Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 16, 2013

M. Nottingham
February 12, 2013

HTTP Origin and Hop Hints
draft-nottingham-http-browser-hints-05

Abstract

Over time, HTTP clients -- especially Web browsers -- have adapted how they use the protocol based upon common server configurations and behaviours. While this is necessary in the common case, it can be detrimental for performance and interoperability.

This document establishes a mechanism whereby both origin servers and intermediaries can make hints available to clients about their preferences and capabilities, without imposing undue overhead on their interactions or requiring support for them.

This is intended to allow clients to safely optimise connections to servers.

Note to Readers

Feedback for this draft should take place on the apps-discuss@ietf.org mailing list <https://www.ietf.org/mailman/listinfo/apps-discuss>.

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 August 16, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

HTTP [HTTP-p1] clients -- especially browsers -- typically use hardcoded values or heuristics to determine how they use TCP, based on common-case server behaviours and limitations.

For example, they often send voluminous request headers (e.g., in User-Agent and Allow) because they fear that changing those headers' values will break some sites that depend upon specific values.

These conservative behaviours are good for interoperability, but potentially bad for performance in certain circumstances.

This document specifies a mechanism whereby a HTTP server can advertise hints for browsers and other clients, so that communication with them can be optimised.

It does so by defining two headers; "OH" (Origin Hints) for end-to-end hints from the Origin Server, and "HH" (Hop Hints) for hop-by-hop hints from the upstream server (origin or proxy). A selection of hints are also defined in this document, and a registry is defined to allow future such hints.

2. Notational Conventions

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 [RFC2119].

This document uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234], and explicitly includes the following rules from it: ALPHA, DIGIT. Additionally, it uses the list rule extension defined in [HTTP-p1], Appendix B.

3. Origin Hints

Origin Hints are applicable to all future requests to the origin [RFC6454] associated with the response, until they are overridden. They are carried in the "OH" header field's value.

```
OH = #hint
```

A hint is considered overridden when an OH header field is seen from the same origin that does not contain the hint, or contains a different value for the hint.

Practically speaking, this means that clients MAY cache origin hint values indefinitely, updating them when new OH header fields are seen from an origin.

For example, an origin may send an OH header on every response, in which case the latest one seen would be the value used, or it could send one sporadically (e.g., upon connection), in which case the origin hints in effect would be the last one seen by the client.

Practically speaking, origins that wish to have their hints available to clients as soon as possible will send them on every response; those that wish to limit the sizes of responses might use some other strategy, knowing that clients will eventually cache their hints.

For example:

```
HTTP/1.1 200 OK
Content-Type: text/html
OH: a,b=6,c
Content-Length: 1234
Cache-Control: max-age=60

...
```

Here, the origin hints "a", "b", and "c" are indicated; the "b" hint has a value of "6". If a subsequent message from the same origin to the same client were:

```
HTTP/1.1 303 See Other
Content-Type: text/html
OH: a,b=2
Content-Length: 1234
Cache-Control: max-age=60

...
```

The client would now consider "a" to still be true, whereas the value of "b" would be 2, and "c" would be false.

Origin hints SHOULD NOT be generated by proxy servers.

4. Hop Hints

Hop Hints are applicable to all future requests on the TCP connection they occur upon, until they are overridden. The are carried in the "HH" header field's value.

```
HH = #hint
```

A hint is considered overridden when an HH header field is seen on the same connection that does not contain the hint, or contains a different value for the hint.

Typically, a server (whether origin or proxy) will send the HH header field on the first response, omitting it from subsequent responses unless it wishes to change a value.

When it occurs in a message, the HH header field MUST be listed in the Connection header's field-value.

For example:

```
HTTP/1.1 404 Not Found
Content-Type: text/plain
HH: x,ya
Content-Length: 5678
Connection: HH
Cache-Control: max-age=60

...
```

Here, the hop hints "x" and "ya" are indicated.

5. Hint Syntax

Both origin hints and hop hints share a common syntax, consisting of a string of alphanumeric characters. This form is designed to be compact without sacrificing readability.

Every hint has a case-sensitive hint identifier.

```
hint = 1*ALPHA [ "=" 1*DIGIT ]
```

Hints are allowed to have a numeric argument. However, wherever possible, they are encouraged to be defined as flags (i.e., as a hint identifier only), so that the hints don't consume too much space in responses.

Hints can be defined as one of two types:

- Boolean - indicated by the presence of the hint identifier. If the hint identifier is absent in the last message containing the relevant hint header field, it is considered false.

- Numeric - value indicated by the digits after "=", up to the first non-digit character.

Note that HTTP/1.1 allows headers with comma-separated values to be conveyed using multiple instances of the same header; as a result, the hints of a given type (origin or hop) are collected from all instances of that header on the message in question before being considered complete.

6. Pre-defined Hints

6.1. Small Request Headers

- Hint Name: s
- Hint Type: origin
- Description: When true, this hint indicates that clients can omit the Accept and Accept-Charset request headers when communicating with the origin, and that they can use a shortened version of the User-Agent header.
- Value Type: boolean
- Contact: mnot@mnot.net

This hint can help reduce request sizes.

