


Template-Driven HTTP CONNECT Proxying for TCP

Ben Schwartz, Meta Platforms Inc.
HTTPBIS @ IETF 119



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

Proxy is identified by a template:

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

In HTTP/2 & HTTP/3:

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

...



Changes and discussions since IETF 118

- Changes in draft-02
 - Added a default template, for use only if plain CONNECT fails.
 - Instruction to use “WWW-Authenticate”, not “Proxy-Authenticate”
 - Discussion of “Alt-Svc”, “Set-Cookie” and other origin-scoped headers.
 - *“Unlike classic HTTP CONNECT proxies, a templated TCP proxy has an unambiguous origin of its own.”*
 - Security Considerations text
- Comments and proposed changes during WGLC
 - Editorial issues (e.g. ambiguity related to “100 (Continue)” in [#2717](#), resolved in [#2718](#))
 - Multiprotocol template inference and “tcp_port” vs. “target_port” ([#2713](#), see [slide 4](#))
 - Proposal to remove “target_host” list feature (see [slide 5](#))
 - Proposal to integrate the Capsule Protocol (see [slide 6](#))



Multiprotocol Templates and “tcp_port”

draft-02: “The names of the variables in the URI Template uniquely identify the capabilities of the proxy. ... https://proxy.example/{?target_host,tcp_port,target_port,target,ipproto,dns}”

Problems:

1. The “connect-ip” variables (target, ipproto) are both optional to include in the template!
2. Nothing explicitly forbids adding a bunch of extra variables with colliding names.

Proposed change:

1. s/tcp_port/target_port/
2. “The contents of the URI Template are not necessarily sufficient to determine its purpose, so clients must determine this in some other way, such as by **probing** or via **a separate usage indication**.”

Alternative: Define formal rules for creating and parsing “multiprotocol proxy templates”.



Happy Eyeballs and “target_host” lists

A classic HTTP CONNECT request can only carry one target IP address. This makes Happy Eyeballs very inefficient with client-side DNS resolution. draft-02 tries to do better:

“If “target_host” is a list ..., the server SHOULD perform the same connection procedure as if these addresses had been returned in response to A and AAAA queries for a domain name.”

Proposed Change: Drop this feature and create a new draft that adds it to “connect-tcp” and “connect-udp” as an extension. (But: **we don’t have an extension mechanism...**).

☆ **Alternative:** Keep the feature (restricted to Level 2 Template syntax to simplify clients). Potentially follow up with a draft to backport it to “connect-udp” (which is not trivial, since UDP doesn’t intrinsically support Happy Eyeballs).



Using the Capsule Protocol

“connect-tcp” doesn’t use the Capsule Protocol. The protocol content is the TCP stream.

Proposed Change: Add optional or mandatory support for the Capsule Protocol in this draft by defining a new “DATA” capsule type for generic data. Potentially UPDATE RFC 9297 to permit streaming capsules.

☆ **Alternative:** Leave the document as-is until we have a need for the Capsule Protocol. If a need arises, it can be negotiated by adding the “Capsule-Protocol: 1” request header.

FIN

