Stream: Internet Engineering Task Force (IETF)
RFC: 9973
Obsoletes: 8773
Category: Standards Track
Published: April 2026
ISSN: 2070-1721
Author: R. Housley
Vigil Security

RFC 9973
TLS 1.3 Extension for Using Certificates with an External Pre-Shared Key

Abstract

This document specifies a TLS 1.3 extension that allows TLS clients and servers to authenticate with certificates and provide confidentiality based on encryption with a symmetric key from the usual key agreement algorithm and an external pre-shared key (PSK). This Standards Track RFC obsoletes RFC 8773, which was an Experimental RFC.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9973.

Table of Contents

1. Introduction

The TLS 1.3 [RFC9846] handshake protocol provides two mutually exclusive forms of server authentication. First, the server can be authenticated by providing a signature certificate and creating a valid digital signature to demonstrate that it possesses the corresponding private key. Second, the server can be authenticated by demonstrating that it possesses a pre-shared key (PSK) that was established by a previous handshake. A PSK that is established in this fashion is called a resumption PSK. A PSK that is established by any other means is called an external PSK.

A TLS 1.3 server that is authenticating with a certificate may optionally request a certificate from the TLS 1.3 client for authentication, as described in Section 4.3.2 of [RFC9846].

This document specifies a TLS 1.3 extension permitting certificate-based authentication and providing an external PSK to be input to the TLS 1.3 key schedule.

Please see Appendix A for a list of changes since the publication of [RFC8773].

2. Requirements Language

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. Motivation and Design Rationale

There are two motivations for using a certificate with an external PSK.

One motivation is confidentiality protection against the future invention of a Cryptographically Relevant Quantum Computer (CRQC), which would pose a serious challenge for the asymmetric cryptographic algorithms that are widely deployed today, including the digital signature algorithms that are used to authenticate the server in the TLS 1.3 handshake protocol and key agreement algorithm used to establish a pairwise shared secret between the client and server. It is an open question whether or not it is feasible to build such a quantum computer and, if so, when that might happen. However, if such a quantum computer is invented, many of the cryptographic algorithms and the security protocols that use them would become vulnerable. In particular, the TLS 1.3 handshake protocol employs key agreement algorithms that could be broken by the invention of a CRQC [PQC]. Including a strong external PSK in the TLS 1.3 key schedule offers confidentiality protection against the long-term quantum computing threat; it requires the attacker to learn the external PSK as well as the shared secret produced by the key agreement algorithm to break confidentiality.

The term "strong external PSK" is used to mean that the PSK that has been generated and distributed in such a way that the invention of a CRQC will not allow the owner of that quantum computer to learn the PSK. While generation and distribution of the PSK are outside the scope of this document, in the context of a CRQC, security of the TLS 1.3 session using the strong external PSK relies on and is implicitly tied to the confidentiality, entropy, and authenticity of the PSK. Thus, the security claims in this document depend on generation and distribution of the strong external PSK.

When a certificate is used for authentication, the authentication is provided by the existing certificate and digital signature mechanisms. This authentication cannot be relied upon if a CRQC is ever invented. The addition of a strong external PSK in the TLS 1.3 key schedule does not offer improvement against the long-term quantum computing threat regarding authentication.

Likewise, a raw public key can be provided, as described in [RFC7250].

Quantum-resistant public-key cryptographic algorithms are becoming standards, but it will take many years for TLS 1.3 ciphersuites that use these algorithms to be developed and deployed. In some environments, deployment of a strong external PSK provides protection until these quantum-resistant algorithms are deployed.

Another motivation is the use of a public key with a factory-provisioned secret value for the initial enrollment of a device in an enterprise network [TLS-POK].

4. Extension Overview

This section provides a brief overview of the "tls_cert_with_extern_psk" extension.

The client includes the "tls_cert_with_extern_psk" extension in the ClientHello message. The "tls_cert_with_extern_psk" extension **MUST** be accompanied by the "key_share", "supported_groups", "psk_key_exchange_modes", and "pre_shared_key" extensions. Since the "tls_cert_with_extern_psk" extension is intended to be used only with initial handshakes, it **MUST NOT** be sent alongside the "early_data" extension. These extensions are all described in Section 4.2 of [RFC9846], which also requires the "pre_shared_key" extension to be the last extension in the ClientHello message.

