Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 26, 2008

P. Hoffman
VPN Consortium
A. Melnikov
Isode Ltd.

January 23, 2008

Security Requirements for HTTP
draft-ietf-httpbis-security-properties-00.txt

Status of this Memo

Copyright Notice

Abstract

Recent IESG practice dictates that IETF protocols must specify mandatory-to-implement security mechanisms, so that all conformant implementations share a common baseline. This document examines all widely deployed HTTP security technologies, and analyzes the trade-offs of each.

Table of Contents

1. Introduction

Recent IESG practice dictates that IETF protocols are required to specify mandatory to implement security mechanisms. "The IETF Standards Process" [RFC2026] does not require that protocols specify mandatory security mechanisms. "Strong Security Requirements for IETF Standard Protocols" [RFC3365] requires that all IETF protocols provide a mechanism for implementors to provide strong security. RFC 3365 does not define the term "strong security".

"Security Mechanisms for the Internet" [RFC3631] is not an IETF procedural RFC, but it is perhaps most relevant. Section 2.2 states:

> We have evolved in the IETF the notion of "mandatory to implement" mechanisms. This philosophy evolves from our primary desire to ensure interoperability between different implementations of a protocol. If a protocol offers many options for how to perform a particular task, but fails to provide for at least one that all must implement, it may be possible that multiple, non-interoperable implementations may result. This is the consequence of the selection of non-overlapping mechanisms being deployed in the different implementations.

This document examines the effects of applying security constraints to Web applications, documents the properties that result from each method, and will make Best Current Practice recommendations for HTTP security in a later document version. At the moment, it is mostly a laundry list of security technologies and tradeoffs.

2. Existing HTTP Security Mechanisms

For HTTP, the IETF generally defines "security mechanisms" as some combination of access authentication and/or a secure transport.

2.1. Forms And Cookies

Almost all HTTP authentication is accomplished through HTML forms, with session keys stored in cookies. For cookies, most implementations rely on the "Netscape specification", which is described loosely in section 10 of "HTTP State Management Mechanism" [RFC2109]. The protocol in RFC 2109 is relatively widely implemented, but most clients don't advertise support for it. RFC 2109 was later updated [RFC2965], but the newer version is not widely implemented.

Forms and cookies have number of properties that make them an excellent solution for some implementors. However, many of those

properties introduce serious security trade-offs.

HTML forms provide a large degree of control over presentation, which is an imperative for many websites. However, this increases user reliance on the appearance of the interface. Many users do not understand the construction of URIs [RFC3986], or their presentation in common clients [[CITATION NEEDED]]. As a result, forms are extremely vulnerable to spoofing.

HTML forms provide acceptable internationalization if used carefully, at the cost of being transmitted as normal HTTP content in all cases (credentials are not differentiated in the protocol).

HTML forms provide a facility for sites to indicate that a password should never be pre-populated. [[More needed here on autocomplete]]

The cookies that result from a successful form submission make it unessecary to validate credentials with each HTTP request; this makes cookies an excellent property for scalability. Cookies are susceptible to a large variety of XSS (cross-site scripting) attacks, and measures to prevent such attacks will never be as stringent as necessary for authentication credentials because cookies are used for many purposes. Cookies are also susceptible to a wide variety of attacks from malicious intermediaries and observers. The possible attacks depend on the contents of the cookie data. There is no standard format for most of the data.

HTML forms and cookies provide flexible ways of ending a session from the client.

HTML forms require an HTML rendering engine, which many protocols have no use for.

2.2. HTTP Access Authentication

HTTP 1.1 provides a simple authentication framework, and "HTTP Authentication: Basic and Digest Access Authentication" [RFC2617] defines two optional mechanisms. Both of these mechanisms are extremely rarely used in comparison to forms and cookies, but some degree of support for one or both is available in many implementations. Neither scheme provides presentation control, logout capabilities, or interoperable internationalization.

2.2.1. Basic Authentication

Basic Authentication (normally called just "Basic") transmits usernames and passwords in the clear. It is very easy to implement, but not at all secure unless used over a secure transport.

Basic has very poor scalability properties because credentials must be revalidated with every request, and because secure transports negate many of HTTP's caching mechanisms. Some implementations use cookies in combination with Basic credentials, but there is no standard method of doing so.

Since Basic credentials are clear text, they are reusable by any party. This makes them compatible with any authentication database, at the cost of making the user vulnerable to mismanaged or malicious servers, even over a secure channel.

Basic is not interoperable when used with credentials that contain characters outside of the ISO 8859-1 repertoire.

2.2.2. Digest Authentication

In Digest Authentication, the client transmits the results of hashing user credentials with properties of the request and values from the server challenge. Digest is susceptible to man-in-the-middle attacks when not used over a secure transport.

Digest has some properties that are preferable to Basic and Cookies.

Credentials are not immediately reusable by parties that observe or receive them, and session data can be transmitted along side credentials with each request, allowing servers to validate credentials only when absolutely necessary. Authentication data session keys are distinct from other protocol traffic.

