HTTP/3
Open Design Issues

Errors and Error Handling

#2718 – Truncated Stream Handling is Aggressive

Spec says:

> These streams carry frames related to the request/response (see Section 4.1). When a stream terminates cleanly, if the last frame on the stream was truncated, this MUST be treated as a connection error (see HTTP_MALFORMED_FRAME in Section 8.1).

- Is a connection error appropriate, or should this be relaxed?

Editor says: Yes, this is an error. Close with no action.

#2711 – Relax prohibition on server-initiated bidirectional streams

Spec says:

> HTTP/3 does not use server-initiated bidirectional streams; clients MUST omit or specify a value of zero for the QUIC transport parameter initial_max_bidi_streams.

- Extensions might want to use bidirectional streams
- Current text adds an extra RTT to extensions wanting to do this
  - Or send a MAX_STREAMS frame immediately
- No specified enforcement (server doesn't reject connections that allow server to open streams)

Editor says: Change this to a SHOULD omit/zero *unless* an extension is being offered

#2699 – Specify handling of QUIC SERVER_BUSY connection failures

Transport spec says:

> If a server isn't currently accepting any new connections, it SHOULD send an Initial packet containing a CONNECTION_CLOSE frame with error code SERVER_BUSY.

HTTP spec says:

> [When using Alt-Svc,] Connectivity problems (e.g. firewall blocking UDP) can result in QUIC connection establishment failure, in which case the client SHOULD continue using the existing connection or try another alternative endpoint offered by the origin.

- Is this sufficient?
- Does the solution to this belong in HTTP/3 or in QUIC?

Editor says: Method for selecting candidate ports to connect to should describe what happens when a connection to a candidate fails. Failure error code is irrelevant.

#2551/#2662 – Replace MALFORMED_FRAME with specific error codes

Spec says:

> HTTP_MALFORMED_FRAME (0x01XX):
>
> An error in a specific frame type. If the frame type is 0xfe or less, the type is included as the last byte of the error code. For example, an error in a MAX_PUSH_ID frame would be indicated with the code (0x10D). The last byte 0xff is used to indicate any frame type greater than 0xfe.

PR says:

- HTTP_BAD_FRAME_SIZE
- HTTP_INVALID_PRIORITY
- HTTP_DUPLICATE_SETTING
- HTTP_LIMIT_EXCEEDED (expanded definition)
- HTTP_DUPLICATE_PUSH (expanded definition)
- HTTP_PUSH_ID_REDUCED
- ~~HTTP_MALFORMED_FRAME~~

Editor says: Let’s do this.

#2516 – Semantics of MAX_HEADER_LIST_SIZE

Spec says:

> An HTTP/3 implementation MAY impose a limit on the maximum size of the header it will accept on an individual HTTP message; encountering a larger message header SHOULD be treated as a stream error of type HTTP_EXCESSIVE_LOAD. If an implementation wishes to advise its peer of this limit, it can be conveyed as a number of bytes in the SETTINGS_MAX_HEADER_LIST_SIZE parameter.

- Many implementations in HTTP/2 don't actually enforce the advertised value
- Inconsistently enforced limits don't provide value
- Not clear when this is intended to be used

Editor says: This is a shortcut; advertise the point at which you would drop an incoming message for excessive size (which might be never). Next hop can early-reject messages on your behalf.

#2498 – Behavior on out-of-range settings

QPACK says:

> SETTINGS_QPACK_MAX_TABLE_CAPACITY (0x1):
>
> An integer with a maximum value of 2^30 - 1. The default value is zero bytes. [...]
>
> SETTINGS_QPACK_BLOCKED_STREAMS (0x7):
>
> An integer with a maximum value of 2^16 - 1. The default value is zero.

HTTP/3 says:

> **crickets**

- QPACK treats a "maximum" value for a setting as a general concept
- HTTP/3 defines no error for an out-of-range setting

Editor says: Expand the definition of HTTP_LIMIT_EXCEEDED

#2412 – Can MAX_PUSH_ID go backward?

HTTP/3 says:

> A MAX_PUSH_ID frame cannot reduce the maximum Push ID; receipt of a MAX_PUSH_ID that contains a smaller value than previously received MUST be treated as a connection error of type HTTP_MALFORMED_FRAME.

QUIC says:

> Loss or reordering can cause a MAX_STREAMS frame to be received which states a lower stream limit than an endpoint has previously received. MAX_STREAMS frames which do not increase the stream limit MUST be ignored.

- MAX_PUSH_ID is sent on the control stream, where there is no loss or reordering
  - If HTTP/4 uses QUIC DATAGRAMs to carry these frames, obviously this changes.
- Reneging is a correctness violation

Editor says: Close with no action

#2410 – Import rules on “malformed requests” from RFC7540

HTTP/2 says:

> Intermediaries that process HTTP requests or responses (i.e., any intermediary not acting as a tunnel) MUST NOT forward a malformed request or response. Malformed requests or responses that are detected MUST be treated as a stream error (Section 5.4.2) of type PROTOCOL_ERROR.
>
> For malformed requests, a server MAY send an HTTP response prior to closing or resetting the stream. Clients MUST NOT accept a malformed response.

HTTP/3 says:

> **crickets*
>
> (though see some of DaanDeMeyer’s PRs for specific varieties of malformed requests)

Editor says: Let’s do this.

PRIORITY and Prioritization

#2697 – SHOULD use PRIORITY

- HTTP/2 says *how to convey* priorities
  - Some implementations, server and client, don’t implement it
- HTTP/3 says *how to convey* priorities
  - Should the spec call implementation a SHOULD?