If the client includes both the "tls_cert_with_extern_psk" extension and the "early_data" extension, then the server **MUST** terminate the connection with an "illegal_parameter" alert.

If the server is willing to use one of the external PSKs listed in the "pre_shared_key" extension and perform certificate-based authentication, then the server includes the "tls_cert_with_extern_psk" extension in the ServerHello message. The "tls_cert_with_extern_psk" extension **MUST** be accompanied by the "key_share" and "pre_shared_key" extensions. If none of the external PSKs in the list provided by the client is acceptable to the server, then the "tls_cert_with_extern_psk" extension is omitted from the ServerHello message.

When the "tls_cert_with_extern_psk" extension is successfully negotiated, the TLS 1.3 key schedule processing includes both the selected external PSK and the (EC)DHE shared secret value. (EC)DHE refers to Diffie-Hellman over either finite fields or elliptic curves. As a result, the Early Secret, Handshake Secret, and Main Secret (previously known as the Master Secret) values all depend upon the value of the selected external PSK. Of course, the Early Secret does not depend upon the (EC)DHE shared secret.

The authentication of the server and optional authentication of the client depend upon the ability to generate a signature that can be validated with the public key in their certificates. The authentication processing is not changed in any way by the selected external PSK. As a result, if a CRQC is ever invented, the digital signature algorithm will need to be replaced with a quantum-resistant one. Failure to do so will result in vulnerable entity and data authentication mechanisms.

Each external PSK is associated with a single hash algorithm, which is required by Section 4.2.11 of [RFC9846]. The hash algorithm **MUST** be set when the PSK is established, with a default of SHA-256.

5. Certificate with External PSK Extension

This section specifies the "tls_cert_with_extern_psk" extension, which **MAY** appear in the ClientHello message and ServerHello message. It **MUST NOT** appear in any other messages. The "tls_cert_with_extern_psk" extension **MUST NOT** appear in the ServerHello message unless the "tls_cert_with_extern_psk" extension appeared in the preceding ClientHello message. If an implementation recognizes the "tls_cert_with_extern_psk" extension and receives it in any other message, then the implementation **MUST** abort the handshake with an "illegal_parameter" alert.

The general extension mechanisms enable clients and servers to negotiate the use of specific extensions. Clients request extended functionality from servers with the extensions field in the ClientHello message. If the server responds with a HelloRetryRequest message, then the client sends another ClientHello message as described in Section 4.1.2 of [RFC9846], including the same "tls_cert_with_extern_psk" extension as the original ClientHello message, or aborts the handshake.

Many server extensions are carried in the EncryptedExtensions message; however, the "tls_cert_with_extern_psk" extension is carried in the ServerHello message. Successful negotiation of the "pre_shared_key" extension enables certificate verification to take place in addition to the inclusion of the external PSK in the key schedule. The external PSK is identified by the "key_share" extension, and the inclusion of the external PSK in the key schedule affects the key used for encryption. The "tls_cert_with_extern_psk" extension is only present in the ServerHello message if the server recognizes the "tls_cert_with_extern_psk" extension and the server possesses one of the external PSKs offered by the client in the "pre_shared_key" extension in the ClientHello message.

The Extension structure is defined in [RFC9846]; it is repeated here for convenience.

```
struct {
    ExtensionType extension_type;
    opaque extension_data<0..2^16-1>;
} Extension;
```

The "extension_type" identifies the particular extension type, and the "extension_data" contains information specific to the particular extension type.

