TLS 1.3

draft-ietf-tls-tls13-19

Eric Rescorla
Mozilla

ekr@rtfm.com
Agenda

- Status
- WGLC issues
- Timeline
Status

- In WGLC#2 with: draft-ietf-tls-tls13-19
  - Modest changes from -18 (more later)
- Quite a few interoperable implementations
  - draft-18 in Firefox Beta (NSS), Chrome Beta (BoringSSL), Cloudflare, OpenSSL, Facebook (Fizz), OpenSSL
  - draft-19 under development with partial interop
Additional Derive-Secret stage to key schedule

... 
| 
| v
| Derive-Secret(., "derived secret", ")
| 
| v
| (EC)DHE -> HKDF-Extract = Handshake Secret
| 
| +-----+ Derive-Secret(., "client handshake traffic secret", ClientHello...ServerHello)
| | = client_handshake_traffic_secret
| ... 

- Added before each HKDF-Extract from a non-0 salt
- Restore extract/expand parity
- Prevent theoretical concern about collisions from chosen “IKM” values
Hash the context value in exporters

- The context value is limited to 255 bytes
- But the context length in 5705 is 16 bits
- Consensus: hash the value before feeding to HKDF
Hash ClientHello1 in transcript when doing HRR

- This makes stateless HRR easier
- Also insert the selected cipher suite in HRR
Add an additional Derive-Secret stage to exporters

- The EKM can be used to compute any exported value
  - This means if you need a long-term exporter the EKM is a threat to other exported value

- Solution: domain separate exporters on label

  $\text{HKDF-Expand-Label(Derive-Secret(Secret, label, ""),}$
  
  "$\text{exporter", Hash(context\_value), key\_length)}$
Change `end_of_early_data` to be a handshake message

- It was goofy to have it an alert
- All other state transitions are handshake messages
- Spec isn’t very clear on how this fits into the transcript
  - Consensus answer:
    - ServerFinished, EOED, [Client Certificate]...
  - -20 will be clearer
PR#768: DH Key Reuse Considerations

- Not that confident of the analysis
- We don’t really want to encourage re-use

- Proposed resolution: drop
PR#762: Short Headers

- Concerns about interop
  - Already seeing some interop problems without this
  - Controlled experiments not very encouraging

- Proposed resolution: drop
Non-X.509 Certificates

• We’ve changed Certificate a lot

• The other certificate format documents assume you replace all of Certificate, which doesn’t work

• Proposed resolution:
  – Deprecate the following for TLS 1.3:
    {client,server}_{certificate_type, user_mapping, cert_type, cached_info}?
  – People can update drafts with new code points if they want
Opting-out of post-handshake client auth

• Olivier Levillain on-list:

  The client can not indeed ignore all this state to answer, since it is
  supposed to answer at least with a Finished message, which will cover
  the CertificateRequest message. Moreover, since each of these Finished
  messages must cover the initial handshake and the current
  CertificateRequest message, it requires a forkable hash implementation,
  which requires more memory.

• Potential options:
  – Remove post-handshake auth
  – Require an extension to opt-in to post-handshake auth
  – Specifically allow ignoring post-handshake
  – Do nothing

• Proposal: Do nothing
Any other issues?

On to draft-20 and IETF-LC