TCP Use TLS Option

Eric Rescorla

Mozilla

ekr@rtfm.com

Background: TLS over TCP

- TLS over TCP is ubiquitous
  - Probably the most deployed Internet security protocol Widely implemented
  - Heavily analyzed and reasonably well understood
- Hard to coordinate
  - Servers which are expecting application data choke on TLS `ClientHello`

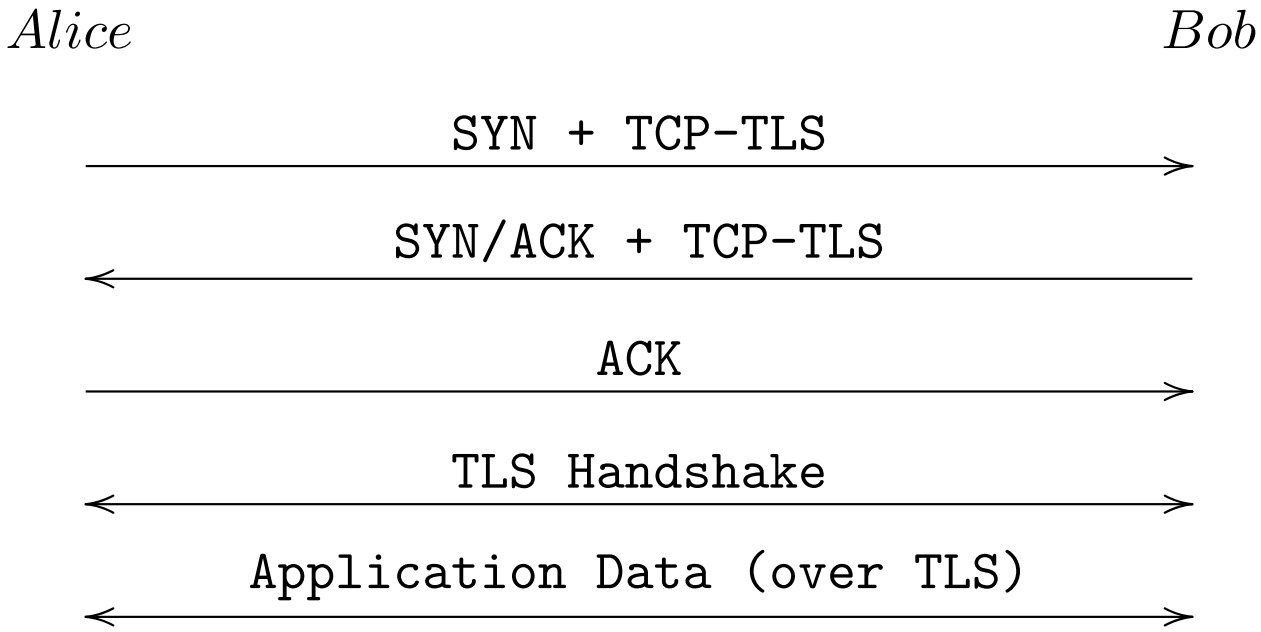
Some Existing Coordination techniques

- External signal to the client (e.g., `https:`)
- Separate ports
- Manual config
- DNS signaling
- Extend the application layer protocol (STARTTLS)
- None of these lend themselves to opportunistic deployment

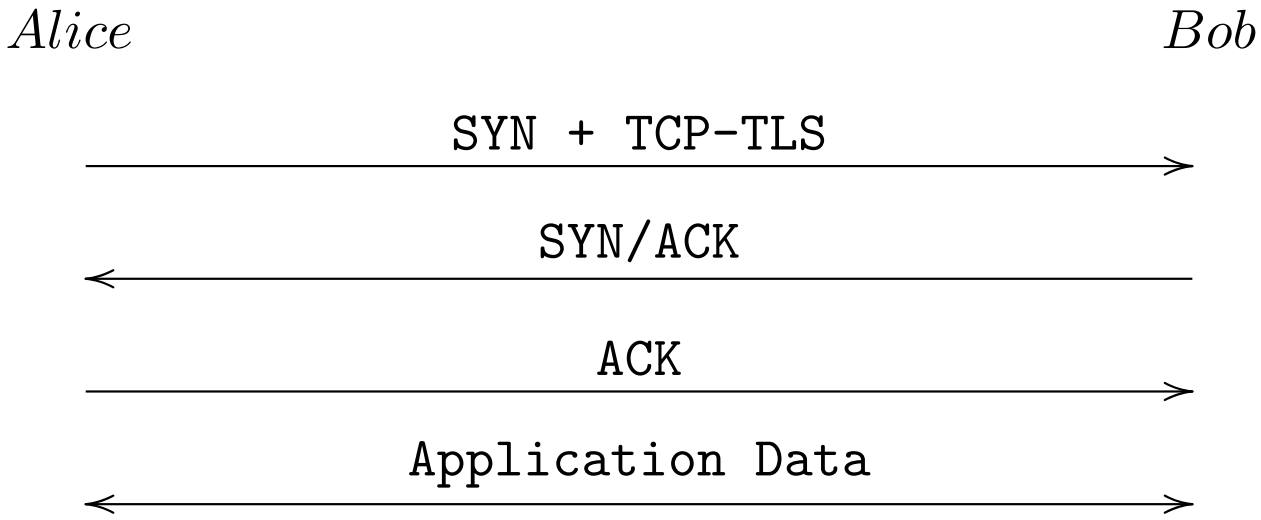
Problem Statement

Add the minimum necessary machinery to TCP to let it opportunistically negotiate TLS when both sides want to.

