A well-known URL for publishing ECHConfigLists

https://datatracker.ietf.org/doc/draft-ietf-tls-wkech/

Stephen Farrell, stephen.farrell@cs.tcd.ie

IETF114, Philadelphia, July 2022

25/07/22 06:56:16

Process Summary

- draft-ietf-tls-wkech-00 posted when submissions re-opened
  - Replaces draft-farrell-tls-wkesni
- Presented at IETF113 dispatch session, then mentioned on that list and the dnsop list before (a bit of) discussion and adoption by the tls WG
  - Changes are expected and welcome
- If you saw the IETF113 presentation, not much has changed but 2nd half of this deck deals with points raised during the adoption call
  - So I’ll try nip along pretty quickly to get to slide 11, let’s see if that works:-)

Technical Summary

- There are a bunch of ECH-enabled web servers at https://defo.ie/ (for interop testing)
- ECH keys are **updated regularly** (hourly, but that doesn't matter here...)
- That DNS setup doesn't use DDNS or provide an API the ECH-enabled frontend (aka the ECH public_name) can use to write to DNS for the backend (aka the "inner" SNI)
  - Other setups do have that or equivalent, or could use CNAME or a static AliasMode: this isn't meant to replace those
- There is a "zonefactory" machine (for the backend) that knows the names of the backend servers and polls the frontend for new ECHConfigLists
  - When it finds new keys it tests those work and if so modifies zone file and re-publishes the backend's DNS zone
- That benefits from a .well-known URL so this aims to specify that

Example

https://cover.defo.ie/.well-known/ech/draft-13.esni.defo.ie.json

^^^ public_name

^^^ name in inner CH SNI

Response at 2022-03-14T13:50: a JSON array...


```
[
    {
        "desired-ttl": 1800,
        "ports":  [ 8413,8414,9413,10413,11413 ],
        "echconfiglist":
"AQD+DQA8AgAgACCuXw02/lUWxgMiwhhZzjkP11LxoTwi4TLxDH/gMtVBIQAEAAEAAQANY292ZXIuZGVmby5pZQAA/
g0APL8AIAAg9yNI2MhNZrf7XJGeOUowNMJCTeVZJ7i+jP+mxds5znMABAABAAEADWNvdmVyLmRlZm8uaWUAAP4NADzoACAAIKhvKLr
j0yWuzZiRJZyYnwoH6EEFXLVr8QI4iEG4wXJCAAQAAQABAA1jb3Zlci5kZWZvLmllAAD+DQA8YQAgACCKnTfgeEF8xz/
SDTHmlcZHThsym3vybQbBFlQ6oaypMQAEAAEAAQANY292ZXIuZGVmby5pZQAA"
    }
]
```

Example HTTPS RR Resulting...

