Enhanced validation of domains for HTTP State Management Cookies using DNS

draft-pettersen-dns-cookie-validate-05

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79. This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on December 23, 2009.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the
Abstract

HTTP State Management Cookies are used for a wide variety of tasks on the Internet, from preference handling to user identification. An important privacy and security feature of cookies is that their information can only be sent to servers in a limited namespace, the domain.

The variation of domain structures that are in use by domain name registries, especially the country code Top Level Domains (ccTLD) namespaces, makes it difficult to determine what is a valid domain, e.g. example.co.uk and example.no, which cookies should be permitted for, and a registry-like domain (subTLDs) like co.uk where cookies should not be permitted.

This document specifies an imperfect method using DNS name lookups for cookie domains to determine if cookies can be permitted for that domain, based on the assumption that most subTLD domains will not have an IP address assigned to them, while most legitimate services that share cookies among multiple servers will have an IP address for their domain name to make the user’s navigation easier by omitting the customary "www" prefix.

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].
Table of Contents

1. Introduction .................................................. 4
2. ABNF for the hostname and domain-attribute ................. 5
3. Domain matching summary ..................................... 6
4. Problem description ........................................... 7
5. A DNS based approach ......................................... 8
   5.1. Foundations ............................................... 8
   5.2. Method for DNS validation of cookie domains .......... 9
   5.3. Incorrect results ......................................... 10
6. IANA Considerations ........................................... 11
7. Security Considerations ....................................... 11
8. References .................................................... 12
   8.1. Normative References ................................. 12
   8.2. Non-normative references ............................. 12
Author’s Address ................................................ 12
1. Introduction

HTTP State Management Cookies are used to maintain a state shared between several HTTP resources within a domain. This state can for example be a login ID, a shopping cart, or user configurable preference settings.

Presently, two somewhat compatible Cookie formats exist: Netscape’s original specification [NETSC], which is currently the most widely deployed version, and [RFC2965] cookies. While syntactically similar, these definitions specify different response headers due to compatibility issues, but use the same request header, with some modifications mandated by [RFC2965].

Cookies are usually sent by an HTTP server to the client as one or more headers in the response to a request, and the client may permit the received cookie(s) to be stored locally on the client machine, so that they may later be returned to the server attached as a header to future requests for HTTP resources within the domain specified by the server, for as long as the cookies are valid. Alternative mechanisms for setting cookies are also available through HTML Meta tags and an ECMAScript interface.

To prevent cookies set by one website from interfering with other, independent websites or leaking sensitive information to such websites, a number of limitations exist for which websites may receive cookies set by a given website.

The primary limitation is the domain attribute of the cookie. This attribute either defines the name of the website that will receive the cookie, or the Internet domain name that must be the suffix of all servers that will receive the cookie. The default is that if no domain attribute is specified by the server the cookie can only be sent to the server that set the cookie.

The domain mechanism does however have certain limitations, limitations that become obvious when cookies are used in the national domains outside the generic top level domains (TLDs): ".com", ".net", ".org", ".gov", ".mil", ".edu" and ".int".

The national domains are organized in various ways, some have a flat structure, like the one used by the .com domain, while others have one or more hierarchical levels that are used to indicate what kind of service the domain is used for, e.g. co.uk is used for commercial domains in the UK, while ac.uk is used for academic institutions.

Many national domains are using a hybrid of these two structures.

These various namespace structures cause problems when a client is
going to decide if a cookie sent by a server can be set. As national
domain administrators are free to organize, and name, their domain
name structures as they wish, there are no general rules available to
tell a client if a given domain is a valid website domain (e.g.
example.co.uk or example.no), or one of the hierarchical subTLDs
(like "co.uk"). Permitting a server to set a cookie for "co.uk"
could compromise the user’s privacy and possibly other issues, such
as interfering with the functionality of other servers.

[NETSC] did try to deal with the problem by requiring two internal
dots in the domain attribute (e.g. example.co.uk) when the TLD is not
one of the specified generic ones. Unfortunately, this rule was
never implemented correctly, and if it had been, it would have made
it impossible to use cookies in the many flat ccTLD domains.

[RFC2965] took another approach, by only permitting a server to set
cookies for its immediate parent domain. While this takes care of
most of the problem, it still makes it possible for the server
"example.co.uk" to set a cookie for the entire co.uk domain. This
document presents a method that supplements the existing domain
matching rules from [NETSC] and [RFC2965] by using the DNS protocol
to decide whether or not to accept the domain specified by the
server.

