draft-ietf-tls-downgrade-scsv-00

“TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks”

Bodo Möller
Adam Langley

TLS@IETF-90, July 2014
Background & Objective

- TLS version negotiation can fail in practice
  - Broken servers, broken middleware
  - \( \approx 1\% \) TLS 1.1 intolerant servers*,
    \( \approx 1\% \) TLS 1.2 intolerant servers*
  - Bugs lie dormant until it’s too late:
    \( \approx 11\% \) TLS 1.3 intolerant servers*
    (which so far look perfectly fine to casual testing)
  *[Ivan Ristić, Nov. 2013]*

- For interoperability, many clients will fall back to a downgraded protocol version
Background & Objective (cont’d)

- Attackers or network glitches can trigger the protocol downgrade
- Earlier protocol, obviously, can be worse
  - e.g., no AEAD before TLS 1.2
  - e.g., bad CBC IVs before TLS 1.1
  - e.g., no ECDHE w/o TLS extensions (and no draft-ietf-tls-encrypt-then-mac-02)
  - e.g., unfixably bad CBC padding before TLS 1.0
- Want to avoid downgrade unless the server actually needs it!
Our approach

- Include explicit signal to the server in the ClientHello: "This is a fallback connection attempt. If I shouldn’t have had to downgrade, please abort."

- Server then aborts if it supports a protocol after `ClientHello.client_version`

- Downgrade strategy directed by client: simple server logic, no server-side heuristics
Specifics

- Our signal is a Signaling Cipher Suite Value (SCSV), `TLS_FALLBACK_SCSV`: works without extension support
  - also, takes less space than empty extension

- Enabled on Google servers, in Google Chrome 33
  - Chrome Stable Channel as of February 2014
Other considerations

- Clients shouldn’t really have to downgrade ...
  - Can’t remove all buggy servers from internet, but see draft-pettersen-tls-version-rollback-removal-03
  - Seems *orthogonal* to our spec
I-D progress

- Since draft-bmoeller-tls-downgrade-scsv-01 (June 2013), only editorial changes
- Next step: Working Group Last Call?