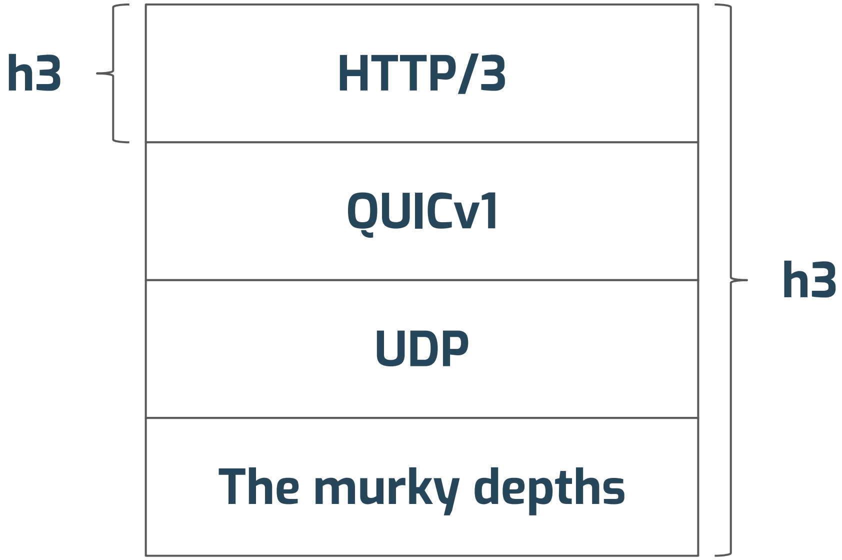
QUIC, HTTP/3, and ALPN

QUIC @IETF 106, Singapore

#3117 How do we identify HTTP/3?

Do we identify the layer, or the stack?



Assumptions/Facts

Both QUIC and HTTP will get new versions

HTTP/3 will very likely work with some QUIC versions >1

Some versions of QUIC will work with each HTTP version

Some versions won’t

- The application protocol interface is not a QUIC invariant

QUICv2 for HTTP/3

Identifying the stack is uncomplicated:

- “h3q2” is unambiguous

Identifying the layer is tricky:

- Compatibility can’t be defined separately from either HTTP/3 or QUICv2
- Failures occur if endpoints could disagree about whether HTTP/3 is allowed with QUICv2

Interaction with Version Negotiation

Are the layers negotiated together, or separately?

The original conception of version negotiation

- QUIC version negotiated first
- Pick an application protocol given QUIC version

Compatible version negotiation (schinazi-quic-version-neg…)

- Maybe transport and application negotiated together
- Except that when versions *aren't* compatible…?

Alt-Svc

Complicated AND flexible

Alt-Svc: h3=”:443”;quic=”1,2”, h4=”:443”;quic=”2”

Or simpler but more rigid

Alt-Svc: h3=”:443”, h3q2=”:443”, h4=”:443”

Layering: Define HTTP/3 dependency on QUICv1

If HTTP/3 is going to define how it can be used with QUICv2

- … it has to be based on abstractions
- It now says that it’s OK if QUIC uses TLS, which is wrong

HTTP/3 depends on transport services from QUIC

- Streams (both bidirectional and unidirectional)
- Does it also depend on the shape of the handshake?
- Can we clearly define the critical services?

Layering(2): Punt to QUICv2

HTTP/3 requires QUICv1

Let QUICv2 say that HTTP/3 can be used with it

What about protocols that are less well known?

Stack: Multi-stage negotiation implications

QUIC version can be negotiated without application protocol

- Could make suboptimal decision

e.g.,

	QUICv1	QUICv2
HTTP/3	✓	✓
HTTP/4	✓	✗

- A preference for HTTP/4 + QUICv1 never gets chosen
- That option might never be known to the server

Decision

This is a straight up trade-off:

- Identifying the layer might be appealing
  - But it is not free
- Identifying the stack is simple
  - But it is a little inflexible