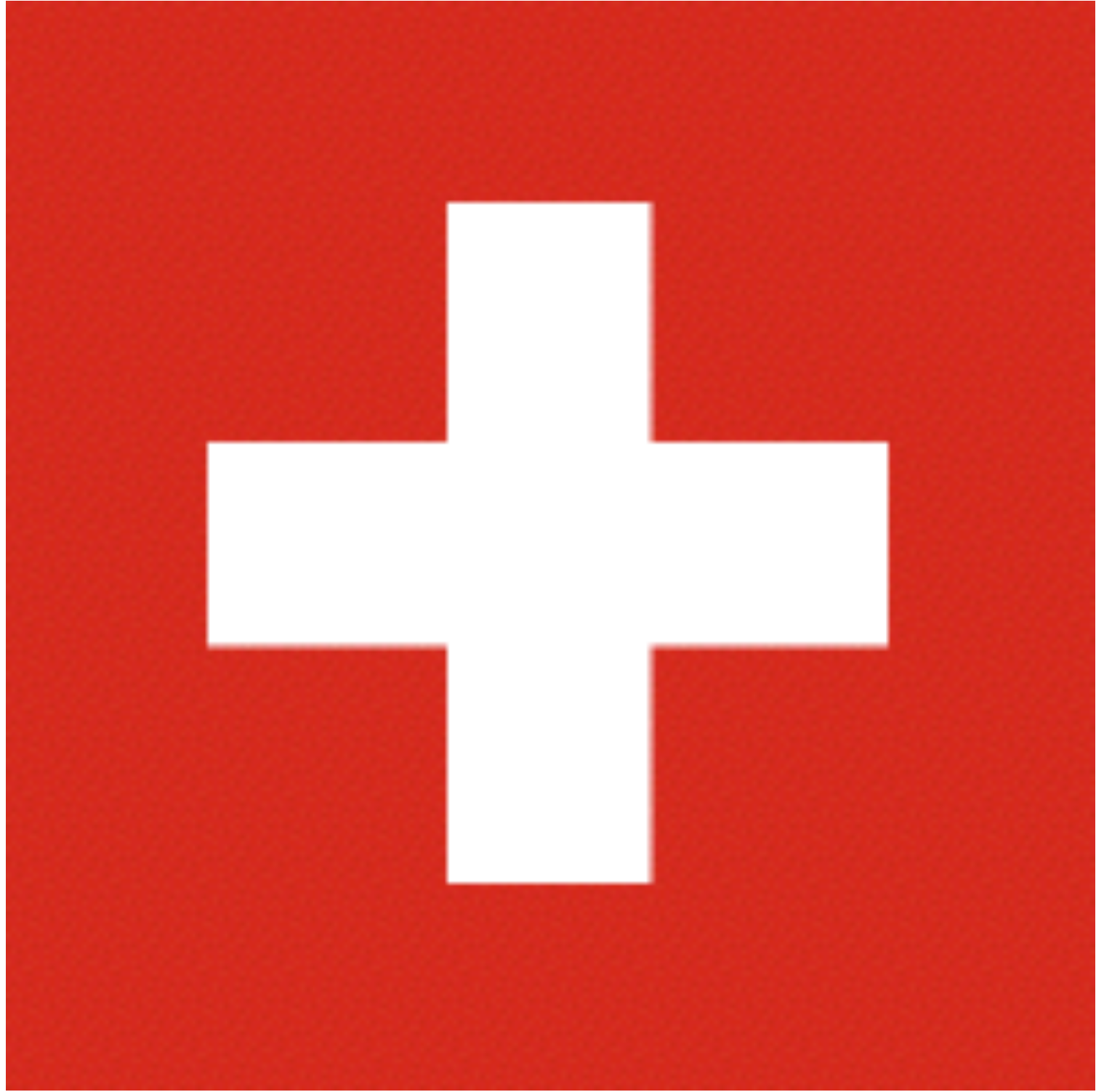


# Path Layer UDP Substrate (PLUS) Technical Considerations

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



**P A R E N T A L**

**ADVISORY**

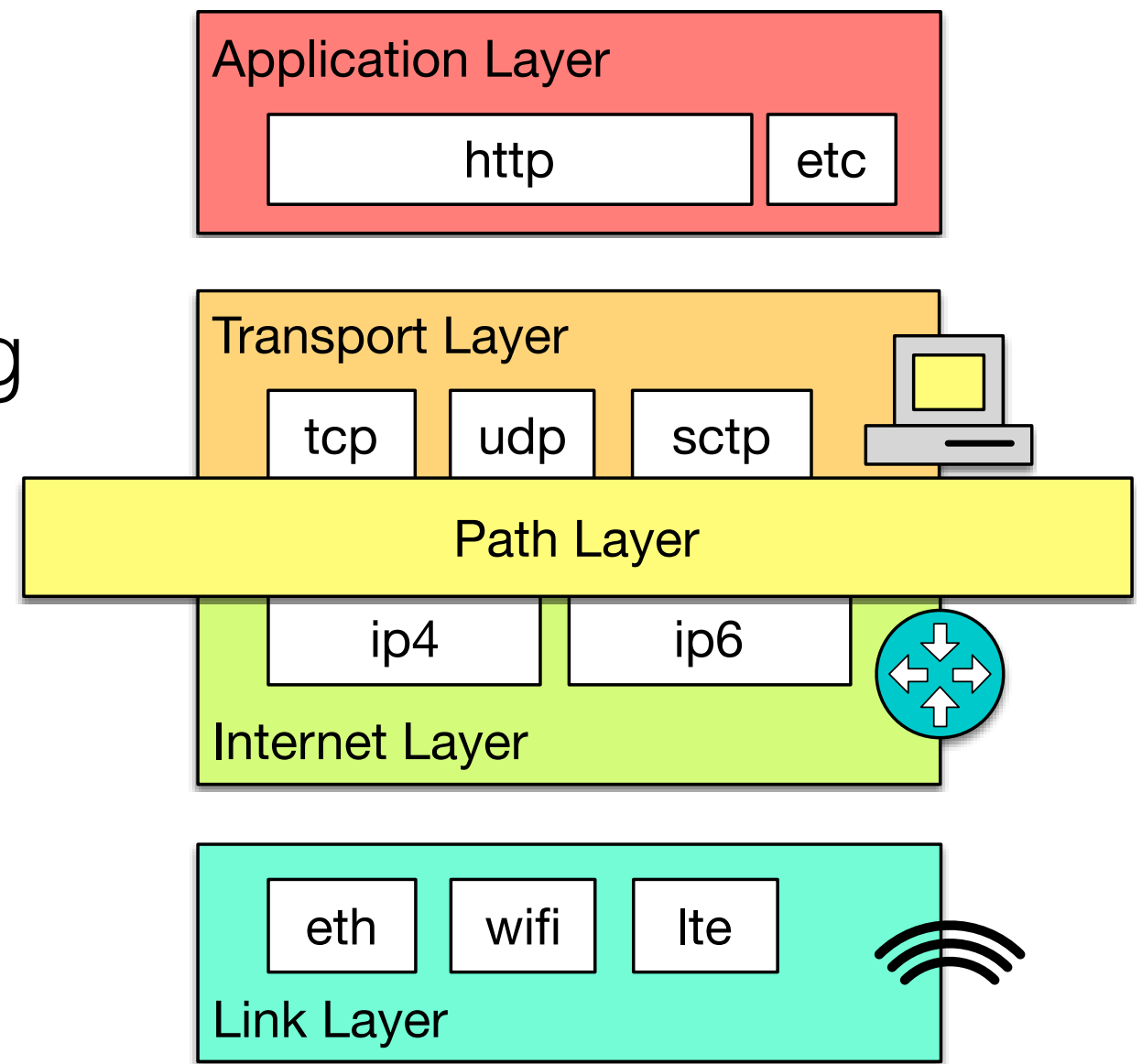
**EXPLICIT COOPERATION**

# Explicit Cooperation

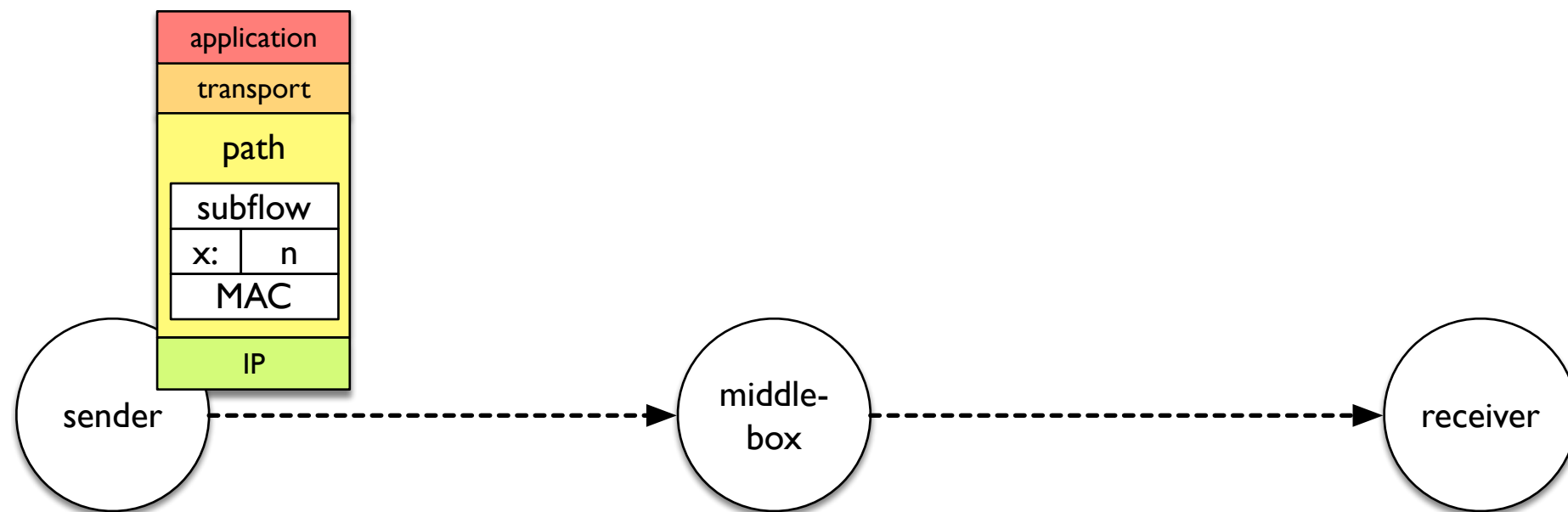
- “Implicit cooperation” between endpoints and middleboxes **already widespread in the Internet**,
  - where “cooperation” may be the wrong term: some hacks and workarounds are quite hostile.
- **Explicit cooperation** under **endpoint control** may be a way to reduce tension in this tussle
  - Declarative, advisory signaling with no trust required between endpoint and path.
- **Encrypt everything devices on path don’t need to see** (including transport headers), to prevent future unauthorized “implicit cooperation”.

