

Network Working Group
Internet-Draft
Expires: April 1, 2002

R. Cunnings
Lectrosonics
R. Salz
Zolera
October 2001

SOAP Extensions: Basic and Digest Authentication
draft-cunnings-salz-soap-auth-01

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

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/lid-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 April 1, 2002.

Copyright Notice

Copyright (C) The Internet Society (2001). All Rights Reserved.

Abstract

This memo defines SOAP Extensions which implement a basic access authentication mechanism and a digest access authentication mechanism for use in request/response message exchange patterns. These mechanisms are adaptations of the corresponding mechanisms defined in [RFC 2617](#).

Internet-Draft

SOAP Authentication Extensions

October 2001

1. Introduction

This specification defines SOAP Extensions [2] which implement a "Basic" access authentication mechanism and a "Digest" access authentication mechanism for use in request/response message exchange patterns. These are adaptations of the corresponding mechanisms presented in [RFC 2617](#)[6]. A SOAP node utilizing one of these extensions will process requests for protected resources only if the client can be "authenticated"; that is, only if the client has presented valid "credentials" in the request message. These credentials are presented in response to a "challenge" sent by the server after a request for access to a protected resource.

The term "client" is defined as the node originating a SOAP request message, and the term "server" is defined as the node responsible for implementing an Authentication Extension. This server node is the target of any header entries in request messages, and source of any header entries in the response message that are defined by this specification. The authentication server may be either a SOAP processing intermediary node, or the ultimate destination node (endpoint) of the request/response message exchange.

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 [RFC 2119](#)[4].

[2. Basic Access Authentication](#)

The Basic mechanism allows a client to authenticate itself by presenting credentials consisting of a username/password pair, directly transmitted in the request message. This is a challenge/response protocol in which the client presents its credentials in response to a challenge from the server, which consists of the name of a security realm within which the requested resource is protected. The realm name serves as a hint to the client for the choice of username and password.

[2.1 Challenge/Response Protocol](#)

When a client wishes to access a protected resource, it presents credentials contained in an instance of the "BasicAuth" header entry, which is included in the SOAP request message sent to the server. If the responsible SOAP actor fails to find this header entry in a request message envelope, or the credentials are invalid, the client is not "authenticated" and a "challenge" is issued to the client in the response. The challenge is contained in a "BasicChallenge" header entry included in a SOAP Fault message. To proceed, the client must resend the request, including a "response" to the challenge in an instance of the header entry "BasicAuth". If the server finds this response to be correct, the client is authenticated, and the request can be processed. Clients may preemptively include the "BasicAuth" header entry in a request message in order to avoid a challenge from the server.

[2.2 Header Entry Element Definitions](#)

The header entry elements described below are defined in the XML Schema document found in [Appendix A](#). These elements are defined in XML namespace "http://soap-authentication.org/2002/01/". This namespace MUST be used by implementations. The nature of this SOAP extension requires that the "SOAP-ENV:mustUnderstand" attribute

SHOULD be present with a value of "1".

[2.2.1](#) BasicChallenge Element

This element is contained in the header of the fault message returned by the server when a request is received with missing or invalid credentials. It is a structure with the following member:

Realm serves as a hint to the client for the choice of user name and password, e.g. the name of a group of users authorized to access a resource. The realm name identifies the protection space associated with the requested resource.

In this example the server is responding to a request which did not contain a "BasicAuth" header entry, or contained invalid credentials, with a SOAP fault message. The "BasicChallenge" header entry element represents the challenge to the client.

```
<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <h:BasicChallenge
      xmlns:h="http://soap-authentication.org/2001/01/"
      SOAP-ENV:mustUnderstand="1">
      <Realm>test@whitemesa.net</Realm>
    </h:BasicChallenge>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>
        Authentication failed: missing, malformed,
        or invalid credentials.
      </faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[2.2.2](#) BasicAuth Element

This element is contained in the header of the request message sent by a client. It is a structure and contains the following members:

Name This member is of type "string" and contains the name identifying the entity for which the authentication is being attempted.

Password This member is of type "string" and contains the password.

In this example the client is resending a request message and including the "BasicAuth" header entry element. This contains its response (the "BasicAuth" element) to a previous challenge from the server.