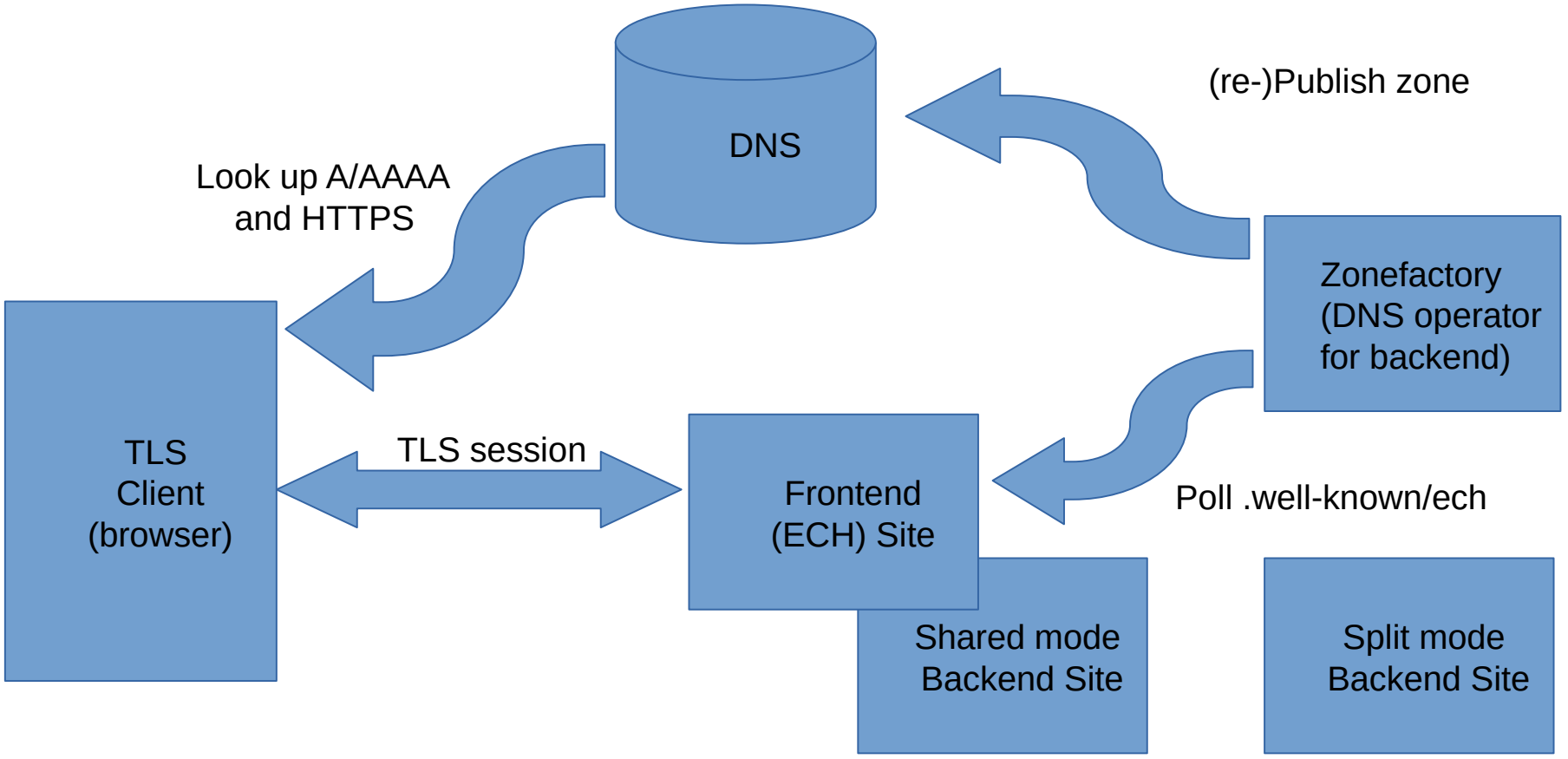
```
$ kdig +short https _8413._https.draft-13.esni.defo.ie
1 . ech=AQD+DQA8AgAgACCuXw02/lUWxgMiwhhZ
        zjkP11LxoTwi4TLxDH/gMtVBIQAEAAEA
        AQANY292ZXIuZGVmby5pZQAA/g0APBwA
        IAAgrAo6aAInG25QPzvDf4Oq3uRfpXNh
        fiyDTIY1Ylb0mm4ABAABAAEADWNvdmVy
        LmRlZm8uaWUAAP4NADxVACAAIKIaUipn
        r6YboBx8JBmA5AA5VMkgnTeuhpcr/QS2
        zrF5AAQAAQABAA1jb3Zlci5kZWZvLmll
        AAD+DQA8fwAgACD0eVURxRleTTCBi4FQ
        KFBKebhHASO2op6ehCj5GSbHAQAEAAEA
        AQANY292ZXIuZGVmby5pZQAA
```

CDN Scenario

- The defo.ie web-sites are small and just for interop testing, so don’t provide the hiding-in-crowds aspect of ECH, but CDNs might also benefit from this (not that I know what CDNs want:-) in a scenario like this…