# Three and a half mechanisms to make the path layer explicit

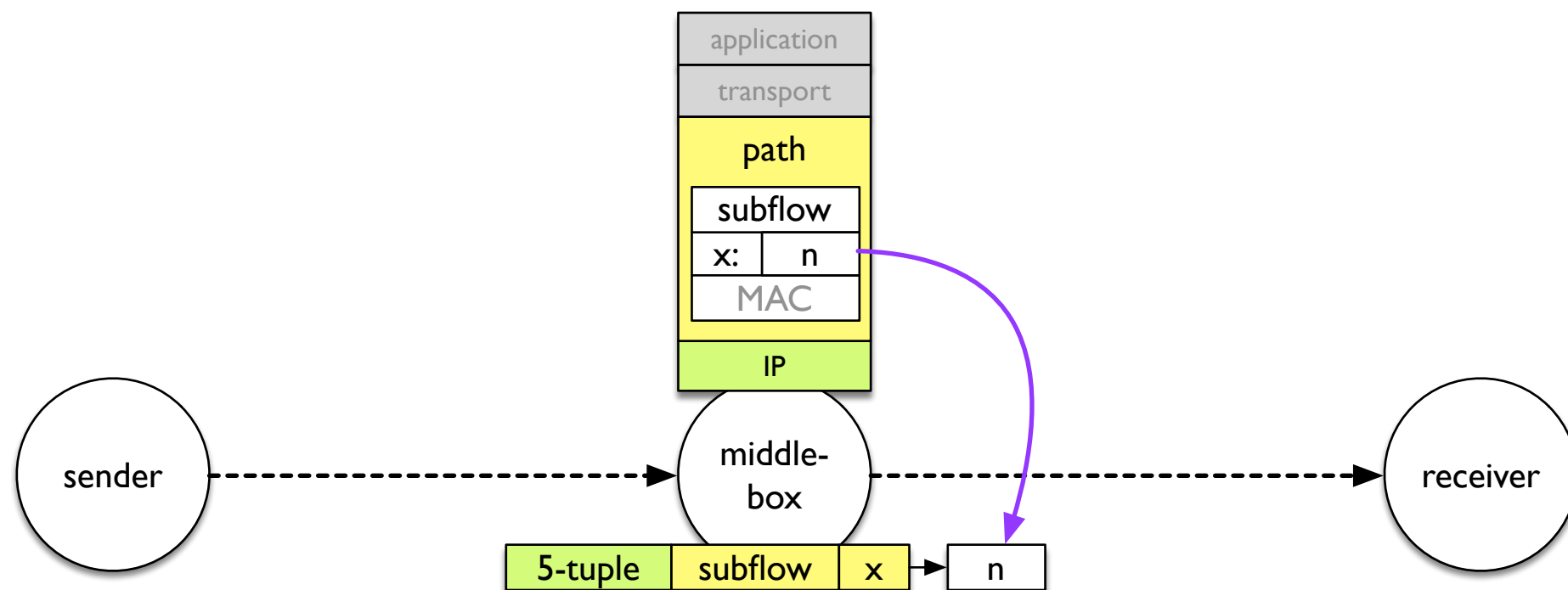
- Sender – Path Signaling
- Path – Receiver Signaling
  - with encrypted feedback to sender
- Direct Path – Sender Signaling



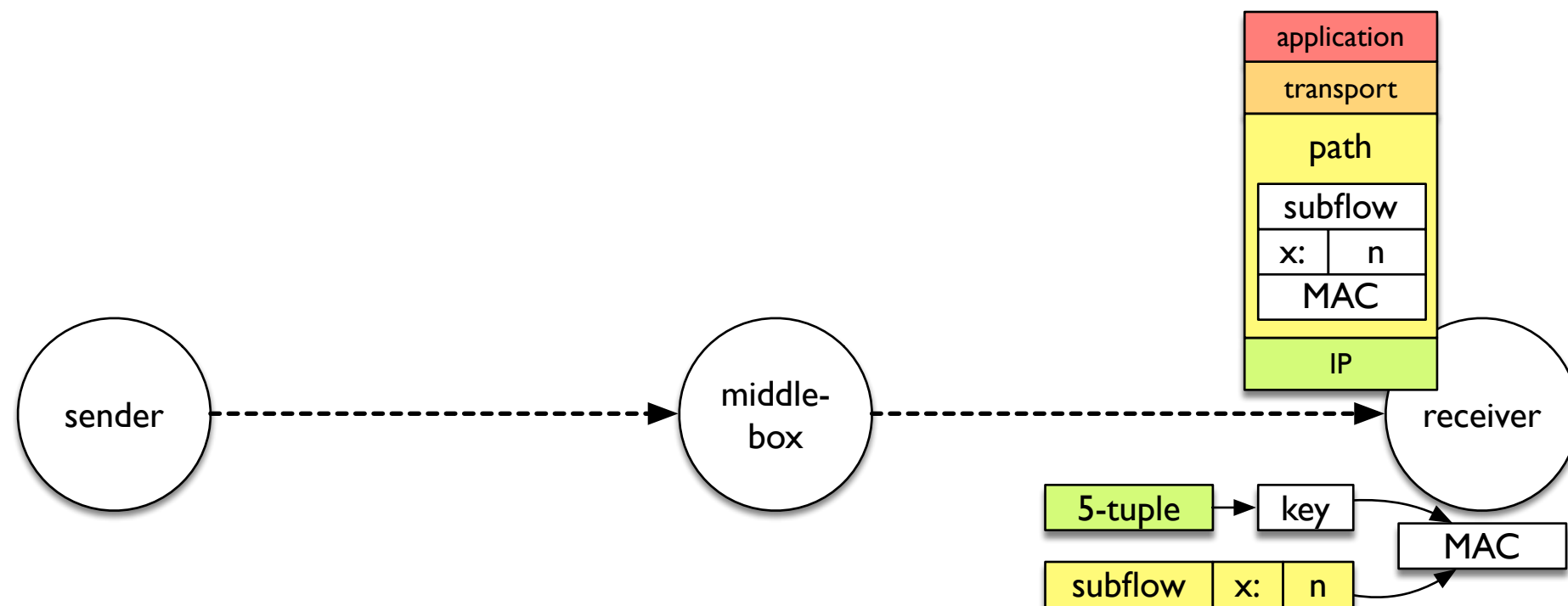
# Sender to Path (sender-side)



# Sender to Path (on-path)



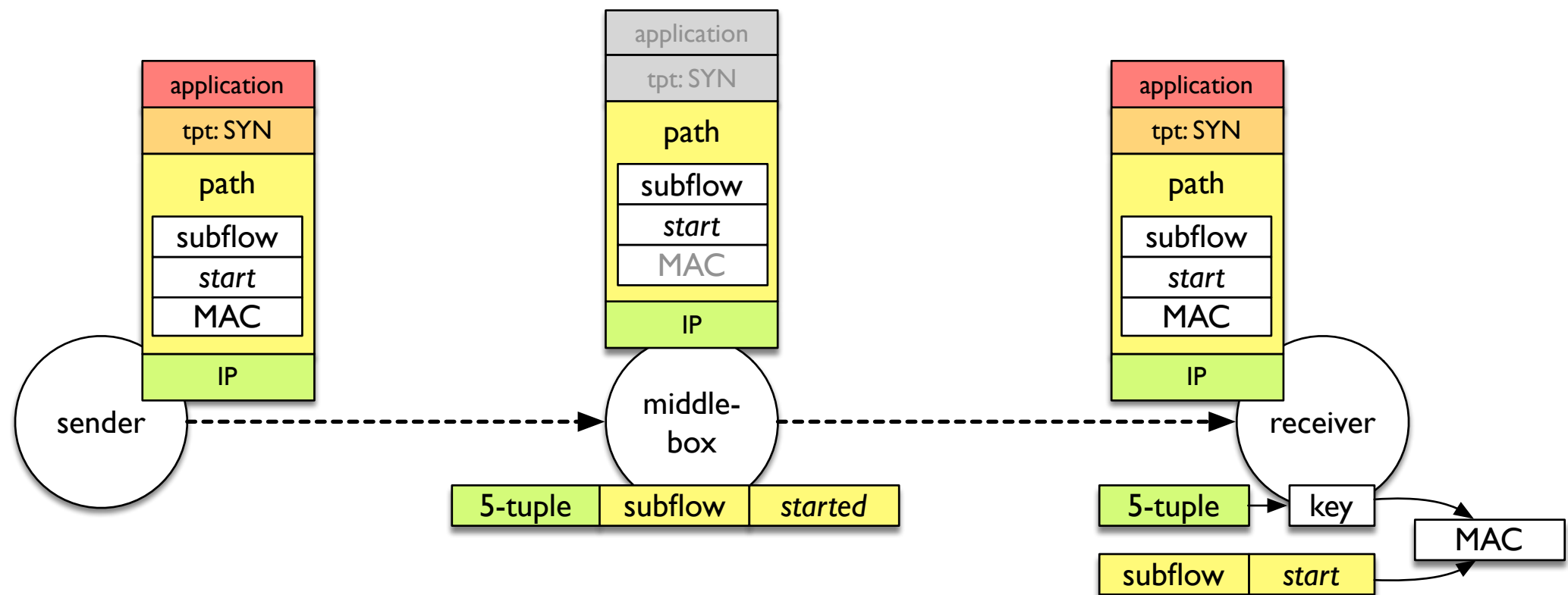
# Sender to Path (receiver-side)



