February 2, 2025

QUIC *vs.* Middleboxes

Lars Eggert, lars@eggert.org, FOSDEM 2025

Mozilla

February 2, 2025

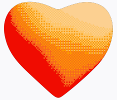
QUIC ❤️ Middleboxes

Lars Eggert, lars@eggert.org, FOSDEM 2025

Agenda

01 QUIC

02 Middleboxes

03 

01

QUIC

QUIC: a fast, secure, evolvable transport

Fast.

Better user experience than TCP/TLS for HTTP/2 and other content.

Evolvable.

Prevent network from ossifying, deploy new QUIC versions quickly.

Secure.

Always-encrypted end-to-end security, resist pervasive monitoring.

Transport.

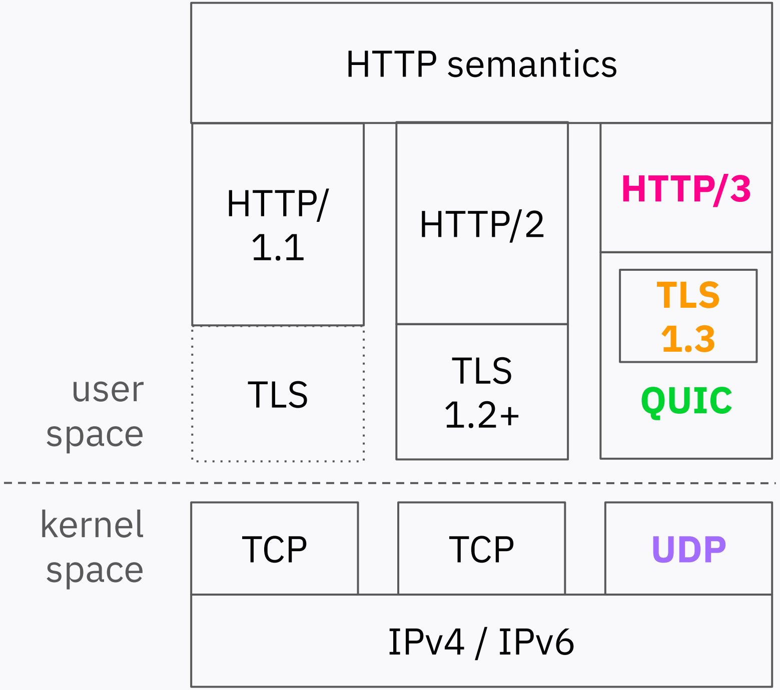
Support all TCP content & more (realtime media, etc.) Provide better abstractions, avoid known TCP issues.

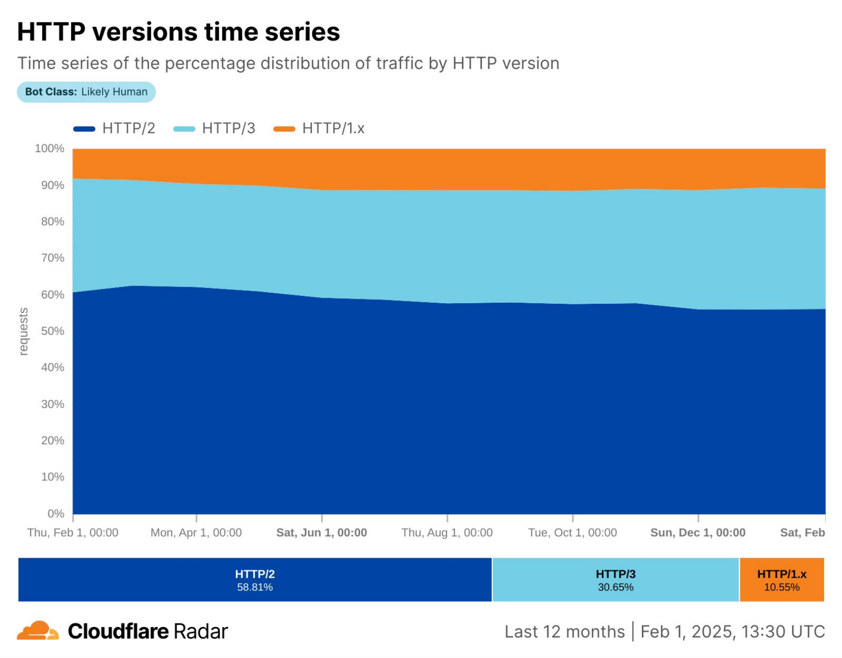
QUIC

2013 Experiment at Google

2016 IETF WG started

2021 RFCs 8999-9002





Why UDP?