```
<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <h:BasicAuth
      xmlns:h="http://soap-authentication.org/2002/01/"
      SOAP-ENV:mustUnderstand="1">
      <Name>admin</Name>
      <Password>broccoli</Password>
    </h:BasicAuth>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:echoString xmlns:m="http://soapinterop.org/">
```

```
        <inputString>This is a test.</inputString>
    </m:echoString>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[2.3](#) Processing Rules

A server receiving a request MUST respond with a SOAP Fault message if the request header entry element "BasicAuth" is not present, is malformed, or contains invalid credentials. The Fault message header MUST contain the header entry element "BasicChallenge" with member "Realm" containing the name of a security realm with which the resource is associated.

A client responding to a challenge from a server MUST include the "BasicAuth" header entry element in the header of the message sent in the next attempt to access the resource. The "Name" and "Password" elements MUST be present.

A client MAY preemptively include a "BasicAuth" header entry element containing its credentials in a request message in order to avoid a challenge from the server. The "Name" and "Password" elements MUST be present.

After receiving a request in which the "BasicAuth" header entry element is present and contains valid credentials, the server MUST allow normal request processing to take place.

[3.](#) Digest Access Authentication

The Digest mechanism allows a client to authenticate itself by presenting credentials consisting of a message digest (such as MD5[3] or a SHA[1] variant) transmitted in a request message. It is based on the principle that the client and server are in possession of a shared secret, a password string. The advantage of this method is that the client password is only used in calculating the digest, so it remains safe from network exposure.

The Digest mechanism is a challenge/response protocol in which the client presents its credentials in response to a challenge from the server, which consists of an opaque data string called a "nonce". This nonce serves as additional input to the digest function, and

allows the server to influence the digest value in a way not controlled by the client; this increases the security of the mechanism. In order to authenticate the client, the server simply compares the digest value received from the client with the value it has computed internally. If the values match, the client must be in possession of the same nonce and password as the server, so the client is "authenticated". The same technique is used in the mutual authentication scenario, where the server authenticates itself to the client by presenting a digest as credentials in response to a challenge from the client. In this case also the challenge consists of a client-produced nonce to be used as input to the digest function, allowing the client to influence the digest value in a way not controlled by the server. The full specification for digest computation is found in in [Section 3.3](#).

[3.1](#) Challenge/response protocol

When a client wishes to access a protected resource, it presents credentials contained in an instance of the "ClientAuth" header entry element, which is included in the SOAP request message sent to the server. If the responsible SOAP actor fails to find this header entry in a request message envelope, or finds an instance containing invalid or expired credentials, the client is not "authenticated", and a "challenge" is issued to the client in the response. The challenge is contained in a "Challenge" header entry included in a SOAP Fault message. To proceed, the client must resend the request, including a "response" to the challenge in an instance of the header entry "ClientAuth". The response consists of a message digest computed as specified in [Section 3.3](#).

If the server finds this response to be correct, the client is authenticated, and the request can be processed. When processing is complete, the response message returned to the client by the server will include a header entry, "NextChallenge", containing the next challenge to the client. If the client wishes to send another

request, it will re-authenticate itself by supplying a response to this new challenge, as part of the "ClientAuth" header entry it includes in the request message. Thus each request/response SOAP message exchange involves a separate authentication of the client to the server.

A mechanism is provided for mutual authentication. If the client

wishes to authenticate the server, it includes a challenge of its own to the server as part of the "ClientAuth" header entry it sends in a request message. The server supplies a response to this challenge in the "NextChallenge" header entry it includes in response message returned to the client. The response consists of a message digest computed as specified in [Section 3.3](#). If the client finds this response to be correct, the server is authenticated.

A method is provided for a client to explicitly solicit a challenge from a server, optionally including a challenge of its own to the server. Including a challenge is useful when the client wishes to immediately determine the authenticity of the server, before sending any further resource request messages for processing. The client initiates the exchange by including the "InitChallenge" header entry in a request message. The server responds with a challenge, contained in an instance of the "NextChallenge" header entry contained in a SOAP Fault message. If the client has included a challenge to the server in the "InitChallenge" header entry, the server supplies a response in the "NextChallenge" header entry returned to the client. This response can then be used by the client determine the authenticity of the server. If the client wishes to proceed with a request at this point, it will authenticate itself by supplying a response to the challenge from the server, as part of the "ClientAuth" header entry it includes in the request message.

