

# Don't be scared of HTTP RR and ECH

Vinicius Fortuna - Jigsaw, Google  
March 20, 2026 - IETF 125 - tls

Goal

Enable **HTTP RR and ECH support**  
in **networking libraries and applications**  
**by default**

Is it safe?

# Does enabling by default break connectivity?

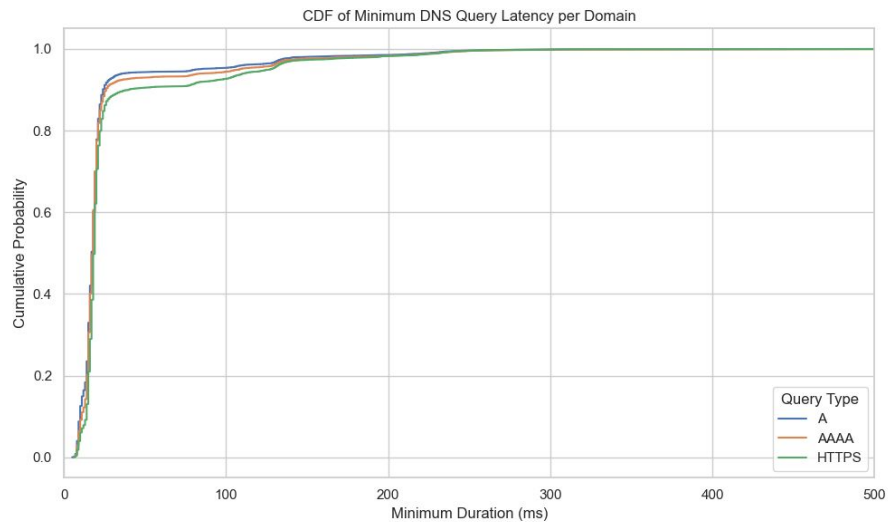
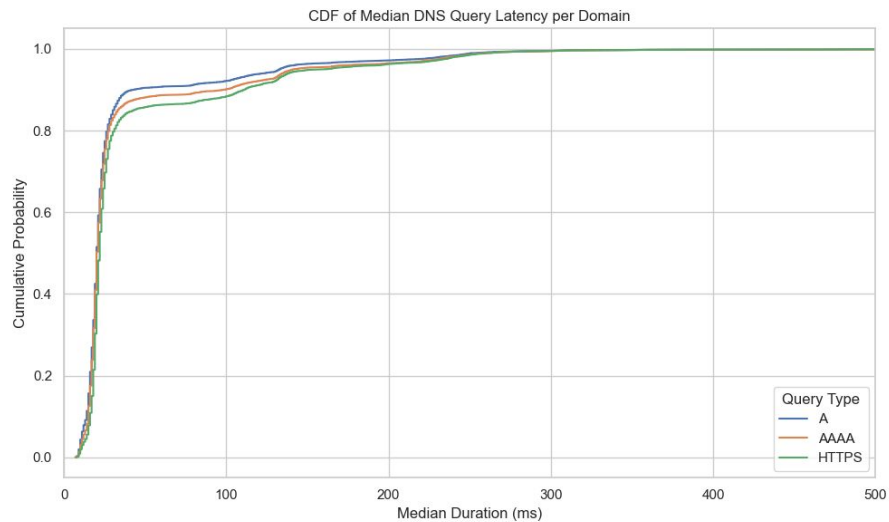
Research questions:

1. How does waiting for the HTTPS RR affect performance and connectivity?
2. Does enabling ECH by default break connectivity?
  - a. Do services break when they don't support ECH?
  - b. Do networks block ECH?

# DNS Test

# HTTPS RR DNS Test

1. Query top 10k Tranco domains for A, AAAA and HTTPS
2. Measure response times
3. Repeat 3x



# HTTPS RR DNS Test

What's the delay if you always wait for the HTTPS record?

