

HTTP
Internet-Draft
Intended status: Standards Track
Expires: February 6, 2021

M. Nottingham
Fastly
August 5, 2020

The Cache-Status HTTP Response Header Field
draft-ietf-httpbis-cache-header-04

Abstract

To aid debugging, HTTP caches often append header fields to a response explaining how they handled the request. This specification codifies that practice and updates it to align with HTTP's current caching model.

Note to Readers

RFC EDITOR: please remove this section before publication

Discussion of this draft takes place on the HTTP working group mailing list (ietf-http-wg@w3.org), which is archived at <https://lists.w3.org/Archives/Public/ietf-http-wg/> [1].

Working Group information can be found at <https://httpwg.org/> [2]; source code and issues list for this draft can be found at <https://github.com/httpwg/http-extensions/labels/cache-header> [3].

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 <https://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, 2021.

Internet-Draft

Cache-Status Header

August 2020

Copyright Notice

Copyright (c) 2020 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 (<https://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	2
1.1.	Notational Conventions	3
2.	The Cache-Status HTTP Response Header Field	3
2.1.	The hit parameter	4
2.2.	The fwd parameter	4
2.3.	The fwd-status parameter	5
2.4.	The ttl parameter	5
2.5.	The stored parameter	5
2.6.	The collapsed parameter	5
2.7.	The key parameter	5
2.8.	The detail parameter	5
3.	Examples	6
4.	Defining New Proxy-Status Parameters	7
5.	IANA Considerations	7
6.	Security Considerations	7
7.	References	8
7.1.	Normative References	8
7.2.	Informative References	9
7.3.	URIs	9
	Author's Address	9

[1.](#) Introduction

To aid debugging, HTTP caches often append header fields to a response explaining how they handled the request. Unfortunately, the semantics of these headers are often unclear, and both the semantics

and syntax used vary greatly between implementations.

This specification defines a single, new HTTP response header field, "Cache-Status" for this purpose.

[1.1.](#) Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

This document uses ABNF as defined in [[RFC5234](#)], along with the "%s" extension for case sensitivity defined in [[RFC7405](#)].

[2.](#) The Cache-Status HTTP Response Header Field

The Cache-Status HTTP response header indicates caches' handling of the request corresponding to the response it occurs within.

Its value is a List [[I-D.ietf-httpbis-header-structure](#)]:

```
Cache-Status = sf-list
```

Each member of the list represents a cache that has handled the request. The first member of the list represents the cache closest to the origin server, and the last member of the list represents the cache closest to the client (possibly including the user agent's cache itself, if it chooses to append a value).

Caches determine when it is appropriate to add the Cache-Status header field to a response. Some might decide to add it to all responses, whereas others might only do so when specifically configured to, or when the request contains a header that activates a debugging mode.

When adding a value to the Cache-Status header field, caches SHOULD preserve the existing contents of the header field, to allow debugging of the entire chain of caches handling the request.

Each list member identifies the cache that inserted that value, and MUST be a String or Token. Depending on the deployment, this might be a product or service name (e.g., ExampleCache or "Example CDN"), a hostname ("cache-3.example.com"), and IP address, or a generated string.

Each member of the list can also have parameters that describe that cache's handling of the request. While all of these parameters are OPTIONAL, caches are encouraged to provide as much information as possible.

This specification defines these parameters:

hit	= sf-boolean
fwd	= sf-token
fwd-status	= sf-integer
tll	= sf-integer
stored	= sf-boolean
collapsed	= sf-boolean
key	= sf-string
detail	= sf-token / sf-string

[2.1.](#) The hit parameter

"hit", when true, indicates that the request was satisfied by the cache; i.e., it did not go forward and the response was obtained from the cache. A response that originally was produced by the origin but was modified by the cache (for example, a 304 or 206 status code) is still considered a hit.

"hit" and "fwd" are exclusive; only one of them should appear on each list member.

[2.2.](#) The fwd parameter

"fwd" indicates that the request went forward towards the origin, and why.

The following values are defined to explain why the request went forward:

- o uri-miss - The cache did not contain any responses that matched

the request URI

- o vary-miss - The cache contained a response that matched the request URI, but could not select a response based upon this request's headers and stored Vary headers.
- o miss - The cache did not contain any responses that could be used to satisfy this request (to be used when an implementation cannot distinguish between uri-miss and vary-miss)
- o stale - The cache was able to select a response for the request, but it was stale
- o request - The cache was able to select a fresh response for the request, but client request headers (e.g., Cache-Control request directives) did not allow its use
- o bypass - The cache was configured to not handle this request

[2.3.](#) The fwd-status parameter

"fwd-status" indicates what status code the next hop server returned in response to the request. Only meaningful when "fwd" is present; if "fwd-status" is not present but "fwd" is, it defaults to the status code sent in the response.

This parameter is useful to distinguish cases when the next hop server sends a 304 Not Modified response to a conditional request, or a 206 Partial Response due to a range request.

[2.4.](#) The ttl parameter

"ttl" indicates the response's remaining freshness lifetime as calculated by the cache, as an integer number of seconds, measured when the response is sent by the cache. This includes freshness assigned by the cache; e.g., through heuristics, local configuration, or other factors. May be negative, to indicate staleness.

