Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 6, 2011

M. Nottingham

August 5, 2010

Considerations for Captive Portals in HTTP
draft-nottingham-http-portal-00

Abstract

"Captive portals" are a commonly-deployed means of obtaining access credentials and/or payment for a network. This memo discusses issues of their use for HTTP applications, and proposes one possible mitigation strategy.

This memo should be discussed on the ietf-http-wg@w3.org mailing list, although it is not a work item of the HTTPbis WG.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on February 6, 2011.

Copyright Notice

Table of Contents

1. Introduction

It has become common for networks to require authentication, payment and/or acceptance of terms of service before granting access. Typically, this occurs when accessing "public" networks such as those in hotels, trains, conference centres and similar networks.

While there are several potential means of providing credentials to a network, these are not yet universally supported, and in some instances the network administrator requires that information (e.g., terms of service, login information) be displayed to end users.

In such cases, it has become widespread practice to use a "captive portal" that diverts HTTP requests to the administrator's web page. Once the user has satisfied requirements (e.g., for payment, acceptance of terms), the diversion is ended and "normal" access to the network is allowed.

Typically, this diversion is accomplished by one of several possible techniques;

- IP interception - all requests on port 80 are intercepted and send to the portal.
- HTTP redirects - all requests on port 80 are intercepted and an HTTP redirect to the portal's URL is returned.
- DNS interception - all DNS lookups return the portal's IP address.

In each case, the intent is that users connecting to the network will open a Web browser and see the portal.

This memo examines the HTTP-related issues that these techniques raise, and proposes a potential mitigation strategy.

2. HTTP Issues Encountered

Since clients cannot differentiate between a portal's response and

that of the HTTP server that they intended to communicate with, a number of issues arise.

One example is the "favicon.ico" <http://en.wikipedia.org/wiki/Favicon> commonly used by browsers to identify the site being accessed. If the favicon for a given site is fetched from a captive portal instead of the intended site (e.g., because the user is unauthenticated), it will often "stick" in the browser's cache (most implementations cache favicons aggressively) beyond the portal session, so that it seems as if the portal's favicon has "taken over" the legitimate site.

Another browser-based issue comes about when P3P <http://www.w3.org/TR/P3P/> is supported. Depending on how it is implemented, it's possible a browser might interpret a portal's response for the p3p.xml file as the server's, resulting in the privacy policy (or lack thereof) advertised by the portal being interpreted as applying to the intended site. Other Web-based protocols such as WebFinger <http://code.google.com/p/webfinger/wiki/WebFingerProtocol>, CORS <http://www.w3.org/TR/cors/> and OAuth <http://tools.ietf.org/html/draft-ietf-oauth-v2> may also be vulnerable to such issues.

Although HTTP is most widely used with Web browsers, a growing number of non-browsing applications use it as a substrate protocol. For example, WebDAV <http://tools.ietf.org/html/rfc4918> and CalDAV <http://www.ietf.org/rfc/rfc4791.txt> both use HTTP as the basis (for network filesystem access and calendaring, respectively). Using these applications from behind a captive portal can result in spurious errors being presented to the user, and might result in content corruption, in extreme cases.

Similarly, other non-browser applications using HTTP can be affected as well; e.g., widgets <http://www.w3.org/TR/widgets/>, software updates, and other specialised software such as Twitter clients and the iTunes Music Store.

It should be noted that it's sometimes believed that using HTTP redirection to direct traffic to the portal addresses these issues. However, since many of these uses "follow" redirects, this is not a

good solution.

3. Proposal

The heart of the issues seen is that the client doesn't understand that a response from the portal does not represent the requested resource.

As such, the response needs to indicate that it is non-authoritative.

In HTTP, response status codes indicate the type of response, and therefore defining a new one is the most appropriate way to do this. Status codes are divided into general classes;

- 1xx - Informational
- 2xx - Successful
- 3xx - Redirection

- 4xx - Client errors
- 5xx - Server errors

Although it's common for captive portals to use redirection status codes, defining a new 3xx code for them isn't practical; current implementations won't recognise the new status code, and therefore won't follow it.

Error status codes, on the other hand, have a nice property in that browsers will generally display the response content if they don't understand the status code. The one exception to this is Internet Explorer, which will display a "friendly" message if the response body is too small; however, this is easy enough to work around, by padding the response message as necessary.

HTTP defines 4xx status codes as those where the error lies in the client; i.e., the client shouldn't retry the same request without changing something. This is arguably more appropriate than using a 5xx error, where the error is said to lie in the server's area of responsibility, because clients might automatically retry a request upon seeing a 5xx error.

In fact, there's already an existing status code with similar (but not quite suitable) semantics; 407 Proxy Authentication Required. What's needed is a new status code with the semantics of "Network Authentication Required."

As such, this memo proposes (but does not yet define) using a new HTTP response status code in the 4xx range with the semantics "Network Authentication Required" to mitigate the risks of captive portals.

Captive portals that deploy this status code will return it for all requests other than those to the actual portal resources (e.g., images). Clients that are unaware of the specific semantics of the new status code will fall back to treating it as a generic 400 error, and browsers will display the portal page to users.

Note that this would make the HTTP redirection technique described above obsolete; the portal page would be served directly with the new status code.

4. What about Non-HTTP Applications and Techniques?

This memo does not address non-HTTP applications, such as IMAP, POP, or even TLS-encapsulated HTTP. Since captive portals almost always target Web browsers (has anyone ever seen one that inserts an e-mail

into your inbox asking you to authenticate?), this is appropriate.

Instead, it is anticipated that well-behaved portals will block all non-HTTP ports (especially port 443) until the user has successfully authenticated.

Overall, there may also be an interesting discussion to be had about improving network access methods to the point where a user interface can be presented for the same purposes, without resorting to intercepting HTTP traffic. However, since such a mechanism would by necessity require modifying the network stack and operating system of the client, this memo takes a more modest approach.

5. Security Considerations

This memo does not (yet) define any protocol elements, and therefore does not (yet) have any security considerations.

6. IANA Considerations

This document has no actions for IANA.

Appendix A. Acknowledgements

The author takes all responsibility for errors and omissions.

Author's Address

Mark Nottingham

Email: mnot@mnot.net
URI: http://www.mnot.net/