Internet Engineering Task Force	P. Hallam-Baker
Internet-Draft	Comodo Group Inc.
Intended status: Standards Track	October 19, 2012
Expires: April 22, 2013	

HTTP Integrity Header
draft-hallambaker-httpintegrity-01

Abstract

The HTTP Integrity header provides a means of authenticating HTTP requests and responses using a Message Authentication Code (MAC). This document defines the HTTP integrity header and specifies its use to authenticate and verify specific parts of an HTTP message. the means by which the symmetric or asymmetric keys used to authenticate the messages is outside the scope of this document.

Status of this Memo

Copyright Notice

Table of Contents

TOC

1. Introduction

The HTTP Integrity header provides a simple and effective means of authenticating components of a HTTP transaction at the HTTP Application layer without disclosure of the secret(s) used as the basis for authentication.

This approach has considerable advantages over the traditional approaches of using HTTP Cookies containing an authentication secret, embedding authentication data within the HTTP message content or relying on TLS integrity checks alone.

The Integrity header provides an equivalent security functionality to the use of authentication cookies without the vulnerabilities intrinsic to the use of static authentication secrets that are disclosed to the verifying party en-clair to effect verification.

Use of a HTTP header to provide integrity information offers a simpler and more flexible approach than use of schemes such as PKCS#7/CMS, XML Signature or JSON Signature. The chief technical challenge in such specifications being to reliably identify the exact scope of the data being signed and the form of encoding. Moving the integrity check to a HTTP header permits the scope of the signature to be defined as the scope of the HTTP content body and the encoding to be the HTTP transport encoding.

Use of the HTTP Intergity header permits the same mutual authentication guarantees provided by TLS client authentication without the need to provision client certificates and with considerably less complexity.

For simplicity, the HTTP Integrity header is strictly limited to identifying a security context, the HTTP transaction item(s) to be authenticated and the resulting authentication value. The means of establishing the keys and algorithms that make up the security context are outside the scope of this document.

In the typical case the security context identifier is a ticket (c.f. Kerberos) that contains the account identifier, shared secret(s) and algorithm identifier required a Web Service protocol or Web Browser Authentication protocol. This approach permits a stateless server design in which the server does not store per-account keys.

TOC

1.1. Use in Web Services

The HTTP Integrity header was originally developed to simplify implemenatation of Web Services. Using the SOAP approach a Web Service message is encoded in XML, wrapped in a SOAP envelope and a WS-Security header with an XML Signature attached. The whole package is then attached to a HTTP message as a content payload.

This approach involves a considerable degree of complexity and in most cases achieves nothing more than attaching an integrity check. Carrying the integrity check as a HTTP header eliminates the need for the SOAP and WS-Security layers entirely. For example, the following example is a HTTP request in the Omnibroker connectin protocol.

```
Post / HTTP/1.1
Host: example.com
Cache-Control: no-store
Content-Type: Application/json;charset=UTF-8
Content-Length: 78
Integrity: content=true;
    value=cjkMkfnnYP8JYWZAbRLvtpqImmOK3rsrOT1XcvAgHDk=;
    id=TUMnorO0SjHHS7D2uFcGlRYJ0Hd3eibwe0ogptoNMQuCYmCHfHAJcJlyvi
      j8WoXDglTSOkctnmoBzl8W0NLSlcgSyZcmsAyoWs8y1Rn2ZlO2WBgoWrFIOqPa4
      oB29dgs/ei6ieINZtmvXNCm2NUkWA==

{
"TicketRequest": {
"ChallengeResponse": "TctLOG74cwpm26YNpEibcQ=="}}
```

A single HTTP message MAY have multiple Integrity headers. This facilitates support for multi-party transactions in which A submits a transaction to B who countersigns it and passes it to C who is required to chek that she has proof of agreement by both A and B.

Use of the Integrity header permits the developer to isolate integrity and authentication checks to a single point of control, as is advised by best security practice. The security monitor examines a HTTP message, verifies that the required integrity data is present and correct and only passes the payload on for processing by the Web Service itself if and only if the verification checks have been passed.

1.2. User Authentication

Use of the HTTP Intgerity header addresses the most challenging problem facing the designers of SAML, OpenID and OAUTH, this being binding the established authentication context to the HTTP messages sent by the authenticated parties. The problem is difficult because the protocols are designed to interoperate with legacy browsers. Providing explicit support for an integrity check in the HTTP protocol would require browsers with native support for the authentication mechanism but greatly reduce the complexity of providing this support.

2. Syntax and options

2.1. Attribute id=[base64(value)]

The ticket attribute identifies the security context under which the authentication value has been generated. The attribute is an opaque sequence of octets in base64 encoding.

2.2. Attribute nonce=[base64(value)]

Specifies a nonce value to be used in combination with the specified scope.

2.3. Attribute value=[base64(value)]

The mac attribute specifies the value resulting from applying the security context and nonce (if present) to the specified scope.

The scope is specified by the start, header and content attributes. The order in which the scope attributes are specified is immaterial. The scope is always constructed in the same order as the elements occur in a HTTP message, i.e. start, headers and content.

2.4. Attribute content=[true|false]

If set true, the specified scope includes the message body. The content transfer encoding (e.g. chunked) is ignored for the purpose of determining the content.

2.5. Attribute start=[true|false]

If set true, the specified scope includes the message start line. This being the request Line in the case of a request and the status line in the case of a response.

2.6. Attribute header=[escaped(headers)]

Specifies HTTP headers to be included in the specified scope following the escaped encoding defined in DKIM.

3. Security Considerations

3.1. Data outside the specified scope is not authenticated

The integrity check only extends to the portions of the message that are within the specified scope.

3.2. Truncated Hash Algorithms

If the security context permits the use of a truncated MAC, it MUST specify the minimum length of the MAC after truncation and verifiers MUST reject MAC values shorter than that length as invalid.

3.3. Randomness of Secret Keys and nonces

The security of any cryptographic protocol relies on the difficulty of guessing secret keys. Secret keys and nonces SHOULD be generated using a mechanism that ensures that the range of possible values is sufficiently large to prevent 'brute force' guessing attacks. (see that RFC of EKR's)

3.4. Weak Ciphers

Specification of the cryptographic algorithms used to construct the Integrity header value is implicit in the Security context identifier and thus outside the scope of this specification.

4. IANA Considerations

Add the 'Integrity' header to the list of provisional HTTP headers.

[Upgrade if/when this becomes an RFC]

5. Normative References

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

Author's Address

TOC

Phillip Hallam-Baker
Comodo Group Inc.
Email: phillip@comodo.com