

Path Layer UDP Substrate (PLUS) Technical Considerations

Brian Trammell, PLUS BoF
IETF 96 Berlin - 21 July 2016

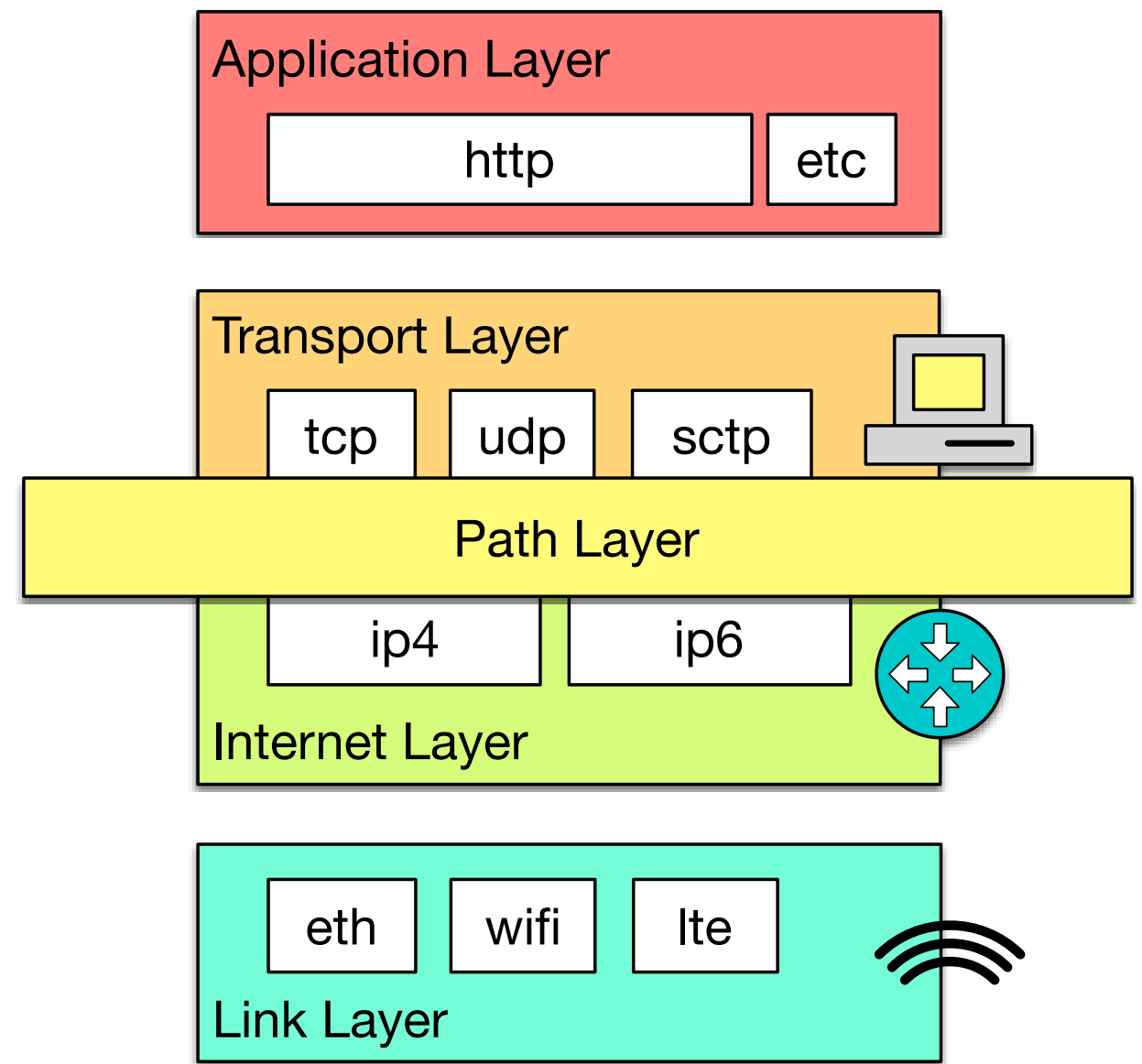
segue

parental advisory explicit
cooperation

Explicit Cooperation

- “Implicit cooperation” between endpoints and middleboxes already widespread in the Internet,
 - where “cooperation” may be the wrong term: some hacks and workarounds are quite hostile.
- We present making this cooperation explicit, and handing control over it to the endpoints, as a way to reduce tension in the end-to-end tussle.
- We declare that everything devices on path don’t need to see (including transport headers) should be encrypted to prevent future unauthorized “implicit cooperation”.