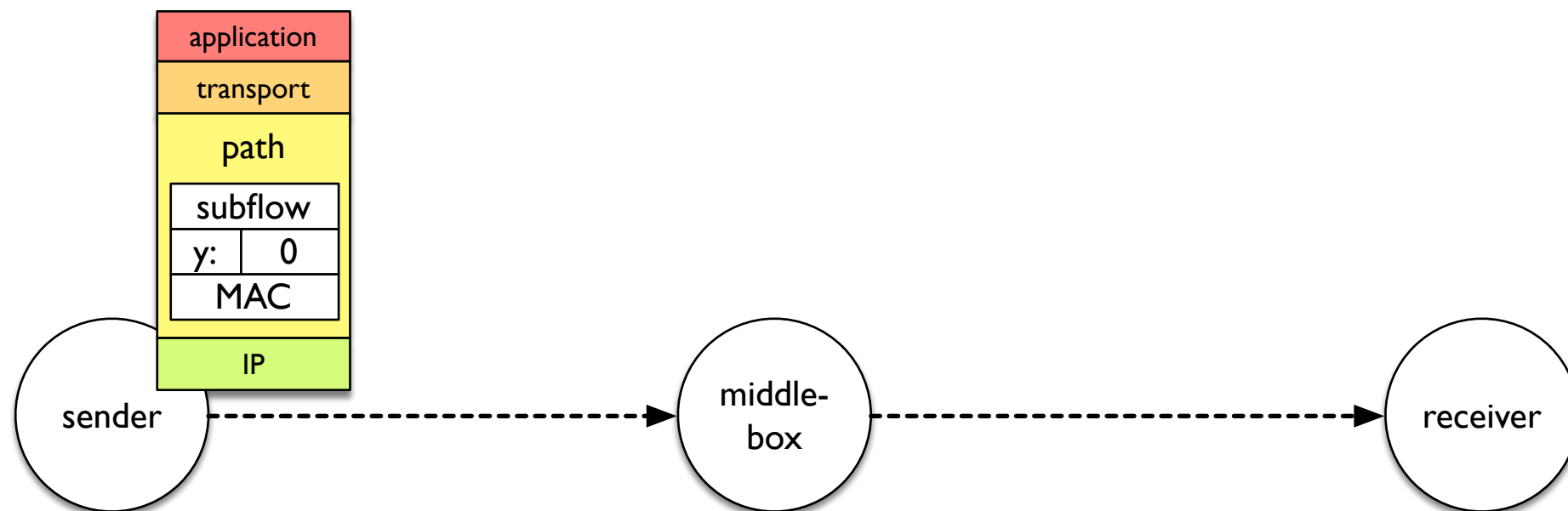


# Sender to Path

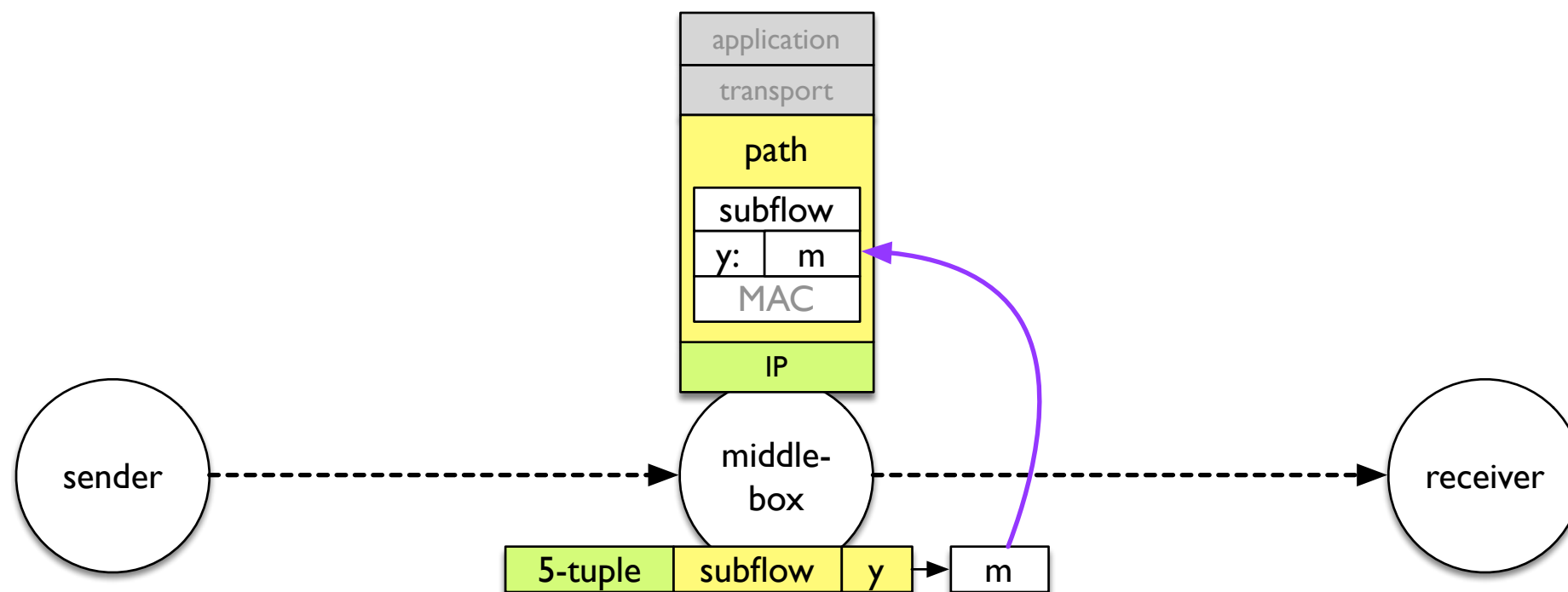
## Transport State Signaling



