HTTP Working Group
Internet-Draft
Expires: January 29, 1998

Koen Holtman, TUE

July 29, 1997

The Safe Response Header Field

draft-holtman-http-safe-02.txt

STATUS OF THIS MEMO

ABSTRACT

This document proposes a HTTP response header field called Safe, which can be used to indicate that repeating the corresponding POST request is safe. Such an indication will allow user agents to present services which use safe POSTs in a more user-friendly way. Improving the user-friendliness of safe POSTs is considered important, because web internationalization will depend for a large part on the use of safe POSTs.

1 Introduction

This document proposes a HTTP response header field called Safe, which can be used to indicate that repeating the corresponding POST request is safe. Such an indication will allow user agents to present services which use safe POSTs in a more user-friendly way. Improving the user-friendliness of safe POSTs is considered important, because web internationalization will depend for a large part on the use of safe POSTs.

2 Background

According to Section 9.1.1 (Safe Methods) of the HTTP/1.1 draft specification [1], POST requests are assumed to be `unsafe' by default. `Unsafe' means `causes side effects for which the user will be held accountable'.

If the POST request is unsafe, explicit user confirmation is necessary before the request is repeated. User agents will repeat POST requests when the user presses the RELOAD button while a POST result is displayed, or when the history function is used to redisplay a POST result which is no longer in the history buffer.

The necessary confirmation dialog often takes the form of a `repost form data?' dialog box. The dialog is confusing to many users, and slows down navigation in any case.

In theory, if the repeated POST request is safe, the user-unfriendly confirmation dialog can be omitted. However, up till now, HTTP has no mechanism by which agents can tell if POST requests are safe. This proposal adds such a mechanism.

Many content authors have managed to avoid the confirmation dialog problem by using GETs for form submission instead of safe POSTs. However, this escape is not possible for forms

a) which are (sometimes) used to submit large amounts of data
b) which are (sometimes) used to submit data in a charset other than ISO-8859-1.

Case b) will be the increasingly common; web internationalization [2] makes it necessary to use the POST method for form submission.

> Note: Actually, according to the authors of [2], web internationalization makes it necessary to use requests which request bodies. This rules out the use of the only methods which are safe under HTTP/1.1: GET and HEAD. A new GET-WITH-BODY method could be defined, but it would be unsafe by default under HTTP/1.1, so GET-WITH-HEAD would also need something like the Safe header.

It is therefore considered important to eliminate the unnecessary confirmation dialogs for safe POSTs as soon as possible. They are a counter-incentive to the upgrading of GET based forms services (like search engines) to internationalized POST based forms services.

3 The Safe response header

This header is proposed as an addition to the HTTP/1.x suite.

The Safe response header field indicates whether repeating the request in the corresponding request message is safe in the sense of Section 9.1.1 (Safe Methods) of the HTTP/1.1 specification [1].

```
Safe        = "Safe" ":" safe-nature
safe-nature = "yes" | "no"
```

An example of the field is:

```
Safe: yes
```

If the Safe header field is absent in the response, the corresponding request must be considered unsafe, unless it is a GET or HEAD request. GET and HEAD requests are safe by definition, user agents must ignore a `Safe: no' header field in GET and HEAD responses.

> Note: User agents can use the received information about safety when repeating an earlier request. If the repeating the request is known to be safe, the request can be repeated automatically without asking for user confirmation.

4 Smooth upgrade path

That the Safe header provides a smooth upgrade path; if a service switches from GETs to safe POSTs, existing browsers will still be able to access the service. Its use requires little re-coding effort for service authors and user agent authors, and is optional in any case.

5 About syntax

This document mainly intends to recommend a _mechanism_; the syntax of the corresponding header is considered less important. One alternative to the addition of a Safe header would be the addition of a `safe' response directive to the Cache-Control header.

6 Alternatives to the Safe header

A number of alternative ways to solve the confirmation dialog problem have been proposed. These alternative solutions would make the introduction of the Safe header unnecessary.

6.1 GET-WITH-BODY

If a new HTTP/1.x GET-WITH-BODY is defined, one would not need the Safe header anymore, one could simply define GET-WITH-BODY as always safe. However, it has been shown that some ugly extensions to the HTML form syntax would be needed to provide the same level of downwards compatibility with existing browsers that Safe can provide, for example

```
<form action="..." method=post preferred_method=get-with-body>
 ....
</form>.
```

One could say that a GET-WITH-BODY manages to keep HTTP clean at the cost of adding extensions to HTML. The author of this draft prefers to keep HTML clean by adding the Safe extension to HTTP.

Note that the Safe header does not block the introduction of a GET-WITH-BODY in the long run.

6.2 Link

The need for a confirmation dialog can also be eliminated by offering the user agent an alternative to resubmitting the POST request altogether. A POST result could for example have a Link header which would contain an URL from which the result can be retrieved again with a GET request. However, this Link URL cannot be very long: it it were too long, request would fail due to user agent, proxy, or origin server limitations. This length restriction would make the Link solution hard to use in the general case.

6.3 Conclusion

The Safe header seems to be the best solution in the current framework. Though the existence of alternative solutions like the ones above has been asserted, the author of this draft has not seen any work on getting alternative solutions standardized.

The Safe header would eliminate a counter-incentive to web internationalization, and the author feels that these counter-incentives need to be eliminated as soon as possible. It is therefore proposed to introduce the Safe header into HTTP/1.x without undue delay.

The counter-incentive can be thought of as an interoperability problem between HTTP/1.1 [1] and [2]. These documents currently both have the Proposed Standard status. It may be possible to add the Safe header to [1] before it goes to Draft Standard.

7 Security considerations

This proposal adds no new HTTP security considerations.

8 References

[1] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext Transfer Protocol -- HTTP/1.1. RFC 2068, HTTP Working Group, January, 1997.

[2] Yergeau et al, Internationalization of the Hypertext Markup Language, Internet-Draft draft-ietf-html-i18n-05.txt, Network Working Group, August 7, 1996

9 Author's address

Koen Holtman
Technische Universiteit Eindhoven
Postbus 513
Kamer HG 6.57
5600 MB Eindhoven (The Netherlands)
Email: koen@win.tue.nl

Expires: January 29, 1998