tls
Internet-Draft
Intended status: Informational
Expires: 8 October 2020

R. Housley
Vigil Security
J. Hoyland
Cloudflare Ltd.
M. Sethi
Ericsson
C.A. Wood
6 April 2020

Guidance for External PSK Usage in TLS
draft-dt-tls-external-psk-guidance-01

Abstract

This document provides usage guidance for external Pre-Shared Keys (PSKs) in TLS. It lists TLS security properties provided by PSKs under certain assumptions and demonstrates how violations of these assumptions lead to attacks. This document also discusses PSK use cases, provisioning processes, and TLS stack implementation support in the context of these assumptions. It provides advice for applications in various use cases to help meet these assumptions.

Note to Readers

Source for this draft and an issue tracker can be found at https://github.com/tlswg/external-psk-design-team (https://github.com/tlswg/external-psk-design-team).

Table of Contents

1. Introduction

This document provides usage guidance for external Pre-Shared Keys (PSKs) in TLS. It lists TLS security properties provided by PSKs under certain assumptions and demonstrates how violations of these assumptions lead to attacks. This document also discusses PSK use cases, provisioning processes, and TLS stack implementation support in the context of these assumptions. It provides advice for applications in various use cases to help meet these assumptions.

The guidance provided in this document is applicable across TLS [RFC8446], DTLS [I-D.ietf-tls-dtls13], and Constrained TLS [I-D.rescorla-tls-ctls].

2. Conventions and Definitions

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. PSK Security Properties

The external PSK authentication mechanism in TLS implicitly assumes one fundamental property: each PSK is known to exactly one client and one server, and that these never switch roles. If this assumption is violated, then the security properties of TLS are severely weakened.

As discussed in Section 5, there are use cases where it is desirable for multiple clients or multiple servers to share a PSK. If this is done naively by having all members share a common key, then TLS only authenticates the entire group, and the security of the overall system is inherently rather brittle. There are a number of obvious weaknesses here:

1. Any group member can impersonate any other group member.

2. If a group member is compromised, then the attacker can impersonate any group member (this follows from property (1)).

3. If PSK without DH is used, then compromise of any group member allows the attacker to passively read all traffic.

In addition to these, a malicious non-member can reroute handshakes between honest group members to connect them in unintended ways, as detailed below.

Let the group of peers who know the key be "A", "B", and "C". The attack proceeds as follows:

1. "A" sends a "ClientHello" to "B".

2. The attacker intercepts the message and redirects it to "C".

3. "C" responds with a "ServerHello" to "A".

4. "A" sends a "Finished" message to "B". "A" has completed the handshake, ostensibly with "B".

5. The attacker redirects the "Finished" message to "C". "C" has completed the handshake, ostensibly with "A".

This attack violates the peer authentication property, and if "C" supports a weaker set of cipher suites than "B", this attack also violates the downgrade protection property. This rerouting is a type of identity misbinding attack [Krawczyk][Sethi]. Selfie attack [Selfie] is a special case of the rerouting attack against a group member that can act both as TLS server and client. In the Selfie attack, a malicious non-member reroutes a connection from the client to the server on the same endpoint.

Entropy properties of external PSKs may also affect TLS security properties. In particular, if a high entropy PSK is used, then PSK-only key establishment modes are secure against both active and passive attack. However, they lack forward security. Forward security may be achieved by using a PSK-DH mode.

In contrast, if a low entropy PSK is used, then PSK-only key establishment modes are subject to passive exhaustive search passive attacks which will reveal the traffic keys. PSK-DH modes are subject to active attacks in which the attacker impersonates one side. The exhaustive search phase of these attacks can be mounted offline if the attacker captures a single handshake using the PSK, but those attacks will not lead to compromise of the traffic keys for that connection because those also depend on the Diffie-Hellman (DH) exchange. Low entropy keys are only secure against active attack if a PAKE is used with TLS.