**HTTPS Delay = T(HTTPS) - HE\_baseline**

**HE\_baseline** is when you would start the first TCP connection with Happy Eyeballs v2 (considering the 50ms HEv2 *Resolution Delay*)

```
# Calculate standard legacy connections baseline, accounting for Happy Eyeballs V2 (RFC 8305)
# Waiting up to a 50ms Resolution Delay for AAAA but only if it contains actual records
def calc_he_baseline(row):
    t_a = row['A']
    t_aaaa = row['AAAA']

    h_a = row['cat_A'] == 'SUCCESS - has answers'
    h_aaaa = row['cat_AAAA'] == 'SUCCESS - has answers'

    if not h_a and not h_aaaa:
        return np.nan

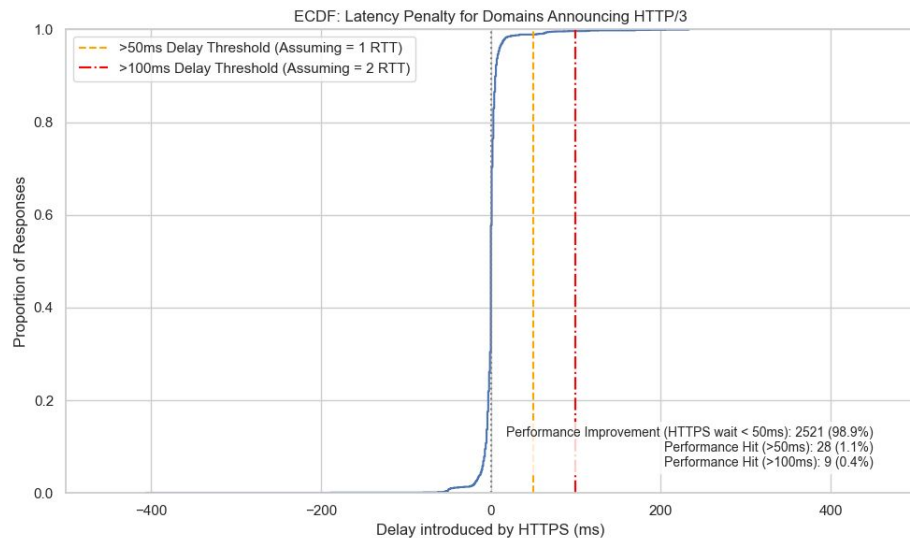
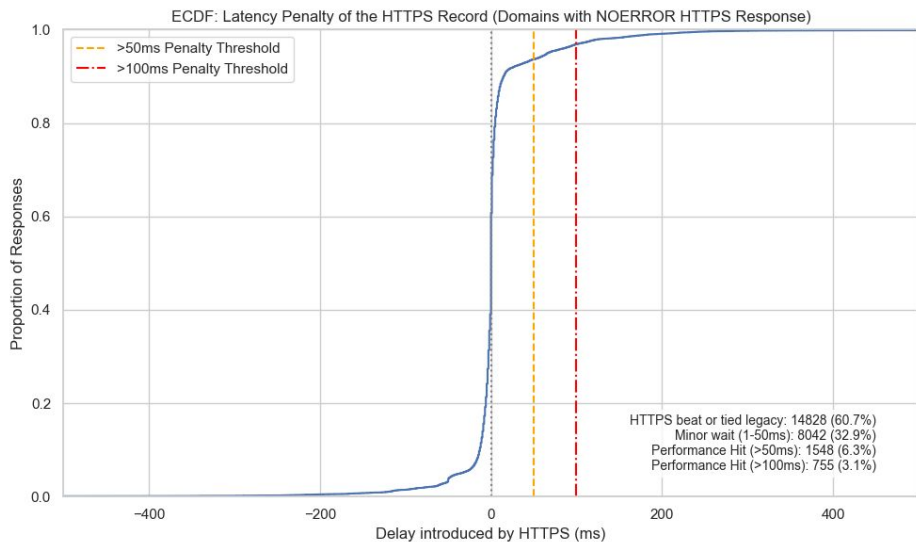
    if h_a and not h_aaaa:
        if pd.isna(t_aaaa): # No response for AAAA yet, wait up to 50ms
            return t_a + 50
        return max(t_a, min(t_a + 50, t_aaaa)) # Wait until we know AAAA is empty, or timer fires

    if not h_a and h_aaaa:
        return t_aaaa # Use AAAA immediately

    # Both have answers
    if pd.isna(t_aaaa):
        return t_a + 50
    if pd.isna(t_a):
        return t_aaaa

    if t_aaaa <= t_a + 50:
        return t_aaaa
    return t_a + 50
```

# Distribution of HTTPS Delay



The delay is only a penalty if it takes longer than the TCP connection.

- You can still apply the config before the TCP connection is established (if config endpoint matches the connected one).
- You save a round trip if QUIC is supported

# Should we wait for the HTTPS RR?

- No or tiny latency penalty
- Significant latency improvement when avoiding the TCP round trip
- Further latency reduction if using IP hints

# Should we wait for the HTTPS RR?

However...