- TCP hard to evolve
- Other protocols blocked by middleboxes (SCTP, etc.)
- **UDP is all we have left**
- Not without problems!
  - Middleboxes ossified on “UDP is for DNS”
  - Enforce short binding timeouts, etc.
  - Short-term issue with NIC offloading
- Also, benefits
  - Can deploy in userspace (no kernel update needed)
  - Can offer alternative transport types (partial reliability, etc.)

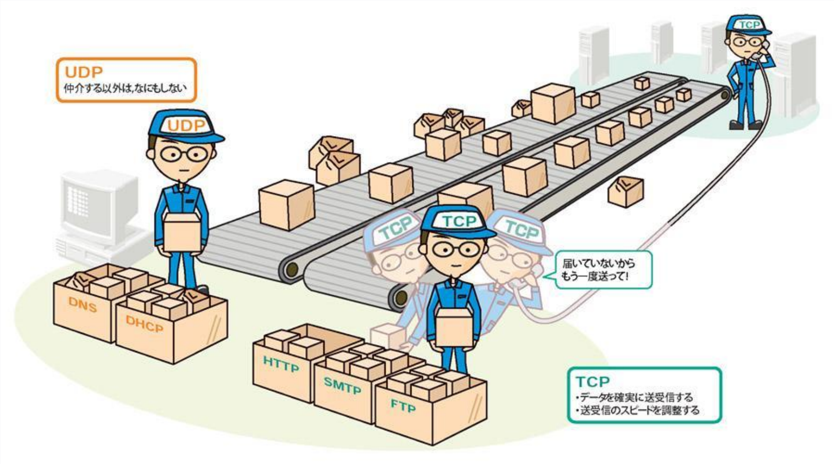


Image from http://itpro.nikkeibp.co.jp

Why congestion control? (Duh)

- Functional CC is **absolute requirement** for operation over real networks
  - UDP has no CC
- First approach: **take what works for TCP, apply to QUIC**
- Consequence: need
  - Segment/packet numbers
  - Acknowledgments (ACKs)
  - Round-trip time (RTT) estimators
  - etc.
- Not an area of large innovation at present
  - This will change

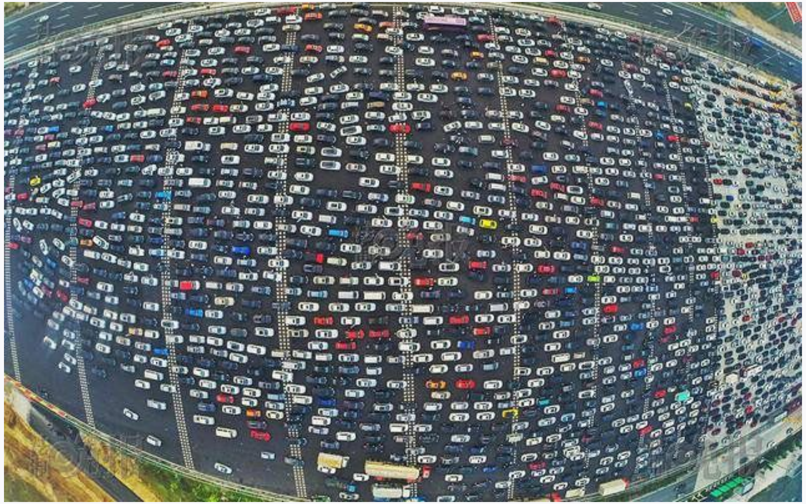
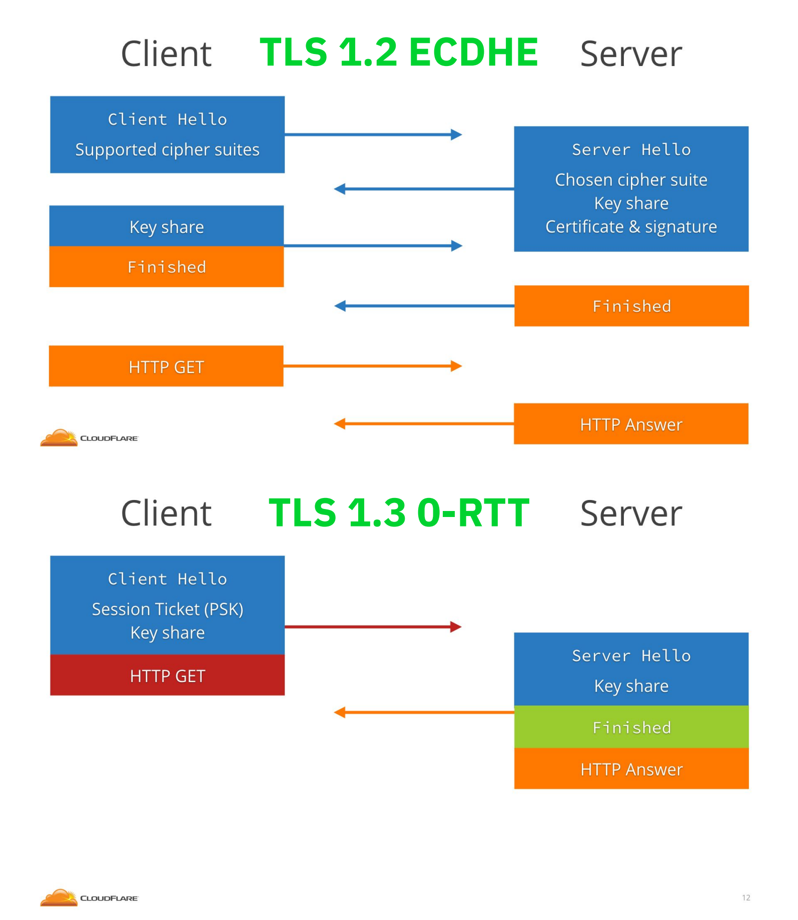