This document specifies the "tls_cert_with_extern_psk" extension, adding one new type to ExtensionType:

```
enum {
    tls_cert_with_extern_psk(33), (65535)
} ExtensionType;
```

The "tls_cert_with_extern_psk" extension is relevant when the client and server possess an external PSK in common that can be used as an input to the TLS 1.3 key schedule. The "tls_cert_with_extern_psk" extension is essentially a flag to use the external PSK in the key schedule, and it has the following syntax:

```
struct {
    select (Handshake.msg_type) {
        case client_hello: Empty;
        case server_hello: Empty;
    };
} CertWithExternPSK;
```

5.1. Companion Extensions

Section 4 lists the extensions that are required to accompany the "tls_cert_with_extern_psk" extension. Most of those extensions are not impacted in any way by this specification. However, this section discusses the extensions that require additional consideration.

The "psk_key_exchange_modes" extension is defined in Section 4.2.9 of [RFC9846]. The "psk_key_exchange_modes" extension restricts the use of both the PSKs offered in this ClientHello and those that the server might supply via a subsequent NewSessionTicket. As a result, when the "psk_key_exchange_modes" extension is included in the ClientHello message, clients **MUST** include psk_dhe_ke mode. In addition, clients **MAY** also include psk_ke mode to support a subsequent NewSessionTicket. When the "psk_key_exchange_modes" extension is included in the ClientHello message, servers **MUST** select the psk_dhe_ke mode for the initial handshake. Servers **MUST** select a key exchange mode that is listed by the client for subsequent handshakes that include the resumption PSK from the initial handshake.

The "pre_shared_key" extension is defined in Section 4.2.11 of [RFC9846]. The syntax is repeated below for convenience. All of the listed PSKs **MUST** be external PSKs. If a resumption PSK is listed along with the "tls_cert_with_extern_psk" extension, the server **MUST** abort the handshake with an "illegal_parameter" alert.

```
struct {
    opaque identity<1..2^16-1>;
    uint32 obfuscated_ticket_age;
} PskIdentity;

opaque PskBinderEntry<32..255>;

struct {
    PskIdentity identities<7..2^16-1>;
    PskBinderEntry binders<33..2^16-1>;
} OfferedPsks;

struct {
    select (Handshake.msg_type) {
        case client_hello: OfferedPsks;
        case server_hello: uint16 selected_identity;
    };
} PreSharedKeyExtension;
```

"OfferedPsks" contains the list of PSK identities and associated binders for the external PSKs that the client is willing to use with the server.

The identities are a list of external PSK identities that the client is willing to negotiate with the server. Each external PSK has an associated identity that is known to the client and the server; the associated identities may be known to other parties as well. In addition, the binder validation (see below) confirms that the client and server have the same key associated with the identity.

The "obfuscated_ticket_age" is not used for external PSKs. As stated in Section 4.2.11 of [RFC9846], clients **SHOULD** set this value to 0, and servers **MUST** ignore the value.

The binders are a series of HMAC [RFC2104] values, one for each external PSK offered by the client, in the same order as the identities list. The HMAC value is computed using the binder_key, which is derived from the external PSK, and a partial transcript of the current handshake. Generation of the binder_key from the external PSK is described in Section 7.1 of [RFC9846]. The partial transcript of the current handshake includes a partial ClientHello up to and including the PreSharedKeyExtension.identities field, as described in Section 4.2.11.2 of [RFC9846].

The "selected_identity" contains the index of the external PSK identity that the server selected from the list offered by the client. As described in Section 4.2.11 of [RFC9846], the server **MUST** validate the binder value that corresponds to the selected external PSK, and if the binder does not validate, the server **MUST** abort the handshake with an "illegal_parameter" alert.

5.2. Authentication

When the "tls_cert_with_extern_psk" extension is successfully negotiated, authentication of the server depends upon the ability to generate a signature that can be validated with the public key. When the server uses a certificate, this is accomplished by the server sending the Certificate and CertificateVerify messages, as described in Sections 4.4.2 and 4.4.3 of [RFC9846]. Alternatively, the server can use a raw public key, as described in [RFC7250].

TLS 1.3 does not permit the server to send a CertificateRequest message when a PSK is being used. This restriction is removed when the "tls_cert_with_extern_psk" extension is negotiated, allowing certificate-based authentication for both the client and the server. If certificate-based client authentication is desired, this is accomplished by the client sending the Certificate and CertificateVerify messages as described in Sections 4.4.2 and 4.4.3 of [RFC9846].