- cdn.example.net is doing ECH for client web-sites (e.g. example.com)
- Some client web-sites don’t use the CDN as their DNS operator
- cdn.example.net wants to regularly update ECHConfigLists for example.com (and ~all other client web-sites)
- example.com’s DNS operator can poll the relevant URL once it knows that example.com uses cdn.example.net for ECH

Picture


DNS
(re-)Publish zone
Look up A/AAAA
and HTTPS
Zonefactory
(DNS operator
for backend)
TLS
Client
(browser)
TLS session
Frontend
(ECH) Site
Poll .well-known/ech
Shared mode
Backend Site
Split mode
Backend Site

It’s a working work-in-progress

- Possible/likely changes:
  - Change stuff as implementations evolve
  - The JSON response details will certainly need work when someone else looks at ‘em, and/or as some other zonefactory DNS tooling is used
  - Do the proper administrivia (.well-known registration, i18n…)
- Hopefully not much work, as it kinda just works
  - As of now, only for defo.ie but changing from ESNI to ECH wasn’t hard at all and an implementation can be done with pretty simple scripting
  - I plan to publish those scripts once tidied up
    - They currently have stuff dating back to ESNI draft-02:-)

Current HTTP response fields

- Top level content is a JSON Array; using >1 array element covers the case where e.g. different keys are used on different ports or with different desired-TTLs (not currently used at defo.ie but was once)
- “desired-TTL” – idea is that backend can use this to help ensure loose sync between key gen lifecycle and TTLs on RRs – nothing stops zonefactory ignoring the value, but that’d seem counterproductive, specifying more than a desired-TTL also seems OTT
- “ports” – I operate a bunch of different frontends on different ports at defo.ie as they use different web servers (apache, nginx, lighttpd, haproxy), not sure if that’s generally useful, and need to check how it’d play with alt-svc and more generic SVCB RRs but this is needed as ports other than 443 are reflected in the owner name for HTTPS RRs so the zonefactory needs port info
  - Allows zonefactory to verify that a new echconfiglist value actually works before publication in DNS
  - Aside: for each port, my zonefactory script explodes the list and checks each “singleton” ECH key works in isolation as well as checking the overall list (my TLS library takes a full list as an input, so this is needed to ensure that each key works by itself)
- “echconfiglist” - the base64 encoding seems right – in my case that contains a real list with current, previous and last-but-one keys (I hope other TLS ECH implementations all work with such lists:-)

Design points arising…

Most substantive comments so far were from Ben Schwartz (thanks!)

Using retry_configs

- An alternative design suggested...
- Use retry_configs result after getting ECH “wrong” - zonefactory could try e.g. a GREASEd ECH to frontend and get back retry_configs with the ServerHello, then check and publish based on that
- Doesn’t work for defo.ie because:
  - No backend-specifics in response – frontend only sees “inner” SNI if ECH worked, in which case no retry_configs will be sent
  - No TTL info
  - No ports info (zonefactory would need to probe based on already-known list or something)
  - My TLS server implementation currently only provides one (the “current”) ECH key in retry_configs and does not provide all loaded ECH keys (there could be **lots** in some scenarios). Requiring more would be a change to draft-ietf-tls-esni and may be unwise e.g. if there’re 1000 loaded keys
- It could be that changing the definition and handling of retry_configs could avoid some of the above problems; I’d prefer we not make such changes to draft-ietf-tls-esni, what do others think?
- I conclude we can assert that that design doesn’t work well enough
  - Or maybe I assert we can conclude that that design doesn’t work well enough:-)

Another RR somewhere...

- Another alternative design…
- Frontend could publish an RR with content like the current HTTP response (so like an SVCB-squared, gulp!) within it's own DNS zone and backend's zonefactory could poll there for the relevant information
- An RR that describes an SVCB seems odd (to me) and I guess could work for the zonefactory, BUT…
  - We'd lose the server authentication of the HTTP response which seems bad
  - It'd require something like this .well-known scheme between the frontend and the frontend's own zonefactory (so wouldn't work for e.g. defo.ie)
- Same conclusion as last one for me…

ECH or more generic?

