Workgroup: TLS
Internet-Draft: draft-farrell-tls-wkesni-03
Published: 24 May 2022
Intended Status: Experimental
Expires: 25 November 2022
Authors: S. Farrell
Trinity College Dublin

A well-known URI for publishing ECHConfigList values.

Abstract

We propose use of a well-known URI at which web servers can publish ECHConfigList values as a way to help get those published in the DNS.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 November 2022.

Table of Contents

1. Introduction

Encrypted ClientHello (ECH) [I-D.ietf-tls-esni] for TLS1.3 [RFC8446] defines a confidentiality mechanism for server names and other ClientHello content in TLS. For many applications, that requires publication of ECHConflgList data structures in the DNS. An ECHConfigList structure contains a list of ECHConfig values. Each ECHConfig value contains the public component of a key pair that will typically be periodically (re-)generated by a web server. Many web infrastructures will have an API that can be used to dynamically update the DNS RR values containing ECHConfigList values. Some deployments however, will not, so web deployments could benefit from a mechanism to use in such cases.

We define such a mechanism here. Note that this is not intended for universal deployment, but rather for cases where the web server doesn't have write access to the relevant zone file (or equivalent). That zone file will eventually include an HTTPS or SVCB RR [I-D.ietf-dnsop-svcb-https] containing an ECHConfigList.

We use the term "zone factory" for the entity that does have write access to the zone file. We assume the zone factory (ZF) can also make HTTPS requests to the web server with the ECH keys.

We propose use of a well-known URI [RFC8615] on the web server that allows ZF to poll for changes to ECHConfigList values. For example, if a web server generates new ECHConfigList values hourly and publishes those at the well-known URI, ZF can poll that URI. When ZF sees new values, it can check if those work, and if they do, then update the zone file and re-publish the zone.

[[The source for this is in https://github.com/sftcd/wkesni/ PRs are welcome there too.]]

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Example use of the well-known URI for ECH

An example deployment could be as follows:

* Web server generates new ECHConfigList values hourly at N past the hour via a cronjob
* ECHConfigList values are "current" for an hour, published with a TTL of 1800, and remain usable for 3 hours from the time of generation
* Web server has a set of "backend" sites - the DNS name for each such site is here represented as $BACKEND, which will end up as an SNI value to be encrypted inside an ECH extension
* Web server has a "front-end" site ($FRONT), where $FRONT will typically be the DNS name used in the ECHConfigList public_name field for ECHConfig version 0xff0d
* A cronjob creates creates a JSON file for each backend site at https://$FRONT/.well-known/ech/$BACKEND.json
* Each JSON file contains an array with the ECHConfigList values values for that particular $BACKEND as shown in Figure 1 - the values in Figure 1 with ellipses are the values we want to eventually see in the DNS
* On the zone factory, a cronjob runs at N+3 past the hour, it knows all the names involved and checks to see if the content at those well-known URIs has changed or not
* If the content has changed the cronjob attempts to use the ECHConfigList values, and for each $BACKEND where that works, it updates the zone file and re-publishes the zone containing only the new ECHConfigList values

4. The ech well-known URI

When a web server ($FRONT) wants to publish ECHConfigList information for a backend site ($BACKEND) then it provides the JSON content defined in Section 5 at: https://$FRONT/.well-known/ech/$BACKEND.json

The well-known URI defined here MUST be an https URL and therefore the zone factory verifies the correct $FRONT is being accessed. If there is any failure in accessing the well-known URI, then the zone factory MUST NOT modify the zone.

5. The JSON structure for ECHConfigList values

[[Since the specifics of the JSON structure in Figure 1 are very likely to change, this is mostly TBD. What is here for now, is what the author has currently implemented simply because it worked ok and was easy to do:-) One issue raised as a result of the dispatch presentation is whether or not anything beyond the ECHConfigList might make sense to represent in the JSON response. One example could be the inner ClientHello ALPN extension, if that might somehow be useful to the TLS cilent (which really should know in that case). The scope in that respect and the correct level of generality to cover here is something to consider as this evolves.]]

```
[
    {
        "desired-ttl": 1800,
        "ports": [ 443, 8413 ],
        "echconfiglist": "AD7+DQA65wAgAC..AA=="
    },
    {
        "desired-ttl": 1800,
        "ports": [ 443, 8413 ],
        "echconfiglist": "AD7+DQA65wAgAC..AA=="
    }
]
```

Figure 1: Sample JSON

The JSON file at the well-known URI MUST contain an array with one or more elements. Each element of the array MUST have these fields:

- *desired-ttl: contains a number indicating the TTL that the web server would like to see used for this RR. The zone factory MUST NOT use a longer TTL.
- *ports: this has a list of the TCP ports on which the web server with the relevant key pair will listen (needed to produce the correct zone file).
- *ECHConfigList: contains the value to be used as a base64 encoded string.

