Encrypted Client Hello – Lessons learned from trying to do something that was probably too complicated

Dr. Stephen Farrell
Trinity College Dublin
stephen.farrell@cs.tcd.ie

OpenSSL Conference
October 2025

This talk’s point-of-view

- The lessons here are from the point-of-view of someone aiming to contribute to upstream projects (like OpenSSL) who is not a maintainer
- There may be implied lessons for maintainers of such projects too, but it’d be presumptive of me to state those
  - Implications may well be implied though:-)
- There are lessons for standards development organisations, but that’d (mostly) be a different talk

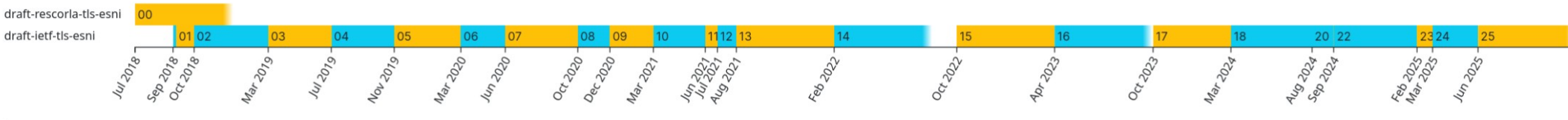
Obvious/Generic “Lessons” (1/2)

- Lesson: Don’t try for huge changes to upstream
  - Turns out, huge/intrusive changes to OpenSSL code are what ECH requires;-(
  - Changes to application upstreams are much more modest though
- Lesson: Upstreams have pointlessly different styles
  - Live with ‘em nonetheless, mostly just an irritant
  - Maintainers do try help though, they know style is local

Obvious/Generic “Lessons” (2/2)

- Lesson: Be nice, maybe even a bit deferential
  - Maintainers will live with the code for a lot longer
- Lesson: Figure out who’s actually in control
  - Not hard, but not all PR comments are equal
  - But don’t disregard non-maintainer comments

What’s ECH?



- SNI is a privacy leak, be nice to encrypt that
- Spec: draft-ietf-tls-esni
  - Started 2018, still not quite done yet (but close)
- Encrypted ClientHello (ECH) allows a cleint/browser to encrypt sensitive parts of the TLS ClientHello, primarily the server name indication (SNI), if the server has published a public-key (ECHConfig) in the DNS

It began simply enough

- RFC 8744 documents Issues and Requirements for Server Name Identification (SNI) Encryption in TLS
- Initially draft-ietf-tls-esni proposed to add a new TLS extension (ESNI) with a ciphertext form of the SNI and a TXT RR for publishing a public key
- I was looking for something to do
- Nobody was coding up ESNI for OpenSSL
- Looks like I started on that around November 2018
- Implementing ESNI wasn’t too hard

DEfO project

- DEfO == “Developing ECH for OpenSSL”
  - https://defo.ie/ started in 2019, ongoing
  - Funded by Open Tech Fund (OTF), who’ve been great
- Goal: encourage ECH implementation and deployment by contributing to existing open-source projects
- Non-goal: creating some new everlasting project
- Lesson: It’s very useful to have deadlines, deliverables and a few quid
- Lesson: Teams do some stuff better
  - We got lucky in DEfO, teaming up with really excellent people (mainly from Guardian Project)
  - E.g. the DEfO CI setup https://github.com/defo-project/

DEfO Project

- Scope includes work to ECH-enable applications
  - Code upstreamed: curl, lighttpd, apache2, OONI
  - PRs/patches: haproxy, nginx, cPython
- Some of the above support multipleTLS libraries, e.g., BoringSSL, AWS-LS, WolfSSL
- Lesson: >1 important TLS library in the world

ESNI -> ECHO -> ECH

- Early ESNI drafts used an ad-hoc way of encrypting the SNI, yet TLSv1.3 has good cryptgraphic proofs, so that wasn’t desirable
- HPKE (RFC9180) was developed partly to regularise that – it provides a good way to encrypt a message “to” a public key
  - RFC9180 has way too many options, arguing against such is generally a losing position
  - At least, initially

OpenSSL PR#17172 (HPKE)

- Opened Nov 2021, took nearly a year, 4.5kLOC
- I learned a lot in doing that
- Followed up with an OTF-funded pentester code review
  - https://7asecurity.com/reports/pentest-report-defo-2.pdf
- Lesson: stick at it
- Lesson: I wouldn’t start from there if I were you
- Lesson: be willing to accept maintainer help when offered

ECH DNS issues

- ESNI used a TXT RR: undesirable for various reasons
  - See `dig txt tcd.ie` output for why:-)
- Development of HTTPS RR, including go-faster-stripes for browsers, was likely required for browser enthusiasm, result: RFC9460
- Lesson: you don’t control the ecosystem but understanding incentives is very useful

ECH feature branch

- OpenSSL ECH feature branch created April 2024, 7 PRs merged since, one currently open (s_client/s_server), maybe 2 more for completion (test code)
  - Not including ECH “split-mode” (TBD later)
- Adds 10kLOC so far;-) And those lines are scattered over a lot of the TLSv1.3 code
- Lesson: maintainer response times are bursty

Overall

- It’s time-consuming and sometimes frustratning to try do something that’s probably too complicated
  - But hey – you walked into it with eyes open:-)
- Despite the above, DEfO is finally getting closer to done, and I can hope to not be working much on ECH in the not too distant future