5.3. Keying Material

Section 7.1 of [RFC9846] specifies the TLS 1.3 key schedule. The successful negotiation of the "tls_cert_with_extern_psk" extension requires the key schedule processing to include both the external PSK and the (EC)DHE shared secret value.

If the client and the server have different values associated with the selected external PSK identifier, then the client and the server will compute different values for every entry in the key schedule, which will lead to the client aborting the handshake with a "decrypt_error" alert.

6. IANA Considerations

IANA has updated the "TLS ExtensionType Values" registry [IANA] entry for the "tls_cert_with_extern_psk" extension to reference this document.

7. Security Considerations

The Security Considerations in [RFC9846] remain relevant.

TLS 1.3 [RFC9846] does not permit the server to send a CertificateRequest message when a PSK is being used. This restriction is removed when the "tls_cert_with_extern_psk" extension is offered by the client and accepted by the server. However, TLS 1.3 does not permit an external PSK to be used in the same fashion as a resumption PSK, and this extension preserves that restriction.

Implementations must protect the external PSK. Compromise of the external PSK will make the encrypted session content vulnerable to the future development of a CRQC. However, the generation, distribution, and management of the external PSKs is out of scope for this specification.

Implementers should not transmit the same content on a connection that is protected with an external PSK and a connection that is not. Doing so may allow an eavesdropper to correlate the connections, making the content vulnerable to the future invention of a CRQC.

Implementations must generate external PSKs with a secure key-management technique, such as pseudorandom generation of the key or derivation of the key from one or more other secure keys. The use of inadequate pseudorandom number generators (PRNGs) to generate external PSKs can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the external PSKs and search the resulting small set of possibilities, rather than brute-force searching the whole key space. The generation of quality random numbers is difficult. [RFC4086] offers important guidance in this area.

Implementations must use a ciphersuite that includes a symmetric encryption algorithm with sufficiently large keys. For protection against the future invention of a CRQC, the symmetric key needs to be at least 128 bits. While Grover's algorithm (described in Section 3.1 of [PQC]) allows a quantum computer to perform a brute force key search using quadratically fewer steps than would be required with classical computers, there are a number of mitigating factors suggesting that Grover's algorithm will not speed up a brute force symmetric key search as dramatically as one might suspect. First, quantum computing hardware will likely be more expensive to build and use than classical hardware. Second, to obtain the full quadratic speedup, all the steps of Grover's algorithm must be performed in series. However, attacks on cryptography use massively parallel processing; the advantage of Grover's algorithm will be smaller.

Implementations must use sufficiently large external PSKs. For protection against the future invention of a CRQC, the external PSK needs to be at least 128 bits.

TLS 1.3 [RFC9846] has received careful security analysis, and the following informal reasoning shows that the addition of this extension does not introduce any security defects in the threat model of a traditional adversary, that is, an adversary that does not have access to a CRQC. This extension requires the use of certificates for authentication, but the processing of certificates is unchanged by this extension. This extension requires an external PSK in the key schedule as part of the computation of the Early Secret. In the initial handshake without an external PSK in [RFC9846], the Early Secret is computed as:

```
Early Secret = HKDF-Extract(0, 0)
```

With this extension, the Early Secret is computed as:

```
Early Secret = HKDF-Extract(0, External PSK)
```

Any entropy contributed by the external PSK can only make the Early Secret better; the external PSK cannot make it worse. Thus, TLS 1.3 continues to meet well-studied confidentiality goals when this extension is used.

Even when the external PSK is not known to any party other than the client and the server, the external PSK **MUST NOT** be the sole basis for authentication. The reasoning is explained in Section 4.2 of [K2016]. The authentication of the server and optional authentication of the client depend upon the ability to generate a signature that can be validated with the public key in their certificates. The authentication processing is not changed in any way by the selected external PSK.