6.2. Relative Referers

- Hint Name: r
- Hint Type: origin
- Description: When true, this hint indicates that the origin prefers a relative URI in the Referer request header.
- Value Type: boolean
- Contact: mnot@mnot.net

This hint can help reduce request sizes.

6.3. Omitting Cookies

- Hint Name: c
- Hint Type: origin
- Description: When true, this hint indicates that all cookies [RFC6265] can be omitted in requests to the origin.
- Value Type: boolean
- Contact: mnot@mnot.net

This hint can help reduce request sizes.

6.4. Sharing Connections

- Hint Name: sc
- Hint Type: hop
- Description: When true, this hint indicates that the server allows clients to reuse persistent connections keyed by IP address, rather than by hostname. Note that all origins that are sharing the connection MUST declare this hint for it to be used, and if a transport-layer certificate is in use (e.g., for TLS [RFC5246]), it MUST be valid for all origins.
- Value Type: boolean
- Contact: mnot@mnot.net
- Specification: [this document]
- Notes: Although it is a Hop Hint, this MUST NOT be set by servers other than origins.

In other words, if both www.example.com and foo.example.org resolve to the address 192.0.2.5, and indicate this hint, then clients can send a request to www.example.com and then a request to foo.example.org on the same TCP connection to that address.

6.5. Pipeline Depth

- Hint Name: p
- Hint Type: hop
- Description: When present, this hint indicates the maximum number of pipelined requests per connection that the server would like clients to use.
- Value Type: numeric
- Contact: mnot@mnot.net

7. Security Considerations

By their nature, hints are both optional and advisory; clients ought to exercise judgement when applying them, as an attacker might use a naive implementation to trick the client into generating abnormal traffic. For example, the "p" hint should not be the only input into determining how deep to pipeline requests.

Hints are also only as secure as the channel they are transmitted upon; if HTTP is used in the clear, then hints might be observed (which typically is not a great risk), and modified (which could be, for a naive client implementation).

The Hop Hints mechanism uses the "Connection" header to scope the hint to a single HTTP hop. A few old implementations have been observed to not properly strip headers indicated by "Connection".

8. IANA Considerations

8.1. The OH HTTP Response Header Field

This document defines the "OH" HTTP header field, and registers it in the Permanent Message Headers registry.

- Header field name: OH
- Applicable protocol: HTTP
- Status: Informational
- Author/Change controller: Mark Nottingham, mnot@mnot.net
- Specification document(s): [this document]
- Related information: for Origin Hints

8.2. The HH HTTP Response Header Field

This document defines the "HH" HTTP header field, and registers it in the Permanent Message Headers registry.

- Header field name: HH
- Applicable protocol: HTTP
- Status: Informational
- Author/Change controller: Mark Nottingham, mnot@mnot.net
- Specification document(s): [this document]
- Related information: for Hop Hints

8.3. The HTTP Hints Registry

This document establishes the HTTP Hints Registry.

New hints are registered using Expert Review (see [RFC5226]), by sending e-mail to <mailto:iana@iana.org> (or using other mechanisms, as established by IANA).

New hints are expected to be implemented in at least one client in common use. The Expert MAY use their judgement in determining what "common" is, and when something is considered to be implemented.

New hints MUST be optional; they cannot place requirements upon implementations.

Specifically, new hints MUST NOT make communication non-conformant with HTTP itself; i.e., this is not a mechanism for changing the HTTP protocol in incompatible ways. For example, if a hint indicates that browsers can compress request headers using GZIP, intermediaries that are interposed are likely to fail.

Registration requests MUST use the following template:

- Hint Name: [name of hint]
- Hint Type: ["origin" or "hop"]
- Description: [description of hint]
- Value Type: ["boolean" or "numeric"]
- Contact: [e-mail address(es)]
- Specification: [optional; reference or URI to more info]
- Notes: [optional]

The initial contents of the registry are defined in Section 6.

9. References

9.1. Normative References

[HTTP-p1] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", draft-ietf-httpbis-p1-messaging-21; (work in progress), Feb 2013.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, December 2011.

9.2. Informative References

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011.

Appendix A. Acknowledgements

Thanks to Mike Belshe, Artur Bergman, William Chan, Jason Duell, Poul-Henning Kamp, Anirban Kundu, Patrick McManus, Ryan Sleevi, Steve Souders, and Martin Thompson for their suggestions and feedback.

The author takes all responsibility for errors and omissions.

Appendix B. Open Issue: Hint Syntax

This revision defines the syntax of hints as a comma-delimited list. This is convenient (especially for delimiting hints with values), but if many hints need to be conveyed, it'll be inefficient.

An alternate syntax could remove the commas, but we'd likely be constrained in the number of hints we'd be able to define.

Yet another approach would be to define a bitfield, and an ASCII representation of that field. However, this would be cumbersome.

Feedback appreciated.

Appendix C. Open Issue: Hint Value Types

Are the defined hint value types sufficient? Feedback appreciated.

Author's Address

Mark Nottingham

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