Network Working Group
Internet-Draft
Updates: draft-ietf-tls-tls13 (if approved)
Intended status: Informational
Expires: December 16, 2018

D. Benjamin
Google

June 14, 2018

Universal PSKs for TLS
draft-davidben-tls-universal-psk-00

Abstract

This document describes universal PSKs (Pre-Shared Keys) for TLS. Universal PSKs abstract the TLS 1.3 requirement that each PSK can only be used with a single hash function. This allows PSKs to be provisioned without depending on details of the TLS negotiation, which may change as TLS evolves. Additionally, this document describes a compatibility profile for using TLS 1.3 with PSKs provisioned for the TLS 1.2 PSK mechanism.

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 December 16, 2018.

Table of Contents

1. Introduction 2
1.1. Requirements Language 3
2. Universal PSKs 3
3. Compatibility with TLS 1.2 PSKs 4
4. Security Considerations 5
5. IANA Considerations 6
6. Acknowledgements 6
7. Normative References 6
Author's Address 7

1. Introduction

TLS 1.3 [I-D.ietf-tls-tls13] provides a PSK mechanism to authenticate connections with symmetric keys provisioned externally to TLS. However, unlike the analogous mechanism in earlier versions of TLS [RFC4279], TLS 1.3 PSKs must be constrained to a single hash function.

While this constraint simplifies the analysis and does not hinder the resumption use case, it is cumbersome for external PSKs. It ties the PSK provisioning process to details of TLS. The application protocol configuring TLS is usually abstracted from TLS's details. In some cases, the underlying TLS implementation may even be updated without changes to the calling application.

Additionally, applications using TLS with PSKs typically require some PSK be negotiated, so parameter selection must follow the hash constraint. In contrast, applications using resumption typically allow the session to be declined in favor of a full handshake, so parameter selection may complete independently of this constraint. Switching the order of the selections for external PSKs adds implementation complexity and complicates analysis of the server's configuration.

This document resolves these issues by adding an extra key derivation step to reuse the same secret for all TLS 1.3 KDF hashes, including hashes to be defined in the future.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Universal PSKs

A universal PSK consists of the following:

- An identity. This is a public opaque byte string.
- A secret. This is a secret opaque byte string.
- A KDF hash function for use with HKDF [RFC5869]. Unless otherwise specified, this is SHA-256 [SHS].

In this section's diagrams, "0" refers to a string of zero bytes with length matching the KDF hash. Derive-Secret refers to the corresponding function defined in TLS 1.3, using the KDF hash.

A universal PSK is advertised in TLS 1.3 by including the identity and index in the "pre_shared_key" extension of the ClientHello and ServerHello, respectively. The binder value is computed as in TLS 1.3, however, the binder key is derived with the universal PSK's secret and KDF hash as follows:

```
extracted = HKDF-Extract(universal_psk, 0)
binder_key = Derive-Secret(extracted, "univ binder", identity)
```

Unlike other PSKs, a universal PSK may be negotiated with any cipher suite, including those using a different KDF hash than the PSK. When negotiated, the universal PSK's secret is used to derive a hash-specific TLS 1.3 PSK as follows:

If the negotiated cipher suite uses a SHA-256 KDF hash, the PSK is derived as follows:

```
extracted = HKDF-Extract(universal_psk, 0)
psk = Derive-Secret(extracted, "sha256 psk", identity)
```

If the negotiated cipher suite uses a SHA-384 KDF hash, the PSK is derived as follows:

```
extracted = HKDF-Extract(universal_psk, 0)
psk = Derive-Secret(extracted, "sha384 psk", identity)
```

These PSKs are used in the key schedule as specified in TLS 1.3, except that they are not used to derive the "binder_key" value, already derived above.

Future KDF hash algorithms added to TLS 1.3 MUST specify how to compute the derived PSK from a universal PSK. Future versions of TLS MUST specify how to negotiate a universal PSK and how to use it when negotiated. Note, however, all versions of TLS using the "pre_shared_key" extension to negotiate PSKs MUST use the same binder derivation, while the derived PSKs SHOULD be version-specific.

Universal PSKs are not defined for use with 0-RTT. 0-RTT requires specifying many negotiated TLS parameters, which is not compatible with the goals of this specification. However, a client MAY choose to offer a universal PSK alongside a resumption-based or other 0-RTT-compatible PSK. The universal PSK is then analogous to the full handshake option when resumption is declined.

