Workgroup: UTA
Internet-Draft:
draft-ietf-uta-tls13-iot-profile-05
Updates: 7925 (if approved)
Published: 6 July 2022
Intended Status: Standards Track
Expires: 7 January 2023
Authors: H. Tschofenig T. Fossati
Arm Limited Arm Limited

TLS/DTLS 1.3 Profiles for the Internet of Things

Abstract

This document is a companion to RFC 7925 and defines TLS/DTLS 1.3 profiles for Internet of Things devices. It also updates RFC 7925 with regards to the X.509 certificate profile.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/thomas-fossati/draft-tls13-iot.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
   1.1. Conventions and Terminology
2. Credential Types
3. Error Handling
4. Session Resumption
5. Compression
6. Perfect Forward Secrecy
7. Keep-Alive
8. Timeouts
9. Random Number Generation
10. Server Name Indication
11. Maximum Fragment Length Negotiation
12. Crypto Agility
13. Key Length Recommendations
14. 0-RTT Data
15. Certificate Profile
   15.1. All Certificates
      15.1.1. Version
      15.1.2. Serial Number
      15.1.3. Signature
      15.1.4. Issuer
      15.1.5. Validity
      15.1.6. subjectPublicKeyInfo
   15.2. Root CA Certificate
   15.3. Subordinate CA Certificate
   15.4. End Entity Certificate
      15.4.1. Client Certificate Subject
16. Certificate Revocation Checks
17. Certificate Overhead
18. Ciphersuites
19. Fault Attacks on Deterministic Signature Schemes
20. Open Issues
21. Security Considerations
22. IANA Considerations
23. References
   23.1. Normative References
   23.2. Informative References

Acknowledgments

Authors' Addresses

1. Introduction

This document defines a profile of DTLS 1.3 [DTLS13] and TLS 1.3 [RFC8446] that offers communication security services for IoT applications and is reasonably implementable on many constrained devices. Profile thereby means that available configuration options and protocol extensions are utilized to best support the IoT environment.

For IoT profiles using TLS/DTLS 1.2 please consult [RFC7925]. This document re-uses the communication pattern defined in [RFC7925] and makes IoT-domain specific recommendations for version 1.3 (where necessary).

TLS 1.3 has been re-designed and several previously defined extensions are not applicable to the new version of TLS/DTLS anymore. This clean-up also simplifies this document. Furthermore, many outdated ciphersuites have been omitted from the TLS/DTLS 1.3 specification.

1.1. Conventions and 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.

2. Credential Types

In accordance with the recommendations in [RFC7925], a compliant implementation MUST implement TLS_AES_128_CCM_8_SHA256. It SHOULD implement TLS_CHACHA20_POLY1305_SHA256.

Pre-shared key based authentication is integrated into the main TLS/DTLS 1.3 specification and has been harmonized with session resumption.

A compliant implementation supporting authentication based on certificates and raw public keys MUST support digital signatures with ecdsa_secp256r1_sha256. A compliant implementation MUST support the key exchange with secp256r1 (NIST P-256) and SHOULD support key exchange with X25519.

A plain PSK-based TLS/DTLS client or server MUST implement the following extensions:

* Supported Versions,
* Cookie,
* Server Name Indication (SNI),

* Pre-Shared Key,
* PSK Key Exchange Modes, and
* Application-Layer Protocol Negotiation (ALPN).

The SNI extension is discussed in this document and the justification for implementing and using the ALPN extension can be found in [RFC7525bis].

For TLS/DTLS clients and servers implementing raw public keys and/or certificates the guidance for mandatory-to-implement extensions described in Section 9.2 of [RFC8446] MUST be followed.

3. Error Handling

TLS 1.3 simplified the Alert protocol but the underlying challenge in an embedded context remains unchanged, namely what should an IoT device do when it encounters an error situation. The classical approach used in a desktop environment where the user is prompted is often not applicable with unattended devices. Hence, it is more important for a developer to find out from which error cases a device can recover from.

4. Session Resumption

TLS 1.3 has built-in support for session resumption by utilizing PSK-based credentials established in an earlier exchange.

