Hypertext Transfer Protocol (HTTP) Client-Initiated Content-Encoding
draft-ietf-httpbis-cice-01

Abstract

   In HTTP, content codings allow for payload encodings such as for
   compression or integrity checks. In particular, the "gzip" content
   coding is widely used for payload data sent in response messages.

   Content codings can be used in request messages as well, however
   discoverability is not on par with response messages. This document
   extends the HTTP "Accept-Encoding" header field for use in responses,
   to indicate that content codings are supported in requests.

Editorial Note (To be removed by RFC Editor before publication)

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

   Working Group information can be found at
   <https://tools.ietf.org/wg/httpbis/> and <http://httpwg.github.io/>;
   source code and issues list for this draft can be found at
   <https://github.com/httpwg/http-extensions>.

   The changes in this draft are summarized in Appendix A.4.

   This Internet-Draft will expire on December 1, 2015.

Copyright Notice

   Copyright (c) 2015 IETF Trust and the persons identified as the
   document authors. All rights reserved.

Table of Contents

1. Introduction

   In HTTP, content codings allow for payload encodings such as for
   compression or integrity checks ([RFC7231], Section 3.1.2).   In
   particular, the "gzip" content coding is widely used for payload data
   sent in response messages.

   Content codings can be used in request messages as well, however
   discoverability is not on par with response messages. This document
   extends the HTTP "Accept-Encoding" header field ([RFC7231], Section
   5.3.4) for use in responses, to indicate that content codings are
   supported in requests.

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 reuses terminology defined in the base HTTP
   specifications, namely Section 2 of [RFC7230] and Section 3.1.2 of
   [RFC7231].

3. Using the 'Accept-Encoding' Header Field in Responses

   Section 5.3.4 of [RFC7231] defines "Accept-Encoding" as a request
   header field only.

   This specification expands that definition to allow "Accept-Encoding"
   as a response header field as well. When present in a response, it
   indicates what content codings the resource was willing to accept in
   the associated request. A field value that only contains "identity"
   implies that no content codings were supported.

   Note that this information is specific to the associated request; the
   set of supported encodings might be different for other resources on
   the same server, and could change over time or depend on other
   aspects of the request (such as the request method).

   Section 6.5.13 of [RFC7231] defines status code 415 (Unsupported
   Media Type) to apply to both media type and content coding related
   problems.

   Servers that fail a request due to an unsupported content coding
   SHOULD respond with a 415 status and SHOULD include an "Accept-
   Encoding" header field in that response, allowing clients to
   distinguish between content coding related issues and media type
   related issues. In order to avoid confusion with media type related

problems, servers that fail a request with a 415 status for reasons
unrelated to content codings SHOULD NOT include the "Accept-Encoding"
header field.

It is expected that the most common use of "Accept-Encoding" in
responses will have the 415 (Unsupported Media Type) status code, in
response to optimistic use of a content coding by clients. However,
the header field can also be used to indicate to clients that content
codings are supported, to optimize future interactions. For example,
a resource might include it in a 2xx response when the request
payload was big enough to justify use of a compression coding, but
the client failed do so.

4. Example

A client submits a POST request using the "compress" content coding
([RFC7231], Section 3.1.2.1):

```
  POST /edit/ HTTP/1.1
  Host: example.org
  Content-Type: application/atom+xml;type=entry
  Content-Encoding: compress

  ...compressed payload...
```

The server rejects request because it only allows the "gzip" content
coding:

```
  HTTP/1.1 415 Unsupported Media Type
  Date: Fri, 09 May 2014 11:43:53 GMT
  Accept-Encoding: gzip
  Content-Length: 68
  Content-Type: text/plain

  This resource only supports the "gzip" content coding in requests.
```

...at which point the client can retry the request with the supported
"gzip" content coding.

Alternatively, a server that does not support any content codings in
requests could answer with:

```
  HTTP/1.1 415 Unsupported Media Type
  Date: Fri, 09 May 2014 11:43:53 GMT
  Accept-Encoding: identity
  Content-Length: 61
  Content-Type: text/plain
```

   This resource does not support content codings in requests.

5. Deployment Considerations

   Servers that do not support content codings in requests already are
   required to fail a request that does use a content coding. Section
   6.5.13 of [RFC7231] recommends using the status code 415 (Unsupported
   Media Type), so the only change needed is to include the "Accept-
   Encoding" header field with value "identity" in that response.

   Servers that do support some content codings are required to fail
   requests with unsupported content codings as well. To be compliant
   with this specification, servers will need to use the status code 415
   (Unsupported Media Type) to signal the problem, and will have to
   include an "Accept-Encoding" header field that enumerates the content
   codings that are supported. As the set of supported content codings
   is usually static and small, adding the header field ought to be
   trivial.

6. Security Considerations

   This specification does not introduce any new security considerations
   beyond those discussed in Section 9 of [RFC7231].

7. IANA Considerations

   HTTP header fields are registered within the "Message Headers"
   registry located at
   <http://www.iana.org/assignments/message-headers>, as defined by
   [BCP90].

   This document updates the definition of the "Accept-Encoding" header
   field, so the "Permanent Message Header Field Names" registry shall
   be updated accordingly:

```
   +-----------------+----------+----------+--------------------------+
   | Header Field    | Protocol | Status   | Reference                |
   | Name            |          |          |                          |
   +-----------------+----------+----------+--------------------------+
   | Accept-Encoding | http     | standard | [RFC7231], Section 5.3.4, |
   |                 |          |          | and Section 3 of this     |
   |                 |          |          | document                 |
   +-----------------+----------+----------+--------------------------+
```

8. References

8.1. Normative References

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
            Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
            RFC2119, March 1997,
            <http://www.rfc-editor.org/info/rfc2119>.

[RFC7230]   Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
            Protocol (HTTP/1.1): Message Syntax and Routing",
            RFC 7230, DOI 10.17487/RFC7230, June 2014,
            <http://www.rfc-editor.org/info/rfc7230>.

[RFC7231]   Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
            Protocol (HTTP/1.1): Semantics and Content", RFC 7231,
            DOI 10.17487/RFC7231, June 2014,
            <http://www.rfc-editor.org/info/rfc7231>.

8.2. Informative References

[BCP90]   Klyne, G., Nottingham, M., and J. Mogul, "Registration
            Procedures for Message Header Fields", BCP 90, RFC 3864,
            September 2004, <http://www.rfc-editor.org/info/bcp90>.

Appendix A. Change Log (to be removed by RFC Editor before publication)

A.1. Since draft-reschke-http-cice-00

Clarified that the information returned in Accept-Encoding is per resource, not per server.

Added some deployment considerations.

Updated HTTP/1.1 references.

A.2. Since draft-reschke-http-cice-01

Restrict the scope of A-E from "future requests" to "at the time of this request".

Mention use of A-E in responses other than 415.

Recommend not to include A-E in a 415 response unless there was actually a problem related to content coding.

A.3. Since draft-reschke-http-cice-02

First Working Group draft; updated boilerplate accordingly.

A.4. **Since draft-ietf-httpbis-cice-00**

   Apply editorial improvements suggested by Mark Nottingham.

Appendix B. **Acknowledgements**

   Thanks go to the members of the and HTTPbis Working Group, namely
   Amos Jeffries, Mark Nottingham, and Ted Hardie.

Author's Address

   Julian F. Reschke
   greenbytes GmbH
   Hafenweg 16
   Muenster, NW 48155
   Germany

   EMail: julian.reschke@greenbytes.de
   URI: http://greenbytes.de/tech/webdav/