- **Significant domains do not respond to HTTPS queries**
  - Includes nih.gov, many other .gov domains and others
- **Waiting will break connectivity to those domains**

Recommendation:

- We have to cap the wait until those domains are fixed
- Domain owners need to fix their authoritative resolver to answer HTTPS queries

	domain	rank	min_duration_ms
	nih.gov	193	3022
	pubmed.ncbi.nlm.nih.gov	500	3020
	usda.gov	1075	3022
	ifilo.net	1872	4029
	al-array.com	3713	4029
	cancer.gov	4369	3030
	novell.com	4427	3136
	nyc.gov	4528	4024
	southwest.com	5437	2020
	microfocus.com	5504	3021
	medu.ir	5587	4107
	my.medu.ir	5938	4108
	mii-beian.gov.cn	5978	4109
	dtvce.com	6188	2020
	wsu.edu	6203	2021
	telefonica.com	6250	2018
	ct.gov	6382	2019
	utah.gov	7114	2018
	boeing.com	7288	1087
	caf.fr	8478	2019
	unm.edu	8749	2020
	globe.com.ph	9446	4232

# ECH GREASE Test

# Does ECH break services?

When ECH is enabled, ECH GREASE is sent to servers that don't support it.  
Do they break?

- Tested top 10k domains from Tranco list
- For each domain, fetch `https://<domain>/` twice:
  - **Control:** ECH disabled
  - **Experiment:** ECH enabled with GREASE

Used ECH-enabled curl and openssl (by DEfO team)

# ECH GREASE did not break services

- 26 domains (**0.26%** of the total) showed differences (various errors)
  - Disappeared on retrial, indicating transient issues
- 235 domains (**2.35%** of the total) exhibited a significant difference (>100ms) in TLS handshake time
  - 210 saw reduced latency with ECH GREASE
  - 25 saw increased latency with ECH GREASE

In short: **no adverse effects**

# Network Test

# Do networks block ECH?

For each mobile network in each country from the SOAX proxy network

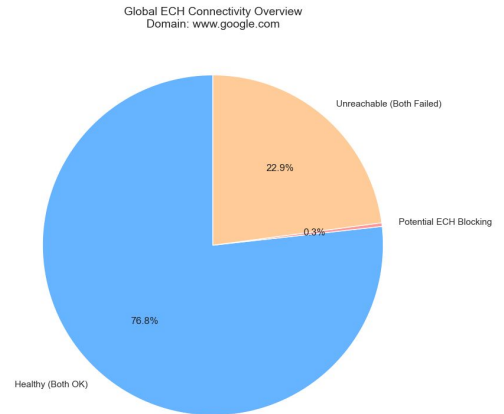
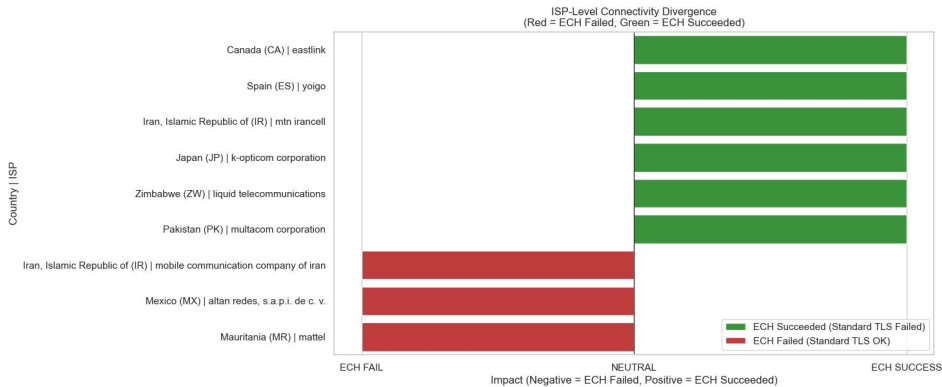
- Fetch <https://www.google.com> via the proxy
  - **Control:** ECH disabled
  - **Experiment:** with ECH GREASE

Used the same ECH-enabled curl

# Do networks block ECH? - no indication

- Total networks: 878
- ECH Failures: 3 (**0.34%**) - may be transient, needs retesting
- Dual (control+experiment) failure: 201 (**22.9%**)

The dual failures need further investigation. May be proxy instability or rate limits. There's also blocking of [www.google.com](http://www.google.com) in China.



# Conclusion

**Enable HTTPS RR and ECH by default**

**Cap the HTTPS RR wait** until top broken domains are fixed

Tools and results will be at

**<https://github.com/Jigsaw-Code/ech-research>**