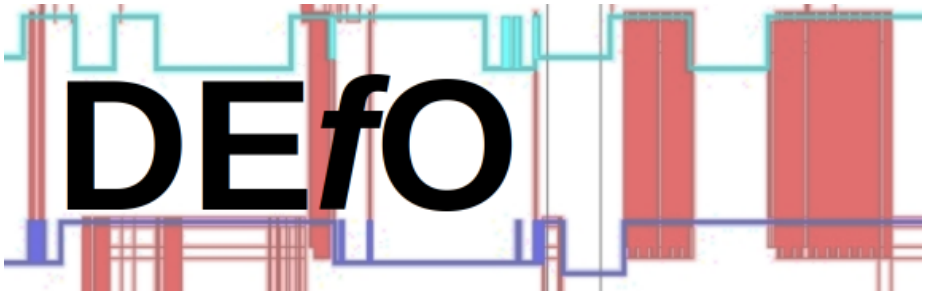


# ECH Interoperability Report

Stephen Farrell
Trinity College Dublin/Tolerant Networks Ltd.
stephen.farrell@cs.tcd.ie

Tuesday 10th December, 2024

**Abstract**

Encrypted Client Hello (ECH) is a privacy-enhancement for the Transport Layer Security (TLS) protocol that underlies all web security and the security of many other Internet protocols. While the specification for ECH is relatively mature, (though not yet an Internet-RFC), and while implementations are already widespread, work remains to ensure that a random client and server can successfully use ECH. To that end, this report describes software libraries, client and server software packages, and Internet services for which ECH is relevant, and configurations of those where ECH currently works, or fails, as part of overall efforts to encourage ECH deployment and hence improve Internet security and privacy. At the time of writing, the overall interoperability story for ECH could be summed up as: if you stick to a simple deployment that works with browsers, that'll be fine, but if you want to push out the boundaries, then something will break, so don't do that. We justify that claim via a test setup involving 67 ECH test URLs (some with broken or non-normal setups), six test clients, and coverage of almost all known ECH-capable server technologies.

Keywords: Encrypted Client Hello (ECH), Interoperability

# 1 Introduction

Deployments of the Transport Layer Security (TLS [1]) protocol expose the name of the server (e.g. the web site DNS name) via the Server Name Indication (SNI) field in the first message sent (the ClientHello). The Encrypted Client Hello (ECH) [2] extension to TLS is a privacy-enhancing scheme that aims to address this leak.

The Open Technology Fund (OTF) [1] have funded the DEfO project [2] to develop ECH implementations for OpenSSL, and to otherwise encourage implementation and deployment of ECH. This report is a deliverable for the DEfO project and describes the current state of ECH interoperability.

As we expect the implementation and deployment environment for ECH to change over time, this report will be updated as events warrant and is currently versioned based on the build-time of this PDF. The latest version can be found at `https://github.com/defo-project/ech-interop-report/blob/main/ech-interop-report.pdf`. Comments, additions or corrections are welcome. Those can be sent via email to the author, or as issues or PRs (preferably for the latex files) in the github repository for this report which is `https://github.com/defo-project/ech-interop-report`.

# 2 Software

This section describes libraries, clients and servers that support ECH.

## 2.1 Libraries

The libraries listed in Table 1 have some support for ECH. Of the libraries listed, ECH support for all of the OpenSSL variants, python and libcurl were developed by the DEfO project.

Table 1: Libraries with ECH

| Name | Details |
|---|---|
| OpenSSL-sftcd | Source: `https://github.com/sftcd/openssl/tree/ECH-draft-13c`<br>Version: 3.4.0-dev fork of master<br>ECH support: client and server, TLS only (no DTLS, no QUIC)<br>Comment: this is the DEfO project's main development branch |
| OpenSSL-defo | Source: `https://github.com/defo-project/openssl/` |