Image from People's Daily, http://people.cn/

Why TLS? (Duh)

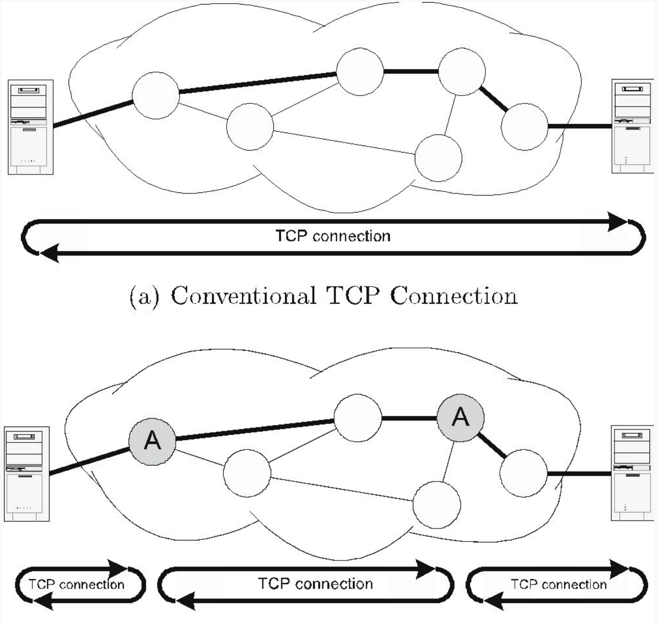
- **End-to-end security is critical**
  - To protect users
  - To prevent network ossification
- TLS is very widely used
  - Can leverage all community R&D
  - Can leverage the PKI
- **Don't want custom security** – too much to get wrong
  - Even TLS keeps having issues
  - But TLS 1.3 removes a lot of cruft
  - And adds new features (0-RTT!)



