[PROPOSED] HTTP Working Group
INTERNET-DRAFT
<draft-ietf-http-digest-aa-00.txt>

Expires SIX MONTHS FROM---> March 10, 1995

Jeffery L. Hostetler
John Franks
Philip Hallam-Baker
Ari Luotonen
Eric W. Sink
Lawrence C. Stewart

A Proposed Extension to HTTP : Digest Access Authentication

Status of this Memo

Distribution of this document is unlimited. Please send comments to the proposed HTTP working group at <http-wg@cuckoo.hpl.hp.com>. Discussions of the working group are archived at <URL:http://www.ics.uci.edu/pub/ietf/http/>. General discussions about HTTP and the applications which use HTTP should take place on the <www-talk@info.cern.ch> mailing list.

Abstract

The protocol referred to as "HTTP/1.0" includes specification for a Basic Access Authentication scheme. This scheme is not considered to be a secure method of user authentication, as the user name and password are passed over the network in an unencrypted form. A specification for a new authentication scheme is needed for future versions of the HTTP protocol. This document provides specification for such a scheme, referred to as "Digest Access Authentication". The encryption method used is the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

Table of Contents

1. Introduction

1.1 Purpose

The protocol referred to as "HTTP/1.0" includes specification for a Basic Access Authentication scheme[1]. This scheme is not considered to be a secure method of user authentication, as the user name and password are passed over the network in an unencrypted form. A specification for a new authentication scheme is needed for future versions of the HTTP protocol. This document provides specification for such a scheme, referred to as "Digest Access Authentication".

The Digest Access Authentication scheme is not intended to be a complete answer to the need for security in the World Wide Web. This scheme provides no encryption of object content. The intent is simply to facilitate secure access authentication.

It is proposed that this access authentication scheme be included in the the proposed HTTP/1.1 specification.

1.2 Overall Operation

Like Basic Access Authentication, the Digest scheme is based on a simple challenge-response paradigm. The Digest scheme challenges using a nonce value. A valid response contains the MD5 checksum of the password and the given nonce value. In this way, the password is never sent in the clear. Just as with the Basic scheme, the username and password must be prearranged in some fashion.

2. Digest Access Authentication Scheme

2.1 Specification

The Digest Access Authentication scheme is conceptually similar to the Basic scheme. The formats of the modified WWW-Authenticate header line and the Authorization header line are specified below. In addition, a new header, Digest-MessageDigest, is specified as well.

Due to formatting constraints, all of the headers are depicted on multiple lines. In actual usage, they are required to be a single line of comma-separated attribute-value pairs, terminated by <CRLF>. Whitespace between the attribute-value pairs is allowed.

If a server receives a request for an access-protected object, and an acceptable Authorizatation header is not sent, the server responds with:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="<realm>",
                            domain="<domain>",
                            nonce="<nonce>",
                            opaque="<opaque>",
                            stale="<TRUE | FALSE>"
```

The meanings of the identifers used above are as follows:

<realm>
A name given to users so they know which username and password to send.

<domain> OPTIONAL
A comma separated list of URIs, as specified for HTTP/1.0. The intent is that the client could use this information to know the set of URIs for which the same authentication information should be sent. The URIs in this list may exist on different servers. If this keyword is omitted or empty, the client should assume that the domain consists of all URIs on the responding server.

<nonce>
A server-specified integer value which may be uniquely generated each time a 401 response is made. Servers may defend themselves against replay attacks by refusing to reuse nonce values. The nonce should be considered opqaue by the client.

<opaque> OPTIONAL
A string of data, specified by the server, which should returned by the client unchanged. It is recommended that this string be base64 or hexadecimal data. Specifically, since the string is passed in the header lines as a quoted string, the double-quote character is not allowed.

<stale>   OPTIONAL
A flag, indicating that the previous request from the client was rejected because the nonce value was stale. If stale is TRUE, the client may wish to simply retry the request with a new encrypted response, without reprompting the user for a new username and password.

The client is expected to retry the request, passing an Authorization header line as follows:

```
Authorization: Digest
           username="<username>",             -- required
           realm="<realm>",                   -- required
           nonce="<nonce>",                   -- required
           uri="<requested-uri>",             -- required
           response="<digest>",               -- required
           message="<message-digest>",        -- OPTIONAL
           opaque="<opaque>"                  -- required if provided by server

        where <digest> := H( H(A1) + ":" + N + ":" + H(A2) )
  and <message-digest> := H( H(A1) + ":" + N + ":" + H(<message-body>) )

        where:

                A1 := U + ':' + R + ':' + P
                A2 := <Method> + ':' + <requested-uri>

                with:
                        N -- nonce value
                        U -- username
                        R -- realm
                        P -- password
                        <Method> -- from header line 0
                        <requested-uri> -- uri sans proxy/routing

    When authorization succeeds, the Server may optionally provide the
    following:

HTTP/1.1 200 OK
Digest-MessageDigest:
              username="<username>",
              realm="<realm>",
              nonce="<nonce>",
              message="<message-digest>"

        The Digest-MessageDigest header indicates that the server wants to
        communicate some info regarding the successful
        authentication (such as a message digest or a
        receipt of some kind).

