Structured Headers

IETF101, London

Reminder: Goals

- Make it easier and more reliable to specify and parse HTTP header fields
- Accommodate future encodings for efficiency (but not specify now)
- Non-Goals:
  - Re-specify existing header fields
  - Affect/handle headers that don’t “opt in” to this spec

Recent History

- -02: After discussion in SIN, agreed to rebase on draft-nottingham-structured-headers
- -03: Refine algorithms (various issues), split numbers into integers and floats (#434), throw error on trailing garbage (#436), etc.
- -04: Lots of editorial work, “labels” → “identifiers”, adjustments to binary type (#495, #473)

Possible Top-Level Types

- Dictionary → `foo=“1”, bar=2`
- List → `foo, bar, “baz”`
- Parameterised List → `foo; a=1, bar; b=“two”`
- Item →

```
foo                             // identifier
1.5                             // float
42                              // integer
“Mary had a little lamb”        // string
*SGVsbG8=*                      // binary
```

Currently, we require a parser to “know” the top-level type

#433: Length Limits

- Right now, we specify limits on how large various types can be.
  - E.g., integers are 64bit signed; strings are max 1024 characters
  - This helps assure interop, and assists optimisations
  - Also means that specifications don’t **have** to spec limits
- Q1: Do we agree that limits are good?
- Q2: Have we chosen the right limits?

#505: Strings and Identifiers

- Like parsing, generating HTTP headers requires knowledge of the top-level type.
- On-wire representation means that data types below that aren't ambiguous.
- But, what about the data inside? E.g. the difference between Identifiers and Strings isn't obvious without extra information about the type.
- Option 1: Require such metadata to be present
- Option 2: Work to make sure that abstract types map to common programming language types / structures

Next Steps

- Have had good review/participation, but more eyeballs on the spec always welcome
- We think we’re about ready for prototype implementations
- Should be ready for WGLC after some implementation experience