02

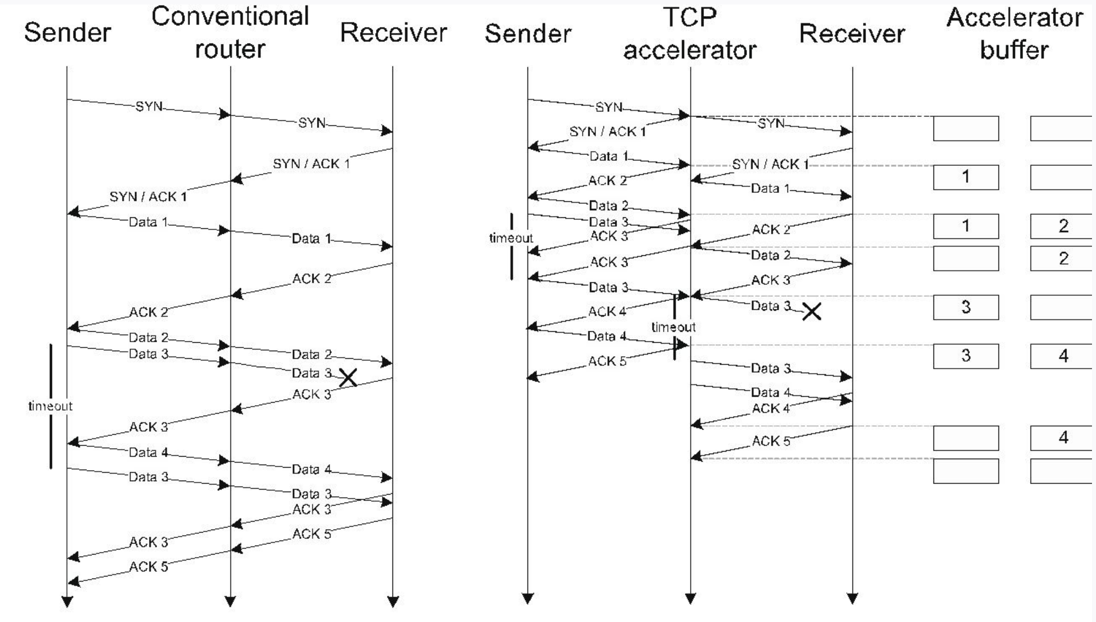
Middleboxes

Middleboxes meddle
e.g., “TCP accelerators”