5. Compression

TLS 1.3 does not have support for compression of application data traffic, as offered by previous versions of TLS. Applications are therefore responsible for transmitting payloads that are either compressed or use a more efficient encoding otherwise.

With regards to the handshake itself, various strategies have been applied to reduce the size of the exchanged payloads. TLS and DTLS 1.3 use less overhead, depending on the type of key confirmations, when compared to previous versions of the protocol. Additionally, the work on Compact TLS (cTLS) [I-D.ietf-tls-ctls] has taken compression of the handshake a step further by utilizing out-of-band knowledge between the communication parties to reduce the amount of data to be transmitted at each individual handshake, among applying other techniques.

6. Perfect Forward Secrecy

TLS 1.3 allows the use of PFS with all ciphersuites since the support for it is negotiated independently.

7. Keep-Alive

The discussion in Section 10 of [RFC7925] is applicable.

8. Timeouts

The recommendation in Section 11 of [RFC7925] is applicable. In particular this document RECOMMENDED to use an initial timer value of 9 seconds with exponential back off up to no less then 60 seconds.

9. Random Number Generation

The discussion in Section 12 of [RFC7925] is applicable with one exception: the ClientHello and the ServerHello messages in TLS 1.3 do not contain gmt_unix_time component anymore.

10. Server Name Indication

This specification mandates the implementation of the Server Name Indication (SNI) extension. Where privacy requirements require it, the Encrypted Client Hello extension [I-D.ietf-tls-esni] prevents an on-path attacker to determine the domain name the client is trying to connect to.

Note: To avoid leaking DNS lookups from network inspection altogether further protocols are needed, including DoH [RFC8484] and DPRIVE [RFC7858] [RFC8094]. Since the Encrypted Client Hello extension requires use of Hybrid Public Key Encryption (HPKE) [I-D.irtf-cfrg-hpke] and additional protocols require further protocol exchanges and cryptographic operations, there is a certain amount of overhead associated with this privacy property.

11. Maximum Fragment Length Negotiation

The Maximum Fragment Length Negotiation (MFL) extension has been superseded by the Record Size Limit (RSL) extension [RFC8449]. Implementations in compliance with this specification MUST implement the RSL extension and SHOULD use it to indicate their RAM limitations.

12. Crypto Agility

The recommendations in Section 19 of [RFC7925] are applicable.

13. Key Length Recommendations

The recommendations in Section 20 of [RFC7925] are applicable.

14. 0-RTT Data

Appendix E.5 of [TLS13] establishes that:

> Application protocols MUST NOT use 0-RTT data without a profile that defines its use. That profile needs to identify which messages or interactions are safe to use with 0-RTT and how to handle the situation when the server rejects 0-RTT and falls back to 1-RTT.

At the time of writing, no such profile has been defined for CoAP [CoAP]. Therefore 0-RTT MUST NOT be used by CoAP applications.

15. Certificate Profile

This section contains updates and clarifications to the certificate profile defined in [RFC7925]. The content of Table 1 of [RFC7925] has been split by certificate "type" in order to clarify exactly what requirements and recommendations apply to which entity in the PKI hierarchy.

15.1. All Certificates

15.1.1. Version

Certificates MUST be of type X.509 v3.

15.1.2. Serial Number

CAs SHALL generate non-sequential Certificate serial numbers greater than zero (0) containing at least 64 bits of output from a CSPRNG (cryptographically secure pseudo-random number generator).

15.1.3. Signature

The signature MUST be ecdsa-with-SHA256 or stronger [RFC5758].

15.1.4. Issuer

Contains the DN of the issuing CA.

15.1.5. Validity

No maximum validity period is mandated. Validity values are expressed in notBefore and notAfter fields, as described in Section 4.1.2.5 of [RFC5280]. In particular, values MUST be expressed in Greenwich Mean Time (Zulu) and MUST include seconds even where the number of seconds is zero.

Note that the validity period is defined as the period of time from notBefore through notAfter, inclusive. This means that a hypothetical certificate with a notBefore date of 9 June 2021 at 03:42:01 and a notAfter date of 7 September 2021 at 03:42:01 becomes valid at the beginning of the :01 second, and only becomes invalid at the :02 second, a period that is 90 days plus 1 second. So for a 90-day, notAfter must actually be 03:42:00.

