Network Working Group
Internet-Draft
Updates: 2109,2965
(if approved)
Intended status: Standards Track
Expires: September 15, 2011

Y. Pettersen
Opera Software ASA

March 14, 2011

Identifying origin server of HTTP Cookies
draft-pettersen-cookie-origin-02

Abstract

HTTP Cookies, as originally defined by Netscape in [NETSC] and as later updated by [RFC2109] , [RFC2965], and [I-D.ietf-httpstate-cookie] did not address the issue of how to restrict for which domains a server is allowed to set a cookie. This is particularly a problem for servers hosted in top-level domains having subdomains that are controlled by registries and not by domain owners, e.g., "co.uk" and "city.state.us" domains. In such situations, unless the client uses some kind of domain black-list, it is possible for a malicious server to set cookies, so they are sent to all servers in a domain the attacker does not control. These cookies may adversely affect the function of servers receiving them. The primary reason this is a problem is that the server receiving the cookie has no way of telling which server originally set it; therefore it is not able to distinguish reliably an invalid cookie from a valid one.

This document proposes a new attribute, "$Origin", that is associated with each cookie and sent in all client cookie headers in the requests sent to the server. Servers recognizing the attribute may then check to see if the cookie was set by a server, which is allowed to set cookies for the server and, if necessary, ignore the cookie.

This document updates RFC 2109 and RFC 2965.

Requirements Language

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 [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

1. Introduction

When originally defined, Netscape's HTTP Cookie specification [NETSC], updated by [I-D.ietf-httpstate-cookie] did not extensively specify how clients should check the domain of cookies. Although [RFC2109] and [RFC2965] did put restrictions on the domains for which a server setting cookies could set cookies, these policies have not been widely implemented and are also not able to protect against all possible abuses.

Clients have attempted to limit this problem by using heuristics and domain blacklists to determine for which domains they can set cookies. However, these workarounds have limits both in terms of correctness, and the amount of data needed to implement them, as well as in the timeliness of updates to the list.

Alternatively, servers have no way to determine whether a cookie it receives from a client is one of the cookies it sent to the client, or, if it came from another server, which server originally set it. The server may include information in the cookie's value to determine correctness. However, this does not guard against a malicious server using a correctly generated cookie that was originally sent to a different client.

A way to allow servers to learn whether received cookies are valid and not set by an unauthorized server is to include the name or URL of the server that originally set the cookie in an attribute, "$Origin", associated with each cookie value in the Cookie header sent to the server. This attribute would either identify the name of the server that set the cookie, or if the name of this server is not known, the domain for which the cookie has been set. This allows the receiving server to remove or ignore cookies set by servers not allowed to set cookies for its domain and also to log the information about the incorrectly set cookies.

2. The $Origin attribute

2.1. General syntax

This specification uses the same syntax as is used by [RFC2109] , [RFC2965] , and [RFC2616]

```
   attr        =     token
   value       =     token | quoted-string
```

2.2. Updated client processing of received cookies

When a client receives a Set-Cookie or Set-Cookie2 header, it will process the header as specified by the appropriate specification, which can be either [NETSC] , [I-D.ietf-httpstate-cookie], [RFC2109], or [RFC2965] . When storing the cookie, it MUST also register information about the host setting the cookie. This information MUST include the hostname and SHOULD include parts of the, or the entire, URI that set the cookie, including the scheme.

2.3. Updated Cookie header syntax

This specification updates the Cookie header as sent by the client by associating each cookie value with a $Origin attribute that specifies where the the cookie came from.

This specification does not change the way cookies are selected for inclusion in the Cookie header.

The syntax for the header field is:

```
cookie          =  "Cookie:" cookie-value 0*(";" cookie-value)
cookie-value    =  NAME "=" VALUE ";" cookie-origin

NAME            =  attr
VALUE           =  value
cookie-origin   =  "$Origin" "=" <"> http_URL <">
```

NAME and VALUE have the same meaning as in[I-D.ietf-httpstate-cookie], [RFC2109] and [RFC2965].

The http_URL value of the $Origin attribute MUST be the URI of the resource setting the cookie, which SHOULD be restricted to the default path (remove the query part and the last path segment). If the client does not know the URI that originally set the cookie,such as when the cookie was received by a version of the client that does not support $Origin, it MUST instead send a generated default URL "http:// "+domainname+"/", where domainname is the name of the domain for which the cookie is set. This domain name MUST be preceded by a single period (".") to differentiate the domain name from a hostname.

The http_URL value MUST be encoded as described in [RFC3986] .

When receiving a cookie header containing $Origin, servers recognizing it SHOULD check if the identified host or domain from the URI in the argument is acceptable to the server. If the cookie is not from an acceptable host or domain, the cookie can be ignored and

optionally reported to the server administrator. The server SHOULD also ignore all cookies that are not followed by a $Origin attribute, if one cookie in the header has a $Origin attribute.

[[Open issue: An option for cases with unknown origin is to send an empty $Origin attribute or no $Origin attribute for that cookie. An argument against having a special dot prefix is that these cookies will only exist for a limited time after a client has been updated to set and send $Origin. The author thinks it is better to provide some information to the server about the domain of the cookie, rather than to provide no information. Either case would require special handling in the server.]]

[[Open issue: An alternative requirement for the URI is to include all of the original URI, except the query portion.]]

[[Open issue: Mention HTTPS URLs? What about including HTTPS URLs in requests to unencrypted HTTP resources? Change to HTTP URL?

3. Examples

http://www.example.com/path1/resource?query sets the cookie:

```
Set-Cookie: foo=value1; domain=.example.com; path=/
```

http://www2.example.com/path2/resource2?query1 sets the cookie:

```
Set-Cookie: bar=value2; domain=.example.com; path=/
```

An unkown server set the cookie:

```
Set-Cookie: xyz=value3;  domain=.example.com; path=/
```

The resulting Cookie header is:

```
Cookie:  foo=value1; $Origin="http://www.example.com/path1/";
         bar=value2; $Origin="http://www2.example.com/path2/";
         xyz=value3; $Origin="http://.example.com/"
```

4. IANA Considerations

This document makes no request of IANA.

Note to the RFC Editor: this section may be removed upon publication as a RFC.

5. Security Considerations

This specification is intended to make the sharing of cookies across domains detectable, whether the sharing is intentional, unintentional, or with malicious intent. It can, therefore, also be used to limit the potential for cookie spoofing, as discussed in the security considerations of [RFC2109] and [RFC2965] . It is, however, still possible for servers within a permitted group of servers to set incorrect or malicioius cookies, which might adversely affect other servers in the domain.

6. Normative References

[I-D.ietf-httpstate-cookie]
Barth, A., "HTTP State Management Mechanism", draft-ietf-httpstate-cookie-23 (work in progress), March 2011.

[NETSC] "Persistent Client State -- HTTP Cookies", <http://www.netscape.com/newsref/std/cookie_spec.html>.

available at <http://www.netscape.com/newsref/std/cookie_spec.html>

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 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.

[RFC2965] Kristol, D. and L. Montulli, "HTTP State Management Mechanism", RFC 2965, October 2000.

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

Author's Address

Yngve N. Pettersen
Opera Software ASA
Waldemar Thranes gate 98
N-0175 OSLO,
Norway

Email: yngve@opera.com