[3.2](#) Header Entry Element Definitions

The header entry elements described below are defined in the XML Schema document found in [Appendix A](#). These elements are defined in XML namespace "http://soap-authentication.org/2002/01/" This namespace MUST be used by implementations. The nature of this SOAP extension requires that the "SOAP-ENV:mustUnderstand" attribute SHOULD be present with a value of "1".

[3.2.1](#) Challenge Element

This element is contained in the header of the fault message returned by the server when a request is received with missing or invalid credentials. It is a structure and contains the following members:

Status This member is of type "string" and contains a code

indicating the reason for the challenge. Status code values are defined in [Section 4](#).

Nonce This member is of type "string" and contains the nonce which represents the challenge issued by the server to the client.

Realm This member is of type "string" and contains a name which serves as a hint to the client for the choice of user id and password, e.g. the name of a group of users authorized to access a resource. The realm name identifies the protection space associated with the requested resource.

In addition, the "Challenge" element may have a "digest" attribute whose value is a URI identifying the digest mechanism to be used. If not present, the identifier for MD5 is assumed.

In this example the server is responding to a request which did not contain a "ClientAuth" header entry with a SOAP fault message. The "Challenge" header entry contains the challenge ("Nonce" member) issued to the client.

```
<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <h:Challenge
      xmlns:h="http://soap-authentication.org/2002/01/"
      SOAP-ENV:mustUnderstand="1">
      <Status>Unauthenticated.NoCredentials</Status>
      <Nonce>950C60A74BAA9BB7EDAC95F02EEC497C</Nonce>
      <Realm>test@whitemesa.net</Realm>
    </h:Challenge>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>
        Authentication failed: missing, malformed,
        or invalid credentials.
      </faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[3.2.2](#) ClientAuth Element

This element is contained in the header of a request message sent by a client. It is a structure and contains the following members:

Nonce This member is of type "string" and echoes the nonce value received from the server representing the challenge to the client.

Auth This member is of type "hexBinary" and contains the response by the client to the challenge received from the server. This is a hex encoded message digest computed as specified in [Section 3.3](#).

UserID This member is of type "string" and contains the User ID identifying the account for which the authentication is being attempted.

Realm This member is of type "string" and identifies the security realm for which the authentication is being attempted.

ClientNonce (Optional) This member is of type "string" and contains the nonce which represents a challenge issued by the client to the server when mutual authentication is desired.

In addition, the "ClientAuth" element may have a "digest" attribute whose value is a URI identifying the digest mechanism to be used. If not present, the identifier for MD5 is assumed.

In this example the client is responding to a challenge by resending the request message and including the "ClientAuth" header entry element. This contains its response ("Auth" member) to the challenge from the server. Also present is the client nonce ("ClientNonce" member) representing a challenge issued by the client to the server.

```
<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <h:ClientAuth
      xmlns:h="http://soap-authentication.org/2002/01/"
      SOAP-ENV:mustUnderstand="1">
      <Nonce>950C60A74BAA9BB7EDAC95F02EEC497C</Nonce>
      <Auth>C48F2DEEC547D9BF590B4C72283445A5</Auth>
      <UserID>admin</UserID>
      <Realm>test@whitemesa.net</Realm>
      <ClientNonce>CEA8A3DB3C06C7970A61B92AE9560A08</ClientNonce>
    </h:ClientAuth>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:echoString xmlns:m="http://soapinterop.org/">
      <inputString>This is a test.</inputString>
    </m:echoString>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[3.2.3](#) NextChallenge Element

This element is contained in the header of a response message sent by the server. It is a structure and contains the following members:

Status This member is of type "string" and contains a code

indicating the outcome of the authentication attempt made by the client in the request. Status code values are defined in [Section 4](#).

Nonce This member is of type "string" and contains the nonce which represents a new challenge issued by the server to the client. It will form the basis of the response by the client in the next request, if any.