Continued on next page

[1] `https://www.opentech.fund/`
[2] `https://defo.ie`

Table 1: Libraries with ECH (Continued)

| Name | Details |
|---|---|
| | Version: 3.5.0-dev fork of master, same ECH code as OpenSSL-sftcd<br>ECH support: client and server, TLS only (no DTLS, no QUIC)<br>Comment: this is used for DEfO project CI builds and tests |
| OpenSSL-feature-branch | Source: `https://github.com/openssl/openssl/tree/feature/ech`<br>Version: 3.5.0-dev feature branch in upstream repo<br>ECH support: stub APIs so far, next PR will have ECH client, then server after<br>Comment: this is the OpenSSL project's target for ECH PRs, and is where ECH code will end up prior to eventually being merged to master |
| boringssl | Source: `https://boringssl.googlesource.com/boringssl`<br>Version: boringssl doesn't do versions, last local build 2024-11-29<br>ECH support: ECH client and server, ECH for TLS, QUIC and (possibly) DTLS<br>Comment: in production use (chromium et al), a little more limited in HPKE suites than OpenSSL - only KEMs are x25519 and p256 |
| NSS | Source: `https://github.com/nss-dev/nss.git`<br>Version: NSS 3.108<br>ECH support: ECH client and server, ECH for TLS (unsure of DTLS/QUIC)<br>Comment: in production use (firefox), a little more limited in HPKE suites than OpenSSL - only KEM is x25519 |
| WolfSSL | Source: `https://github.com/wolfSSL/wolfssl.git`<br>Version: 5.7.4<br>ECH support: ECH client and server, ECH for TLS (unsure of DTLS/QUIC)<br>Comment:<br>- ECH not built by default (needs "–enable-ech")<br>- fails when HelloRetryRequest seen - `https://github.com/wolfSSL/wolfssl/issues/6802` |
| gnuTLS | Source: `https://gitlab.com/gnutls/gnutls/blob/master/README.md` |

Continued on next page

Table 1: Libraries with ECH (Continued)

| Name | Details |
|---|---|
| | Version: work-in-progress<br>ECH support: interop untested (by DEfO-project) at this time<br>Comment: an ECH merge request exists but has yet to be merged `https://gitlab.com/gnutls/gnutls/-/merge_requests/1748` |
| golang | Source: `https://go.googlesource.com/go` or `https://github.com/golang/go/`<br>Version: 1.23 or later required<br>ECH support: client only, server coming in 1.24 (server code is merged)<br>Comment:<br>- golang tests were just (2024-12-02) added to our smokeping tests |
| rustls | Source: `https://github.com/rustls/rustls/`<br>Version: 0.23.19<br>ECH support: client only<br>Comment:<br>- rustls tests were just (2024-12-03) added to our smokeping tests |
| python | Source: `https://github.com/defo-project/cpython`<br>Version: python 3.13/3.14<br>ECH support: TLS client only<br>Comment:<br>- python tests were just (2024-12-05) added to our smokeping tests |
| libcurl | Source: part of `https://github.com/curl/curl`<br>Version: 8.10.0-DEV<br>ECH support: TLS client only<br>Comment: not well tested other than via command line curl |

## 2.2 Clients

Current versions of firefox (e.g. 133.0, used in our smokeping tests) and chromium (version 131.0.6778.85 is used in our tests) support ECH by default. Other chromium-derived browsers also support ECH, while we don't include them in our ongoing tests described below, we have manually verified that current versions of vivaldi, brave and opera also have ECH support. The Tor browser (a

firefox derivative) doesn't seem to support ECH by default. To test if a browser supports ECH one can visit `https://test.defo.ie/` or, for much more detail `https://test.defo.ie/iframe_tests.html`.

Browsers in general are still somewhat quirky in terms of whether or not ECH will succeed, even when ECH is properly setup. This seems at least partly due to the delay that can inherently occur between the browser receiving DNS answers for A/AAAA records, and the time at which the browser receives a DNS answer for an HTTPS record (containing ECH information). It seems to be the case that browsers may be "impatient" in that they start the TLS session without attempting ECH if that delay is too long, and that that happens sufficiently often to be noticeable. (Note however, that this report does not in itself yet justify that claim, but work is ongoing.)

The only command line tool supporting ECH of which we're aware is curl. The DEfO project contributed ECH code as an experimental feature for curl that was merged to the master branch in April 2024. As an experimental feature, ECH is not built by default nor part of a release, so is only currently available to those who build from source. Curl supports using OpenSSL, boringssl or Wolfssl libraries for ECH. Build instructions are at `https://github.com/curl/curl/blob/master/docs/ECH.md`.