This external PSK preserves some confidentiality and authentication even if the (EC)DH key agreement is broken by cryptanalysis or the future invention of a CRQC. As long as the attacker does not know the PSK and the key derivation algorithm remains unbroken, the attacker cannot derive the session secrets, even if the attacker is able to compute the (EC)DH shared secret. While the ephemeral (EC)DH private key used during a given TLS 1.3 session is destroyed before the end of a session, the (EC)DH private key would nevertheless be recoverable due to the break

of the (EC)DH algorithm. However, a more general notion of "secrecy after key material is destroyed" would still be achievable using external PSKs, if they are managed in a way that ensures their destruction when they are no longer needed, and with the assumption that the symmetric algorithms remain safe against the invention of a CRQC.

The forward-secrecy advantages traditionally associated with ephemeral (EC)DH keys are not easily replaced by external PSKs. The confidentiality and authentication provided by the external PSK depend on whether the external PSK is used for more than one TLS 1.3 session and the parties that know the external PSK. Assuming the (EC)DH key agreement is broken:

- If the external PSK is used for a single TLS 1.3 session and it is known only by the client and server, then the usual TLS 1.3 confidentiality and authentication is provided, including the cryptographic separation between TLS 1.3 sessions. Of course, this places a significant burden on the generation and distribution of external PSKs.
- If the external PSK is used for more than one TLS 1.3 session and it is known only by the client and server, then the confidentiality is limited to the client and server, but there is no cryptographic separation between TLS 1.3 sessions.
- If the external PSK is used for more than one TLS 1.3 session and it is known by the client, server, and others, then the confidentiality is limited to the group that knows the external PSK, but there is no cryptographic separation between TLS 1.3 sessions.

This specification does not require that the external PSK is known only by the client and server. The external PSK may be known to a group. Since authentication depends on the public key in a certificate, knowledge of the external PSK by other parties does not enable impersonation. The authentication of the server and optional authentication of the client depend upon the ability to generate a signature that can be validated with the public key in their certificates. The authentication processing is not changed in any way by the selected external PSK.

Confidentiality depends on the shared secret from (EC)DH, so knowledge of the external PSK by other parties does not enable eavesdropping. However, group members can record the traffic of other members and then decrypt that traffic if they ever gain access to a CRQC. Also, when many parties know the external PSK, there are many opportunities for theft of the external PSK by an attacker. Once an attacker has the external PSK, if they ever gain access to a CRQC, they can decrypt stored traffic in the same manner as a legitimate group member.

TLS 1.3 key derivation makes use of the HMAC-based Key Derivation Function (HKDF) algorithm, which depends upon the HMAC [RFC2104] construction and a hash function. This extension provides the desired protection for the session secrets, as long as HMAC with the selected hash function is a pseudorandom function (PRF) [GGM1986].

TLS 1.3 [RFC9846] takes a conservative approach to PSKs; they are bound to a specific hash function and KDF. By contrast, TLS 1.2 [RFC5246] allows PSKs to be used with any hash function and the TLS 1.2 PRF. Thus, the safest approach is to use a PSK exclusively with TLS 1.2 or exclusively with TLS 1.3. Given one PSK, one can derive a PSK for exclusive use with TLS 1.2 and derive another PSK for exclusive use with TLS 1.3 using the mechanism specified in [RFC9258].

8. Privacy Considerations

Appendix F.6 of [RFC9846] discusses identity-exposure attacks on PSKs. Also, Appendix C.4 of [RFC9846] discusses tracking prevention. The guidance in these sections remain relevant.

If an external PSK identity is used for multiple connections, then an observer will generally be able track clients and/or servers across connections. The rotation of the external PSK identity or the use of the "encrypted_client_hello" extension [RFC9849] can mitigate this risk.

This extension makes use of external PSKs to improve resilience against attackers that gain access to a CRQC in the future and provides authentication for initial enrollment of devices in an enterprise network. This extension is always accompanied by the "pre_shared_key" extension to provide the PSK identities in plaintext in the ClientHello message. Passive observation of the these PSK identities will aid an attacker in tracking users or devices that make use of this extension.

9. References

9.1. 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>.

[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/info/rfc7250>.

[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>.