ClientNonce (Optional) This member is of type "string" and echoes the nonce value received from the client, representing a challenge issued by the client to the server. If no challenge was received from the client, this member is omitted.

ServerAuth (Optional) This member is of type "hexBinary" and contains the response by the server to a challenge received from the client. This is a hex encoded message digest computed as given in [Section 3.3](#). If no challenge was received from the client, this member is omitted.

In addition, the "NextChallenge" element may have a "digest" attribute whose value is a URI identifying the digest mechanism to be used. If not present, the identifier for MD5 is assumed.

In this example the server has received a request message containing a correct response to the last challenge issued to the client. It responds with this message containing the "NextChallenge" header entry which includes its response ("ServerAuth" member) to the challenge issued by the client in the request message. Also included is the next challenge issued to the client ("Nonce" member).

<SOAP-ENV:Envelope

 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <SOAP-ENV:Header>

 <h:NextChallenge

 xmlns:h="http://soap-authentication.org/2002/01/"
 SOAP-ENV:mustUnderstand="1">

 <Status>Authenticated</Status>

 <Nonce>574F38FFDE076F9006AC0014146DFD14</Nonce>

```

        <ClientNonce>CEA8A3DB3C06C7970A61B92AE9560A08</ClientNonce>
        <ServerAuth>CA834D49323368101AC51CA15E745DBF</ServerAuth>
    </h:NextChallenge>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
    <m:echoStringResponse xmlns:m="http://soapinterop.org/">
        <return>This is a test.</return>
    </m:echoStringResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

[3.2.4](#) InitChallenge Element

This element is contained in the header of a request message sent by a client. It is a structure and contains the following members:

UserID This member is of type "string" and contains the User ID identifying the account for which the authentication is being attempted.

Realm This member is of type "string" and identifies the security realm for which the authentication is being attempted.

ClientNonce (Optional) This member is of type "string" and contains the nonce which represents a challenge issued by the client to the server when immediate authentication of the server is desired.

In addition, the "InitChallenge" element may have a "digest" attribute whose value is a URI identifying the digest mechanism to be used. If not present, the identifier for MD5 is assumed.

In this example the client is soliciting a challenge from the server by sending a request message including the "InitChallenge" header entry element. This contains the client nonce ("ClientNonce" member) representing a challenge issued by the client to the server.

```

<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

```

```

<SOAP-ENV:Header>
  <h:InitChallenge
    xmlns:h="http://soap-authentication.org/2002/01/"
    SOAP-ENV:mustUnderstand="1">
    <UserID>admin</UserID>
    <Realm>test@whitemesa.net</Realm>
    <ClientNonce>CEA8A3DB3C06C7970A61B92AE9560A08</ClientNonce>
  </h:InitChallenge>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
  <m:echoString xmlns:m="http://soapinterop.org/">
    <inputString></inputString>
  </m:echoString>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

The server responds with a fault message which includes the "NextChallenge" header entry element. This includes its response ("ServerAuth" member) to the challenge issued by the client in the request message. Also included is the challenge issued by the server to the client ("Nonce" member).

```

<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <h:NextChallenge
      xmlns:h="http://soap-authentication.org/2002/01/"
      SOAP-ENV:mustUnderstand="1">
      <Status>Unauthenticated.NoCredentials</Status>
      <Nonce>574F38FFDE076F9006AC0014146DFD14</Nonce>
      <ClientNonce>CEA8A3DB3C06C7970A61B92AE9560A08</ClientNonce>
      <ServerAuth>CA834D49323368101AC51CA15E745DBF</ServerAuth>
    </h:NextChallenge>