As a scripting tool, often used as part of a TLS client, Python also fits here. The DEfO project Cpython build adds ECH support to python via the OpenSSL library. That build is available at `https://github.com/defo-project/cpython` with the new APIs required for ECH described at `https://irl.github.io/cpython/library/ssl.html#encrypted-client-hello`.

## 2.3 Servers

The servers listed in Table 2 support ECH. In each case, the ECH integration, using OpenSSL, was developed as part of the DEfO project.

Table 2: Servers supporting ECH

| Name | Details |
|---|---|
| lighttpd | Source: `https://github.com/defo-project/lighttpd`<br>Version: 1.4.76.81<br>Comment: only shared-mode ECH |
| nginx | Source: `https://github.com/defo-project/nginx`<br>Version: 1.27.3.4<br>Comment: both shared-mode and split-mode ECH |
| apache2 | Source: `https://github.com/defo-project/apache-httpd`<br>Version: 2.5.0.ech.351 |

Continued on next page

Table 2: Servers supporting ECH (Continued)

| Name | Details |
|---|---|
| | Comment: only shared-mode ECH |
| haproxy | Source: `https://github.com/defo-project/haproxy`<br>Version: 3.2.dev0.62<br>Comment: both shared-mode and split-mode ECH |
| OpenSSL s_server | Source: `https://github.com/defo-project/openssl`<br>Version: 3.5.0-dev<br>Comment: shared-mode only but can easily "force" HelloRetryRequest (HRR) |

# 3 Services

## 3.1 Test Services

We can separate services into test and operational services. Table 3 lists known test services supporting ECH. All of those below .ie and my-own.net were setup by the DEfO project.

Table 3: Test Services with ECH

| Name | Details |
|---|---|
| defo.ie | `https://defo.ie/ech-check.php` is often used to check ECH<br>ECH keys are rotated hourly, private usable for 3 hours, ECHConfigList in DNS contains only latest public |
| draft-13.esni.defo.ie | different server technology (as listed in Table 2) instances on different ports as listed at `https://defo.ie/`, e.g., `https://draft-13.esni.defo.ie:10413` is served by nginx<br>ECH keys are rotated hourly, private usable for 3 hours, ECHConfigList in DNS contains only latest public |
| test.defo.ie | hosts a number of ECH server setups with good and variously bad configurations - see `https://test.defo.ie/iframes_tests` which describes those and allows a browser to attempt connections to each via Iframes<br>ECH keys/configs are static for these setups |
| foo.ie | `https://foo.ie/ech-check.php` was setup used to check the defo.ie setup was easily replicated<br>ECH keys are rotated hourly, private usable for 3 hours, ECHConfigList in DNS contains only latest public |

Continued on next page

Table 3: Test Services with ECH (Continued)

| Name | Details |
|---|---|
| my-own.net | this was to test the impact of having the same ECH keys on port 443 (`https://my-own.net/ech-check.php`) and another port (`https://my-own.net:8443/ech-check.php`) - at one point that made a difference to browsers<br><br>ECH keys are rotated hourly, private usable for 3 hours, ECHConfigList in DNS contains only latest public |
| tls-ech.dev | `https://tls-ech.dev/` was setup by the boringssl developers as a test server that uses boringssl<br><br>ECH keys/configs seem to be static for this setup |
| Cloudflare | `https://cloudflare-ech.com/cdn-cgi/trace` is a test page setup by cloudflare that reports on ECH success/failure<br><br>apparently, the server implementation and infrastucrure are part of Cloudflare's normal setup<br><br>a similar test service used to be available at `https://crypto.cloudflare.com/cdn-cgi/trace` but that was turned off around the time that ECH was re-enabled for Cloudflare customers<br><br>based on one test on 2024-12-09, new ECH keys are published roughly hourly with a 300 second TTL; old ECH public values seem usable for approximately 4 or up to 5 hours; and the ECHConfigList published in DNS contains only one public value |
| rfc5746.mywaifu.best | this ECH-enabled web page (`https://rfc5746.mywaifu.best/`) seems to have been setup as an ECH test site by someone using DEfO artefacts (nginx and documentation) but without any contact having happened between the person who set that up and any of the DEfO-project participants<br><br>the person who set this up documented some of that at `https://ckcr4lyf.github.io/tech-notes/services/nginx/nginx-ech.html` |