[2.5.](#) The stored parameter

"stored" indicates whether the cache stored the response; a true

value indicates that it did. Only meaningful when fwd is present.

[2.6.](#) The collapsed parameter

"collapsed" indicates whether this request was collapsed together with one or more other forward requests; if true, the response was successfully reused; if not, a new request had to be made. If not present, the request was not collapsed with others. Only meaningful when fwd is present.

[2.7.](#) The key parameter

"key" conveys a representation of the cache key used for the response. Note that this may be implementation-specific.

[2.8.](#) The detail parameter

"detail" allows implementations to convey additional information not captured in other parameters; for example, implementation-specific states, or other caching-related metrics.

For example:

```
Cache-Status: ExampleCache; hit; detail=MEMORY
```

The semantics of a detail parameter are always specific to the cache that sent it; even if a member of details from another cache shares the same name, it might not mean the same thing.

This parameter is intentionally limited. If an implementation needs to convey additional information, they are encouraged to register extension parameters (see [Section 4](#)) or define another header field.

[3.](#) Examples

The most minimal cache hit:

```
Cache-Status: ExampleCache; hit
```

... but a polite cache will give some more information, e.g.:

Cache-Status: ExampleCache; hit; ttl=376

A stale hit just has negative freshness:

Cache-Status: ExampleCache; hit; ttl=-412

Whereas a complete miss is:

Cache-Status: ExampleCache; fwd=uri-miss

A miss that successfully validated on the back-end server:

Cache-Status: ExampleCache; fwd=stale; fwd-status=304

A miss that was collapsed with another request:

Cache-Status: ExampleCache; fwd=uri-miss; collapsed

A miss that the cache attempted to collapse, but couldn't:

Cache-Status: ExampleCache; fwd=uri-miss; collapsed=?0

Going through two layers of caching, both of which were hits, and the second collapsed with other requests:

Cache-Status: OriginCache; hit; ttl=1100; collapsed,
"CDN Company Here"; hit; ttl=545

[4.](#) Defining New Proxy-Status Parameters

New Cache-Status Parameters can be defined by registering them in the HTTP Cache-Status Parameters registry.

Registration requests are reviewed and approved by a Designated Expert, as per [\[RFC8126\], Section 4.5](#). A specification document is appreciated, but not required.

The Expert(s) should consider the following factors when evaluating requests:

- o Community feedback
- o If the value is sufficiently well-defined
- o Generic parameters are preferred over vendor-specific, application-specific or deployment-specific values. If a generic value cannot be agreed upon in the community, the parameter's name should be correspondingly specific (e.g., with a prefix that identifies the vendor, application or deployment).

Registration requests should use the following template:

- o Name: [a name for the Cache-Status Parameter that matches key]
- o Description: [a description of the parameter semantics and value]
- o Reference: [to a specification defining this parameter]

See the registry at <https://iana.org/assignments/http-cache-status> [4] for details on where to send registration requests.

5. IANA Considerations

Upon publication, please create the HTTP Cache-Status Parameters registry at <https://iana.org/assignments/http-cache-statuses> [5] and populate it with the types defined in [Section 2](#); see [Section 4](#) for its associated procedures.

6. Security Considerations

Information about a cache's content can be used to infer the activity of those using it. Generally, access to sensitive information in a cache is limited to those who are authorised to access that information (using a variety of techniques), so this does not represent an attack vector in the general sense.

However, if the Cache-Status header field is exposed to parties who

are not authorised to obtain the response it occurs within, it could expose information about that data.

For example, if an attacker were able to obtain the Cache-Status header field from a response containing sensitive information and access were limited to one person (or limited set of people), they could determine whether that information had been accessed before. This is similar to the information exposed by various timing attacks, but is arguably more reliable, since the cache is directly reporting its state.

Mitigations include use of encryption (e.g., TLS [[RFC8446](#)])) to protect the response, and careful controls over access to response header fields (as are present in the Web platform). When in doubt, the Cache-Status header field can be omitted.

[7.](#) References

[7.1.](#) Normative References

- [I-D.ietf-httpbis-header-structure] Nottingham, M. and P. Kamp, "Structured Field Values for HTTP", [draft-ietf-httpbis-header-structure-19](#) (work in progress), June 2020.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, [RFC 5234](#), DOI 10.17487/RFC5234, January 2008, <<https://www.rfc-editor.org/info/rfc5234>>.
- [RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", [RFC 7405](#), DOI 10.17487/RFC7405, December 2014, <<https://www.rfc-editor.org/info/rfc7405>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

7.2. Informative References

- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", [RFC 8446](#), DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

7.3. URIs

- [1] <https://lists.w3.org/Archives/Public/ietf-http-wg/>
- [2] <https://httpwg.org/>
- [3] <https://github.com/httpwg/http-extensions/labels/cache-header>
- [4] <https://iana.org/assignments/http-cache-status>
- [5] <https://iana.org/assignments/http-cache-statuses>

Author's Address

Mark Nottingham
Fastly
made in
Pahran, VIC
Australia

Email: mnot@mnot.net
URI: <https://www.mnot.net/>

Nottingham

Expires February 6, 2021

[Page 9]