Digest includes many modes of operation, but only the simplest modes enjoy any degree of interoperability. For example, most implementations do not implement the mode that provides full message integrity. Additionally, implementation experience has shown that the message integrity mode is impractical because it requires servers to analyze the full request before determining whether the client knows the shared secret.

Digest is extremely susceptible to offline dictionary attacks, making it practical for attackers to perform a namespace walk consisting of a few million passwords [[CITATION NEEDED]].

Many of the most widely-deployed HTTP/1.1 clients are not compliant when GET requests include a query string [Apache_Digest].

Digest either requires that authentication databases be expressly designed to accomodate it, or requires access to cleartext passwords. As a result, many authentication databases that chose to do the former are incompatible, including the most common method of storing passwords for use with Forms and Cookies.

Many Digest capabilities included to prevent replay attacks expose the server to Denial of Service attacks.

Digest is not interoperable when used with credentials that contain characters outside of the ISO 8859-1 repertoire.

2.2.3. Other Access Authentication Schemes

There are many niche schemes that make use of the HTTP Authentication framework, but very few are well documented. Some are bound to transport layer connections.

2.2.3.1. Negotiate (GSS-API) Authentication

[[A discussion about "SPNEGO-based Kerberos and NTLM HTTP

Authentication in Microsoft Windows" [RFC4559] goes here.]]

2.3. Centrally-Issued Tickets

Many large Internet services rely on authentication schemes that center on clients consulting a single service for a time-limited ticket that is validated with undocumented heuristics. Centralized ticket issuing has the advantage that users may employ one set of credentials for many services, and clients don't send credentials to many servers. This approach is often no more than a sophisticated application of forms and cookies.

All of the schemes in wide use are proprietary and non-standard, and usually are undocumented. There are many standardization efforts in progress, as usual.

2.4. Web Services

Many security properties mentioned in this document have been recast in XML-based protocols, using HTTP as a substitute for TCP. Like the amalgam of HTTP technologies mentioned above, the XML-based protocols are defined by an ever-changing combination of standard and vendor-produced specifications, some of which may be obsoleted at any time [WS-Pagecount] without any documented change control procedures. These protocols usually don't have much in common with the Architecture of the World Wide Web. It's not clear why term "Web" is used to group them, but they are obviously out of scope for HTTP-based application protocols.

2.5. Transport Layer Security

[[A discussion of HTTP over TLS needs to be added here.]]

[[Discussion of connection confidentiality should be separate from the discussion of access authentication based on mutual authentication with certificates in TLS.]]

3. Revisions To HTTP

Is is possible that HTTP will be revised in the future. "HTTP/1.1"

[RFC2616] and "Use and Interpretation of HTTP Version Numbers" [RFC2145] define conformance requirements in relation to version numbers. In HTTP 1.1, all authentication mechanisms are optional, and no single transport substrate is specified. Any HTTP revision that adds a mandatory security mechanism or transport substrate will have to increment the HTTP version number appropriately. All widely used schemes are non-standard and/or proprietary.

4. Security Considerations

This entire document is about security considerations.

5. Normative References

[Apache_Digest]
Apache Software Foundation, "Apache HTTP Server - mod_auth_digest", <http://httpd.apache.org/docs/1.3/mod/mod_auth_digest.html>.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2109] Kristol, D. and L. Montulli, "HTTP State Management Mechanism", RFC 2109, February 1997.

[RFC2145] Mogul, J., Fielding, R., Gettys, J., and H. Nielsen, "Use and Interpretation of HTTP Version Numbers", RFC 2145, May 1997.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[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.

[RFC2965] Kristol, D. and L. Montulli, "HTTP State Management

Mechanism", RFC 2965, October 2000.

[RFC3365] Schiller, J., "Strong Security Requirements for Internet Engineering Task Force Standard Protocols", BCP 61, RFC 3365, August 2002.

[RFC3631] Bellovin, S., Schiller, J., and C. Kaufman, "Security Mechanisms for the Internet", RFC 3631, December 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4559] Jaganathan, K., Zhu, L., and J. Brezak, "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", RFC 4559, June 2006.

[WS-Pagecount]
Bray, T., "WS-Pagecount", September 2004, <http://www.tbray.org/ongoing/When/200x/2004/09/21/WS-Research>.

Appendix A. Acknowledgements

Much of the material in this document was written by Rob Sayre, who first promoted the topic.

Appendix B. Document History

[This entire section is to be removed when published as an RFC.]

B.1. Changes between draft-sayre-http-security-variance-00 and draft-ietf-http-security-properties-00

Changed the authors to Paul Hoffman and Alexey Melnikov, with permission of Rob Sayre.

Made lots of minor editorial changes.

Removed what was section 2 (Requirements Notation), the reference to RFC 2119, and any use of 2119ish all-caps words.

In 3.2.1 and 3.2.2, changed "Latin-1 range" to "ISO 8859-1 repertoire" to match the defintion of "TEXT" in RFC 2616.

Added minor text to the Security Considerations section.

Added URLs to the two non-RFC references.

Authors' Addresses

Paul Hoffman
VPN Consortium

Email: paul.hoffman@vpnc.org

Alexey Melnikov
Isode Ltd.

Email: alexey.melnikov@isode.com

Full Copyright Statement

Intellectual Property

ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).