- Is an ECH-specific well-known URL as proposed here more or less useful than something more generic?
- Are ECHConfigLists the main things likely to be changed frequently in SVCB/HTTPS RR values?
- My guess: the ECH frontend/backend thing is specific and simple enough that an ECH-specific mechanism is maybe correct
  - Trying to tackle the full generality of SVCB/HTTPS RRs this way seems… wrong, esp for SVCB cases where there just isn’t a frontend/backend distinction with frequent RR value changes

Add “alpn” to JSON?

- Including “alpn” as a (comma-sep) field in the JSON array elements (and hence in HTTPS or SVCB RRs) seems sensible **if** those would be used by applications doing ECH (my TLS library has an API allowing just that)
  - Do e.g. browsers make use of such ALPNs in inner CH from HTTPS or SVCB when doing ECH?
  - Not sure what to say about “no-default-alpn”
- IMO if answer is “yes” or “maybe” then worth including that here too
- Side note: ALPNs from the JSON are used in inner CH – ALPNs for the outer CH are up to the client; anything in DNS for outer CH ALPNs really ought be inside the ECHConfigContent.extensions, alongside the public_name (that’s confusing isn’t it;-)
  - Basis: the outer ALPNs should be the same for all backends so belong alongside the public_name and ECH key rather than differing from zone to zone where that ECHConfigList is used

Other inner CH content...

- Related question: what else might browsers or other clients consume from HTTPS/SVCB RRs that gets directly reflected in inner CH when doing ECH?
  - ...pauses for comment/suggestions...
- If that’s an empty list, great, we’re done (sooner)
- If not there’s probably a good case to allow inclusion of (some of) those in JSON array elements
  - IIUC some browsers replicate as much as possible in outer and inner CH so some care needed to not end up creating possible mis-matches affecting interop or leaks affecting confidentiality
- Suggests making the JSON array elements at least a bit extensible
  - My take: meh, but maybe yeah;-)
- I’m not sure how best that’d be done, if we want to do it (just text? An IANA registry?…)
- Caution: for zonefactory to be able to verify things will work, how do we handle case where zonefactory doesn’t know extension <foo> but frontend wants outer CH to use <foo> stuff?

The fullness of SVCBing...

- I’ve only done this for v. simple HTTPS RRs for very simple web servers (so no use of complicated SVCB options etc. so far)
- Need to think through and test more complex SVCB and HTTPS cases
- Need to think through and test more complex web setups (esp. CDN-like things)
- Given SCVB allows >1 way to do the same thing, I’d hope we don’t really end up reflecting too much of that in this spec
- Personally, I wonder how well clients will do with more complex SVCB cases, anyone got ideas there?
- Again though, we may be ok to only consider in-scope cases where the RRs published in the backend’s zone change frequently and affect ECH – in that case we’d e.g. not need to figure out how to tell the backend’s zonefactory to use a CNAME etc.
- I’ll definitely need help there but hope to get it as we go

Other points arising...

- Rob Sayre: Maybe don’t go mentioning shared/split mode “topologies”
- mnot: This doesn’t need .well-known (I think it’s nicer for zonefactory to only have to configure names rather than URLs)
- Ben Schwartz: The path in the URL is “wrong” (I don’t yet get why.)
- Rob Sayre (offlist): Make root JSON object a JSON Object {} for better extensibility if needed
- Me: HTTPS RRs can have a port-prefixed QNAME or have “port=” in the SvcParams – does that matter in a way that might affect our JSON?
- Me: not sure about IP address hints

Next steps...

- Side-meeting with those who commented during the adoption call to see if that set of people can bottom out on some of the topics above
  - The plan is that meeting just happened over lunch, but best laid plans and all that...
- Bring suggestions back to WG via a draft-01?
  - Either: not enough change yet to go to full githubbery issue tracking mode ‘till after -01, or else, there’ll be such major change in -01 that issue tracking could get in the way
  - So that plan’d call for getting stuff below the tlswg github org starting from draft-01
- Work on implementation/trials – who’d like to play? :-)
- Ultimately – aim to have this ready for WGLC whenever publication is requested for ECH?