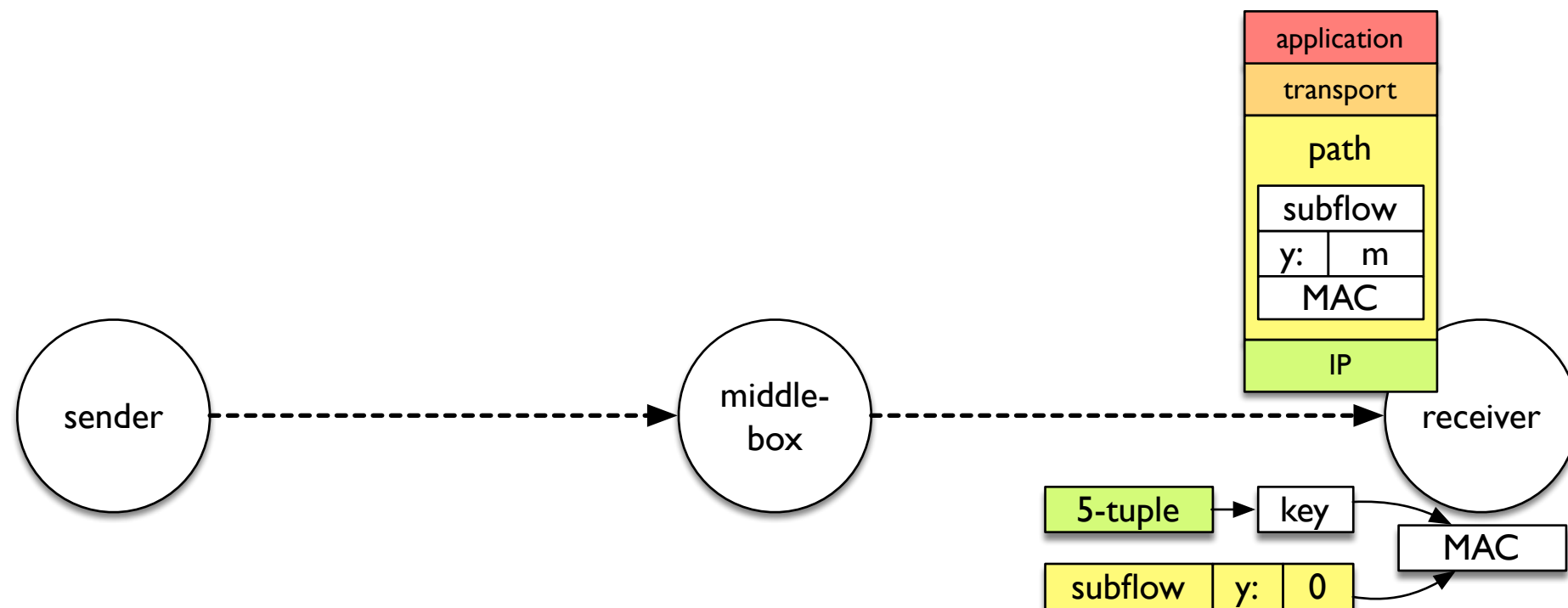
# Path to Receiver (sender-side)



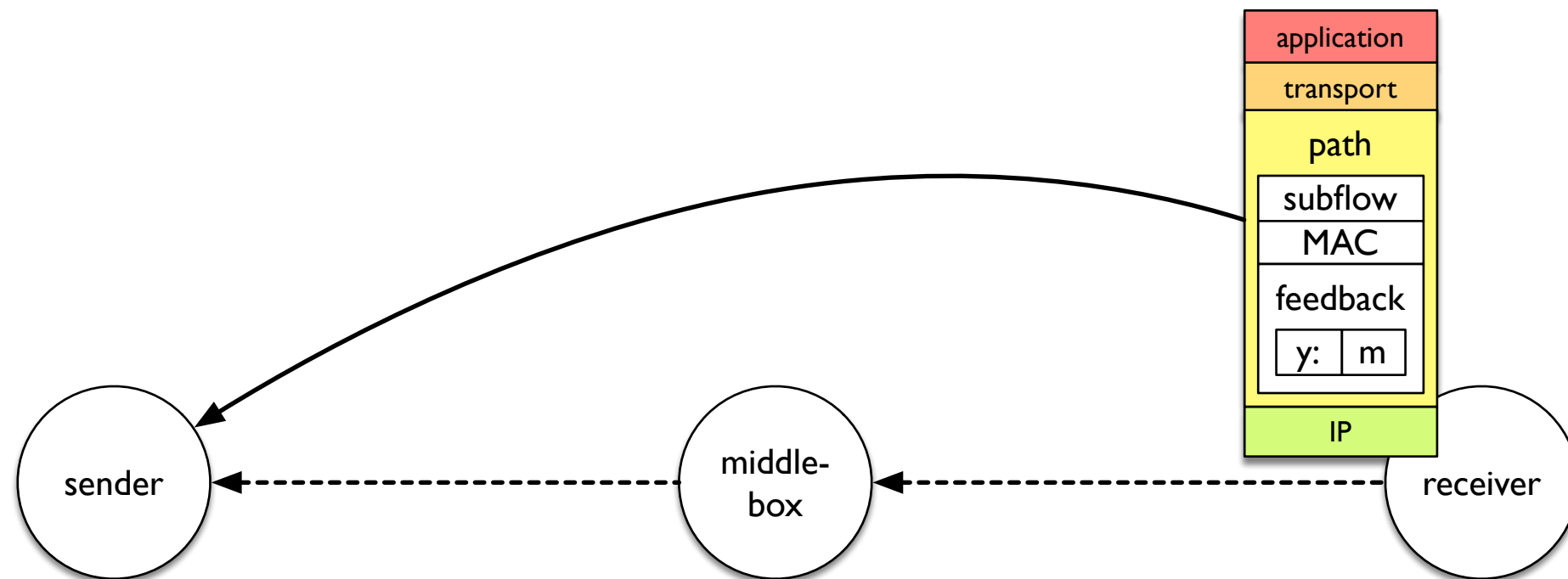
# Path to Receiver (on-path)