In many cases it is necessary to indicate that a certificate does not expire. This is likely to be the case for manufacturer-provisioned certificates. RFC 5280 provides a simple solution to convey the fact that a certificate has no well-defined expiration date by setting the notAfter to the GeneralizedTime value of 99991231235959Z.

Some devices might not have a reliable source of time and for those devices it is also advisable to use certificates with no expiration date and to let a device management solution manage the lifetime of all the certificates used by the device. While this approach does not utilize certificates to its widest extent, it is a solution that extends the capabilities offered by a raw public key approach.

15.1.6. subjectPublicKeyInfo

The SubjectPublicKeyInfo structure indicates the algorithm and any associated parameters for the ECC public key. This profile uses the id-ecPublicKey algorithm identifier for ECDSA signature keys, as defined and specified in [RFC5480].

15.2. Root CA Certificate

* basicConstraints MUST be present and MUST be marked critical. The cA field MUST be set true. The pathLenConstraint field SHOULD NOT be present.
* keyUsage MUST be present and MUST be marked critical. Bit position for keyCertSign MUST be set.
* extendedKeyUsage MUST NOT be present.

15.3. Subordinate CA Certificate

* basicConstraints MUST be present and MUST be marked critical. The cA field MUST be set true. The pathLenConstraint field MAY be present.
* keyUsage MUST be present and MUST be marked critical. Bit position for keyCertSign MUST be set.
* extendedKeyUsage MUST NOT be present.

15.4. End Entity Certificate

* extendedKeyUsage MUST be present and contain at least one of id-kp-serverAuth or id-kp-clientAuth.
* keyUsage MAY be present and contain one of digitalSignature or keyAgreement.
* Domain names MUST NOT be encoded in the subject commonName, instead they MUST be encoded in a subjectAltName of type DNS-ID. Domain names MUST NOT contain wildcard (*) characters. subjectAltName MUST NOT contain multiple names.

15.4.1. Client Certificate Subject

The requirement in Section 4.4.2 of [RFC7925] to only use EUI-64 for client certificates is lifted.

If the EUI-64 format is used to identify the subject of a client certificate, it MUST be encoded in a subjectAltName of type DNS-ID as a string of the form HH-HH-HH-HH-HH-HH-HH-HH where 'H' is one of the symbols '0'-'9' or 'A'-'F'.

16. Certificate Revocation Checks

The considerations in Section 4.4.3 of [RFC7925] hold.

Since the publication of RFC 7925 the need for firmware update mechanisms has been reinforced and the work on standardizing a secure and interoperable firmware update mechanism has made substantial progress, see [I-D.ietf-suit-architecture]. RFC 7925 recommends to use a software / firmware update mechanism to provision devices with new trust anchors.

The use of device management protocols for IoT devices, which often include an onboarding or bootstrapping mechanism, has also seen considerable uptake in deployed devices and these protocols, some of which are standardized, allow provision of certificates on a regular basis. This enables a deployment model where IoT device utilize end-entity certificates with shorter lifetime making certificate revocation protocols, like OCSP and CRLs, less relevant.

Hence, instead of performing certificate revocation checks on the IoT device itself this specification recommends to delegate this task to the IoT device operator and to take the necessary action to allow IoT devices to remain operational.

17. Certificate Overhead

In a public key-based key exchange, certificates and public keys are a major contributor to the size of the overall handshake. For example, in a regular TLS 1.3 handshake with minimal ECC certificates and no subordinate CA utilizing the secp256r1 curve with mutual authentication, around 40% of the entire handshake payload is consumed by the two exchanged certificates.