## 3.2 Operational Services

The only operational ECH service we know of is Cloudflare's deployment. However, that is non-negligible. Cloudflare earlier enabled ECH but disabled it soon after in October 2023 [3] as it caused some back-end issues. They then re-enabled ECH in October 2024. Our understanding of Cloudflare's deployment is that ECH is enabled by default for their "free" tier customers, but that paying customers have to take action to enable ECH.

[3] `https://community.cloudflare.com/t/early-hints-and-encrypted-client-hello-ech-are-currently-disabled-globall`
`567730`

In contrast, non-paying customers are not provided with a control to disable only ECH - they have to downgrade to TLSv1.2 in order to disable ECH. This is something that has been seen in recent weeks after the Russian government started blocking use of ECH to Cloudflare. [4]

### 3.3 Domain Probe Data

As part of our DEfO test setup, we have a web page where one can enter a host name and port and a script on our server will check if ECH is enabled for a web server at the name and port. (That's at `https://test.defo.ie/domainechprobe.php`.) Data is only stored if there is an HTTPS resource record for the name and port, if there is not we store nothing. In cases where this is an HTTPS resource record, we only store the name and port, whether the HTTPS value includes an "ech=" field, if there is, whether or not ECH worked using our ECH-enabled curl as the client. We also store the HTTPS resource record value, unless there are CNAMES or other kinds of re-direction involved.

We can use this data to get some further insight into services for which ECH is enabled. For now, we are not publishing the raw data - while data for the most recent 50 queries is shown on the web page, if we wanted to publish the raw data, we'd have to enable some form of consent for users, and it's not clear that'd be a) easy or b) worthwhile. (We currently take no steps to try record who made which requests to this page.)

Figure 1 shows the (small) numbers of queries made since August 2024. We can see a noticeable uptick in accesses just-before and after Russia blocked Cloudflare's ECH service. We also see some usage (at least 15 cases) that appear to indicate some web site owners may be using this service to check whether or not they have successfully disabled ECH. (The pattern is two quick accesses to the same name/port, the first of which indicates ECH worked, and the second that the HTTPS RR no longer contains an "ech=" value.)

The 1244 queries involved 517 unique names (117 of which are under the ".ru" ccTLD). The most commonly queried name is "youtube.com" (131 times), for which ECH is not currently enabled. Some 354 names were only queried once, and 86 were queried twice.

Of the 1244 queries, 660 referred to a name that had an "ech=" in the relevant HTTPS record, referring to 298 unique names. Within the 660 HTTPS values that contain an "ech=" value, we see 276 unique ECHConfigList values (4 of which are corrupt values published for our DEfO test services). Of the 298 unique names, 271 seem to be served by Cloudflare (based on the relevant name server names), 15 relate to test services described in Table 3. At the time of writing, 10 of the names result in an NXDOMAIN or SERVFAIL response, 1 seems to correspond to a possible hobbyist site and 2 seem to (now) be parked domains or part of some advertising campaign. None of of those last 3 currently publish an HTTPS record.

[4] `https://betanews.com/2024/11/20/encrypted-client-hello-didnt-solve-censorship-but-still-may-have-a-role-to-p`

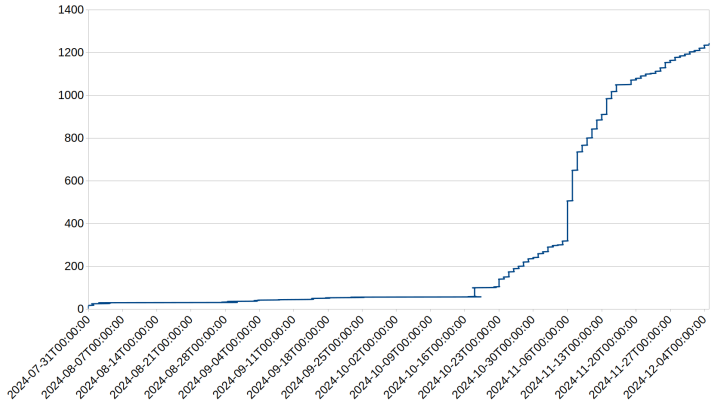


Figure 1: Cumulative number of queries seen at `https://test.defo.ie/domainechprobe.php` versus time. The total number of queries is 1244.

