Network Working Group
Request for Comments: 2964
BCP: 44
Category: Best Current Practice

K. Moore
University of Tennessee
N. Freed
Innosoft

October 2000

Use of HTTP State Management

Status of this Memo

Copyright Notice

IESG Note

The IESG notes that this mechanism makes use of the .local top-level domain (TLD) internally when handling host names that don't contain any dots, and that this mechanism might not work in the expected way should an actual .local TLD ever be registered.

Abstract

The mechanisms described in "HTTP State Management Mechanism" (RFC-2965), and its predecessor (RFC-2109), can be used for many different purposes. However, some current and potential uses of the protocol are controversial because they have significant user privacy and security implications. This memo identifies specific uses of Hypertext Transfer Protocol (HTTP) State Management protocol which are either (a) not recommended by the IETF, or (b) believed to be harmful, and discouraged. This memo also details additional privacy considerations which are not covered by the HTTP State Management protocol specification.

1. Introduction

The HTTP State Management mechanism is both useful and controversial. It is useful because numerous applications of HTTP benefit from the ability to save state between HTTP transactions, without encoding such state in URLs. It is controversial because the mechanism has been used to accomplish things for which it was not designed and is not well-suited. Some of these uses have attracted a great deal of public criticism because they threaten to violate the privacy of web

users, specifically by leaking potentially sensitive information to third parties such as the Web sites a user has visited. There are also other uses of HTTP State Management which are inappropriate even though they do not threaten user privacy.

This memo therefore identifies uses of the HTTP State Management protocol specified in RFC-2965 which are not recommended by the IETF, or which are believed to be harmful and are therefore discouraged.

This document occasionally uses terms that appear in capital letters. When the terms "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" appear capitalized, they are being used to indicate particular requirements of this specification. A discussion of the meanings of the terms "MUST", "SHOULD", and "MAY" appears in [RFC-1123]; the terms "MUST NOT" and "SHOULD NOT" are logical extensions of this usage.

2. Uses of HTTP State Management

The purpose of HTTP State Management is to allow an HTTP-based service to create stateful "sessions" which persist across multiple HTTP transactions. A single session may involve transactions with multiple server hosts. Multiple client hosts may also be involved in a single session when the session data for a particular user is shared between client hosts (e.g., via a networked file system). In other words, the "session" retains state between a "user" and a "service", not between particular hosts.

It's important to realize that similar capabilities may also be achieved using the "bare" HTTP protocol, and/or dynamically-generated HTML, without the State Management extensions. For example, state information can be transmitted from the service to the user by embedding a session identifier in one or more URLs which appear in HTTP redirects, or dynamically generated HTML; and the state information may be returned from the user to the service when such URLs appear in a GET or POST request. HTML forms can also be used to pass state information from the service to the user and back, without the user being aware of this happening.

However, the HTTP State Management facility does provide an increase in functionality over ordinary HTTP and HTML. In practice, this additional functionality includes:

(1) The ability to exchange URLs between users, of resources accessed during stateful sessions, without leaking the state information associated with those sessions. (e.g. "Here's the URL for the FooCorp web catalog entry for those sandals that you wanted.")

(2) The ability to maintain session state without "cache-busting". That is, separating the session state from the URL allows a web cache to maintain only a single copy of the named resource. If the state is maintained in session-specific URLs, the cache would likely have to maintain several identical copies of the resource.

(3) The ability to implement sessions with minimal server configuration and minimal protocol overhead, as compared to other techniques of maintaining session state.

(4) The ability to associate the user with session state whenever a user accesses the service, regardless of whether the user enters through a particular "home page" or "portal".

(5) The ability to save session information in stable storage, so that a "session" can be maintained across client invocations, system reboots, and client or system crashes.

2.1. Recommended Uses

Use of HTTP State Management is appropriate whenever it is desirable to maintain state between a user and a service across multiple HTTP transactions, provided that:

(1) the user is aware that session state is being maintained and consents to it,

(2) the user has the ability to delete the state associated with such a session at any time,

(3) the information obtained through the ability to track the user's usage of the service is not disclosed to other parties without the user's explicit consent, and

(4) session information itself cannot contain sensitive information and cannot be used to obtain sensitive information that is not otherwise available to an eavesdropper.

This last point is important because cookies are usually sent in the clear and hence are readily available to eavesdroppers.

An example of such a recommended use would be a "shopping cart", where the existence of the shopping cart is explicitly made known to the user, the user can explicitly "empty" his or her shopping cart (either by requesting that it be emptied or by purchasing those

items) and thus cause the shared state to be discarded, and the service asserts that it will not disclose the user's shopping or browsing habits to third parties without the user's consent.

Note that the HTTP State Management protocol effectively allows a service provider to refuse to provide a service, or provide a reduced level of service, if the user or a user's client fails to honor a request to maintain session state. Absent legal prohibition to the contrary, the server MAY refuse to provide the service, or provide a reduced level of service, under these conditions. As a purely practical consideration, services designed to utilize HTTP State Management may be unable to function properly if the client does not provide it. Such servers SHOULD gracefully handle such conditions and explain to the user why the full level of service is not available.

2.2. Problematic Uses

The following uses of HTTP State Management are deemed inappropriate and contrary to this specification:

2.2.1. Leakage of Information to Third Parties

HTTP State Management MUST NOT be used to leak information about the user or the user's browsing habits to other parties besides the user or service, without the user's explicit consent. Such usage is prohibited even if the user's name or other externally-assigned identifier are not exposed to other parties, because the state management mechanism itself provides an identifier which can be used to compile information about the user.