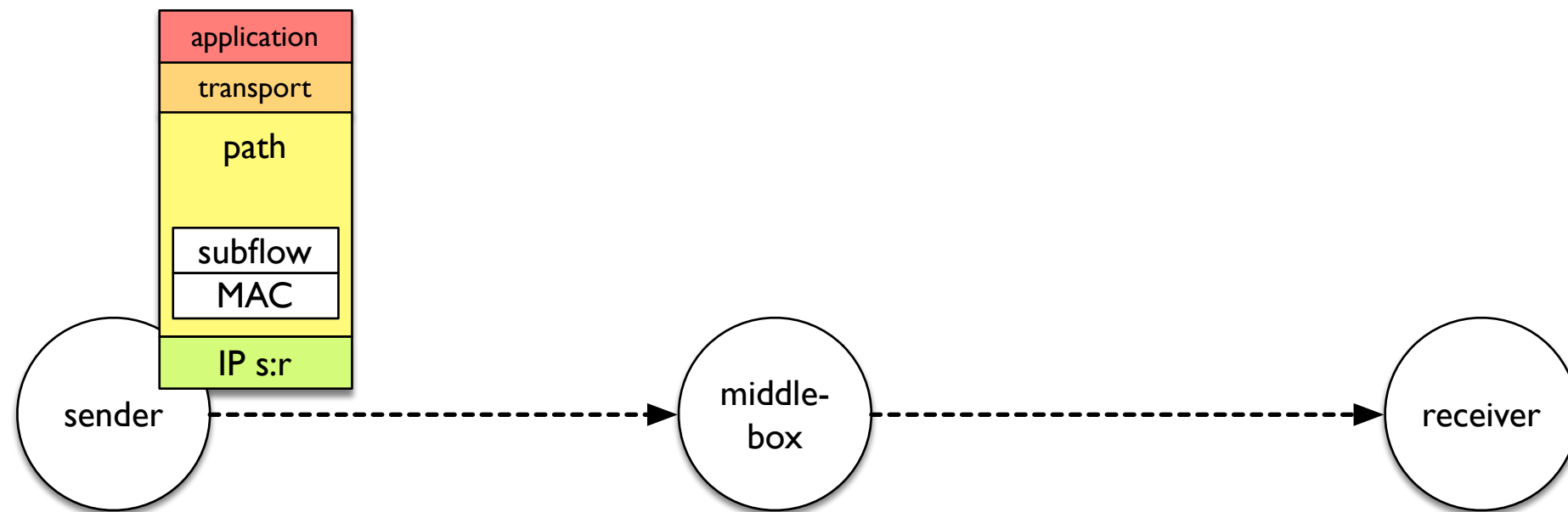
# Path to Receiver (receiver-side)



