WIKIPEDIA

Chunked transfer encoding

Chunked transfer encoding is a streaming data transfer mechanism available in version 1.1 of the Hypertext Transfer Protocol (HTTP). In chunked transfer encoding, the data stream is divided into a series of non-overlapping "chunks". The chunks are sent out and received independently of one another. No knowledge of the data stream outside the currently-being-processed chunk is necessary for both the sender and the receiver at any given time.

Each chunk is preceded by its size in bytes. The transmission ends when a zero-length chunk is received. The *chunked* keyword in the Transfer-Encoding header is used to indicate chunked transfer.

An early form of the chunked transfer encoding was proposed in 1994.[1] Chunked transfer encoding is not supported in HTTP/2, which provides its own mechanisms for data streaming.[2]

Contents

Rationale

The introduction of chunked encoding provided various benefits:

- Chunked transfer encoding allows a server to maintain an HTTP persistent connection for dynamically generated content. In this case, the HTTP Content-Length header cannot be used to delimit the content and the next HTTP request/response, as the content size is not yet known. Chunked encoding has the benefit that it is not necessary to generate the full content before writing the header, as it allows streaming of content as chunks and explicitly signaling the end of the content, making the connection available for the next HTTP request/response.
- Chunked encoding allows the sender to send additional header fields after the message body. This is important in cases where values of a field cannot be known until the content has been produced, such as when the content of the message must be digitally signed. Without chunked encoding, the sender would have to buffer the content until it was complete in order to calculate a field value and send it before the content.

Applicability

For version 1.1 of the HTTP protocol, the chunked transfer mechanism is considered to be always and anyway acceptable, even if not listed in the TE (transfer encoding) request header field, and when used with other transfer mechanisms, should always be applied last to the transferred data and never more than one time. This transfer coding method also allows additional entity header fields to be sent after the last chunk if the client specified the "trailers" parameter as an argument of the TE field. The origin server of the response can also decide to send additional entity trailers even if the client did not specify the "trailers" option in the TE request field, but only if the metadata is optional (i.e. the client can use the received entity without them). Whenever the trailers are used, the server should list their names in the Trailer header field; three header field types are specifically prohibited from appearing as a trailer field: Transfer-Encoding, Content-Length and Trailer.

Format

If a `Transfer-Encoding` field with a value of "`chunked`" is specified in an HTTP message (either a request sent by a client or the response from the server), the body of the message consists of an unspecified number of chunks, a terminating chunk, trailer, and a final CRLF sequence (i.e. carriage return followed by line feed).

Each chunk starts with the number of octets of the data it embeds expressed as a hexadecimal number in ASCII followed by optional parameters (*chunk extension*) and a terminating CRLF sequence, followed by the chunk data. The chunk is terminated by CRLF.

If chunk extensions are provided, the chunk size is terminated by a semicolon and followed by the parameters, each also delimited by semicolons. Each parameter is encoded as an extension name followed by an optional equal sign and value. These parameters could be used for a running message digest or digital signature, or to indicate an estimated transfer progress, for instance.

The terminating chunk is a regular chunk, with the exception that its length is zero. It is followed by the trailer, which consists of a (possibly empty) sequence of entity header fields. Normally, such header fields would be sent in the message's header; however, it may be more efficient to determine them after processing the entire message entity. In that case, it is useful to send those headers in the trailer.

Header fields that regulate the use of trailers are *TE* (used in requests), and *Trailers* (used in responses).

Use with compression

HTTP servers often use compression to optimize transmission, for example with `Content-Encoding: gzip` or `Content-Encoding: deflate`. If both compression and chunked encoding are enabled, then the content stream is first compressed, then chunked; so the chunk encoding itself is not compressed, and the data in each chunk is not compressed individually. The remote endpoint then decodes the stream by concatenating the chunks and uncompressing the result.

Example

Encoded data

In the following example, three chunks of length 4, 6 and 14 (hexadecimal "E") are shown. The chunk size is transferred as a hexadecimal number followed by \r\n as a line separator, followed by a chunk of data of the given size.

```
4\r\n (bytes to send)
Wiki\r\n (data)
6\r\n (bytes to send)
pedia \r\n (data)
E\r\n (bytes to send)
in \r\n
\r\n
chunks.\r\n (data)
0\r\n (final byte - 0)
\r\n (end message)
```

Note: the chunk size indicates the size of the chunk data and excludes the trailing CRLF ("\r\n"). In this particular example, the CRLF following "in" are counted as two octets toward the chunk size of 0xE (14). The CRLF in its own line are also counted as two octets toward the chunk size. The period character at the end of "chunks" is the 14th character, so it is the last data character in that chunk. The CRLF following the period is the trailing CRLF, so it is not counted toward the chunk size of 0xE (14).

Decoded data

```
Wikipedia in

chunks.
```

See also

- List of HTTP header fields

References

1. Connolly, Daniel (27 September 1994). "Content-Transfer-Encoding: packets for HTTP" (http://1997.webhistory.org/www.lists/www-talk.1994q3/1147.html). Retrieved 13 September 2013.
2. Belshe, Mike; Thomson, Martin; Peon, Roberto (May 2015). "Hypertext Transfer Protocol Version 2 (HTTP/2)" (https://tools.ietf.org/html/rfc7540). *tools.ietf.org*. Retrieved 2017-11-17. "HTTP/2 uses DATA frames to carry message payloads. The "chunked" transfer encoding defined in Section 4.1 of [RFC7230] MUST NOT be used in HTTP/2"

- See RFC 7230 section 4.1 (http://tools.ietf.org/html/rfc7230#section-4.1) for further details of chunked encoding.
- The previous (obsoleted) version is at RFC 2616 section 3.6.1 (https://tools.ietf.org/html/rfc2616#section-3.6.1).

Retrieved from "https://en.wikipedia.org/w/index.php?title=Chunked_transfer_encoding&oldid=1010842601"