# Template-Driven HTTP CONNECT Proxying for TCP

Ben Schwartz, Meta Platforms Inc.
HTTPBIS @ IETF 125

# Template-driven TCP Transport Proxy (i.e. MASQUE for TCP)

Proxy is identified by a template:

```
https://proxy.example/tcp
{?target_host,target_port}
```

In HTTP/1.1:

```
GET /tcp?
    target_host=192.0.2.1&
    tcp_port=443 HTTP/1.1
Host: proxy.example:443
Connection: Upgrade
Upgrade: connect-tcp
capsule-protocol = ?1
```

In HTTP/2 & HTTP/3:

```
:method = CONNECT
:protocol = connect-tcp
:scheme = https
:authority = proxy.example:443
:path = /tcp?
        target_host=192.0.2.1&
        target_port=443
capsule-protocol = ?1
...
```

# Changes since -09 (presented @ IETF 123)

- Changes to “abrupt closure” signaling requirements in HTTP/1.1
  - Discussed at IETF 123
  - Github PR #3141
  - This is a very minor point!
    - Ordinary FIN is represented by the FINAL_DATA capsule, independent of HTTP version.
    - This change only affects “RST after FIN” (i.e. half-close succeeded, full close failed)
    - The draft discourages use of HTTP/1.1 anyway.
- Editorial changes

# HTTP/1.1 Abrupt Closure: -09 vs. -10

- *HTTP/1.1 over TLS: a TLS Error Alert*

- *HTTP/1.1 over TLS: TCP shutdown without a TLS closure alert; see [RFC8446], Section 6.1.*

Test implementation (Golang) showed that sending a TLS Error Alert is difficult, but closing the TCP socket without close_notify is easy.

A TLS Error Alert would still be allowed.

# Reasons to avoid HTTP/1.1: -09 vs. -10

- *It may be difficult to implement the recommended unclean shutdown signals (Section 3.4), as many TLS libraries do not support injecting TLS Alerts.*

- *The graceful and abrupt closure signals (Section 3.4) are more likely to be missing or corrupted:*
  - *Some implementations may be unable to emit the recommended abrupt closure signals, due to limitations in their TCP and TLS subsystems.*
  - *Faulty implementations may fail to send a TLS closure alert during graceful shutdown, or fail to report an error when the expected closure alert is not received. These misbehaviors are not compliant with [RFC8446], but they are common nonetheless among HTTP/1.1 implementations today.*

# New Problem: *Proxy-Status Trailers*

**RFC 9209**

*Proxy-Status MAY be sent as an HTTP trailer field. …*

`Proxy-Status: ThisProxy; error=read_timeout`

*… For example, if the forward connection abruptly closes, an intermediary might add Proxy-Status with an appropriate error as a trailer field.*

Supports debugging: can convey an error message for “ran out of quota” vs. “server is rebooting” vs. “connection error” vs. “connection timeout” vs. “error propagated from a further gateway” …

# draft-ietf-connect-tcp-10 Section 4.2

*Proxies implementing this specification SHOULD include a "Proxy-Status" response header [RFC9209] in any success or failure response (i.e., status codes 101, 2XX, 4XX, or 5XX) to support advanced client behaviors and diagnostics. In HTTP/2 or HTTP/3, proxies MAY additionally send a "Proxy-Status" trailer in the event of an abrupt stream closure.*

Just a harmless reminder about Proxy-Status trailers, right?

# Problem: No Trailers Allowed with CONNECT

**RFC 9113 (HTTP/2)**

*Frame types other than DATA or stream management frames (RST_STREAM, WINDOW_UPDATE, and PRIORITY) MUST NOT be sent on a connected stream and MUST be treated as a stream error (Section 5.4.2) if received.*

**RFC 9114 (HTTP/3)**

*Once the CONNECT method has completed, only DATA frames are permitted to be sent on the stream. Extension frames MAY be used if specifically permitted by the definition of the extension. Receipt of any other known frame type MUST be treated as a connection error of type H3_FRAME_UNEXPECTED.*

# Possible Ways Forward

1. s/MAY/MUST NOT/. No Trailers with connect-tcp.
    a. Could follow up with a new Capsule draft to convey this info
        i. “FIN”? “PROXY_STATUS”? “ERROR”? “HEADER”? “TRAILER”?
2. Interpret “connect-tcp” as an “extension” that allows trailers:
    a. RFC 9114: *“Extension frames MAY be used if specifically permitted by the definition of the extension”.*
    b. RFC 9113 is silent but we can presume extensibility is allowed.
    c. But what about “connect-udp”?
3. “Update” Extended CONNECT (RFC 8441, RFC 9220) to allow trailers
    a. Either generally, or only if an “upgrade token” opts-in.
    b. Theoretically a breaking change but probably nothing would break…
4. ~~“Update” HTTP/2 and HTTP/3 to allow trailers on CONNECT~~

**Need WG input**