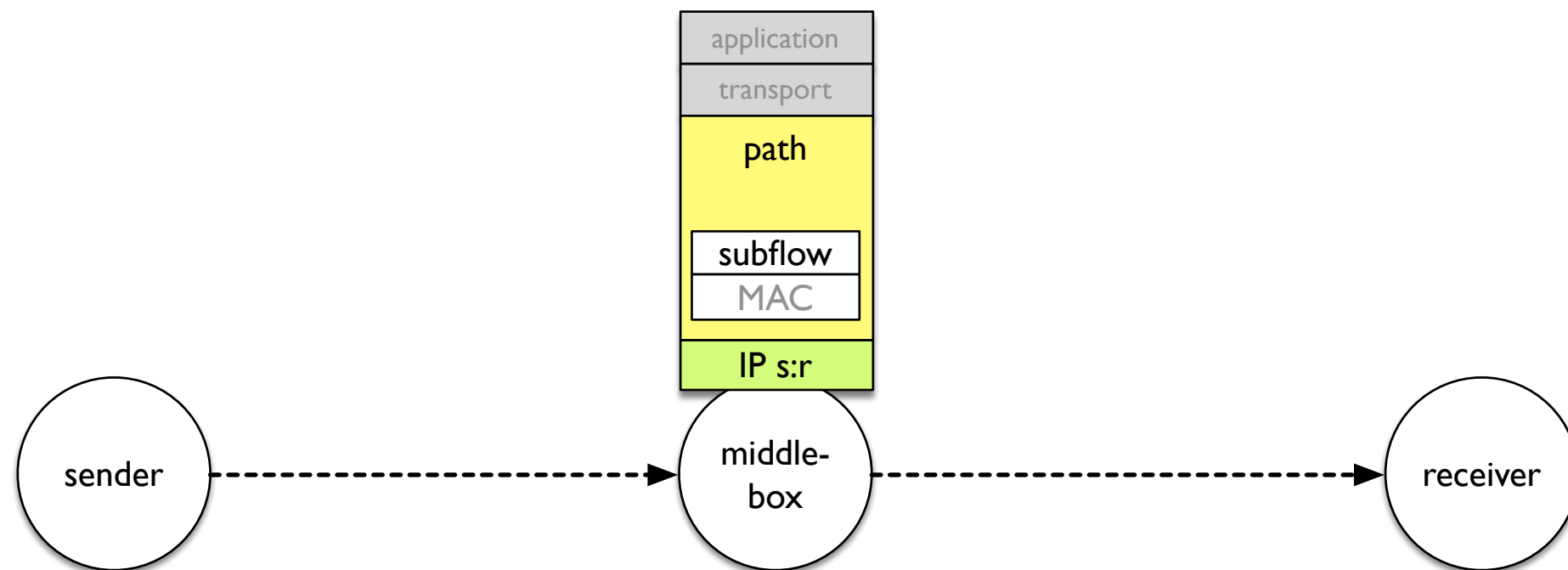
# Receiver Feedback



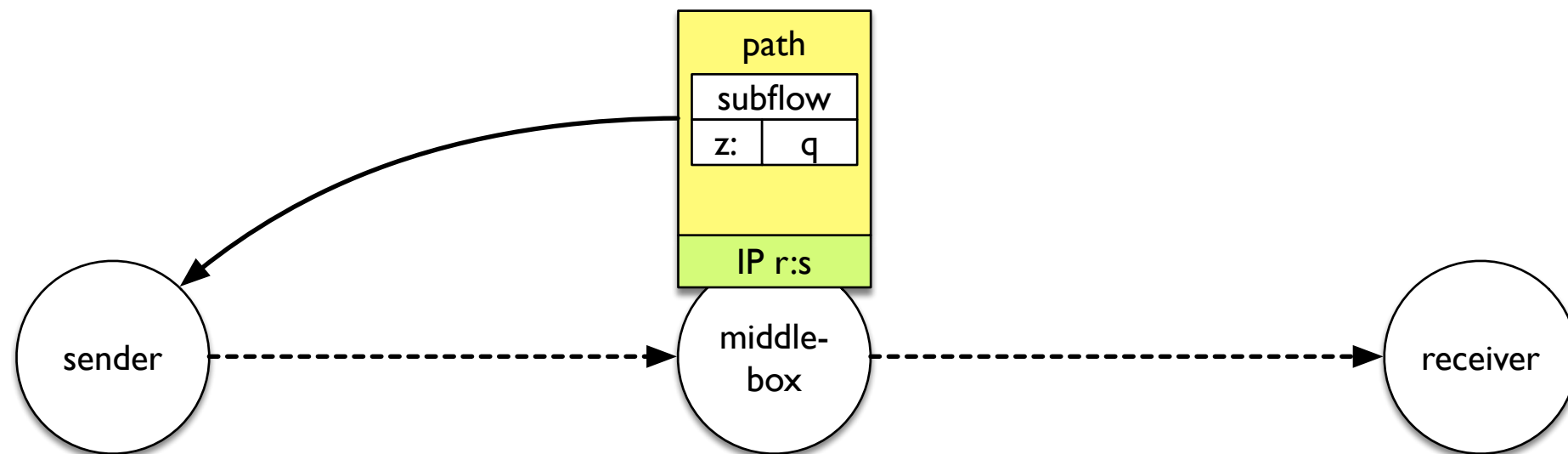
# Path Direct to Sender (sender-side)



# Path Direct to Sender (on-path)

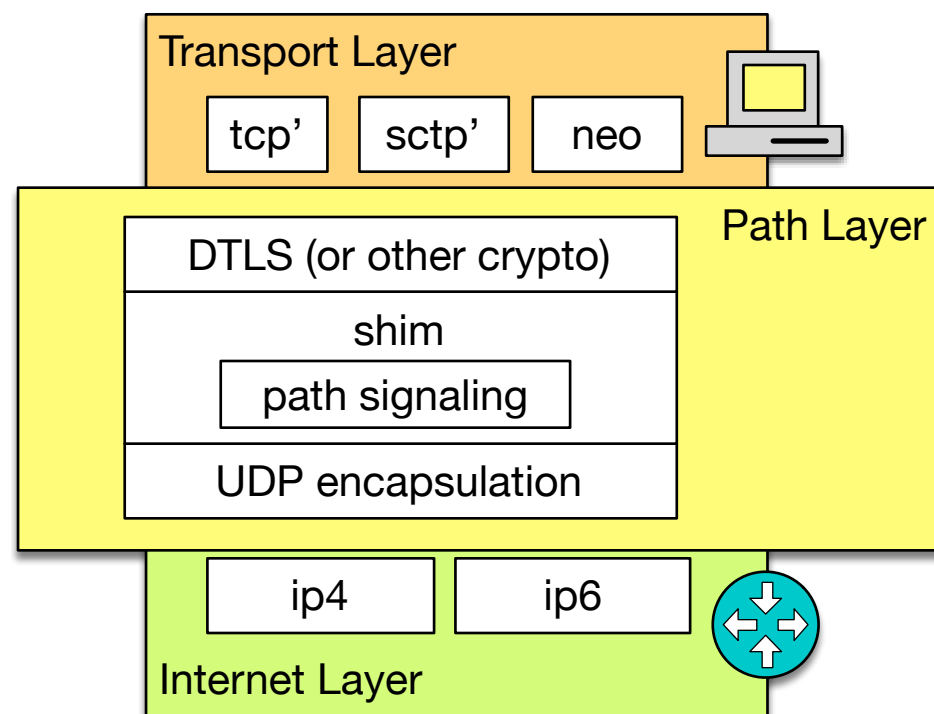


# Path Direct to Sender (feedback)





# Anatomy of the Path Layer



- UDP encapsulation
  - userspace implementation
  - ports for NAT
  - ~95% deployable today
- encoding for signaling mechanisms
- crypto to protect transport headers and above

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

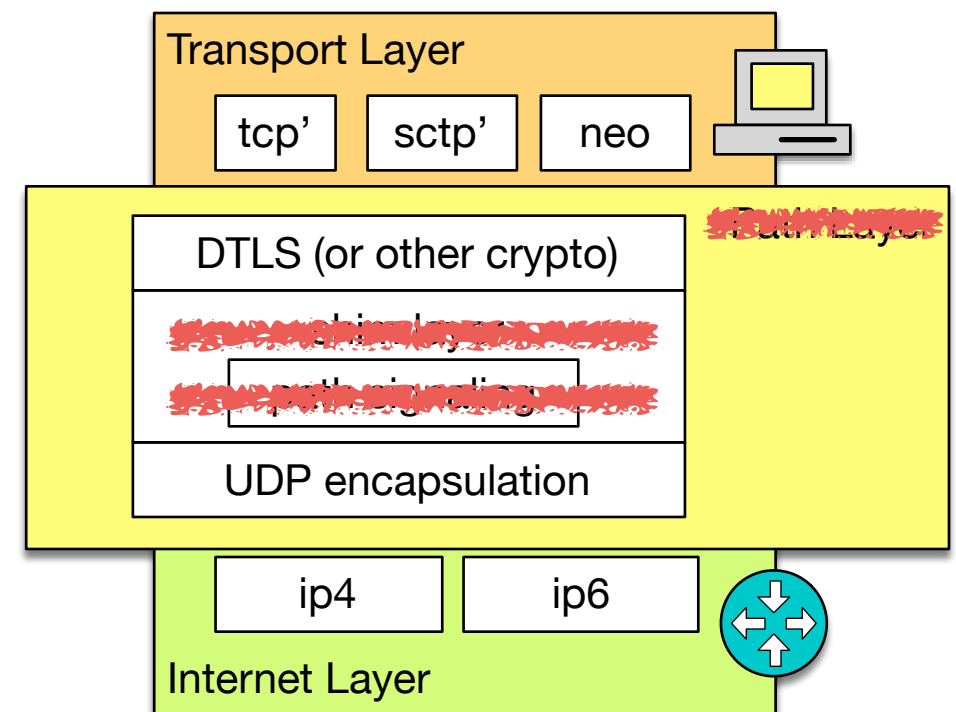
# Is this a user tracking and network neutrality violation machine?

