HTTP Working Group
Internet-Draft
Expires: 10 October 1997

Jeffrey Mogul, DECWRL
Paul J. Leach, Microsoft

25 March 1997

Simple Hit-Metering and Usage-Limiting for HTTP

draft-ietf-http-hit-metering-02.txt

STATUS OF THIS MEMO

Distribution of this document is unlimited. Please send comments to the HTTP working group at <http-wg@cuckoo.hpl.hp.com>. Discussions of the working group are archived at <URL:http://www.ics.uci.edu/pub/ietf/http/>. General discussions about HTTP and the applications which use HTTP should take place on the <www-talk@w3.org> mailing list.

ABSTRACT

This document proposes a simple extension to HTTP, using a new ‘‘Meter’’ header, which permits a limited form of demographic information (colloquially called ‘‘hit-counts’’) to be reported by caches to origin servers, in a more efficient manner than the ‘‘cache-busting’’ techniques currently used. It also permits an origin server to control the number of times a cache uses a cached response, and outlines a technique that origin servers can use to capture referral information without ‘‘cache-busting.’’

TABLE OF CONTENTS

1 Introduction

For a variety of reasons, content providers want to be able to collect information on the frequency with which their content is accessed. This desire leads to some of the "cache-busting" done by existing servers. ("Cache-busting" is the use by servers of techniques intended to prevent caching of responses; it is unknown exactly how common this is.) This kind of cache-busting is done not for the purpose of maintaining transparency or security properties, but simply to collect demographic information. Some cache-busting is also done to provide different advertising images to appear on the same page (i.e., each retrieval of the page sees a different ad).

This proposal supports a model similar to that of publishers of hard-copy publications: such publishers (try to) report to their advertisers how many people read an issue of a publication at least once; they don't (try to) report how many times a reader re-reads an issue. They do this by counting copies published, and then try to estimate, for their publication, on average how many people read a single copy at least once. The key point is that the results aren't exact, but are still useful. Another model is that of coding inquiries in such a way that the advertiser can tell which publication produced the inquiry.

1.1 Goals, non-goals, and limitations

HTTP/1.1 already allows origin servers to prevent caching of responses, and evidence exists [8] that at least some of the time, this is being done for the sole purpose of collecting counts of the number of accesses of specific pages. Some of this evidence is inferred from the study of proxy traces; some is based on explicit statements of the intention of the operators of Web servers. Information collected this way might or might not be of actual use to the people who collect it; the fact is that they want to collect it, or already do so.

The goal of this proposal is to provide an optional performance optimization for this use of HTTP/1.1.

This specification is:

- Optional: no server or proxy is required to implement it.

- Proxy-centered: there is no involvement on the part of end-client implementations.

- Solely a performance optimization: it provides no information or functionality that is not already available in HTTP/1.1. The intent is to improve performance overall, and reduce latency for almost all interactions; latency might be increased for a small fraction of HTTP interactions.

- Best-efforts: it does not guarantee the accuracy of the reported information, although it does provide accurate results in the absence of persistent network failures or host crashes.

- Neutral with respect to privacy: it reveals to servers no information about clients that is not already available through the existing features of HTTP/1.1.

The goals of this specification do not include:

- Solving the entire problem of efficiently obtaining extensive information about requests made via proxies.

- Improving the protection of user privacy (although our proposal may reduce the transfer of user-specific information to servers, it does not prevent it).

- Preventing or encouraging the use of log-exchange mechanisms.

- Avoiding all forms of "cache-busting", or even all cache-busting done for gathering counts.

This design has certain potential limitations:

- If it is not deployed widely in both proxies and servers, it will provide little benefit.

- It may, by partially solving the hit-counting problem, reduce the pressure to adopt more complete solutions, if any become available.

- Even if widely deployed, it might not be widely used, and so might not significantly improve performance.

These potential limitations might not be problems in actual practice.

1.2 Brief summary of the design

This section is included for people not wishing to read the entire document; it is not a specification for the proposed design, and over-simplifies many aspects of the design.

The goal of this design is to eliminate the need for origin servers to use "cache-busting" techniques, when this is done just for the purpose of counting the number of users of a resource. (Cache-busting includes techniques such as setting immediate Expiration dates, or sending "Cache-control: private" in each response.)

The design adds a new "Meter" header to HTTP; the header is always protected by the "Connection" header, and so is always hop-by-hop. This mechanism allows the construction of a "metering subtree", which is a connected subtree of proxies, rooted at an origin server. Only those proxies that explicitly volunteer to join in the metering subtree for a resource participate in hit-metering, but those proxies that do volunteer are required to make their best effort to provide accurate counts. When a hit-metered response is forwarded outside of the metering subtree, the forwarding proxy adds "Cache-control: proxy-maxage=0", so that other proxies (outside the metering subtree) are forced to forward all requests to a server in the metering subtree.

NOTE: the HTTP/1.1 specification does not currently define a "proxy-maxage" Cache-control directive. A separate proposal has been made, on various grounds, to add such a directive to the next revision of the HTTP/1.1 specification [6].

The Meter header carries zero or more directives, similar to the way that the Cache-control header carries directives. Proxies may use certain Meter directives to volunteer to do hit-metering for a resource. If a proxy does volunteer, the server may use certain directives to require that a response be hit-metered. Finally, proxies use a "count" Meter directive to report the accumulated hit counts.

The Meter mechanism can also be used by a server to limit the number of uses that a cache may make of a cached response, before revalidating it.

The full specification includes complete rules for counting "uses" of a response (e.g., non-conditional GETs) and "reuses" (conditional GETs). These rules ensure that the results are entirely consistent in all cases, except when systems or networks fail.

1.3 Terminology

This document uses terms defined and explained in the HTTP/1.1 specification [3], including ‘‘origin server,’’ ‘‘resource,’’ ‘‘hop-by-hop,’’ ‘‘unconditional GET,’’ and ‘‘conditional GET.’’ The reader is expected to be familiar with the HTTP/1.1 specification and its terminology.

2 Overview

The design described in this document introduces several new features to HTTP:

- Hit-metering: allows an origin server to obtain reasonably accurate counts of the number of clients using a resource instance via a proxy cache, or a hierarchy of proxy caches.

- Usage-limiting: allows an origin server to control the number of times a cached response may be used by a proxy cache, or a hierarchy of proxy caches, before revalidation with the origin server.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase®
Smarter legal research.