RFC 7230 - HTTP (HTTP/1.1): Message Syntax and Routing

2. Architecture

HTTP was created for the World Wide Web (WWW) architecture

2.1 Client/Server Messaging

HTTP is a stateless request/response protocol that operates by exchanging messages across a reliable transport- or session-layer connection.

HTTP Client - program that establishes a connection to a server for the purpose of sending one or more HTTP requests

HTTP Server - program that accepts connections in order to service HTTP requests by sending HTTP responses.

user agent - refers to any of the various client programs that initiate a request, including (but not limited to) browsers, spiders (web-based robots), command-line tools, custom applications, and mobile apps.

Most HTTP communication consists of a retrieval request (GET) for a representation of some resource identified by a URI

2.3 Intermediaries

three common forms of HTTP intermediary:

- proxy - message-forwarding agent that is selected by the client, usually via local configuration rules, to receive requests for some type(s) of absolute URI and attempt to satisfy those requests via translation through the HTTP interface.
- gateway - aka reverse proxy - acts as an origin server for the outbound connection but translates received requests and forwards them inbound to another server or servers
- tunnel - acts as a blind relay between two connections without changing the messages. Used in virtual connections such as TLS

2.4. Caches

a local store of previous response messages and the subsystem that controls its message storage, retrieval, and deletion.

The effect of a cache is that the request/response chain is shortened if one of the participants along the chain has a cached response applicable to that request

```
    >             >
          UA ========== A ========== B - - - - - - C - - - - - - - O
```

A response is only cacheable if the response can be used in answering subsequent requests

2.5. Conformance and Error Handling

An implementation is considered conformant if it complies with all of the requirements associated with the roles it partakes in HTTP. This could include both syntax and semantics defined by the ABNF rules.

HTTP does not have specific length limitations but at a minimum, it must be able to parse and process protocol element lengths that are at least as long as the values that it generates for those same protocol elements in other messages.

A recipient MUST interpret a received protocol element according to the semantics defined for it by this specification, including extensions to this specification (i.e. Accept-Encoding header field)

HTTP does not define specific error handling mechanisms except when they have a direct impact on security, since different applications of the protocol require different error handling strategies

2.6. Protocol Versioning

HTTP uses a "<major>.<minor>" numbering scheme to indicate versions of the protocol. This RFC specifies version 1.1

When an HTTP/1.1 message is sent to an HTTP/1.0 recipient the HTTP/1.1 message is constructed such that it can be interpreted as a valid HTTP/1.0 message if all of the newer features are ignored.

Intermediaries that process HTTP messages (i.e., all intermediaries other than

those acting as tunnels) MUST send their own HTTP-version in forwarded messages.
A Client SHOULD send a request version equal to the highest version to which the client is conformant
A Server SHOULD send a response version equal to the highest version to which the server is conformant less than or equal to the one received in the request.

2.7. Uniform Resource Identifiers

used throughout HTTP as the means for identifying resources (Section 2 of RFC 7231)
URI references are used to target requests, indicate redirects, and define relationships.

2.7.1. http URI Scheme

http-URI = "http:" "//" authority path-abempty ["?" query] ["#" fragment]
example - https://en.wikipedia.org/wiki/Fragment_identifier?name=adam#Examples
origin server - identified by the authority component, which includes a host identifier and optional TCP port en.wikipedia.org is the origin server in example
hierarchical path component - serve as an identifier for a potential target resource within that origin server's name space. wiki/Fragment_identifier is the path component
optional query component - name=adam is the query
fragment - optional component which allows for indirect identification of a secondary resource, independent of the URI scheme. Examples is the query
Although HTTP is independent of the transport protocol, the "http" URI scheme is specific to TCP-based services because the name delegation process depends on TCP for establishing authority.

2.7.2. https URI Scheme

All requirements listed above are the same for https schemes. Except the TCP port 443 is the default. Also, the "https" URI scheme depends on both TLS and TCP for establishing authority.

2.7.3. http and https URI Normalization and Comparison

The normalization algorithm defined in RFC3986 section 6, is used to compare similar looking URIs to check for equivalence. For example, the following three URIs are equivalent:

http://example.com:80/~smith/home.html
http://EXAMPLE.com/%7Esmith/home.html
http://EXAMPLE.com:/%7esmith/home.html

3. Message Format