(a) Conventional TCP Connection

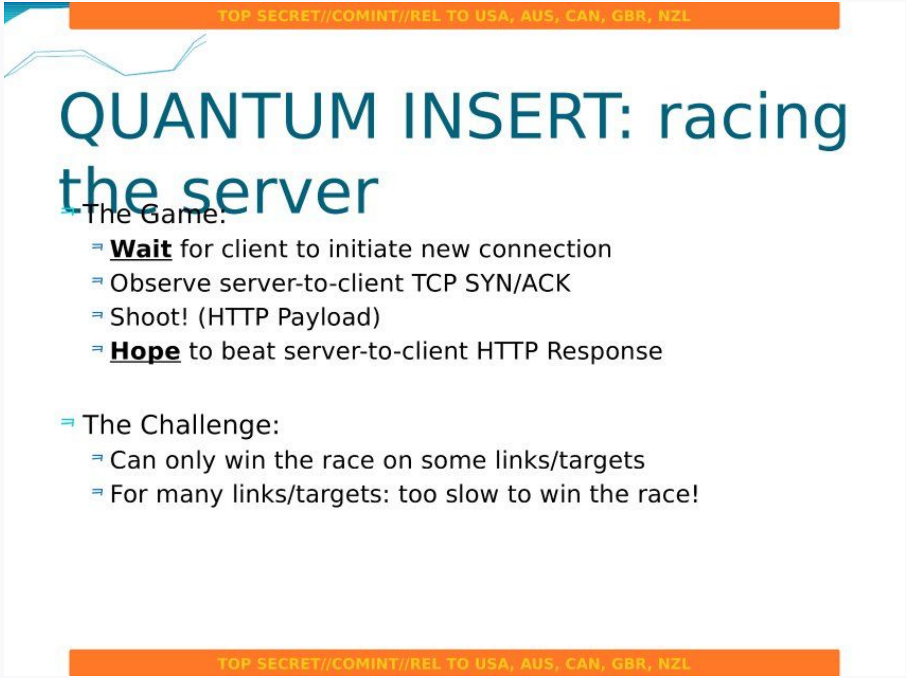
(b) Accelerated TCP Connection

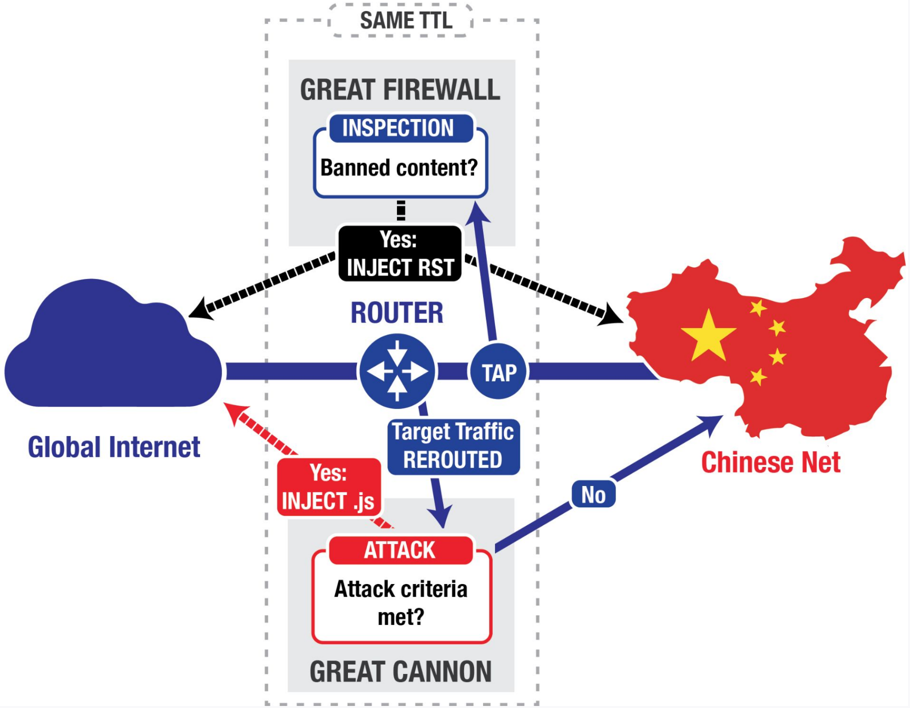


(a) Conventional TCP Connection (b) Accelerated TCP Connection

Sameer Ladiwala, Ramaswamy Ramaswamy, and Tilman Wolf. Transparent TCP acceleration. Computer Communications, Volume 32, Issue 4, 2009, pages 691-702.

Middleboxes meddle
e.g., nation states as attackers





QFIRE Pilot Lead. NSA/Technology Directorate. QFIRE pilot report. 2011.

B. Marczak, N. Weaver, J. Dalek, R. Ensafi, D. Fifield, S. McKune, A. Rey, J. Scott-Railton, R. Deibert, and V. Paxson. An Analysis of China's "Great Cannon". 5th USENIX FOCI Workshop, 2015.

RFC 7528 Pervasive monitoring is an attack

- IETF (& wider) community consensus that pervasive monitoring is an attack
- Agreement to mitigate pervasive monitoring
- What does “mitigate” mean?
- To many, ”encrypt as much as possible”

- **But what else could we do?**

TLS extension randomization

- TLS extensions in the client hello are sent in some order
- This aids TLS stack fingerprinting
- Solution: **randomize that order**
- Easily (partially) defeated by canonical reordering :-(
- Par for the course (= do it anyway)

03

RFC 8701 Grease

Internet Engineering Task Force (IETF) D. Benjamin
Request for Comments: 8701 Google LLC
Category: Informational
Published: January 2020
ISSN: 2070-1721

Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility

Abstract

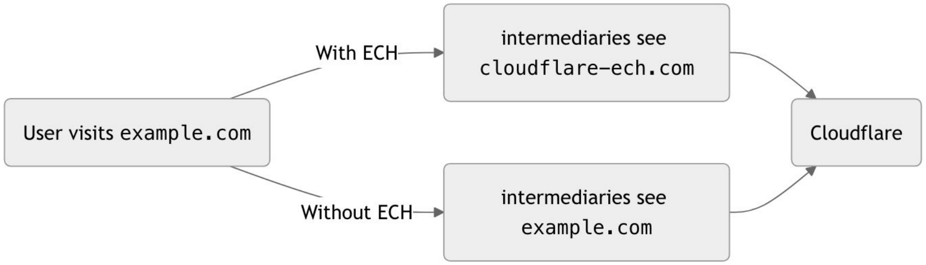
This document describes GREASE (Generate Random Extensions And Sustain Extensibility), a mechanism to prevent extensibility failures in the TLS ecosystem. It reserves a set of TLS protocol values that may be advertised to ensure peers correctly handle unknown values.

- "MUST be set to zero on send, and ignored on receive" - **NO MORE**
- Instead, "grease" unused codepoints by setting them to random on send
- For codepoint registries, include (many) (non-contiguous) ranges of to-be-ignored grease codepoints
  - "All [version] codepoints that follow the pattern 0x?a?a?a?a are reserved, MUST NOT be assigned by IANA, and MUST NOT appear in the listing of assigned values."
  - "Each [transport parameter] value of the form 31 * N + 27 for integer values of N (that is, 27, 58, 89, ...) are reserved; these values MUST NOT be assigned by IANA and MUST NOT appear in the listing of assigned values."