[RFC9846] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 9846, DOI 10.17487/RFC9846, January 2026, <https://www.rfc-editor.org/info/rfc9846>.

9.2. Informative References

[Err7598] RFC Errata, Erratum ID 7598, RFC 8773, <https://www.rfc-editor.org/errata/eid7598>.

[GGM1986] Goldreich, O., Goldwasser, S., and S. Micali, "How to construct random functions", Journal of the ACM (JACM), vol. 33, no. 4, pp. 792-807, DOI 10.1145/6490.6503, August 1986, <https://dl.acm.org/doi/10.1145/6490.6503>.

[IANA] IANA, "TLS ExtensionType Values", <https://www.iana.org/assignments/tls-extensiontype-values>.

[K2016] Krawczyk, H., "A Unilateral-to-Mutual Authentication Compiler for Key Exchange (with Applications to Client Authentication in TLS 1.3)", Cryptology ePrint Archive, Paper 2016/711, 2016, <https://eprint.iacr.org/2016/711>.

[PQC] Banerjee, A., Reddy.K, T., Schoinianakis, D., Hollebeek, T., and M. Ounsworth, "Post-Quantum Cryptography for Engineers", Work in Progress, Internet-Draft, draft-ietf-pquip-pqc-engineers-14, 25 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-pquip-pqc-engineers-14>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC8773] Housley, R., "TLS 1.3 Extension for Certificate-Based Authentication with an External Pre-Shared Key", RFC 8773, DOI 10.17487/RFC8773, March 2020, <https://www.rfc-editor.org/info/rfc8773>.

[RFC9258] Benjamin, D. and C. A. Wood, "Importing External Pre-Shared Keys (PSKs) for TLS 1.3", RFC 9258, DOI 10.17487/RFC9258, July 2022, <https://www.rfc-editor.org/info/rfc9258>.

[RFC9849] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", RFC 9849, DOI 10.17487/RFC9849, March 2026, <https://www.rfc-editor.org/info/rfc9849>.

[TLS-POK] Friel, O. and D. Harkins, "Bootstrapped TLS Authentication with Proof of Knowledge (TLS-POK)", Work in Progress, Internet-Draft, draft-ietf-emu-bootstrapped-tls-11, 1 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-emu-bootstrapped-tls-11>.

Appendix A. Changes Since RFC 8773

The status elevation from Experimental RFC to Standards Track RFC is the most significant change in this document.

In addition to minor editorial updates, which include a change to the title, the following changes were made:

- Correct the order of the arguments to HKDF-Extract when an external PSK is present.
- The client must include the "supported_groups" extension in the ClientHello message.
- Expand the motivation discussion to talk about protection against the future development of a CRQC and enrollment in enterprise networks.
- Separate the discussion of confidentiality and authentication. The inclusion of the external PSK offers some confidentiality protection against the future invention of a CRQC, but the external PSK does not improve authentication.
- Correct RFC Erratum 7598 [Err7598].
- Add a discussion of TLS Encrypted Client Hello to the Privacy Considerations section.
- Adopt terminology that has become widely accepted, such as CRQC and Main Secret (instead of Master Secret).
- Provide URLs for all references.

Acknowledgments

Many thanks to Liliya Akhmetzyanova, Roman Danyliw, Christian Huitema, Ben Kaduk, Geoffrey Keating, Hugo Krawczyk, Mirja Kühlewind, Nikos Mavrogiannopoulos, Nick Sullivan, Martin Thomson, and Peter Yee for their review and comments on the Internet-Drafts that eventually became RFC 8773; their efforts have improved the document.

Many thanks to Mike Bishop, Deb Cooley, Owen Friel, Britta Hale, Dan Harkins, Christian Huitema, Joe Mandel, John Preuß Mattsson, Eric Rescorla, Joe Salowey, Muhammad Usama Sardar, Paul Wouters, and Peter Yee for their review and comments on the updates to RFC 8773 that became this document; their efforts have improved the document.

Author's Address

Russ Housley
Vigil Security, LLC
516 Dranesville Road
Herndon, VA 20170
United States of America
Email: housley@vigilsec.com