Hence, it is not surprising that there is a strong desire to reduce the size of certificates and certificate chains. This has lead to various standardization efforts. Here is a brief summary of what options an implementer has to reduce the bandwidth requirements of a public key-based key exchange:

* Use elliptic curve cryptography (ECC) instead of RSA-based certificate due to the smaller certificate size.
* Avoid deep and complex CA hierarchies to reduce the number of subordinate CA certificates that need to be transmitted.
* Pay attention to the amount of information conveyed inside certificates.
* Use session resumption to reduce the number of times a full handshake is needed. Use Connection IDs [DTLS-CID], when possible, to enable long-lasting connections.
* Use the TLS cached info [RFC7924] extension to avoid sending certificates with every full handshake.
* Use client certificate URLs [RFC6066] instead of full certificates for clients.
* Use certificate compression as defined in [I-D.ietf-tls-certificate-compression].
* Use alternative certificate formats, where possible, such as raw public keys [RFC7250] or CBOR-encoded certificates [I-D.ietf-cose-cbor-encoded-cert].

The use of certificate handles, as introduced in cTLS [I-D.ietf-tls-ctls], is a form of caching or compressing certificates as well.

Whether to utilize any of the above extensions or a combination of them depends on the anticipated deployment environment, the availability of code, and the constraints imposed by already deployed infrastructure (e.g., CA infrastructure, tool support).

18. Ciphersuites

Section 4.5.3 of [DTLS13] flags AES-CCM with 8-octet authentication tags (CCM_8) as unsuitable for general use with DTLS. In fact, due to its low integrity limits (i.e., a high sensitivity to forgeries), endpoints that negotiate ciphersuites based on such AEAD are susceptible to a trivial DoS. (See also Section 5.3 and 5.4 of [I-D.irtf-cfrg-aead-limits] for further discussion on this topic, as well as references to the analysis supporting these conclusions.)

Specifically, [DTLS13] warns that:

> "TLS_AES_128_CCM_8_SHA256 MUST NOT be used in DTLS without additional safeguards against forgery. Implementations MUST set usage limits for AEAD_AES_128_CCM_8 based on an understanding of any additional forgery protections that are used."

Since all the ciphersuites mandated by [RFC7925] and [CoAP] are based on CCM_8, there is no stand-by ciphersuite to use for applications that want to avoid the security and availability risks associated with CCM_8 while retaining interoperability with the rest of the ecosystem.

In order to ameliorate the situation, this document RECOMMENDS that implementations support the following two ciphersuites:

* TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
* TLS_ECDHE_ECDSA_WITH_AES_128_CCM

and offer them as their first choice. These ciphersuites provide confidentiality and integrity limits that are considered acceptable in the most general settings. For the details on the exact bounds of both ciphersuites see Section 4.5.3 of [DTLS13]. Note that the GCM-based ciphersuite offers superior interoperability with cloud services at the cost of a slight increase in the wire and peak RAM footprints.

When the GCM-based ciphersuite is used with TLS 1.2, the recommendations in Section 6.2.1 of [RFC7525bis] related to deterministic nonce generation apply. In addition, the integrity limits on key usage detailed in Section 4.4 of [RFC7525bis] also apply.

19. Fault Attacks on Deterministic Signature Schemes

A number of passive side-channel attacks as well as active fault-injection attacks (e.g., [Ambrose2017]) have been demonstrated that allow a malicious third party to gain information about the signing key if a fully deterministic signature scheme (e.g., [RFC6979] ECDSA or EdDSA [RFC8032]) is used.

Most of these attacks assume physical access to the device and are therefore especially relevant to smart cards as well as IoT deployments with poor or non-existent physical security.

In this security model, it is recommended to combine both randomness and determinism, for example, as described in [I-D.mattsson-cfrg-det-sigs-with-noise].

20. Open Issues

A list of open issues can be found at https://github.com/thomas-fossati/draft-tls13-iot/issues

21. Security Considerations

This entire document is about security.

22. IANA Considerations

This document makes no requests to IANA.

23. References

23.1. Normative References

[DTLS13] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/rfc/rfc9147>.

[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/rfc/rfc2119>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <https://www.rfc-editor.org/rfc/rfc5480>.

[RFC5758] Dang, Q., Santesson, S., Moriarty, K., Brown, D., and T. Polk, "Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA", RFC 5758, DOI 10.17487/RFC5758, January 2010, <https://www.rfc-editor.org/rfc/rfc5758>.

