# hpack Documentation

*Release 2.3.0*

Cory Benfield

Aug 30, 2020

# Contents

hpack provides a simple Python interface to the HPACK compression algorithm, used to compress HTTP headers in HTTP/2. Used by some of the most popular HTTP/2 implementations in Python, HPACK offers a great Python interface as well as optional upgrade to optimised C-based compression routines from nghttp2.

Using hpack is easy:

```
from hpack import Encoder, Decoder

e = Encoder()
encoded_bytes = e.encode(headers)

d = Decoder()
decoded_headers = d.decode(encoded_bytes)
```

hpack will transparently use nghttp2 on CPython if it's available, gaining even better compression efficiency and speed, but it also makes available a pure-Python implementation that conforms strictly to RFC 7541.

# CHAPTER 1

# Contents

## 1.1 Installing hpack

hpack is trivial to install from the Python Package Index. Simply run:

```
$ pip install hpack
```

Alternatively, feel free to download one of the release tarballs from our GitHub page, extract it to your favourite directory, and then run

```
$ python setup.py install
```

hpack has no external dependencies.

### 1.1.1 Using nghttp2

If you want to use nghttp2 with hpack, all you need to do is install it along with its Python bindings. Consult nghttp2's documentation for instructions on how to install it.

## 1.2 hpack API

This document provides the HPACK API.

**class** hpack.**Encoder**

An HPACK encoder object. This object takes HTTP headers and emits encoded HTTP/2 header blocks.

**encode**(*headers*, *huffman=True*)

Takes a set of headers and encodes them into a HPACK-encoded header block.

**Parameters**

- **`headers`** – The headers to encode. Must be either an iterable of tuples, an iterable of `HeaderTuple`, or a `dict`.

  If an iterable of tuples, the tuples may be either two-tuples or three-tuples. If they are two-tuples, the tuples must be of the format `(name, value)`. If they are three-tuples, they must be of the format `(name, value, sensitive)`, where `sensitive` is a boolean value indicating whether the header should be added to header tables anywhere. If not present, `sensitive` defaults to `False`.

  If an iterable of `HeaderTuple`, the tuples must always be two-tuples. Instead of using `sensitive` as a third tuple entry, use `NeverIndexedHeaderTuple` to request that the field never be indexed.

  > **Warning:** HTTP/2 requires that all special headers (headers whose names begin with `:` characters) appear at the *start* of the header block. While this method will ensure that happens for `dict` subclasses, callers using any other iterable of tuples **must** ensure they place their special headers at the start of the iterable.
  >
  > For efficiency reasons users should prefer to use iterables of two-tuples: fixing the ordering of dictionary headers is an expensive operation that should be avoided if possible.

- **`huffman`** – (optional) Whether to Huffman-encode any header sent as a literal value. Except for use when debugging, it is recommended that this be left enabled.

**Returns** A bytestring containing the HPACK-encoded header block.

**`header_table_size`**
Controls the size of the HPACK header table.

**`class`** `hpack.`**`Decoder`**(*max_header_list_size=65536*)
An HPACK decoder object.

Changed in version 2.3.0: Added `max_header_list_size` argument.

**Parameters** **`max_header_list_size`** (`int`) – The maximum decompressed size we will allow for any single header block. This is a protection against DoS attacks that attempt to force the application to expand a relatively small amount of data into a really large header list, allowing enormous amounts of memory to be allocated.

If this amount of data is exceeded, a *OversizedHeaderListError <hpack.OversizedHeaderListError>* exception will be raised. At this point the connection should be shut down, as the HPACK state will no longer be useable.

Defaults to 64kB.

**`decode`**(*data*, *raw=False*)
Takes an HPACK-encoded header block and decodes it into a header set.

**Parameters**

- **`data`** – A bytestring representing a complete HPACK-encoded header block.
- **`raw`** – (optional) Whether to return the headers as tuples of raw byte strings or to decode them as UTF-8 before returning them. The default value is False, which returns tuples of Unicode strings

**Returns** A list of two-tuples of `(name, value)` representing the HPACK-encoded headers, in the order they were decoded.

**Raises** ***`HPACKDecodingError`*** – If an error is encountered while decoding the header block.

**header_table_size**
Controls the size of the HPACK header table.

**class** `hpack.`**HeaderTuple**
A data structure that stores a single header field.

HTTP headers can be thought of as tuples of `(field name, field value)`. A single header block is a sequence of such tuples.

In HTTP/2, however, certain bits of additional information are required for compressing these headers: in particular, whether the header field can be safely added to the HPACK compression context.

This class stores a header that can be added to the compression context. In all other ways it behaves exactly like a tuple.

**class** `hpack.`**NeverIndexedHeaderTuple**
A data structure that stores a single header field that cannot be added to a HTTP/2 header compression context.

**class** `hpack.`**HPACKError**
The base class for all `hpack` exceptions.

**class** `hpack.`**HPACKDecodingError**
An error has been encountered while performing HPACK decoding.

**class** `hpack.`**InvalidTableIndex**
An invalid table index was received.

**class** `hpack.`**OversizedHeaderListError**
A header list that was larger than we allow has been received. This may be a DoS attack.

New in version 2.3.0.

## 1.3 Vulnerability Notifications

This section of the page contains all known vulnerabilities in the HPACK library. These vulnerabilities have all been reported to us via our vulnerability disclosure policy.

### 1.3.1 Known Vulnerabilities

| # | Vulnerability | Date Announced | First Version | Last Version | CVE |
|---|---|---|---|---|---|
| 1 | HPACK Bomb | 2016-08-04 | 1.0.0 | 2.2.0 | CVE-2016-6581 |

# Index