4. Privacy Properties

PSK privacy properties are orthogonal to security properties described in Section 3. Traditionally, TLS does little to keep PSK identity information private. For example, an adversary learns information about the external PSK or its identifier by virtue of it appearing in cleartext in a ClientHello. As a result, a passive adversary can link two or more connections together that use the same external PSK on the wire. Applications should take precautions when using external PSKs to mitigate these risks.

In addition to linkability in the network, external PSKs are intrinsically linkable by PSK receivers. Specifically, servers can link successive connections that use the same external PSK together. Preventing this type of linkability is out of scope, as PSKs are explicitly designed to support mutual authentication.

5. External PSK Use Cases and Provisioning Processes

Pre-shared Key (PSK) ciphersuites were first specified for TLS in 2005. Now, PSK is an integral part of the TLS version 1.3 specification [RFC8446]. TLS 1.3 also uses PSKs for session resumption. It distinguishes these resumption PSKs from external PSKs which have been provisioned out-of-band (OOB). Below, we list some example use-cases where pair-wise external PSKs with TLS have been used for authentication.

* Device-to-device communication with out-of-band synchronized keys. PSKs provisioned out-of-band for communicating with known identities, wherein the identity to use is discovered via a different online protocol.

* Intra-data-center communication. Machine-to-machine communication within a single data center or PoP may use externally provisioned PSKs, primarily for the purposes of supporting TLS connections with fast open (0-RTT data).

* Certificateless server-to-server communication. Machine-to-machine communication may use externally provisioned PSKs, primarily for the purposes of establishing TLS connections without requiring the overhead of provisioning and managing PKI certificates.

* Internet of Things (IoT) and devices with limited computational capabilities. [RFC7925] defines TLS and DTLS profiles for resource-constrained devices and suggests the use of PSK ciphersuites for compliant devices. The Open Mobile Alliance Lightweight Machine to Machine Technical Specification [LwM2M] states that LwM2M servers MUST support the PSK mode of DTLS.

* Use of PSK ciphersuites are optional when securing RADIUS [RFC2865] with TLS as specified in [RFC6614].

* The Generic Authentication Architecture (GAA) defined by 3GGP mentions that TLS-PSK can be used between a server and user equipment for authentication [GAA].

* Smart Cards. The electronic German ID (eID) card supports authentication of a card holder to online services with TLS-PSK [SmartCard].

There are also use cases where PSKs are shared between more than two entities. Some examples below (as noted by Akhmetzyanova et al.[Akhmetzyanova]):

* Group chats. In this use-case, the membership of a group is confirmed by the possession of a PSK distributed out-of-band to the group participants. Members can then establish peer-to-peer connections with each other using the external PSK. It is important to note that any node of the group can behave as a TLS client or server.

* Internet of Things (IoT). In this use-case, resource-constrained IoT devices act as TLS clients and share the same PSK. The devices use this PSK for quickly establishing connections with a central server. Such a scheme ensures that the client IoT devices are legitimate members of the system. To perform rare system specific operations that require a higher security level, the central server can request resource-intensive client authentication with the usage of a certificate after successfully establishing the connection with a PSK.

5.1. Provisioning Examples

* Many industrial protocols assume that PSKs are distributed and assigned manually via one of the following approaches: typing the PSK into the devices, or via web server masks (using a Trust On First Use (TOFU) approach with a device completely unprotected before the first login did take place). Many devices have very limited UI. For example, they may only have a numeric keypad or even less number of buttons. When the TOFU approach is not suitable, entering the key would require typing it on a constrained UI. Moreover, PSK production lacks guidance unlike user passwords.

* Some devices provision PSKs via an out-of-band, cloud-based syncing protocol.

* Some secrets may be baked into or hardware or software device components. Moreover, when this is done at manufacturing time, secrets may be printed on labels or included in a Bill of Materials for ease of scanning or import.

5.2. Provisioning Constraints

PSK provisioning systems are often constrained in application-specific ways. For example, although one goal of provisioning is to ensure that each pair of nodes has a unique key pair, some systems do not want to distribute pair-wise shared keys to achieve this. As another example, some systems require the provisioning process to embed application-specific information in either PSKs or their identities. Identities may sometimes need to be routable, as is currently under discussion for EAP-TLS-PSK.

