANA has
   assigned the following:

```
+------------------------+-----------------------+
| placeholder            | value assigned by IANA |
+------------------------+-----------------------+
| Digest-Response        | 103                   |
| Digest-Realm           | 104                   |
| Digest-Nonce           | 105                   |
| Digest-Nextnonce       | 106                   |
| Digest-Response-Auth   | 107                   |
| Digest-Method          | 108                   |
| Digest-URI             | 109                   |
| Digest-Qop             | 110                   |
| Digest-Algorithm       | 111                   |
| Digest-Entity-Body-Hash | 112                  |
| Digest-CNonce          | 113                   |
| Digest-Nonce-Count     | 114                   |
| Digest-Username        | 115                   |
| Digest-Opaque          | 116                   |
| Digest-Auth-Param      | 117                   |
| Digest-AKA-Auts        | 118                   |
| Digest-Domain          | 119                   |
| Digest-Stale           | 120                   |
| Digest-HA1             | 121                   |
| SIP-AOR                | 122                   |
+------------------------+-----------------------+
```

                              Table 2

8. Security Considerations

   The RADIUS extensions described in this document enable RADIUS to
   transport the data that is 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.

8.1. Denial of Service

   RADIUS clients implementing the extension described in this document
   may authenticate HTTP-style requests received over the Internet. As
   compared with the use of RADIUS to authenticate link-layer network
   access, attackers 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 RADIUS
   server MUST check whether 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.

8.2. Confidentiality and Data Integrity

   The attributes described in this document are sent in cleartext.
   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.

   Some parameter combinations require the protection of RADIUS packets
   against eavesdropping and tampering. Implementations SHOULD try to
   determine automatically whether IPsec is configured to protect
   traffic between the RADIUS client and the RADIUS server. If this is
   not possible, the implementation checks a configuration parameter
   telling it whether IPsec will protect RADIUS traffic. The default
   value of this configuration parameter tells the implementation that
   RADIUS packets will not be protected.

   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 could use an
equally secure connection to the RADIUS server. There are several
ways to achieve this, for example:

o The RADIUS client may reject HTTP-style requests received over TLS
   or IPsec.

o The RADIUS client may 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. Acknowledgements

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

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

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

   [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson,
              "Remote Authentication Dial In User Service (RADIUS)", RFC
              2865, June 2000.

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

   [RFC3579] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication
              Dial In User Service) Support For Extensible
              Authentication Protocol (EAP)", RFC 3579, September 2003.

   [RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC
              3966, December 2004.

10.2. Informative References

   [SIP-APP] Garcia-Martin, M., "Diameter Session Initiation Protocol
              (SIP) Application", Work in Progress), April 2006.

   [RFC1994] Simpson, W., "PPP Challenge Handshake Authentication
              Protocol (CHAP)", RFC 1994, August 1996.

   [RFC2069] Franks, J., Hallam-Baker, P., Hostetler, J., Leach, P.,
              Luotonen, A., Sink, E., and L. Stewart, "An Extension to
              HTTP : Digest Access Authentication", RFC 2069, January
              1997.

   [RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.1", RFC 4346, April 2006.

   [RFC3851] Ramsdell, B., "Secure/Multipurpose Internet Mail
              Extensions (S/MIME) Version 3.1 Message Specification",
              RFC 3851, July 2004.

   [RFC3310] Niemi, A., Arkko, J., and V. Torvinen, "Hypertext Transfer
              Protocol (HTTP) Digest Authentication Using Authentication
              and Key Agreement (AKA)", RFC 3310, September 2002.

   [RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J.
              Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

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
   Deutsche Telekom Allee 7
   Darmstadt 64295
   Germany

   EMail: beckw@t-systems.com