[RFC7525bis] Sheffer, Y., Saint-Andre, P., and T. Fossati, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", Work in Progress, Internet-Draft, draft-ietf-uta-rfc7525bis-09, 30 June 2022, <https://

datatracker.ietf.org/doc/html/draft-ietf-uta-rfc7525bis-09>.

[RFC7925] Tschofenig, H., Ed. and T. Fossati, "Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", RFC 7925, DOI 10.17487/RFC7925, July 2016, <https://www.rfc-editor.org/rfc/rfc7925>.

[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/rfc/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/rfc/rfc8446>.

[RFC8449] Thomson, M., "Record Size Limit Extension for TLS", RFC 8449, DOI 10.17487/RFC8449, August 2018, <https://www.rfc-editor.org/rfc/rfc8449>.

[TLS13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

23.2. Informative References

[Ambrose2017] Ambrose, C., Bos, J. W., Fay, B., Joye, M., Lochter, M., and B. Murray, "Differential Attacks on Deterministic Signatures", 2017, <https://eprint.iacr.org/2017/975.pdf>.

[CoAP] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/rfc/rfc7252>.

[DTLS-CID] Rescorla, E., Ed., Tschofenig, H., Ed., Fossati, T., and A. Kraus, "Connection Identifier for DTLS 1.2", RFC 9146, DOI 10.17487/RFC9146, March 2022, <https://www.rfc-editor.org/rfc/rfc9146>.

[I-D.ietf-cose-cbor-encoded-cert]
Mattsson, J. P., Selander, G., Raza, S., Höglund, J., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-03, 10 January 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-cbor-encoded-cert-03>.

[I-D.ietf-suit-architecture] Moran, B., Tschofenig, H., Brown, D., and M. Meriac, "A Firmware Update Architecture for Internet of Things", Work in Progress, Internet-Draft, draft-ietf-suit-architecture-16, 27 January 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-suit-architecture-16>.

[I-D.ietf-tls-certificate-compression] Ghedini, A. and V. Vasiliev, "TLS Certificate Compression", Work in Progress, Internet-Draft, draft-ietf-tls-certificate-compression-10, 6 January 2020, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-certificate-compression-10>.

[I-D.ietf-tls-ctls] Rescorla, E., Barnes, R., and H. Tschofenig, "Compact TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-ctls-05, 7 March 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-ctls-05>.

[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://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-14>.

[I-D.irtf-cfrg-aead-limits] Günther, F., Thomson, M., and C. A. Wood, "Usage Limits on AEAD Algorithms", Work in Progress, Internet-Draft, draft-irtf-cfrg-aead-limits-04, 7 March 2022, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aead-limits-04>.

[I-D.irtf-cfrg-hpke] Barnes, R. L., Bhargavan, K., Lipp, B., and C. A. Wood, "Hybrid Public Key Encryption", Work in Progress, Internet-Draft, draft-irtf-cfrg-hpke-12, 2 September 2021, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-hpke-12>.

[I-D.mattsson-cfrg-det-sigs-with-noise] Mattsson, J. P., Thormarker, E., and S. Ruohomaa, "Deterministic ECDSA and EdDSA Signatures with Additional Randomness", Work in Progress, Internet-Draft, draft-mattsson-cfrg-det-sigs-with-noise-04, 15 February 2022, <https://datatracker.ietf.org/doc/html/draft-mattsson-cfrg-det-sigs-with-noise-04>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/rfc/rfc6066>.

[RFC6979]
Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 6979, DOI 10.17487/RFC6979, August 2013, <https://www.rfc-editor.org/rfc/rfc6979>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <https://www.rfc-editor.org/rfc/rfc7250>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/rfc/rfc7858>.

[RFC7924] Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", RFC 7924, DOI 10.17487/RFC7924, July 2016, <https://www.rfc-editor.org/rfc/rfc7924>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/rfc/rfc8032>.

[RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <https://www.rfc-editor.org/rfc/rfc8094>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/rfc/rfc8484>.

Acknowledgments

We would like to thank Ben Kaduk and John Mattsson.

Authors' Addresses

Hannes Tschofenig
Arm Limited

Email: Hannes.Tschofenig@gmx.net

Thomas Fossati
Arm Limited

Email: Thomas.Fossati@arm.com