Problem: TLS SNI observability

- SNI = server name indication
- Basically, the DNS name of the server you’re connecting to
- Range of ASCII bytes in client hello
- Easily extractable/observable

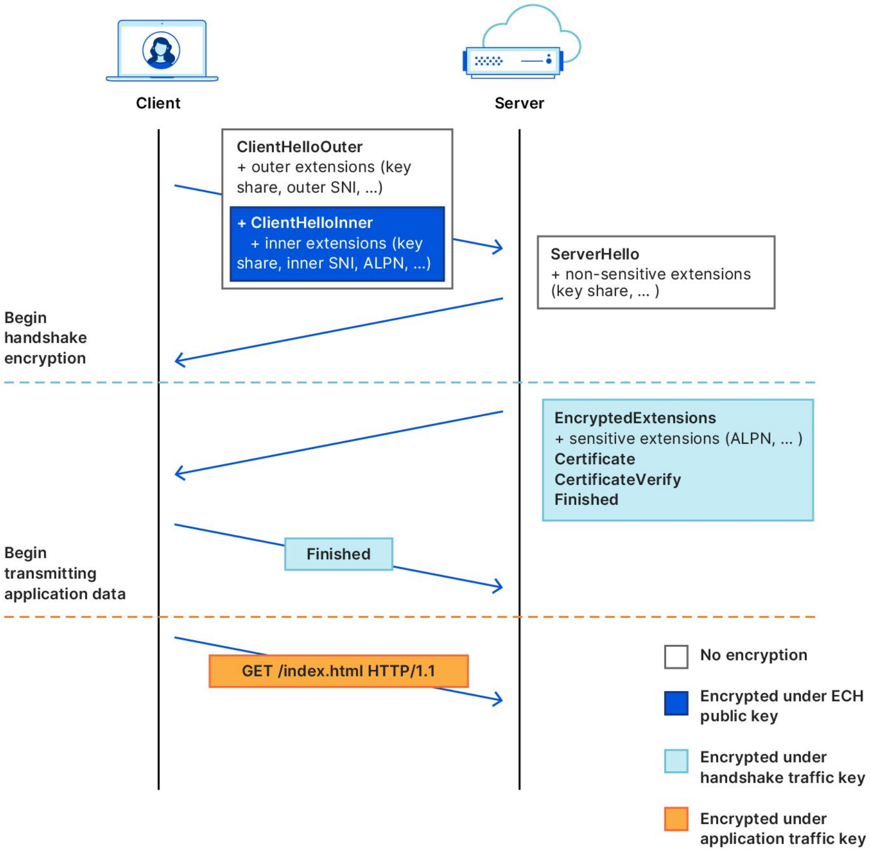
Encrypted client hello



Encrypts the actual SNI

Observers see outer SNI of `cloudflare-ech.com` for all TLS connections

Victory? No :-(



Images from https://blog.cloudflare.com/encrypted-client-hello/

SNI obfuscation

- QUIC carries TLS1.3 handshake data in “CRYPTO frames”
- That means we can split the data, and reorder the chunks
- For example, we can split the data in the middle of the SNI
- [...]mozilla.com[...] becomes [...]a.com[...]mozill[...]

- Bonus: post-quantum crypto (e.g. MLKEM) use multi-packet client hellos – make middleboxes hold state

Public name masquerade for ECH

- Replace `cloudflare-ech.com` with a unique name for each client (ideally)
- Idea: use outer SNI to indicate anonymity set to server, TLS retry to make progress from that

- draft-thomson-tls-ech-pnmasq-latest

0 for fun and profit

- ...the [Great Firewall of China] exempts a connection if the fraction of bits set in the client's first data packet deviates from half. This corresponds to a crude measure of entropy: random (encrypted) data will have close to half of the bits set to 1, while other protocols usually have fewer 1 bits per byte due to plaintext or zero-padded protocol headers.

How the Great Firewall of China Detects and Blocks Fully Encrypted Traffic. Mingshi Wu, Jackson Sippe, Danesh Sivakumar, Jack Burg, Peter Anderson, Xiaokang Wang, Kevin Bock, Amir Houmansadr, Dave Levin, Eric Wustrow. *USENIX Security Symposium 2023.*

Thank you

Help us build this!

https://github.com/mozilla/neqo

Lars Eggert, lars@eggert.org, FOSDEM 2025

@lars.social.secret-wg.org