So at least for the names entered to our domain probe page, we can seemingly conclude that Cloudflare have the only operational service at present.

# 4 Interoperability

Our primary interoperability testing is based on our "smokeping"-like tests. While smokeping measures latency for names/addreses, our tests aim to establish whether and which ECH configurations interoperate or fail to do so.

We mainly do this via a set of hourly cronjobs running on the test.defo.ie VM that attempt to access 67 URLs (some deliberately with broken configurations) from six different clients (described in Table 4. We report results for the most recent 6 runs at `https://test.defo.ie/smokeping-summary.php`.

Table 4: Smokeping clients

| Name | Details |
|---|---|
| chromium | headless browser tests via selenium<br>Version: 131.0.6778.85<br>Python script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/selenium_test.py`<br>ECH implementation based on boringssl |
| firefox | headless browser tests via selenium<br>Version: 133.0 |

Continued on next page

Table 4: Smokeping clients (Continued)

| Name | Details |
|---|---|
| | Python script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/selenium_test.py`<br>ECH implementation based on NSS |
| curl | bash script using curl<br>Version: 8.11.1-DEV<br>Bash script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_curl.sh`<br>curl currently only pays attention the the first HTTPS resource record seen in DNS answers<br>ECH implementation based on OpenSSL |
| golang | custom golang programme and bash script<br>Version: golang 1.23<br>Golang programme: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/ech_url.go`<br>Bash script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_go.sh`<br>ECH implementation developed for golang |
| rustls | custom rustls programme and bash script<br>Version: rustls 0.23.19<br>Rustls programme: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/ech_url.rs`<br>Bash script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_rs.sh`<br>ECH implementation developed for rustls |
| python | custom Cpython build and python script<br>Version: python 3.13<br>Python script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/ech_url.py`<br>Bash script: `https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_py.sh`<br>ECH implementation based on OpenSSL |

There are some notes on these tests worth calling out in case someone wants to reproduce them:

- Some of the headless browser tests (e.g. with badly encoded ECHConfigList values) will cause selenium to throw exceptions, so test scripts need to catch those in order to properly determine that we got an ex-

pected failure

- Firefox will not attempt ECH to a name when the hostname of the test machine/VM is beneath the 2LD for that name, this causing unexpected failures if that hostname matches the origin of a test URL. In our case, we run both servers and test clients on the same VM ("test.defo.ie") so we have to set the hostname on the VM to something "peculiar" for tests to run as expected. (That of course causes "sudo" to complain which could disturb logs.)

- In the course of testing ECH in recent years, we have repeatedly made one misconfiguration error: using "." as the target name in HTTPS RRs for ports other then 443, when the correct thing do is to include the origin as the target name in such cases. That is when making an HTTPS RR for `https://example.com:12345/` one needs to publish an HTTPS RR at "_12345._https.example.com" and the (presentation) value of that record needs to start with "1 example.com ..." where 1 is the priority and example.com is that targetName. So simply copying the value of the HTTPS RR for port 443 and re-publishing that for another port is not sufficient for interoperability. (But also does work for some clients in some cases, making it easier to accidentally do this and not notice.)

## 4.1 Results

The set of 67 test URLs covers all of the server technologies listed in Table 2) though as all of those use OpenSSL for ECH, we direct most of the URLs towards one nginx instance. On our test VM, we have an haproxy instance (not doing an ECH) listening on port 443, that routes TLS sessions to the ECH-enabled web servers, namely, nginx (port 15443), apache (15444), lighttpd (15445) and two instances of "openssl s_server", one with a standard configuration on port 15447, and one, that only supports the p-384 curve for TLS key exchange on port 15448. That last server's configuration will trigger TLS HelloRetryRequest for many TLS clients. The back-end ports for all servers (e.g. 15443 for nginx) are also open. Our test URLs and associated HTTPS RRs can therefore be exercised both for port 443 and the relevant back-end port, but in most cases we have only set those up for the nginx back-end instance as testing the same setup on different web server instances wouldn't be likely to reveal much new information. The test set also include URLs for the test services listed in Table 3. Some test URLs have "broken" configurations, e.g. with badly encoded HTTPS resource records, or have specification-conformant configurations that we expect may not work.