TLS TCP Option



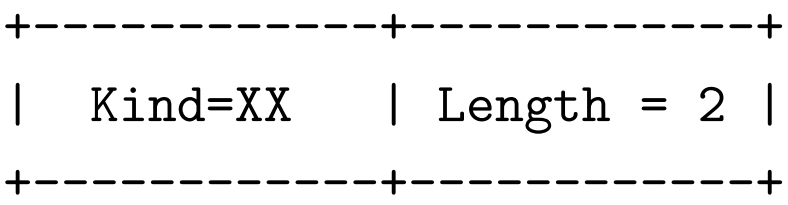
Bob Doesn't Support TLS



What do we need to signal?

- That I want to do TLS
  - Signaled by option present
- TLS roles (client vs. server)
- Obvious for non simultaneous open case
  - Let's ignore simultaneous open (or do an optional tiebreaker)

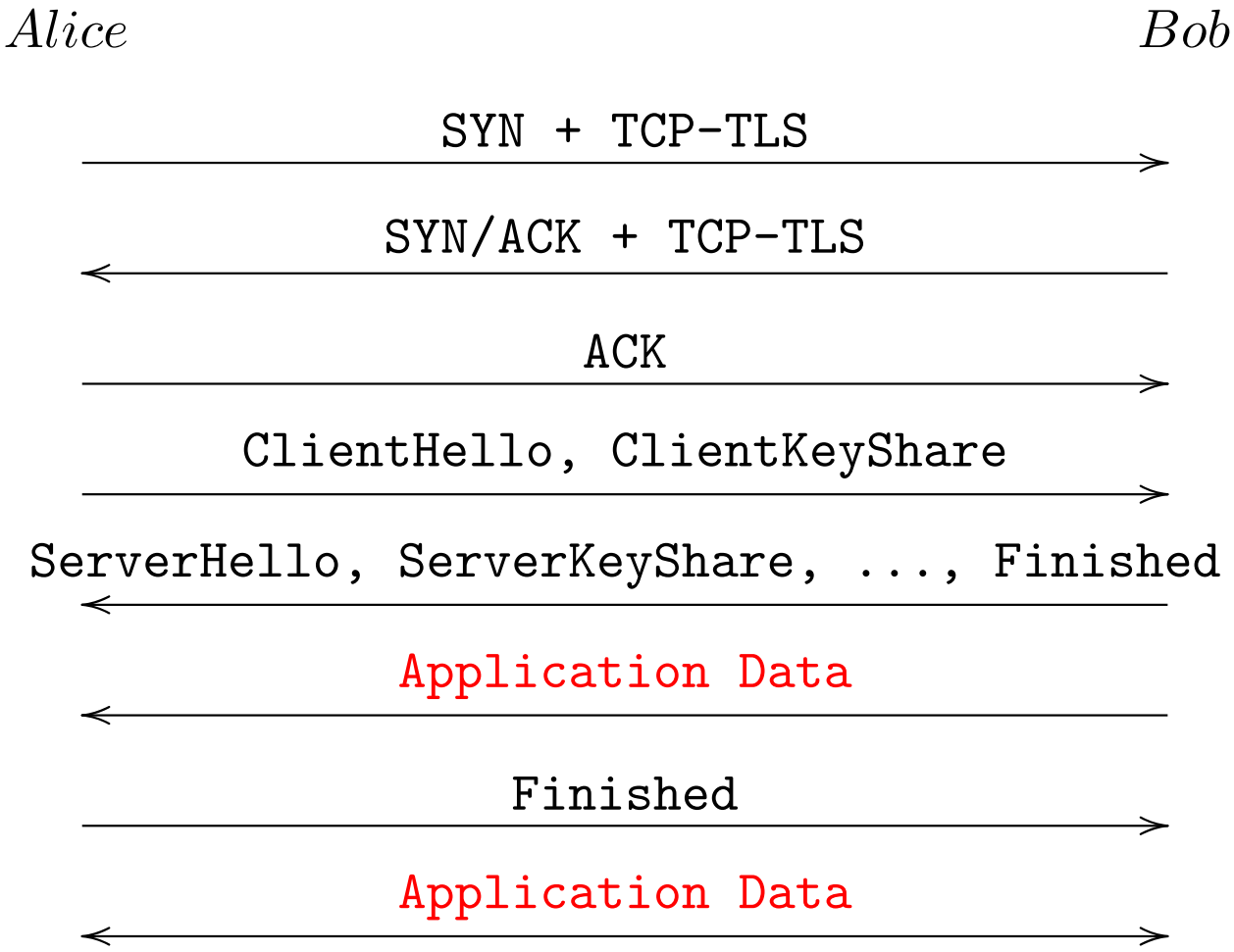
Minimal Option



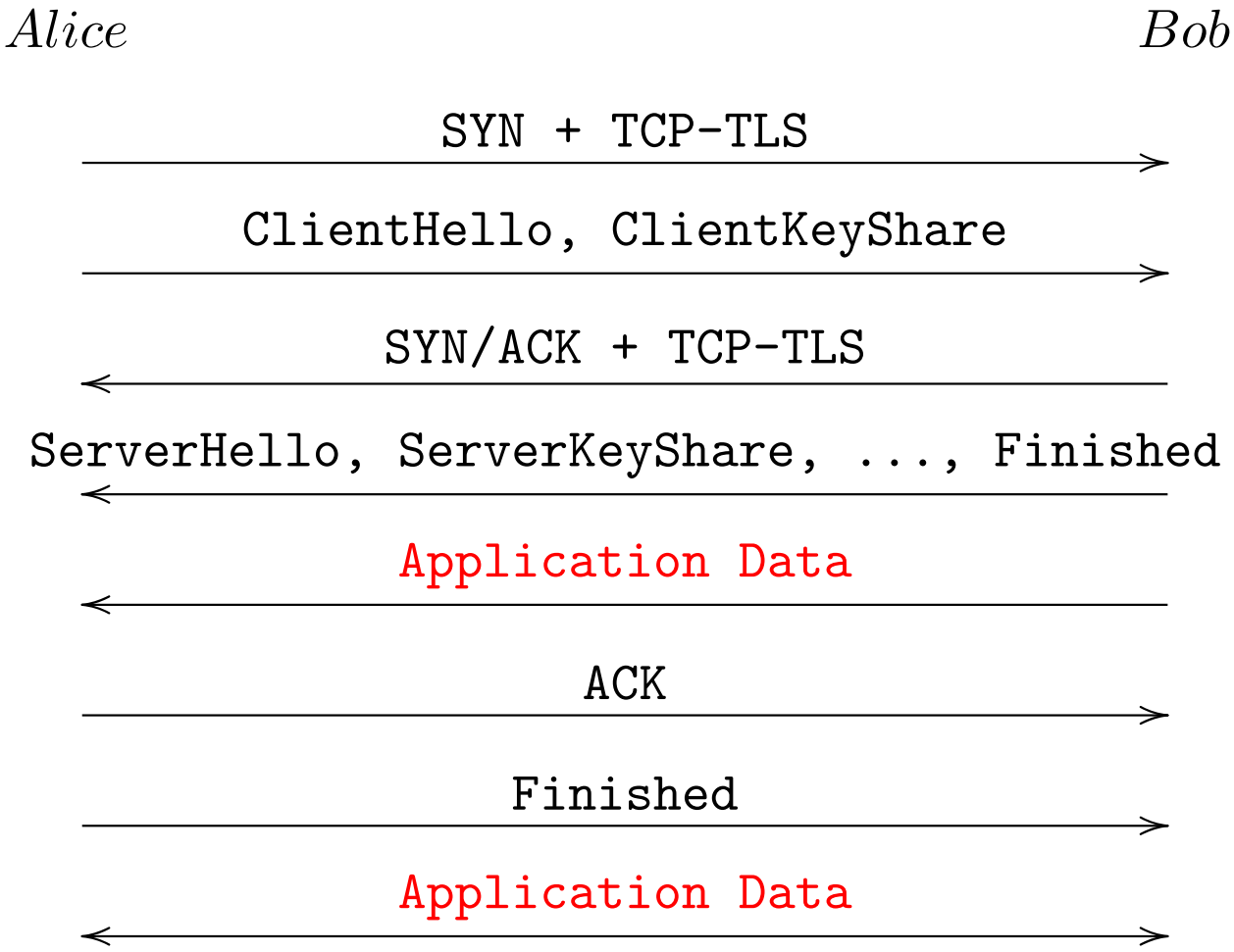
End of Connection (not in draft)

- TLS already has a connection close (`close_notify`)
- Half-closed state not supported
  - Could modify TLS if needed

Setup latency (detail, TLS 1.3, no data in SYN)



Setup latency (detail, TLS 1.3, TFO or data in SYN)



TLS Complexity/Profiling

- TLS is ~~complicated~~powerful
  - Though TLS 1.3 is removing a lot of stuff
- The necessary subset for this is not that complicated
- And it's a pretty obvious subsetting exercise

Comparison to Integrated Designs (e.g. tcpcrypt)

- Advantages
  - Easy to specify and implement
  - Leverage the work that has alredy gone into TLS
    - Looks like existing TLS over TCP on the wire
- Disadvantages
  - Imports TLS history; may want to profile
  - Less optimized, especially when you want to do anti-DoS
  - TLS records can span segment boundaries
    - Easy to manage with attention to MTU

Questions?

Backup Slide: Handling Simultaneous Open