Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 18, 2017

V. Krasnov
CloudFlare Inc.
July 17, 2016

Compression Dictionaries for HTTP/2
draft-vkrasnov-h2-compression-dictionaries-00

Abstract

The HTTP/2 [RFC7540] protocol encourages the use of many small assets for CSS/JS/HTML, due to its multiplexed nature. Prior to HTTP/2, asset inlining was encouraged, resulting in fewer, larger assets per website.

The nature of the compression algorithms, such as DEFLATE [RFC1951] and Brotli [BROTLI], used with HTTP in practice, require a certain "window" of data to perform backward matching. Therefore, larger files have much better compression ratio. These algorithms also allow the use of custom compression dictionaries which can be used as the initial window for backward matches.

This document specifies a new HTTP/2 frame type and a new HTTP/2 setting value that would allow a compression algorithm to use previously sent data as a compression dictionary, resulting in an improved compression ratio.

Note to Readers

A study performed on a actual set of websites in CloudFlare, produced up to 1.50X smaller files, when using DEFLATE (zlib compression level 8) with a dictionary, compared to DEFLATE alone. On average, 1.10X smaller files were produced.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The HTTP protocols allow for transmitted data to be compressed with a lossless compression algorithm. The algorithm used is specified in the "Content-Encoding" header field. For example "Content-Encoding: br" means the data was compressed using the Brotli format.

The compression, especially of dynamic resources, is a compute-heavy operation, where investing more compute power results in diminishing returns (in terms of compression ratio). One technique known to improve compression ratio significantly, and is supported by many compression formats is the "Compression Dictionary". A "Compression

Dictionary" allows the compressor to use a chunk of agreed upon data as the initial sliding window for a given algorithm.

This document introduces a mechanism for supplying such a dictionary over HTTP/2 to be used with an underlying compression algorithm.

1.1. Requirements Notation

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].

2. HTTP/2 Extension

2.1. HTTP/2 SETTINGS_ENABLE_DICTIONARIES Setting

HTTP/2 SETTINGS_ENABLE_DICTIONARIES (0xTBA): This setting can be used to enable the use of Compression Dictionaries for a given connection. The value indicates how many dictionaries are permitted. The initial value is 0, the maximal value is 8.

2.2. The SET_DICTIONARY frame

The SET_DICTIONARY frame (type=0xTBA).

```
+-------------+-------------+
| Dict ID (8) |   Size (8)  |
+-------------+-------------+
```

The payload of a SET_DICTIONARY frame contains the following fields:

Dict ID: An 8-bit value that specifies the slot for this dictionary. Dict ID must be in the range [0..SETTINGS_ENABLE_DICTIONARIES - 1].

Size: An 8-bit field indicating the size of the dictionary used. The actual size of the dictionary would be 2^Size. The maximal value of Size is 16, with the corresponding window size of 65536 octets.

2.3. The USE_DICTIONARY frame

The USE_DICTIONARY frame (type=0xTBA).

```
+-------------+
| Dict ID (8) |
+-------------+
```

The payload of a USE_DICTIONARY frame contains the following fields:

Dict ID: An 8-bit value that identify the dictionary slot, as set by the SET_DICTIONARY frame. Dict ID must be in the range [0..SETTINGS_ENABLE_DICTIONARIES - 1].

2.4. Server Behavior

The server can send the SET_DICTIONARY frame on any stream, before sending the initial DATA frame for that stream.

The server may then use the first 2^Size uncompressed octets of that stream as a Compression Dictionary for any subsequent stream.

In a typical scenario a server may set a dictionary for each content type, or use the initial stream as a dictionary for all other streams.

For every stream compressed with a Compression Dictionary, the server MUST send a USE_DICTIONARY frame, prior to any DATA frame on that stream.

The server MAY send several SET_DICTIONARY frames with the same ID. In that case the old Compression Dictionary is replaced by the new one.

2.5. Client Behavior

Upon receiving a SET_DICTIONARY frame, the client will reserve a slot for a dictionary with a given size. After receiving (and potentially decompressing) the DATA for a given stream, it will store the first 2^Size octets of the decompressed DATA in the dictionary. If 2^Size is greater than the size of the decompressed DATA, as many octets as are available will be used.

When receiving a USE_DICTIONARY frame, the client will use the specified dictionary to decompress the DATA.

A given stream may receive a SET_DICTIONARY and GET_DICTIONARY with the same ID. In that case the stream is decompressed with the old dictionary and then used as the new dictionary.

Due to the multiplexed nature of HTTP/2, it may be that a stream with a dictionary will arrive after a stream that uses it. This needs to be taken into account when setting priorities and stream window sizes.

3. IANA Considerations

This draft currently has no requirements for IANA. If the draft is standardized, the corresponding frames and settings will need to be assigned a type ID.

4. Security Considerations

Using any kind of compression over TLS may leak information about the plaintext. In that regard using a Compression Dictionary can potentially leak more information than regular use of compression. A special care should be taken when compressing sensetive data.

5. References

[BROTLI] Alakuijala, J. and J. Szabadka, "Brotli Compressed Data Format", May 2016.

[RFC1951] Deutsch, P., "DEFLATE Compressed Data Format Specification version 1.3", RFC 1951, May 1996.

[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>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

Author's Address

Vlad Krasnov
CloudFlare Inc.

Email: vlad@cloudflare.com