We have 6 clients testing hourly against 67 URLs, giving us 402 measurements per hour. The full set of URLs are too long to explain in detail here so we'll just show current results and describe a few cases where those results are more "interesting". The full set of URLs and the last 6 hours of test outcomes at `https://test.defo.ie/smokeping-summary.php` or web pages with (most of) the test URLs loaded via an Iframe can be accessed at

https://test.defo.ie/iframe_tests_sort.html. That web page also includes some explanatory text about each test URL.

Table 5 below shows, for each test URL and each client, the ratio of expected and total results for a recent interval. [5]

Table 5: Interop tests from 2024-12-09 17:04:01.233448 to 2024-12-10 17:04:01.233448

| num | url | chromium | curl | firefox | golang | rustls | python |
|---|---|---|---|---|---|---|---|
| 1 | https://2thenp-ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.96/24 | 1.00/24 |
| 2 | https://2thenp-ng.test.defo.ie:15443/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 3 | https://ap.test.defo.ie/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 |
| 4 | https://ap.test.defo.ie:15444/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 5 | https://badalpn-ng.test.defo.ie/echstat.php?format=json | 0.96/24 | 1.00/24 | 0.00/24 | 1.00/24 | 0.96/24 | 1.00/24 |
| 6 | https://badalpn-ng.test.defo.ie:15443/echstat.php?format=json | 0.96/24 | 0.96/24 | 0.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 7 | https://bk1-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.25/24 | 0.25/24 |
| 8 | https://bk1-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.25/24 | 0.25/24 |
| 9 | https://bk2-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.25/24 | 0.25/24 |
| 10 | https://bk2-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.25/24 | 0.25/24 |
| 11 | https://bv-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.25/24 | 0.25/24 |
| 12 | https://bv-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.25/24 | 0.25/24 |
| 13 | https://cloudflare-ech.com/cdn-cgi/trace | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 0.00/24 |
| 14 | https://curves1-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.62/24 | 0.96/24 |
| 15 | https://curves1-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.58/24 | 1.00/24 |
| 16 | https://curves2-ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.33/24 | 0.96/24 |
| 17 | https://curves2-ng.test.defo.ie:15443/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.50/24 | 0.96/24 |
| 18 | https://curves3-ng.test.defo.ie/echstat.php?format=json | 0.00/24 | 1.00/24 | 0.00/24 | 0.00/24 | 0.54/24 | 1.00/24 |
| 19 | https://curves3-ng.test.defo.ie:15443/echstat.php?format=json | 0.00/24 | 1.00/24 | 0.00/24 | 0.00/24 | 0.38/24 | 1.00/24 |
| 20 | https://defo.ie/ech-check.php | 0.96/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 |
| 21 | https://draft-13.esni.defo.ie:10413/ | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 22 | https://draft-13.esni.defo.ie:11413/ | 0.92/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.92/24 | 1.00/24 |
| 23 | https://draft-13.esni.defo.ie:12413/ | 0.00/24 | 1.00/24 | 0.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 24 | https://draft-13.esni.defo.ie:12414/ | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 |
| 25 | https://draft-13.esni.defo.ie:8413/stats | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 26 | https://draft-13.esni.defo.ie:8414/stats | 0.88/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 |
| 27 | https://draft-13.esni.defo.ie:9413/ | 0.75/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 28 | https://h1alpn-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 |
| 29 | https://h1alpn-ng.test.defo.ie:15443/echstat.php?format=json | 0.88/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 30 | https://h2alpn-ng.test.defo.ie/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 31 | https://h2alpn-ng.test.defo.ie:15443/echstat.php?format=json | 0.88/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 32 | https://hidden.hoba.ie/ | 0.92/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 33 | https://longalpn-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 34 | https://longalpn-ng.test.defo.ie:15443/echstat.php?format=json | 0.33/24 | 1.00/24 | 1.00/24 | 0.92/24 | 1.00/24 | 0.96/24 |
| 35 | https://ly.test.defo.ie/echstat.php?format=json | 0.88/24 | 0.96/24 | 1.00/24 | 0.88/24 | 0.92/24 | 0.92/24 |
| 36 | https://many-ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 37 | https://many-ng.test.defo.ie:15443/echstat.php?format=json | 0.00/24 | 1.00/24 | 0.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 38 | https://min-ng.test.defo.ie/echstat.php?format=json | 0.88/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 39 | https://min-ng.test.defo.ie:15443/echstat.php?format=json | 0.88/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 40 | https://mixedalpn-ng.test.defo.ie/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |

Continued on next page

[5] Table 5 is produced using https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smokeping-summary-report.py.

Table 5: Interop tests from 2024-12-09 17:04:01.233448 to 2024-12-10 17:04:01.233448 (Continued)

| num | url | chromium | curl | firefox | golang | rustls | python |
|---|---|---|---|---|---|---|---|
| 41 | https://mixedalpn-ng.test.defo.ie:15443/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.96/24 | 1.00/24 |
| 42 | https://mixedmode-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 0.00/24 | 1.00/24 | 0.21/24 | 0.25/24 |
| 43 | https://mixedmode-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 0.00/24 | 1.00/24 | 0.21/24 | 0.25/24 |
| 44 | https://my-own.net/ech-check.php | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 45 | https://my-own.net:8443/ech-check.php | 0.79/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 46 | https://myechtest.site/ | 0.00/24 | 1.00/24 | 0.00/24 | 1.00/24 | 0.25/24 | 0.92/24 |
| 47 | https://ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 48 | https://ng.test.defo.ie:15443/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.92/24 | 1.00/24 |
| 49 | https://noaddr-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.25/24 | 0.92/24 |
| 50 | https://noaddr-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.25/24 | 0.92/24 |
| 51 | https://p256-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 0.12/24 | 0.83/24 | 0.00/24 | 0.00/24 | 0.42/24 |
| 52 | https://p256-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 0.12/24 | 0.83/24 | 0.00/24 | 0.00/24 | 0.42/24 |
| 53 | https://pthen2-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 54 | https://pthen2-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.92/24 |
| 55 | https://ss.test.defo.ie/stats | 0.96/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |
| 56 | https://sshrr.test.defo.ie/stats | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 57 | https://tls-ech.dev/ | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.00/24 |
| 58 | https://v1-ng.test.defo.ie/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.96/24 | 1.00/24 |
| 59 | https://v1-ng.test.defo.ie:15443/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 60 | https://v2-ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 61 | https://v2-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 62 | https://v3-ng.test.defo.ie/echstat.php?format=json | 0.88/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 63 | https://v3-ng.test.defo.ie:15443/echstat.php?format=json | 0.96/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 |
| 64 | https://v4-ng.test.defo.ie/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 0.21/24 | 0.25/24 |
| 65 | https://v4-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 0.25/24 | 0.25/24 |
| 66 | https://withext-ng.test.defo.ie/echstat.php?format=json | 0.92/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 | 1.00/24 |
| 67 | https://withext-ng.test.defo.ie:15443/echstat.php?format=json | 1.00/24 | 1.00/24 | 1.00/24 | 0.96/24 | 1.00/24 | 1.00/24 |

The set of "interesting" things arising from Table 5 include:

**TODO: Check text still applies after we've updated Table 5**

- Lines 5 and 6 - `https://badalpn-ng.test.defo.ie/echstat.php?format=json` and it's equivalent off port 443, work fine with all clients except Firefox and never work for firefox. The relevant DNS RR has an illegal alpn value (a '+' character). Hard to see this as other than firefox being overfussy, though there could of course be some good reason to reject such alpn values.

- Lines 13 and 57 - our python test client throws an exception for these (the Cloudflare operational, and boringssl test, servers)

  ```
  [SSL: SSLV3_ALERT_ILLEGAL_PARAMETER] ssl/tls alert
         illegal parameter (_ssl.c:1024)
  ```

  This is being investigated.