        <message-digest> is computed as given above for
        the client.  this allows the client to verify that
        the message body has not been changed en-route.

        (The server would probably only send this when it
         has the document and can compute it (like the
         content-length field); the server would probably
         not bother generating this header for CGI output.)


   Upon receiving the Authorization information, the server may check its
```

validity by looking up its known password which corresponds to the submitted
<username>. Then, the server must perform the same MD5 operation performed
by the client, and compare the result to the given <response>.

Note that the HTTP server does not actually need to know the user's
clear text password. As long as H(A1) is available to the server, the
validity of an Authorization header may be verified.

All keyword-value pairs must be expressed in characters from the
US-ASCII character set, excluding control characters.

A client may remember the username, password and nonce values, so that
future requests within the specified <domain> may include the Authorization
line preemptively. The server may choose to accept the old Authorization
information, even though the nonce value included might not be fresh.
Alternatively, the server could return a 401 response with a new nonce
value, causing the client to retry the request. By specifying stale=TRUE
with this response, the server hints to the client that the request should
be retried with the new nonce, without reprompting the user for a new
username and password.

The <opaque> data is useful for transporting state information around.
For example, a server could be responsible for authenticating content
which actual sits on another server. The first 401 response would include
a <domain> which includes the URI on the second server, and the <opaque>
for specifying state information. The client will retry the request, at
which time the server may respond with a 301/302 redirection, pointing
to the URI on the second server. The client will follow the redirection,
and pass the same Authorization line, including the <opaque> data which
the second server may require.

As with the basic scheme, proxies must be completely transparent in
the Digest access authentication scheme. That is, they must forward the
WWW-Authenticate, Digest-MessageDigest and Authorization headers untouched.
If a proxy wants to authenticate a client before a request is forwarded to
the server, it can be done using the Proxy-Authenticate and
Proxy-Authorization headers.

2.2 Security Protocol Negotiation

It is useful for a server to be able to know which security schemes
a client is capable of handling. It is recommended that the HTTP extension
mechanism proposed by Dave Kristol [2] be used. If the client includes
the following header line with the request, then a server can safely assume
that the client can handle Digest authentication.

```
Extension: Security/Digest
```

If this proposal is accepted as a required part of the HTTP/1.1
specification, then a server may assume Digest support when a client
identifies itself as HTTP/1.1 compliant.

It is possible that a server may want to require Digest as its authentication method, even if the server does not know that the client supports it. A client is encouraged to fail gracefully if the server specifies any authorization scheme it cannot handle.

2.3 Example

The following example assumes that an access-protected document is being requested from the server. The URI of the document is "http://www.nowhere.org/simp/".

Both client and server know that the username for this document is "eric", and the password is "spyglass".

The first time the client requests the document, no Authorization header is sent, so the server responds with:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest    realm="testrealm",
                            nonce="72540723369",
                            opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

The client may prompt the user for the username and password, after which it will respond with a new request, including the following Authorization header:

```
Authorization: Digest       username="eric",
                            realm="testrealm",
                            nonce="72540723369",
                            uri="/simp/",
                            response="e966c932a9242554e42c8ee200cec7f6",
                            opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

3. Acknowledgments

Source code in C for the RSA Data Security, Inc. MD5 Message-Digest Algorithm is available free of charge from RSA Data Security, Inc.

4. References

[1] T. Berners-Lee, R. T. Fielding, H. Frystyk Nielsen. "Hypertext Transfer Protocol -- HTTP/1.0" Internet-Draft (work in progress), UC Irvine, <URL:http://ds.internic.net/internet-drafts/draft-fielding-http-spec-01.txt>, December 1994.

[2] D. Kristol. "A Proposed Extension Mechanism for HTTP" <URL:http://www.research.att.com/~dmk/extend.txt>, December 1994.

5. Authors Addresses

John Franks
john@math.nwu.edu
Professor of Mathematics
Department of Mathematics
Northwestern University
Evanston, IL 60208-2730

Phillip M. Hallam-Baker
hallam@w3.org
European Union Fellow
CERN
Geneva
Switzerland

Jeffery L. Hostetler
jeff@spyglass.com
Senior Software Engineer
Spyglass, Inc.
1800 Woodfield Drive
Savoy, IL 61874

Ari Luotonen
luotonen@netscape.com
Member of Technical Staff
501 East Middlefield Road
Mountain View, CA 94043, USA

Eric W. Sink
eric@spyglass.com
Senior Software Engineer
Spyglass, Inc.
1800 Woodfield Drive
Savoy, IL 61874

Lawrence C. Stewart
stewart@OpenMarket.com
Open Market, Inc.
215 First Street
Cambridge, MA 02142

--
Eric W. Sink, Senior Software Engineer -- eric@spyglass.com

http://www.spyglass.com/~eric/home.htm