Note that whether a PSK is a universal PSK is not explicitly negotiated in TLS. It is provisioned alongside the secret itself when the PSK is pre-shared. This would typically be specified in the application protocol.

3. Compatibility with TLS 1.2 PSKs

Universal PSKs are only defined for use with TLS 1.3 and future versions of TLS. New protocols using TLS and PSKs SHOULD require TLS 1.3 or later. However, this may not be possible for existing protocols already using PSKs with TLS 1.2. This section describes a compatibility profile for upgrading to TLS 1.3.

A PSK provisioned for TLS 1.2 and earlier MUST NOT be used either as a universal PSK secret or directly as a TLS 1.3 PSK. This would invalidate security analysis of the two protocols individually. Instead, these PSKs MAY be used to derive a universal PSK. The identity is the TLS 1.2 PSK's identity. The secret is derived using the TLS 1.2 PRF function described in Section 5 of [RFC5246] with SHA-256 as the hash function, as follows:

```
universal_psk = PRF(pre_master_secret, "universal psk", "")[:32]
```

"pre_master_secret" is specified with the structure below, setting "psk" to TLS 1.2 PSK and "other_secret" to a string of all zeroes of the same length as the TLS 1.2 PSK.

```
struct {
  opaque other_secret<0..2^16-1>
  opaque psk<0..2^16-1>
}
```

Note this encoding and derivation aligns with the PSK's conversion to a premaster secret and then a master secret in [RFC5246].

Applications using this derivation are necessarily impacted by portions of TLS 1.2. New applications without a TLS 1.2 legacy SHOULD NOT use this derivation and instead SHOULD provision universal PSKs directly. Applications using it SHOULD migrate to this state after migrating to TLS 1.3.

4. Security Considerations

The security analysis for TLS 1.3 relies on each PSK having a single use. Using a TLS 1.3 PSK with two different hashes or with TLS 1.2 means the same secret is used with different KDF functions, invalidating that analysis. Universal PSKs instead derive independent PSKs using different KDF labels, so each derived PSK continues to have only a single use. The PSK identity is additionally included in each derivation to give a stronger connection between the identity and PSK.

TLS 1.3's analysis also depends on the KDF and MAC used to compute the PSK binder being collision-resistant. This document uses the same derivation as TLS 1.3, but with a different label and initial secret, so the collision-resistance properties carry over.

In [RFC5246], TLS 1.2 PSKs are used in premaster secret to master secret derivation. Section 3 aligns with that derivation, using a different label so the secret is derived independently. Note, however, that TLS 1.2 PSKs are not always associated with a single hash function, so they depend on stronger assumptions about hash functions than TLS 1.3 PSKs. The compatibility derivation is unavoidably dependent on this as well. It uses SHA-256, but some TLS 1.2 cipher suites use SHA-384, and earlier versions of TLS use an MD5 and SHA-1 concatenation.

Additionally, labels in the TLS 1.2 PRF function are not delimited from the seed parameter when concatenated. The labels in use thus must not only be distinct, but also prefix-free. This document registers its new TLS 1.2 label in the TLS Exporter Label registry. This registry is required by [RFC5705] to be prefix-free.

5. IANA Considerations

This document updates the note in the TLS Exporter Label registry <https://www.iana.org/assignments/tls-parameters> to read as follows:

Note: (1) These entries are reserved and MUST NOT be used for the purpose described in RFC 5705, in order to avoid confusion with similar, but distinct, use in the referenced document.

It additionally registers the label "universal psk". The "Note" column is marked with (1).

6. Acknowledgements

The author would like to thank Karthikeyan Bhargavan, Matt Caswell, Eric Rescorla, and Victor Vasiliev for discussions and feedback which led to this design.

7. Normative References

[I-D.ietf-tls-tls13]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-28 (work in progress), March 2018.

[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>.

[RFC4279] Eronen, P., Ed. and H. Tschofenig, Ed., "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, DOI 10.17487/RFC4279, December 2005, <https://www.rfc-editor.org/info/rfc4279>.

[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>.

[RFC5705] Rescorla, E., "Keying Material Exporters for Transport Layer Security (TLS)", RFC 5705, DOI 10.17487/RFC5705, March 2010, <https://www.rfc-editor.org/info/rfc5705>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/info/rfc5869>.

[SHS] Dang, Q., "Secure Hash Standard", National Institute of Standards and Technology report, DOI 10.6028/nist.fips.180-4, July 2015.

Author's Address

David Benjamin
Google
355 Main St
Cambridge, MA 02142
USA

Email: davidben@google.com