```

```

</SOAP-ENV:Header>
<SOAP-ENV:Body>
  <SOAP-ENV:Fault>
    <faultcode>SOAP-ENV:Client</faultcode>
    <faultstring>
      Authentication failed: missing, malformed,
      or invalid credentials.
    </faultstring>
  </SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

[3.3](#) Digest Computation

The response to a challenge is computed as follows:

- o "HASHMECH(concat(secret, ":", sn, ":", cn))" When mutual authentication is required.
- o "HASHMECH(concat(secret, ":", sn))" When only client authentication is required.

Where:

secret (string) represents the hex encoding of the digest "HASHMECH(concat(uid, ":", realm, ":", pwd))" where:

uid (string) represents the User ID.

realm (string) represents the Realm name.

pwd (string) represents the password.

sn (string) represents the nonce serving as the challenge from the

server to the client.

cn (string) represents the client nonce serving as the challenge from the client to the server.

concat(stringA, stringB, ...) represents the string concatenation operation.

`HASHMECH(string)` represents the digesting operation.

All strings, including literals, are presented as the UTF-8[5] encodings of Unicode strings.

The Status Code values defined by this specification are:

Authenticated

Unauthenticated

Unauthenticated.NoCredentials

Unauthenticated.InvalidResponse

Unauthenticated.ExpiredNonce

Unauthenticated.InvalidUser

Unauthenticated.InvalidRealm

Interop.UnsupportedDigest

These codes consist of dot (".") separated values, and are interpreted by considering the value to the left of a dot to be more general in meaning than the value to the right. For instance, the value "Unauthenticated.NoCredentials" represents the extension of the value "Unauthenticated" by the value "NoCredentials". In this manner applications may extend the existing set of codes in a backwards compatible way by appending additional values (e.g. "Unauthenticated.ExpiredNonce.ServerReboot", which could be substituted for "Unauthenticated.ExpiredNonce" where appropriate).

[5.](#) Digest Mechanisms

Digest mechanisms are identified by URI. The URI "http://www.w3.org/2000/09/xmlsig#md5" is used to indicate MD5[3]. The URI "http://soap-authentication.org/2002/01/#sha-1" is used to indicate the SHA-1[1] (160-bit) digest; other standards may, e.g., define SHA-384.

For compatibility with this document's base specifications, the default digest mechanism is MD5[3].

6. Processing Rules

A client MAY include the "InitChallenge" header entry element in a request message, in order to explicitly solicit a challenge from a server. The "digest" attribute, if present, specifies the digest mechanism to be used by the server in generating its response; if omitted, MD5 MUST be used. The "ClientNonce" member MAY contain a nonce serving as a challenge to the server. security realm with which the resource is associated, and the "UserID" member MUST contain a valid value for the specified realm.

A server receiving an explicit request for a challenge (indicated by the presence of the "InitChallenge" header entry element in the request message), MUST respond with a SOAP Fault message containing a challenge. If the "Realm" and "UserID" members of the "InitChallenge" header entry element are valid for the requested resource, and the server implements the desired digest mechanism, the Fault message MUST include the "NextChallenge" header entry element with a newly generated nonce in the "Nonce" member, and a "Status" member value of "Unauthenticated.NoCredentials". The "digest" attribute, if present, mechanism to be used by the client to generate a response; if omitted, MD5 MUST be used. If the request message included a challenge from the client, the "ServerAuth" member MUST be present and contain the response to the challenge. The server MUST respond using the same digest mechanism used by the "InitChallenge" header entry, and the "ClientNonce" member MUST be present echoing the value of the "ClientNonce" member of the received "InitChallenge" header entry element.

If the "Realm" or "UserID" members of the "InitChallenge" header entry element are not valid for the requested resource, or the "digest" attribute specifies a digest mechanism not supported by the server, the Fault message MUST include the "Challenge" header entry element with a newly generated nonce in the "Nonce" member. The "digest" attribute, if present, specifies the digest mechanism to use for the reply; if omitted, MD5 MUST be used. The "Realm" member MUST contain the name of a security realm with which the resource is associated. The value of the "Status" member is determined by the nature of the fault:

Unauthenticated.InvalidUser if the "InitChallenge" header entry contains an invalid "UserID" member value.

Unauthenticated.InvalidRealm if the "InitChallenge" header entry contains an invalid "Realm" member value.

Interop.UnsupportedDigest if the "InitChallenge" header entry "digest" attribute specifies an unsupported digest mechanism, or if the attribute is omitted and MD5 is not supported.

A server receiving a request without an explicit solicitation of a challenge (indicated by the absence of the "InitChallenge" header entry element in the request message), but containing invalid credentials (the "ClientAuth" header entry element is not present, is malformed, uses an unsupported digest mechanism, or authentication fails), MUST respond with a SOAP Fault message containing a challenge. The Fault message header MUST contain the header entry "Challenge" with a newly generated nonce in the "Nonce" member. The "digest" attribute, if present, specifies the digest mechanism for the client to use in its response; if omitted, MD5 MUST be used. The "Realm" member MUST contain the name of a security realm with which the resource is associated. The value of the "Status" member is determined by the nature of the fault:

Unauthenticated.NoCredentials if the request made by the client contained no "ClientAuth" header entry.

Unauthenticated.InvalidResponse if the "ClientAuth" header entry contains an incorrect response to a challenge.

Unauthenticated.ExpiredNonce if the "ClientAuth" header entry contains a response to an expired challenge.

Unauthenticated.InvalidUser if the "ClientAuth" header entry contains an invalid "UserID" member value.

Unauthenticated.InvalidRealm if the "ClientAuth" header entry contains an invalid "Realm" member value.

Interop.UnsupportedDigest if the "ClientAuth" header entry "digest" attribute specifies an unsupported digest mechanism, or the attribute is omitted and MD5 is not supported.

A client responding to a challenge from a server in the form of the "Challenge" header entry element MUST include the "ClientAuth" header entry element in the header of the message sent in the next attempt to access the resource. The client MUST respond using the same digest mechanism used in the "Challenge" header entry, and properly identify it. If the client does not support the digest mechanism given in the "Challenge" header entry, it MAY re-send the original request specifying an alternate digest mechanism. The "Auth" member MUST contain a response to the challenge, and the "ClientNonce" member MAY contain a nonce serving as a challenge to the server. This challenge MUST assume the use of the digest mechanism specified in the enclosing "ClientAuth" header entry element. The "Realm" and "Nonce" members MUST echo the values received in the members of the same name in the "Challenge" header entry element in the prior response message from the server. The "UserID" member MUST contain a valid value for the specified realm.

A client responding to a challenge from a server in the form of the "NextChallenge" header entry element MUST include the "ClientAuth" header entry element in the header of the message sent in the next attempt to access the resource. The client MUST respond using the same digest mechanism given in the "NextChallenge" header entry, and properly identify it. The "Auth" member MUST contain a response to the challenge, and the "ClientNonce" member MAY contain a nonce serving as a challenge to the server. This challenge MUST assume the use of the digest mechanism specified in the enclosing "ClientAuth" header entry element. The "Nonce" member MUST echo the value received in the member of the same name in the "NextChallenge" header entry element received in the prior response message from the server. The "Realm" member MUST contain the name of a security realm with which the resource is associated, and the "UserID" member MUST contain a valid value for the specified realm.

A server receiving a request in which the "ClientAuth" header entry element is present, specifies a supported digest mechanism, and contains valid credentials, MUST allow normal request processing to take place. Once this processing is complete, the server MUST include the header entry element "NextChallenge" in the response message header, with a "Status" member value of "Authenticated". The "Nonce" member MUST contain a newly generated nonce value representing the next challenge to the client. The "digest" attribute, if present, specifies the digest mechanism to be used by

the client to generate a response; if omitted, MD5 MUST be used. If the request message included a challenge from the client, the "ServerAuth" member MUST be present and contain the response to the challenge. The server MUST respond using the digest mechanism given in the "ClientAuth" header entry, and the "ClientNonce" member MUST be present echoing the value of the "ClientNonce" member of the "ClientAuth" header entry element received.

[7.](#) Security Considerations

This specification provides Authentication facilities only. Message integrity or confidentiality requirements are not satisfied by these SOAP Extensions. The Basic Authentication mechanism is vulnerable to network snooping, as the password is present in the header entry. It is RECOMMENDED in this case that communication be done over a transport protocol that provides privacy, such as SSL/TLS. This weakness is avoided in the Digest Authentication mechanism.

The use of a client supplied nonce (optional) and one-time server nonces (required) increases the resistance of this mechanism to certain kinds of attacks. Well chosen nonce values are important to the quality of the protection afforded. An useful analysis of security considerations for Basic and Digest access authentication mechanisms is found in [\[6\]](#).

References

- [1] US DoC/NISC, "US Federal Information Processing Standard 180-1, Secure Hash Standard (SHS)", April 1995,
<<http://csrc.nist.gov/cryptval/shs.html>>.
- [2] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H. F., Thatte, S. and D. Winer, "Simple Object Access Protocol (SOAP) 1.1", May 2000,
<<http://www.w3.org/TR/SOAP/>>.

- [3] Rivest, R., "The MD5 Message-Digest Algorithm", [RFC 1321](#), April 1992.
- [4] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [RFC 2119](#), March 1997.
- [5] Yergeau, F., "UTF-8, a transformation format of ISO 10646", [RFC 2279](#), January 1998.
- [6] 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.

Authors' Addresses

Robert Cunnings
Lectrosonics, Inc.
581 Laser Road
Rio Rancho, NM 87124
US

Phone: +1-505-892-4501
EMail: cunnings@lectrosonics.com
URI: <http://www.lectrosonics.com>

Rich Salz
Zolera Systems, Inc.
600 West Cummings Park
Suite 2000
Woburn, MA 01801
US

Phone: +1-781-759-1122
EMail: rsalz@zolera.com
URI: <http://www.zolera.com/>