Because such practices encourage users to defeat HTTP State Management mechanisms, they tend to reduce the effectiveness of HTTP State Management, and are therefore considered detrimental to the operation of the web.

2.2.2. Use as an Authentication Mechanism

It is generally inappropriate to use the HTTP State Management protocol as an authentication mechanism. HTTP State Management is not designed with such use in mind, and safeguards for protection of authentication credentials are lacking in both the protocol specification and in widely deployed HTTP clients and servers. Most HTTP sessions are not encrypted and "cookies" may therefore be exposed to passive eavesdroppers. Furthermore, HTTP clients and servers typically store "cookies" in cleartext with little or no protection against exposure. HTTP State Management therefore SHOULD

NOT be used as an authentication mechanism to protect information from being exposed to unauthorized parties, even if the HTTP sessions are encrypted.

The prohibition against using HTTP State Management for authentication includes both its use to protect information which is provided by the service, and its use to protect potentially sensitive information about the user which is entrusted to the service's care. For example, it would be inappropriate to expose a user's name, address, telephone number, or billing information to a client that merely presented a cookie which had been previously associated with the user.

Similarly, HTTP State Management SHOULD NOT be used to authenticate user requests if unauthorized requests might have undesirable side-effects for the user, unless the user is aware of the potential for such side-effects and explicitly consents to such use. For example, a service which allowed a user to order merchandise with a single "click", based entirely on the user's stored "cookies", could inconvenience the user by requiring her to dispute charges to her credit card, and/or return the unwanted merchandise, in the event that the cookies were exposed to third parties.

Some uses of HTTP State Management to identify users may be relatively harmless, for example, if the only information which can be thus exposed belongs to the service, and the service will suffer little harm from the exposure of such information.

3. User Interface Considerations for HTTP State Management

HTTP State Management has been very controversial because of its potential to expose information about a user's browsing habits to third parties, without the knowledge or consent of the user. While such exposure is possible, this is less a flaw in the protocol itself than a failure of HTTP client implementations (and of some providers of HTTP-based services) to protect users' interests.

As implied above, there are other ways to maintain session state than using HTTP State Management, and therefore other ways in which users' browsing habits can be tracked. Indeed, it is difficult to imagine how the HTTP protocol or an HTTP client could actually prevent a service from disclosing a user's "click trail" to other parties if the service chose to do so. Protection of such information from inappropriate exposure must therefore be the responsibility of the service. HTTP client implementations inherently cannot provide such protection, though they can implement countermeasures which make it more difficult for HTTP State Management to be used as the mechanism by which such information is exposed.

It is arguable that HTTP clients should provide more protection in general against inappropriate exposure of tracking information, regardless of whether the exposure were facilitated by use of HTTP State Management or by some other means. However, issues related to other mechanisms are beyond the scope of this memo.

3.1. Capabilities Required of an HTTP Client

A user's willingness to consent to use of HTTP State Management is likely to vary from one service to another, according to whether the user trusts the service to use the information appropriately and to limit its exposure to other parties. The user therefore SHOULD be able to control whether his client supports a service's request to use HTTP State Management, on a per-service basis. In particular:

(1) Clients MUST NOT respond to HTTP State Management requests unless explicitly enabled by the user.

(2) Clients SHOULD provide an effective interface which allows users to review, and approve or refuse, any particular requests from a server to maintain state information, before the client provides any state information to the server.

(3) Clients SHOULD provide an effective interface which allows users to instruct their clients to ignore all requests from a particular service to maintain state information, on a per-service basis, immediately in response to any particular request from a server, before the client provides any state information to the server.

(4) Clients SHOULD provide an effective interface which allows a user to disable future transmission of any state information to a service, and/or discard any saved state information for that service, even though the user has previously approved a service's request to maintain state information.

(5) Clients SHOULD provide an effective interface which allows a user to terminate a previous request not to retain state management information for a given service.

3.2. Limitations of the domain-match algorithm

The domain-match algorithm in RFC-2965 section 2 is intended as a heuristic to allow a client to "guess" whether or not two domains are part of the same service. There are few rules about how domain names can be used, and the structure of domain names and how they are delegated varies from one top-level domain to another (i.e. the client cannot tell which part of the domain was assigned to the

service). Therefore NO string comparison algorithm (including the domain-match algorithm) can be relied on to distinguish a domain that belongs to a particular service, from a domain that belongs to another party.

As stated above, each service is ultimately responsible for ensuring that user information is not inappropriately leaked to third parties. Leaking information to third parties via State Management by careful selection of domain names, or by assigning domain names to hosts maintained by third parties, is at least as inappropriate as leaking the same information by other means.

4. Security Considerations

This entire memo is about security considerations.

5. Authors' Addresses

Keith Moore
University of Tennessee Computer Science Department
1122 Volunteer Blvd, Suite 203
Knoxville TN, 37996-3450

EMail: moore@cs.utk.edu

Ned Freed
Innosoft International, Inc.
1050 Lakes Drive
West Covina, CA 81790

EMail: ned.freed@innosoft.com

6. References

[RFC 1123] Braden, R., "Requirements for Internet Hosts -- Application and Support", STD 3, RFC 1123, October 1989.

[RFC 2965] Kristol, D. and L. Montulli, "HTTP State Management Mechanism", RFC 2965, October 2000.

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

7. Full Copyright Statement

Acknowledgement

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