Introducing the Path Layer



Mechanism: Endpoint to Path

- pretty picture here

Mechanism: Path to Receiver

- pretty picture here

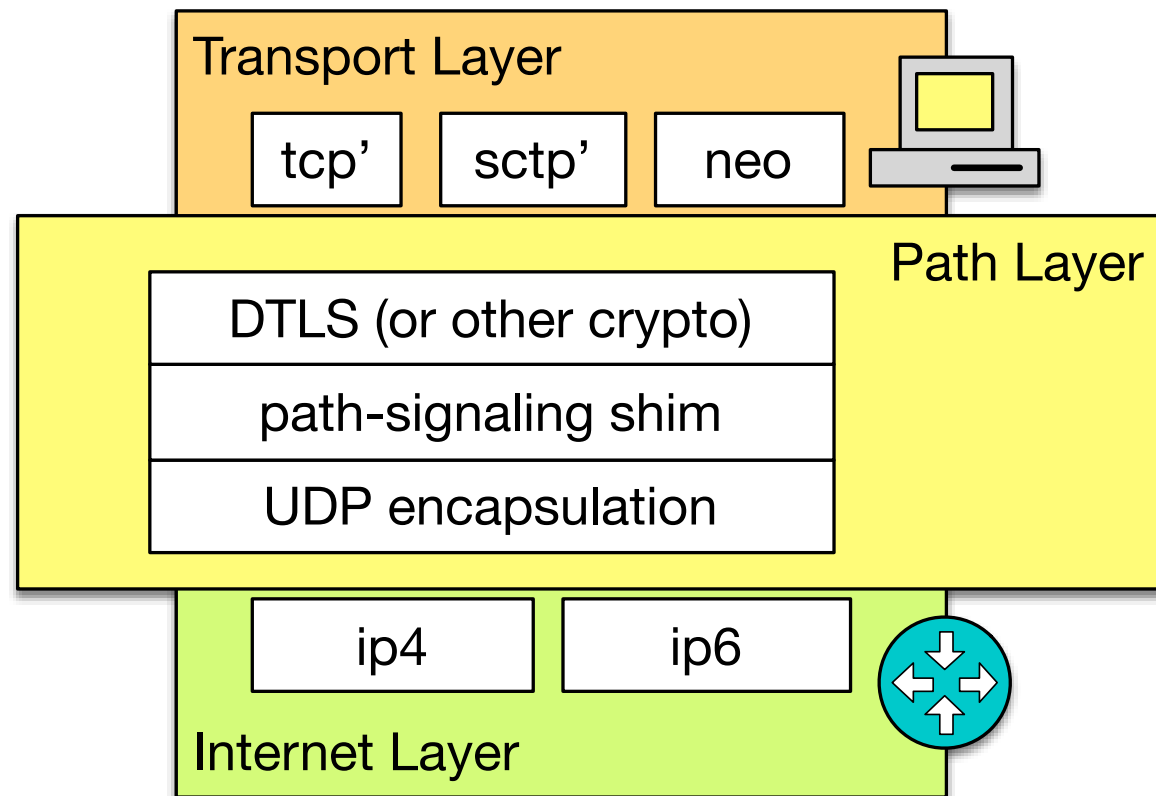
Mechanism: Receiver Feedback

- pretty picture here

Mechanism: Path Direct to Sender

- pretty picture here

The Path Layer runs on UDP



meanwhile, on the
spud@ietf.org list...

Is this a user tracking and network neutrality violation machine?

- Will it be possible for a middlebox to insert user identifiers in the server-bound stream of a client-server protocol?
 - **No**, unless the client specifically requests it.
 - (This is also possible without PLUS, today)
- Will it be possible to use PLUS to require a client to insert a particular kind of metadata into a stream?
 - Bad news: yes; no technical solution exists here.
 - Worse news: also many ways to do this without PLUS.
 - Good news: brings **transparency** to this behavior.

Can we use IPv6 extension headers?

- IPv6 extension headers can be used to implement PLUS mechanisms
 - It is safe to ignore IPv4 in future deployments
 - DO to expose to path: a deployable hack
 - HBH to expose to / communicate with path: architecturally cleaner, deployment issues
- Defined EH already supported in most socket APIs
 - may meet “userspace implementability” req’t.
- but, more impaired in the current Internet than UDP

Can we make transport innovation work without cooperation?

- **draft-herbert-transport-over-udp**
 - Standardize x over DTLS over UDP stack.
 - Fix transport innovation problem with crypto.
 - Break all middleboxes except NATs, as a feature.
- For cases where cooperation is not useful, this is easily realized with PLUS as envisioned if neither endpoint decides to expose anything to the path.

