Proxying Encrypted Transports?

IETF 103, TSV AREA

5 November 2018

Performance Enhancing Proxies

- In the olden days of TCP there used to be a family of middleboxes called PEP
  - RFC 3135 "Performance Enhancing Proxies intended to mitigate link layer degradation"
  - RFC 3449 "TCP Performance Implications of Network Path Asymmetry"

Problems PEP solve

- Improve performance of the transport when crossing low-quality links (high-loss, variable latency) or highly asymmetric links (narrow upstream bandwidth)
  - Examples: satellite, mobile netork, wireless
- Put a PEP on one, sometimes both, end(s) of the weak link, let it split or spoof the TCP connection and perform a number of tricks, including:
  - ACK manipulation (suppression, reconstruction, compaction)
  - rwin size manipulation
  - Header compression
  - split ACKing

Mobile example

- Quite often, the mobile link “freezes” from the point of view of the transport while its data link protocols are busy retransmitting after a loss[1]
- The freeze may last longer than the typical RTT $\rightarrow$ sender thinks the path has lost its segment(s) $\rightarrow$ retransmits
- This is the cliché of a spurious retransmission: the loss has been repaired by the link layer in the meantime
- Here the PEP provides impedance matching at the point where wired and mobile links meet: it ACKs the sender in lieu of the UE, buffers the data and smooths it out to the UE when the link is back to normal
- Without the PEP, the sender would throttle (throughput & goodput decrease)

[1] This is not a congestive loss, it's just wireless physics.

Problems PEP create

- Typical middlebox, so the usual caveat applies (RFC6182):

  > [. . .] All these middleboxes *optimize current applications at the expense of future applications.* In effect, future applications will often need to behave in a similar fashion to existing ones, in order to increase the chances of successful deployment. Further, *the precise behavior of all these middleboxes is not clearly specified,* and implementation errors make matters worse, raising the bar for the deployment of new technologies

- Nasty when it completely breaks the end-to-end path, but even nastier when the breakage is only partial: e.g., when it ends up "eating" unknown (to the box) TCP options

PEP and encrypted traffic

- Transport header protection (e.g., QUIC, IPsec ESP / AH) means transport headers can't be modified and/or forged, therefore PEP is completely inhibited:
  - NO header compression
  - NO ACK tricks
  - NO rwin tricks

PEP and encrypted traffic (cont)

- Encryption inhibits PEP - and therefore solves the problems PEP create -, but it doesn't make the problem PEP address go away. . .

Questions

- Is PEP still a valid approach?
  - Are the problems it solves still relevant?
  - Can we solve the same problems using other techniques?
- If PEP are needed:
  - How does a "modern" PEP look like from the perspective of the endpoints?
  - Should new transports take PEP into consideration at design time?

Backup

Helium / HiNT and PEP

- There seems to be a space that is worth exploring to understand what can be achieved with HiNT / Helium in this context, in particular:
  - Does it provide the right primitives? If not, what is needed?
  - Understanding the interaction between the tunnelled and the outer congestion controllers under diverse links and user mobility models
  - In-band control channel, how can it be used?