Editor says: We can recommend all we want, sure.

#2502/#2690 – Priority inversion from reordering

HTTP/3 says:

> Due to reordering between streams, an element can also be prioritized which is not yet in the tree. Such elements are added to the tree with the requested priority.
>
> When a prioritized element is first created, it has a default initial weight of 16 and a default dependency. Requests and placeholders are dependent on the root of the priority tree….

- Editorial: Needs to say that a dependency on a stream that doesn't exist yet causes that parent to be added to the tree
- Problem: These newly-added streams depend on the root, which makes them the most important things in the tree!

PR says:

> The tree also contains an orphan placeholder. This placeholder cannot be reprioritized, and no resources should be allocated to descendants of the orphan placeholder if progress can be made on descendants of the root.
> […]
> When a prioritized element is first created, it has a default initial weight of 16 and a default dependency. Requests and placeholders are dependent on the orphan placeholder….

Editor says: Briefly under-prioritized is probably better than briefly over-prioritized

# ???/#2700 – Strict Priorities

PR proposes changes to the priority scheme:

- Streams cannot depend on other streams; only on placeholders
- Elements have both a priority and a dependency
    - Bandwidth allocated to a placeholder is used to service the highest-priority child
    - Children of equal priority are handled:
        - One at a time in any order if no weight is set
        - Weight-based allocation of bandwidth if weight is set
        - 50/50 split of bandwidth between unweighted and weighted groups if mixed

Editor says: Interesting. Is this in scope?

Structural Changes

#2678 – Use unidirectional streams for everything!

The control stream carries:

- SETTINGS (only once, must be first)
- PRIORITY (ordered amongst themselves)
- MAX_PUSH_ID (ordered amongst themselves)
- CANCEL_PUSH (unordered)
- GOAWAY (ordered amongst themselves / unordered)

Since there are no ordering requirements cross-type, these could be separate unidirectional streams so that lost packets containing one don't block others.

#2418 separately proposes using unidirectional streams for MAX_PUSH_ID

Editor says: Big change need big reason.

#2526 – PUSH_ID frame

- SETTINGS and PRIORITY create precedent for frames which can only (or MUST) be the first frame on a stream
- Push ID is carried as an extension of the unidirectional stream header on push streams, then frames begin
- Would be more consistent as a PUSH_ID frame

Editor says: `6 == (12/2)`

#2632 – Symmetric GOAWAY

HTTP/3 says:

> The GOAWAY frame … carries a QUIC Stream ID for a client-initiated bidirectional stream encoded as a variable-length integer.
>
> The GOAWAY frame indicates that client-initiated requests on lower stream IDs were or might be processed in this connection, while requests on the indicated stream ID and greater were rejected.

- Does not indicate anything about client-initiated unidirectional streams
- Client cannot indicate to server which push / extension streams were processed before end of connection

Editor says: Extensions can define their own shutdown mechanisms if needed. Close with no action.

Relationship to TCP, Alt-Svc, etc.

#2488 – Embed address validation token in Alt-Svc

- HTTP/3 defines a QUIC version-negotiation Alt-Svc extension which can save a round trip
- Providing the token for the QUIC Initial packet could also save a round trip

Editor says: Doesn't need to be in the HTTP/3 spec.

(Of course, neither does the other one, and mix/match of unknown Alt-Svc extensions gets messy.)

#2439 – http:// URIs over HTTP/3

RFC 8164 says:

> For various reasons, it is possible that the server might become confused about whether requests' URLs have an "http" or "https“ scheme (see Section 4.4). To ensure that the alternative service has opted into serving "http" URLs over TLS, clients are required to perform additional checks before directing "http" requests to it.

HTTP/3 says:

> **crickets**

- Should HTTP/3 require similar opt-in, or should it mandate proper handling of scheme?

Editor says: You’re implementing a new protocol – get it right.

#2223 – Coalescing rules

HTTP/2 says:

> A connection can be reused as long as the origin server is authoritative (Section 10.1). For TCP connections without TLS, this depends on the host having resolved to the same IP address.
>
> For "https" resources, connection reuse additionally depends on having a certificate that is valid for the host in the URI. The certificate presented by the server MUST satisfy any checks that the client would perform when forming a new TLS connection for the host in the URI.

HTTP/3 says:

> The client MAY send any requests for which the client considers the server authoritative.
>
> An authoritative HTTP/3 endpoint is typically discovered because the client has received an Alt-Svc record from the request's origin which nominates the endpoint as a valid HTTP Alternative Service for that origin. As required by [RFC7838], clients MUST check that the nominated server can present a valid certificate for the origin before considering it authoritative. Clients MUST NOT assume that an HTTP/3 endpoint is authoritative for other origins without an explicit signal.

Editor says: Pending an updated definition of authority from http-core.

#253 – HTTP/3 without Alt-Svc

Discussion in Tokyo / Prague:

- Resource identification is increasingly distinct from the actual retrieval server / port / protocol (URIs vs. URLs)
- Any client *can* ask any server for any resource over any connection
  - Client has to decide whether to trust server's answer for that resource (if not, don't ask or discard response)
  - Server has to decide whether it's willing to answer requests for that resource (if not, 421)
- H2/H3 explicitly carry scheme/authority/path to enable this behavior
- Client needs algorithm to derive candidate server connections from URI
  - Alt-Svc expands the set of candidates / hints at server preferences

Where does this text belong?