2. ABNF for the hostname and domain-attribute

NOTE: In this syntax the leading dot of the domain-attribute that is
required by [NETSC] and [RFC2965] is not included. Domain names MUST
be IDNA compliant [RFC3490]. ABNF syntax as defined by [RFC2616]

hostname = local-server | ip-address | full-hostname
domain-attribute = full-domain | "local"
full-hostname = ownername "." full-domain
full-domain = domainname "." toplevelname
domainname = namecomponent *( "." namecomponent)
toplevelname= generic-domain | national-toplevelname; (except "local")
generic-domain = "com" | "net" | "org" | "gov" | "mil" | "edu" | "int"
national-toplevelname = flat-national-domainname | hier-nationalname
hier-nationalname = (1*(subdomain-component ".") national-domainname)
flat-national-domainname = national-domainname
national-domainname = <any namecomponent, except those that are
part of generic-domain>
3. Domain matching summary

Deciding whether or not to permit a cookie to be set depends on matching the hostname of the server setting the cookie with the domain-attribute provided by the server. This domain matching is done according to rules laid out in [NETSC] and [RFC2965].

- If no domain-attribute is provided by the server the cookie is only accepted for the server that set the cookie; it may not be sent to any other server.

- If the hostname is an IP address, the domain-attribute MUST be an exact match of the hostname.

- If the hostname is a local-server name, the domain attribute may be "local", in which case all local-servers may receive the cookie. Otherwise, if the domain-attribute is an exact match with the hostname, it is accepted for the server identified by hostname, and only sent to that server. If there is no match between the domain attribute and the hostname, the cookie MUST be discarded.

For all other hostnames and domain-attributes a set of rules exists:

The primary rule is that the full-domain part of the full-hostname MUST match the domain-attribute exactly.

Second, while [NETSC] does not define any rules for the ownername part of a full-hostname, [RFC2965] specifies that it MUST contain only a single namecomponent, and a server can therefore only set a cookie for its own parent domain, not the grandparent domain or higher, as is permitted by [NETSC].

[NETSC] included as a third rule that all national-toplevelnames must be a hier-nationalname. However, as mentioned above, this rule has never been properly implemented by most clients.

If the cookie’s domain-attribute and the host’s hostname match according to these rules and restrictions, the cookie is accepted and will be returned to all servers that are located within the domain-
attribute’s namespace.

4. Problem description

As mentioned above, a national domain namespace can be organized as

1. A flat namespace where names are assigned as namecomponent "." flat-national-domainname, as is done in the generic domain.

2. A hierarchical namespace where names are assigned as namecomponent "." hier-national-domainname.

3. A combination of both 1 and 2.

With respect to cookies, the domain-attribute cannot be a name classified as a toplevelname domain, as that would permit a server to set cookies that can be sent to all servers within the namespace of the toplevelname domain, which might result in privacy violations such as cross domain tracking of users, or security related problems such as improper influence on the function of servers in another domain.

For domains in the generic-domain namespaces it is easy to make this distinction as a valid full-domain will always have at least two namecomponents, and the rightmost namecomponent (the toplevelname) must match one of the generic-domain alternatives.

Within the national-toplevelname namespace it is not possible to make this distinction between a valid full-domain and a national-toplevelname solely by examination of the toplevelname, UNLESS a detailed list of all names that are part of the hier-nationalname namespace is available to the client.

However, creating a list of all valid hier-nationalnames is an immense task. According to an incomplete list maintained by [GOVCOM] at least half of the 250+ national TLDs listed there use a full or partially hierarchical namespace organization. Many of the subdomain-components have names based on local naming conventions, as well as geographical areas (such as states, provinces, counties, and cities).

While it may be possible for a vendor to assemble such a list, assembling it will require massive amounts of time and resources, and it will never be complete, and must continually be updated as the namespaces are reorganized, or new nations come into existence.

Asking the user in these cases would become tedious and cause endless
irritation for the user.

A stopgap solution could be to use a list of the most common subdomain-component names, but this will leave large areas of the namespace unprotected.

5. A DNS based approach

5.1. Foundations

An HTTP client that understands cookies will, as part of its normal operation, have access to the DNS name resolution system, which it uses to convert a hostname to a network IP address.

The proposed method uses this DNS system to resolve (or attempt to resolve) the domain-attribute specified by the sending server. If the domain-attribute resolves to a valid IP address, we accept the domain-attribute as valid; if it does not resolve to a valid IP address, we assume that the domain-attribute is not a valid full-domain.

This method is based on the following assumptions:

1. It is unlikely that a national-toplevelname will be registered with an IP address. Such domains do however exist.

2. It is far more likely, although not certain, that full-domain will be registered with an IP address as an alias for www.full-domain. Many services have dispensed with the "www" part of their hostname in URIs and are using full-domain as the only active name of their service.

3. It is also likely that a service that will need to share cookies between multiple servers will have so many visitors that the administrators will set up full-domain as a valid host to make access easier for their visitors, e.g. in case they forget to use the www form of the name when entering the site’s URL into their client.