```

<schema targetNamespace="http://soap-authentication.org/2002/01/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap-auth="http://soap-authentication.org/2002/01/"
  xmlns="http://www.w3.org/2001/XMLSchema">

  <import namespace="http://schemas.xmlsoap.org/soap/envelope/"
    schemaLocation="http://schemas.xmlsoap.org/soap/envelope/" />

  <element name="BasicChallenge" type="soap-auth:BasicChallenge_t"/>
  <complexType name="BasicChallenge_t">
    <sequence>
      <element name="Realm" type="xsd:string"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
  </complexType>

  <element name="BasicAuth" type="soap-auth:BasicAuth_t"/>
  <complexType name="BasicAuth_t">
    <sequence>
      <element name="Name" type="xsd:string"/>
      <element name="Password" type="xsd:string"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
  </complexType>

  <element name="Challenge" type="soap-auth:Challenge_t"/>
  <complexType name="Challenge_t">
    <sequence>
      <element name="Status" type="xsd:string"/>
      <element name="Nonce" type="xsd:string"/>
      <element name="Realm" type="xsd:string"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
    <attribute name="digest" type="anyURI" use="optional"/>
  </complexType>

  <element name="ClientAuth" type="soap-auth:ClientAuth_t"/>
  <complexType name="ClientAuth_t">
    <sequence>
      <element name="Nonce" type="xsd:string"/>
      <element name="Auth" type="xsd:hexBinary"/>
    </sequence>
  </complexType>

```

```
        <element name="UserID" type="xsd:string"/>
        <element name="Realm" type="xsd:string"/>
        <element name="ClientNonce" type="xsd:string" minOccurs="0"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
    <attribute name="digest" type="anyURI" use="optional"/>
</complexType>

<element name="NextChallenge" type="soap-auth:NextChallenge_t"/>
<complexType name="NextChallenge_t">
    <sequence>
        <element name="Status" type="xsd:string"/>
        <element name="Nonce" type="xsd:string"/>
        <element name="ClientNonce" type="xsd:string" minOccurs="0"/>
        <element name="ServerAuth" type="xsd:hexBinary" minOccurs="0"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
    <attribute name="digest" type="anyURI" use="optional"/>
</complexType>

<element name="InitChallenge" type="soap-auth:InitChallenge_t"/>
<complexType name="InitChallenge_t">
    <sequence>
        <element name="UserID" type="xsd:string"/>
        <element name="Realm" type="xsd:string"/>
        <element name="ClientNonce" type="xsd:string" minOccurs="0"/>
    </sequence>
    <attribute ref="SOAP-ENV:actor" use="optional"/>
    <attribute ref="SOAP-ENV:mustUnderstand" use="optional"/>
    <attribute name="digest" type="anyURI" use="optional"/>
</complexType>

</schema>
```

[Appendix B](#). Changes since the last version

Added support for multiple digest mechanisms, the "digest" attribute, and the SHA-1 URI.

Consolidated everything into the new
"http://soap-authentication.org/2002/01/" namespace.

Internet-Draft

SOAP Authentication Extensions

October 2001

Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS 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.

Acknowledgement

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