Can we use UDP Options?

- **draft-touch-tsvwg-udp-options**
 - add option space to UDP in a “gap” between the UDP and IP lengths of a packet.
 - Allows optional data to be added to existing UDP applications in a backward compatible manner.
- Proposal: use this option space for PLUS
- Are these the same problem at all?
 - No advantage over a UDP-based shim layer.
 - Needs kernel support: Userland implementation impossible.
 - No fast-path recognition or packet/property association.

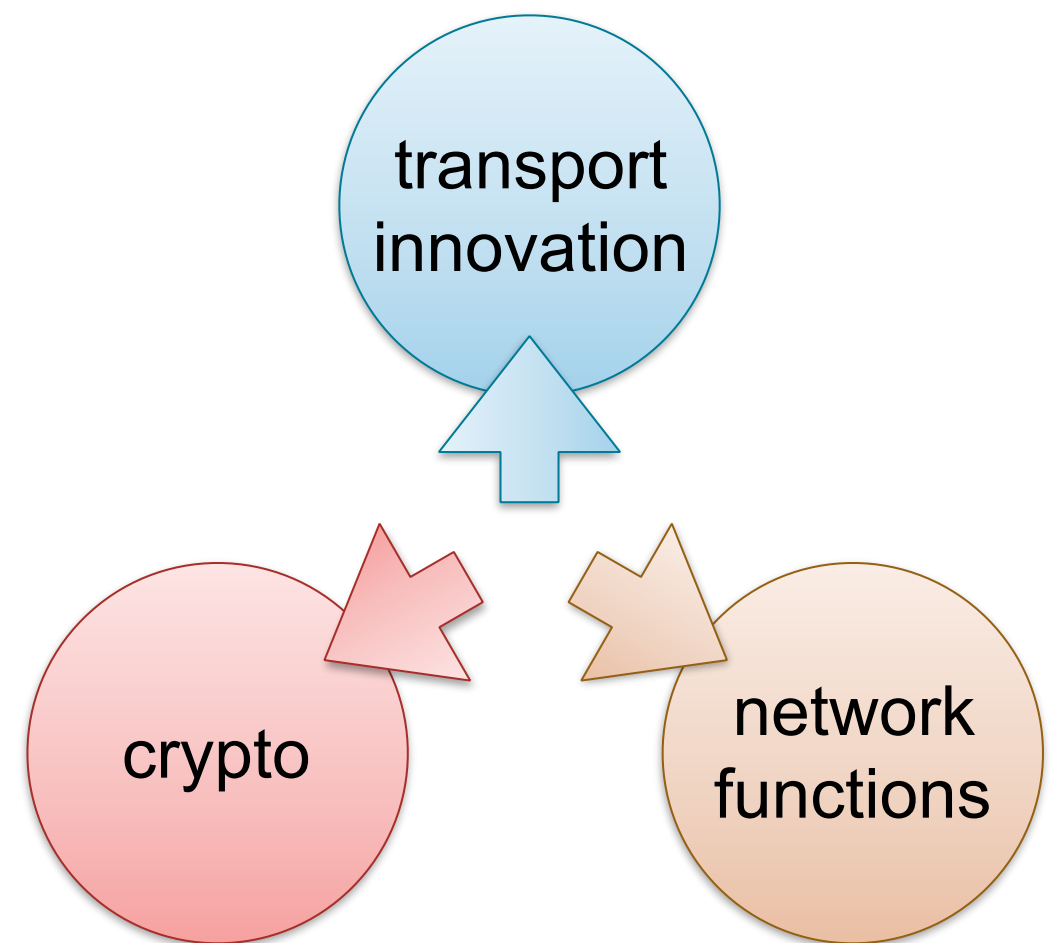
Things we need

- in conclusion...
- explicit coop
- endpoint control
- clear encryption boundary
- deployability

junkyard slides

A three-way tussle

- Transport innovation impossible due to middleboxes
- Crypto to enable innovation breaks in-network functionality
- Some in-network functionality useful, necessary



declarative signaling

Question:

(in this talk)

- If we believe...
 - the set of requirements we think we have...
(draft-trammell-spuq-req as starting point,
for elaboration in a future WG)
 - describe constraints on a solution we think...
 - addresses the problems we want to solve...
(see Ted's and Natasha's talks,
use draft-kuehlewind-spud-use-cases as starting point,
for elaboration in a future WG)
- ... then can we show that a technical approach exists
to implement these requirements?