Maintaining HTTP

Support for HTTP is a key priority in terms of ensuring the continued success of Node.js as captured in the project's technical priorities.

The current high level strategy is based on the discussion in the Next-10 mini-summit on modern HTTP which was held on Jan 27 2022.

High level strategy

The key elements of our strategy for future HTTP APIs are:

- APIs should be HTTP protocol independent (support HTTP1, HTTP2, etc.).
- APIs should be transport protocol independent (TCP, QUIC, etc.).
- APIs should provide good defaults that perform well.
- Client/server APIs should be consistent and allow easy integration.
- Common requirements like piping out from client API to server APIs should be easy.
- For both the Client and Server there is a need for multiple APIs, with each targeting a different level of abstraction.

Unfortunately our existing HTTP APIs (HTTP, HTTPS, and HTTP2) do not align with our high level strategy. While these APIs are widely used and we do not plan to deprecate or remove them, they are not the focus of active development or performance improvements. Bug fixes however are still important for all of these APIs.

With respect to the HTTP protocols themselves, our current assessment in terms of priority within existing or new APIs is:

- HTTP 2 is in “maintenance mode” for both protocol and APIs.
- HTTP 1 is a stable protocol, but innovation is still possible with the APIs.
- HTTP 3 is an important protocol and we need to add support for it.

The current strategy is to build out 2 new client APIs and 2 new Server APIs in line with the high level strategy above.

While transport-level APIs are important (e.g. socket level), the HTTP APIs should not be tied to a specific transport-level API. Therefore, transport-level APIs are out of the scope of our HTTP strategy/maintenance information.

Client APIs

For client APIs we want a high-level API and a low-level API when more control is required. The current plan is for the following APIs:

- High-level API - Fetch based API built on top of undici.
- Low-level API - a subset of the APIs exposed by undici. The exact shape and set of these APIs is still to be worked out. The current plan is to pull undici into Node.js core without exposing its APIs in the Node.js API so that it can initially be used to support the higher level Fetch-based API. As this gets worked out we will discuss which APIs to expose in the Node.js API surface.

Server APIs

For the server APIs we do not yet have a clear path, other than wanting to align them with the APIs built for the client.

Maintaining the HTTP APIs

HTTP, HTTPS

The low-level implementation of the HTTP and HTTPS APIs are maintained in the llttp repository. Updates are pulled into Node.js under deps/llhttp

The low-level implementation is made available in the Node.js API through JavaScript code in the lib directory and C++ code in the src directory.

HTTP2

The low-level implementation of HTTP2 is based on nghttp2. Updates are pulled into Node.js under deps/nghttp2 as needed.

The low-level implementation is made available in the Node.js API through JavaScript code in the lib directory and C++ code in the src directory.