IETF 95 - Thoughts on HTTP Header Field Parsing

Julian Reschke, greenbytes

Again?
Background

- HTTPbis WG Work on Content-Disposition ([RFC 6266](https://tools.ietf.org/html/rfc6266))
- Various HTTPbis WG issues, such as [231: Considerations for new headers](https://tools.ietf.org/html/rfc231)
- General Discussions about header compression in the context of HTTP/2

Problem Statement

- The parsing of many HTTP header fields is *hard*!
- Implementations *do* get it wrong.
- Extension points not well understood.
- I18N not well understood and frequently considered too late.
- We can't fix the past, but we can try to do better.

*Most of these slides were done for IETF 81; we haven't made a lot of progress since!*
Example: the List Production and repeating Header Field instances

```
Foo: a
Foo: b
```

is equivalent to

```
Foo: a, b
```

- This is fine for simple stuff like method names.
- It falls apart when people who define new header fields do not get it (Example: Set-Cookie).
- It helps for folding multiple instances into one, but *not* for parsing.

```
If-Match: "strong", W/"weak", "oops, a "comma"
```
Example: the List Production and repeating Header Field instances

Combining list production with structured field syntax:

```plaintext
WWW-Authenticate = 1#challenge
challenge = auth-scheme [ 1*SP ( token68 / #auth-param ) ]
auth-param = token BWS "=" BWS ( token / quoted-string )
```

Example:

```
WWW-Authenticate: Newauth realm="newauth";
    test="oh, a "comma""; foo=a'b'c, Basic realm="basic"
```
Example: Parameters - Whitespace, Quoting

\[ \text{param} = \text{token } "=\" ( \text{token} / \text{quoted-string} ) \]

\text{foo}=\text{bar}; \text{foo}='\text{bar}'; \text{foo}="\text{bar}"; \text{foo} = "\text{bar}"

- Whitespace sometimes allowed, sometimes not (partly due to confusion about implied LWS).
- Lots of confused parsers.
- Single quote \textit{is} used in \textit{token} values, thus is \textit{not} available for quoting.
- Definitions special-case the right hand side for individual parameter names, generic parsers can't do that (example: RFC 5988 disallows token form for title, uses double quotes for quoted-	extit{mt} without making it a quoted-	extit{string}).
- Empty parameters ("; ;") usually not allowed, but accepted in practice.
Proposals (2011)

- Test Cases. Examples. Lots.
- Make existing syntax more consistent where we can (fix mistakes where possible, discourage generating useless whitespace, require recipients to deal with it nevertheless).
- Encourage authors of new header fields to re-use existing syntax and to think about extensibility. *(done in RFC 7231)*
Proposals (2014)

For existing header fields (including those in the base specs):

- Write test cases.
- Raise bug reports.
- Try to refactor parsing code everywhere to increase the amount of shared code between header fields.
- Feed back the results of this into the RFC723*bis revision process.
Proposals (2014) (continued)

Thought experiment in draft-reschke-http-jfv: what if header field values would use JSON?

```json
WWW-Authenticate: { "Newauth" : {
    "realm": "newauth",
    "test": "oh, a \"comma\"",
    "foo": "a'b'c" },
  { "Basic" : { "realm": "basic" }}
```

- unified data model: JSON array (implied "[ ... ]")
- single parser
- I18N solved once for all
- list syntax a friend, not an interop problem
- potential wins in new HTTP wire formats

But:
• Chatty when compared to homegrown syntax: maybe a case for a more concise notation for JSON?

• An alternative would be "JSON object" with implied "{ .. }", but that variant loses the list notation win.
Links

- JSON Encoding for Header Field Values - [draft-reschke-http-jfv-03](https://www.ietf.org/mail-archive/text/draft-reschke-http-jfv-03.txt)