- Will it be possible for a middlebox to use PLUS to insert user identifiers in the server-bound stream of a client-server protocol?
  - **No**, unless the client specifically requests it.
  - (Note: possible without PLUS, out of band, 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: PLUS brings **transparency** to this behavior.

# Can we make transport innovation work without explicit cooperation?

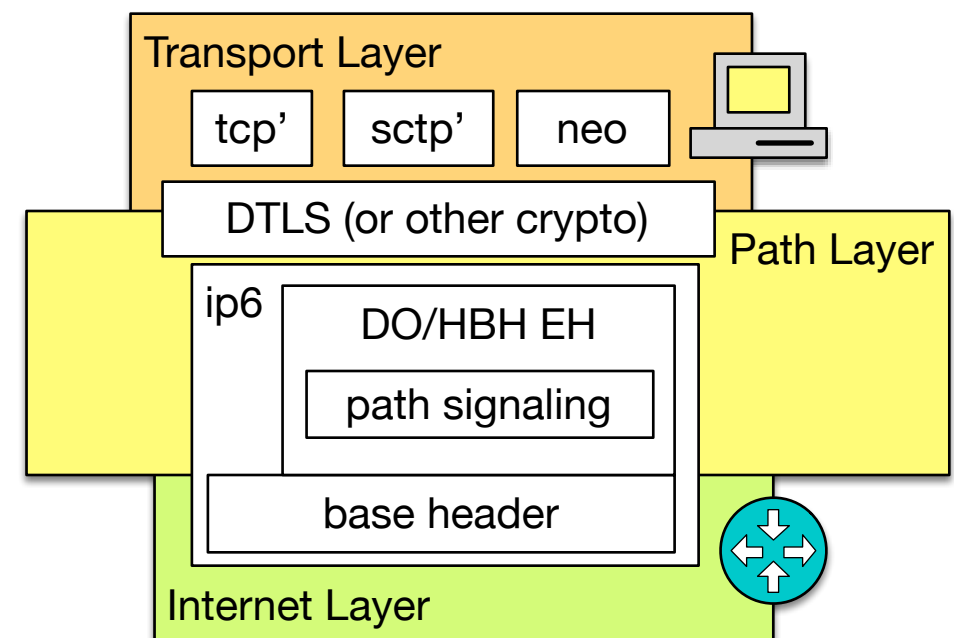
- **draft-herbert-transport-over-udp**

- x over DTLS over UDP.
- Make transport innovation possible with crypto.
- Breaks middleboxes.
  - This is a feature.
- Equivalent to PLUS when neither endpoint decides to expose anything to the path.



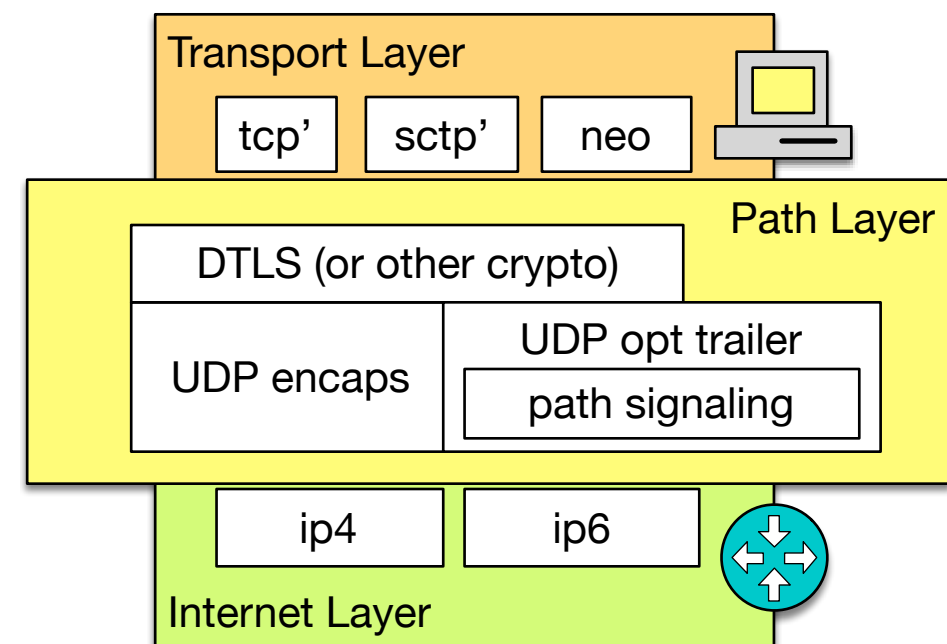
# Can we use IPv6 extension headers?

- IPv6 extension headers can be used to implement PLUS mechanisms
  - Ignore IPv4 in future deployments
  - DO to expose to path: hack, but more deployable
  - HBH to communicate with path: cleaner, but less deployable
- DO/HBH already supported in most socket APIs
- Together with draft-herbert-transport-over-udp, meets PLUS requirements?
- But: more impaired than UDP  
(draft-ietf-v6ops-ipv6-ehs-in-real-world-02)



# 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?
  - Must be in-kernel: no userspace implementation.



Do we need  
to choose now?

and in conclusion...



# Things we need

- A mechanism for making widespread cooperation between endpoints and middleboxes explicit
- Endpoint control over explicit cooperation
- A clear boundary between what the path can see and what it cannot, enforced by encryption
- A design for this facility that deploys on the endpoints from day zero
- All this without requiring a trust relationship between the endpoints and middleboxes