# RFC 9211 : The Cache-Status HTTP Response Header Field

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 9 juin 2022

Date de publication du RFC : Juin 2022

https://www.bortzmeyer.org/9211.html

---

Ce RFC normalise enfin un en-tête HTTP pour permettre aux relais HTTP qui mémorisent les réponses reçues, les "*caches*", d'indiquer au client comment ils ont traité une requête et si, par exemple, la réponse vient de la mémoire du relais ou bien s'il a fallu aller la récupérer sur le serveur HTTP d'origine.

Il existait déjà des en-têtes de débogage analogues mais non-standards, variant aussi bien dans leur syntaxe que dans leur sémantique. Le nouvel en-tête standard, `Cache-Status` : va, espérons-le, mettre de l'ordre à ce sujet. Sa syntaxe suit les principes des en-têtes structurés du RFC 8941[1]. Sa valeur est donc une liste de relais qui ont traité la requête, séparés par des virgules, dans l'ordre des traitements. Le premier relais indiqué dans un `Cache-Status` : est donc le plus proche du serveur d'origine et le dernier relais étant le plus proche de l'utilisateur. Voici un exemple avec un seul relais, nommé `cache.example.net` :

```
Cache-Status: cache.example.net; hit
```

et un exemple avec deux relais, `OriginCache` puis `CDN Company Here` :

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

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc8941.txt`

La description complète de la syntaxe figure dans la section 2 de notre RFC. D'abord, il faut noter qu'un relais n'est pas obligé d'ajouter cet en-tête. Il le fait selon des critères qui lui sont propres. Le RFC déconseille fortement d'ajouter cet en-tête si la réponse a été générée localement par le relais (par exemple un 400 qui indique que le relais a détecté une requête invalide). En tout cas, le relais qui s'ajoute à la liste ne doit pas modifier les éventuels relais précédents. Chaque relais indique son nom, par exemple par un nom de domaine mais ce n'est pas obligatoire. Ce nom peut ensuite être suivi de paramètres, séparés par des point-virgules.

Dans le premier exemple ci-dessus, `hit` était un paramètre du relais `cache.example.net`. Sa présence indique qu'une réponse était déjà mémorisée par le relais et qu'elle a été utilisée ("*hit*" : succès, on a trouvé), le serveur d'origine n'ayant pas été contacté. S'il y avait une réponse stockée, mais qu'elle était rassise (RFC 9111, section 4.2) et qu'il a fallu la revalider auprès du serveur d'origine, `hit` ne doit pas être utilisé.

Le deuxième paramètre possible est `fwd` (pour "*Forward*") et il indique qu'il a fallu contacter le serveur HTTP d'origine. Plusieurs raisons sont possibles pour cela, entre autres :

— `bypass`, quand le relais a été configuré pour que cette requête soit systématiquement relayée,
— `method` parce que la méthode HTTP utilisée doit être relayée (c'est typiquement le cas d'un `PUT`, cf. RFC 9110, section 9.3.4),
— `miss`, un manque, la ressource n'était pas dans la mémoire du relais (notez qu'il existe aussi deux paramètres plus spécifiques, `uri-miss` et `vary-miss`),
— `stale`, car la ressource était bien dans la mémoire du relais mais, trop ancienne et rassise, a dû être revalidée (RFC 9111, section 4.3) auprès du serveur d'origine.

Ainsi, cet en-tête :

```
Cache-Status: ExampleCache; fwd=miss; stored
```

indique que l'information n'était pas dans la mémoire de `ExampleCache` et qu'il a donc dû faire suivre au serveur d'origine (puis qu'il a stocké cette information dans sa mémoire, le paramètre `stored`).

Autre paramètre utile, `ttl`, qui indique pendant combien de temps l'information peut être gardée (cf. RFC 9111, section 4.2.1), selon les calculs du relais. Quant à `detail`, il permet de donner des informations supplémentaires, de manière non normalisée (la valeur de ce paramètre ne peut être interprétée que si on connait le programme qui l'a produite).

Les paramètres sont enregistrés à l'IANA `<https://www.iana.org/assignments/http-cache-status/http-cache-status.xml#http-cache-status>`, dans un nouveau registre, et de nouveaux paramètres peuvent y être ajoutés, suivant la politique « Examen par un expert » du RFC 8126. La recommandation est de n'enregistrer que des paramètres d'usage général, non spécifiques à un programme particulier, sauf à nommer le paramètre d'une manière qui identifie clairement sa spécificité.

Questions mises en œuvre, Squid sait renvoyer cet en-tête (cf. le code qui commence à `SBuf cacheStatus(uniqu` dans le fichier `src/client_side_reply.cc`). Ce code n'est pas encore présent dans la version 5.2 de Squid, qui n'utilise encore que d'anciens en-têtes non-standards, il faut attendre de nouvelles versions :

```
< X-Cache: MISS from myserver
< X-Cache-Lookup: NONE from myserver:3128
< Via: 1.1 myserver (squid/5.2)
```

Je n'ai pas regardé les autres logiciels.