Based on this, it should be possible to perform a DNS lookup for the domain-attribute’s name, and based on the result decide whether or not to accept the cookie. If the DNS lookup succeeds and a valid IP address is retrieved, the cookie can be accepted for the given domain; if it fails, the cookie can either be discarded or the client can remove the domain attribute and continue as if that attribute had never been received, and only send the cookie to the server that sent the cookie.
The primary drawback of this solution is the fact that some sites will require domainwide cookies to function properly, but haven’t defined an IP address for the domain. In such cases the client may encounter problems that can only be solved by user intervention, such as by defining override filters or asking the service to define an IP address for the domain.

In some cases a client does not have a DNS service available that will properly resolve the domain name, even if it actually is registered with an IP address. This is usually the case when the client is located on an isolated network whose only access to the outside network is through an HTTP proxy. In such cases, when the client would use a proxy to retrieve resources, the client can use an alternative validation method by performing an HTTP HEAD request instead of a DNS request to the full-domain in order to determine its status as a valid domain.

5.2. Method for DNS validation of cookie domains

After the normal domain rules specified by the relevant specification (as discussed in Section 3) have been applied, the proposed method works as follows:

When to test:

- The domain-attribute and hostname syntax rules defined in the above rules must be obeyed.
- A domain-attribute that matches the hostname is accepted without testing.
- The rules for local-server names and IP-addresses are enforced as above, and if the cookie is acceptable by those rules the cookie can be accepted, otherwise it must be discarded.
- It is not necessary to apply the test to domain-attributes that are in the namespace of the generic-domains.
- While it is recommended that all domains that are left are tested, as a minimum the domain MUST be tested if
  - The domainname part of the full-domainname and the toplevelname each have only one namecomponent (that is, it is a flat-national-domainname), or
  - The ownername has at least one internal dot (i.e. there are multiple namecomponents in the ownername, and thus full-domainname is not the host’s parent domain)
How to test the domain attribute:

- Testing is done by performing a DNS lookup for the domain-attribute. If the lookup succeeds, and returns a valid IP address, the cookie is accepted for the given domain. If, on the other hand, the lookup fails, or returns an invalid address, the cookie is either rejected or the domain-attribute is removed from the cookie, and processing continues as if the domain attribute had never been specified - the cookie is thus only accepted for the server sending the cookie.

- If general DNS lookup is not available (e.g because the client is located in an isolated network and has to work through a proxy/gateway that is the sole access point to the Internet) the client should send HTTP HEAD requests for one or more of the following URLs:

  1. Only if the original URL was an HTTPS URL: https://domain:port/

  2. Only if the original URL was an HTTP URL: http://domain:port/

  3. Only if the original URL was an HTTPS URL: https://domain/


The port variations should only be used if a non-standard port is used. If one of these requests results in a 200- or 300-series response code, or a 401 response code (407 proxy authentication response codes are handled as they normally would have been) the lookup is considered successful, and the cookie can be accepted for the specified domain-attribute. If none of the accepted response codes are returned for any of the requests, the lookup is considered to have failed, and the domain-attribute is removed from the cookie parameters and the processing continues as mentioned in the previous step.

A user agent should not repeat this test for an alleged domain more than once every 24 hours, but it need not keep the information about failed and successful lookups between individual runs of the user agent.

5.3. Incorrect results

There are primarily two types of incorrect results that can be encountered with this method:
1. The domain-attribute is a valid full-domain, i.e. it is not a national-toplevelname, but fails the test because no IP address has been registered for the domain-attribute. In many cases this will not cause any problem, but when it does, the owner of the domain can easily fix this by adding an IP address for the full-domain in his or her DNS database, usually the same IP-address as the main server of the domain. This is a common practice among many domain owners.

2. The domain-attribute is actually a hier-nationalname, but passes the test because an IP address has been defined for the domain. This possibility may occur because a network provider or TLD registry wants to provide user friendly "unknown host" messages, or a directory service. This could be a serious problem for the visitors and website owners in the top level domain, and can only be solved by removing the DNS IP-address entry for the domain.

A third incorrect result also exists, where the full-domain is shared between many different website owners who do not want to set up, or cannot afford, a website with a full-domain owned by the website owner with all the associated administrative problems.

The method described in this document is not able to handle the second or third possibilities. Handling these cases would require that the domain owner is able to specify a policy for which servers or subdomains within the domain may set which kind of cookies. Such a policy could limit which domains or paths a given server can set cookies for. The specification of this is outside the scope of this document.

6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

The methods discussed in this document rely on the DNS system for information, and are vulnerable both to misleading information entered into the DNS system by well-meaning service providers, and to various forms of DNS related attacks, like DNS poisoning.

A DNS resolution that incorrectly permits a cookie to be set, could result in a privacy problem for the user, or a security problem on
servers receiving the incorrectly set cookie. This situation is, however, no worse than it would have been without the DNS validation routine.

The DNS lookups may reveal to attackers analyzing traffic data that the client may have received a cookie from a server in domain, and what the domain is, but will reveal no further information about the cookie, and the revealed information is ambiguous.

8. References

8.1. Normative References


8.2. Non-normative references

Author’s Address

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

Email: yngve@opera.com