- Line 14 - `https://curves1-ng.test.defo.ie/echstat.php?format=json` has two HTTPS RRs published with the same priority, one that contains

an x25519 public value and another that only have a p256 public value. This seems to cause failure for our rustls client about 50% of the time as that client only supports x25519 (presumably). Browsers, while similarly not supporting NIST curves, presumably either select the "working" HTTPS RR value, and/or use the retry-configs fallback and manage to connect with ECH.

- Line 15 becomes a puzzle given line 14 - seemingly our rustls client does always work when not on port 443! We have verified that the HTTPS RRs published for curves1-ng.test.defo.ie and _15443._https.curves1-ng.test.defo.ie are as expected. Being investigated.

- Lines 18 and 19 may behave similarly to the above, being alike except that the HTTPS RR value priority for the p256 case is lower than for the 25519 one.

- Line 23 - browsers seem to dislike `https://draft-13.esni.defo.ie:12413/`. We suspect that's due to our test result handlers. That test case is an ECH shared-mode hapoxy instance with lighttpd as the backend. As lighttpd is the backend it doesn't produce an HTTP response indicates ECH worked as ECH was terminated with TLS at the haproxy intance. It's unclear how we can handle this test case when browsers don't indicate ECH success or failure. (Between haproxy and lighttpd we may be able to engineer some HTTP response header that indicates ECH success or failure - that'd not be secure but would be fine for testing. That's not been attempted yet.)

- Line 37 - it's not clear why publishing many (20) identical HTTPS RRs would work for port 443 for browsers but not off port 443. To be investigated.

**TODO: based on the above, re-generate Table 5 in some hours**

## 4.2 Recently Fixed Issues with Test Setup

It may be illustrative to describe some self-caused test failures seen as we were preparing this report, and how those were resolved:

- Line 2 - `https://2thenp-ng.test.defo.ie:15443/echstat.php?format=json` works for all clients except chromium where it never works, yet the same configuration on port 443 works 100% of the time on all clients. That configuration has an ECHConfigList with two entries - the first an x25519 key and the second a p256 key - and the same ECHConfigList value is published for both ports. The error here however is in the configuration - the targetName for the port 15443 RR is "." rather than the hostname, so chromium here is correct and our test setup is wrong. We fixed a bug in `https://github.com/defo-project/ech-dev-utils/`

blob/main/test-cases/test_cases_gen.py addressing this on 20241208.

- Line 7 and elsewhere - fixed a bug in https://github.com/defo-project/ech-dev-utils/blob/main/scripts/selenium_test.py where expected failures from tests using the "echstat_stat()" handler were being logged as test failures when they should be logged as expected.
- Line 7 and elsewhere - fixed same bug in both https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_py.sh and https://github.com/defo-project/ech-dev-utils/blob/main/scripts/smoke_ech_rs.sh where we were assuming all non-zero returns were errors, which didn't match with the configured expected values. (Also tweaked a few of those expected values for rustls and python too.)
- Line 13 - our "cf_check" result handler hadn't been changed to be applied with the change in Cloudflare's ECH test URL from "crypto.cloudflare.com" to "cloudflare-ech.com" which caused erroneous failure reports.
- Line 20 and similar - our python test client needed a tweak to not fail for tests not using the "echstat.php" server side script - fix was pushed on 20241209
- Line 32 - somehow certbot was no longer installed on test.defo.ie so we got a cert expiry for hidden.hoba.ie - re-instaled certbot to fix, which leads to some redundancy as we end up with more than one cert for some names (more or less all other name on that VM are below *.test.defo.ie)
- Line 35 - our python test client hits a certificate validation error for this lighttpd server - a change to the server configuration seems to fix this, the change being to send the full cert chain to clients (browsers don't need that but our other test client do).
- Lines 51 and 52: these needed an edit for the expected values, which were wrong.

# 5 Conclusions

Conclude


# Acknowledgements

Thanks to DEfO folks, OTF...

# References

[1] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, Aug. 2018. [Online]. Available: https://www.rfc-editor.org/info/rfc8446

[2] E. Rescorla, K. Oku, N. Sullivan, and C. A. Wood, "TLS Encrypted Client Hello," Internet Engineering Task Force, Internet-Draft draft-ietf-tls-esni-22, Sep. 2024, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-tls-esni/22/