

# meantime: non-consensual http user tracking using caches

From WordNet (r) 1.6 [wn]:

mean

- 2: characterized by malice; "a hateful thing to do"; "in a mean mood"; "told spiteful stories about the fat lady" [syn: hateful, spiteful]
- 3: having or showing a meanspirited lack of honor or morality;
- 4: (slang) excellent; "famous for a mean backhand"

time

- 5: the continuum of experience in which events pass from the future through the present to the past

## executive summary

HTTP cache-control headers such as If-Modified-Since allow servers to track individual users in a manner similar to cookies, but with less constraints. This is a problem for user privacy against which browsers currently provide little protection.

## introduction

Some people would like to be anonymous as they use the web, and other people would like to prevent anonymous access for various reasons. Consider, for example, an internet marketing company that wants to chain together visits to various web sites by a user so as to build a fuller profile of their interests and usage patterns. Conversely, a web user might not wish to leak such information to a site because they are looking at controversial information, desire a good negotiating position, or see privacy as a moral right.

An arms race in techniques for providing and stripping away anonymity has developed over the years. This black paper discusses what is believed to be a new technique for tracking clients and possible responses.

## problem statement

*Alice is browsing the web; Bob runs a number of otherwise-unrelated web servers. Alice makes several requests to Bob's servers over time. Bob would like to tie together as many as possible of the requests made by Alice to learn more about Alice's usage patterns and identity: we call this **identifying the request chain**. Alice would like to access Bob's servers but not give away this information.*

There are many perfectly good reasons why in a particular situation B might want to know A's identity, or at least a unique pseudonym. If B explains the reasons why tracking is required, then A can consent to and allow tracking in various ways. There are several less savory possibilities when A does not consent to the tracking or does not realize that a single chain can be found across apparently unrelated servers controlled by B.

The scenario poses an interesting information-theory and game-theory challenge in anonymity. It is also immediately practical: there is a good deal of development being done in aid of both Alice and Bob.

## existing approaches

## cookies

The standard approach for associating user requests across several responses is the HTTP 'Cookie' state-management extension. The Cookie response header allows a server to ask the client to store arbitrary short opaque data, which should be returned for future requests of that server matching particular criteria. Cookies are commonly used to store per-user form defaults, to manage web application sessions, and to associate requests between executions of the user agent.

The user agent always has the option to just ignore the Set-Cookie response header, but most implementations default to obeying it to preserve functionality. Cookies can optionally specify an expiry time after which they should no longer be used, that they should persist on disk between client session, or that they should only be passed over transmission-level-secure connections.

The privacy implications of cookies have been [extensively discussed](#), and several problems have been found and rectified in the past. One example of privacy compromise through cookies is the use of cookies attached to banner images downloaded from a central banner server: the same cookie is used within images linked from several servers, and so the user can be tracked as they move around.

## other approaches

An obvious means to associate requests is by source IP address. Over the short term this will generally work quite well, as a client is likely to use a single IP address during a browsing session. Even then it is complicated by proxies acting for multiple clients, network address translation, or multiuser machines. Over a longer term, the information is convolved by dynamically-assigned IPs, mobile computers moving between networks, dialup pools and the like. Indeed, cookies were proposed in large part to allow legitimate stateful applications to cope with the impossibility of uniquely identifying users by IP address.

Within a single site, state may be maintained by generating dynamic URLs that include session identification either within the hostname (<http://d9128309812.crackmonkey.org/>) or path (<http://crackmonkey.org/d213213213/faq.html>). However, this does not allow tracking between sites and causes a significant loss of functionality because URLs cannot be shared between users or bookmarked.

Single links can be identified by the HTTP [Referer](#) header. There are some limitations here, however: this only identifies the immediately preceding resource, and the link is lost if the user re-enters a URL by hand or retrieves it from a bookmarks file.

## countermeasures

Users caring to preserve their privacy have taken various countermeasures against these techniques.

To reassure end-users about [cookie privacy issues](#), user agents such as [Netscape Navigator](#), [Microsoft Internet Explorer](#) and [Lynx](#) allow the user some control. The most basic control is to enable or disable cookies altogether; some user agents allow this to be specified for particular domains. There may be more fine-grained controls, such as only accepting cookies from the same server as the top-level page currently viewed and not from servers for subsidiary requests such as images or frames.

The broadest protection is afforded by the use of a proxy local to the browsers machine, such as [Internet Junkbuster](#). This software rewrites the request to strip out identifying browser and cookie information, in addition to attempting to remove advertising banners.

Various proxying solutions are available to prevent identification by IP address, such as [anonymizer.com](#) and [CROWDS](#).

A similar but more powerful attack is possible through the cache-management headers proposed in [draft-mogul-http-delta-02](#).

## caching in http

To make access faster and reduce network usage, browsers generally keep a copy of resources such as pages and images that they download. When a client has a cached copy of a page, it can decide either to use the cached copy as is, or to send a request to the server to check that it is up-to-date.

When the client sends a request for the copy it has in cache, it sends a *conditional request* describing the cached copy and asking the server to only transfer the body of the resource if it is newer than the cached copy.

The most common means of checking this currently in use is the `Last-Modified` date header. The server supplies a date in the metadata of the response, and the client returns the same date when sending a conditional request.

Other techniques, such as checking the length of the resource body, its MD5 hash, and a unique `ETag` cookie have also been used.

## the meantime exploit

The fundament of the meantime exploit is that the server wishes to 'tag' the client with some information that will later be reported back, allowing the server to identify a chain. Cookies are a good approach to this, but their privacy implications are well known and so Bob requires a more surreptitious approach.

The HTTP cache-control headers are perfect for this: the data is provided by the server, stored but not verified by the client, and then provided verbatim back to the server on the next matching request.

Two headers in particular are useful: `Last-Modified` and `ETag`. Both are designed to help the client and server negotiate whether to use a cached copy or fetch the resource again.

The general approach of *meantime* is that rather than using the headers for their intended purpose, Bob's servers will instead send down a unique tag for the client.

`Last-Modified` is constrained to be a date, and therefore is somewhat inflexible. Nevertheless, the server can reasonably choose any second since the Unix epoch, which allows it to tag on the order of one billion distinct clients.

`ETag` allows an arbitrary short string to be stored and passed. It is not so commonly implemented in user agents at the moment, and so not such a good choice.

In both cases the tag will be lost if the client discards the resource from its cache, or if it does not request the exact same resource in the future, or if the request is unconditional. (For example, Netscape sends an unconditional response when the user presses Shift+Reload.) Bob has less control over this than he has with cookies, which can be instructed to persist for an arbitrarily long period.

The date is only sent back for the exact same URL, including any query parameters. By contrast, cookies can be returned for all resources in a site or section of a site. This makes Bob's job a little harder.

Bob therefore should make sure that all pages link to a small common resource: perhaps a one-pixel image. This image is generated by a script that supplies and records a unique timestamp to each client, and records whatever is already present.

## intermediate proxies.