The JSON file contains an array for a couple of reasons:

- *As TLS authentication doesn't really distinguish ports, servers on the same host could in any case cheat on one another, so we may as well just read one JSON file per name.
- *Different ports could map to different sets of ECHConfig values
- *As ECHConfigList is (regrettably:-) an extensible structure, it may be necessary to publish different ECHConfigList values to get best interoperability.

6. Zone factory behaviour

The zone factory SHOULD check that the presented ECHConfigList values work with the $BACKEND server before publication. A "special" TLS client may be needed for this check, that does not require the ECHConfigList value to have already been published in the DNS. [[I guess that calls for the zone factory to know of a "safe" URL on $BACKEND to try, or maybe it could use HTTP HEAD? Figuring that out is TBD. The ZF could also try a GREASEd ECH and see if the retry-configs it gets back is one of the ECHConfig values in the ECHConfigList.]]

A careful zone factory could explode the ECHConfigList value presented into "singleton" values with one public key in each and test each for each port claimed.

The zone factory SHOULD publish all the ECHConfigList values that are presented in the JSON file, and that pass the check above.

The zone factory SHOULD only publish ECHConfigList values that are in the latest version of the JSON file. This leaves the control of "expiry" with the web server, so long as the ECHConfigList values presented actually work. [[An alternative could be to have the new values just be appended to the zone, but that'd require some form of "notAfter" value in the JSON file which seems unnecessary and more complex.]]

The SCVB and HTTPS RR specification [I-D.ietf-dnsop-svcb-https] defines how and where the ECHConfigList values for $BACKEND needs to be published in the DNS. The zone factory is assumed to be in control of how ECHConfigList values are included in such RRs.

A possibly interesting (unintended) consequence of this design is that once a TLS client has first gotten an ECHConfigList from the DNS for $BACKEND with the ECHConfigList structure containing the public_name field, the TLS client would know both $FRONT and $BACKEND and so could later probe for this .well-known as an alternative to doing so via DoT/DoH. Probably not something a web browser might do, but could be fun for other applications maybe.

[[The extent to which retry-configs could be used for a similar purpose might be worth considering. But the JSON stuff here may still be needed if implementations (such as mine:-) tend to only return one ECHConfig in retry-configs.]]

7. Security Considerations

This document defines another way to publish ECHConfigList values. If the wrong keys were read from here and published in the DNS, then clients using ECH would do the wrong thing, likely resulting in

denial of service, or a privacy leak, or worse, when TLS clients attempt to use ECH with a backend web site. So: Don't do that:-)

8. Acknowledgements

Thanks to Niall O'Reilly for a quick review of -00.

9. IANA Considerations

[[TBD: IANA registration of a .well-known. Also TBD - how to handle I18N for $FRONT and $BACKEND within such a URL.]]

10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[I-D.ietf-tls-esni] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft, draft-ietf-tls-esni-14, 13 February 2022, <https://www.ietf.org/archive/id/draft-ietf-tls-esni-14.txt>.

[I-D.ietf-dnsop-svcb-https] Schwartz, B., Bishop, M., and E. Nygren, "Service binding and parameter specification via the DNS (DNS SVCB and HTTPS RRs)", Work in Progress, Internet-Draft, draft-ietf-dnsop-svcb-https-09, 6 May 2022, <https://www.ietf.org/archive/id/draft-ietf-dnsop-svcb-https-09.txt>.

Appendix A. Change Log

[[RFC editor: please remove this before publication.]]

From -02 to -03:

*noted scope issue

From -01 to -02:

*General changes from ESNI to ECH.

From -00 to -01:

*Re-structured a bit after re-reading rfc8615

Author's Address

Stephen Farrell
Trinity College Dublin
Dublin
2
Ireland

Phone: +353-1-896-2354
Email: stephen.farrell@cs.tcd.ie