6. Recommendations for External PSK Usage

Applications MUST use external PSKs that adhere to the following requirements:

1. Each PSK SHOULD be derived from at least 128 bits of entropy, MUST be at least 128 bits long, and SHOULD be combined with a DH exchange for forward secrecy. As discussed in Section 3, low entropy PSKs, i.e., those derived from less than 128 bits of entropy, are subject to attack and SHOULD be avoided. Low entropy keys are only secure against active attack if a Password Authenticated Key Exchange (PAKE) is used with TLS.

2. Each PSK MUST NOT be shared between with more than two logical nodes. As a result, an agent that acts as both a client and a server MUST use distinct PSKs when acting as the client from when it is acting as the server.

3. Nodes SHOULD use external PSK importers [I-D.ietf-tls-external-psk-importer] when configuring PSKs for a pair of TLS client and server.

4. Where possible the master PSK (that which is fed into the importer) SHOULD be deleted after the imported keys have been generated. This protects an attacker from bootstrapping a compromise of one node into the ability to attack connections between any node; otherwise the attacker can recover the master key and then re-run the importer itself.

6.1. Stack Interfaces

Most major TLS implementations support external PSKs. Stacks supporting external PSKs provide interfaces that applications may use when supplying them for individual connections. Details about existing stacks at the time of writing are below.

* OpenSSL and BoringSSL: Applications specify support for external PSKs via distinct ciphersuites. They also then configure callbacks that are invoked for PSK selection during the handshake. These callbacks must provide a PSK identity and key. The exact format of the callback depends on the negotiated TLS protocol version with new callback functions added specifically to OpenSSL for TLS 1.3 [RFC8446] PSK support. The PSK length is validated to be between [1, 256] bytes. The PSK identity may be up to 128 bytes long.

* mbedTLS: Client applications configure PSKs before creating a connection by providing the PSK identity and value inline.

Servers must implement callbacks similar to that of OpenSSL. Both PSK identity and key lengths may be between [1, 16] bytes long.

* gnuTLS: Applications configure PSK values, either as raw byte strings or hexadecimal strings. The PSK identity and key size are not validated.

* wolfSSL: Applications configure PSKs with callbacks similar to OpenSSL.

6.1.1. PSK Identity Encoding and Comparison

Section 5.1 of [RFC4279] mandates that the PSK identity should be first converted to a character string and then encoded to octets using UTF-8. This was done to avoid interoperability problems (especially when the identity is configured by human users). On the other hand, [RFC7925] advises implementations against assuming any structured format for PSK identities and recommends byte-by-byte comparison for any operation. TLS version 1.3 [RFC8446] follows the same practice of specifying the PSK identity as a sequence of opaque bytes (shown as opaque identity<1..2^16-1> in the specification). [RFC8446] also requires that the PSK identities are at least 1 byte and at the most 65535 bytes in length. Although [RFC8446] does not place strict requirements on the format of PSK identities, we do however note that the format of PSK identities can vary depending on the deployment:

* The PSK identity MAY be a user configured string when used in protocols like Extensible Authentication Protocol (EAP) [RFC3748]. gnuTLS for example treats PSK identities as usernames.

* PSK identities MAY have a domain name suffix for roaming and federation.

* Deployments should take care that the length of the PSK identity is sufficient to avoid obvious collisions.

6.1.2. PSK Identity Collisions

It is possible, though unlikely, that an external PSK identity may clash with a resumption PSK identity. The TLS stack implementation and sequencing of PSK callbacks influences the application's behaviour when identity collisions occur. When a server receives a PSK identity in a TLS 1.3 ClientHello, some TLS stacks execute the application's registered callback function before checking the stack's internal session resumption cache. This means that if a PSK identity collision occurs, the application will be given precedence over how to handle the PSK.

7. Security Considerations

It is NOT RECOMMENDED to share the same PSK between more than one client and server. However, as discussed in Section 5, there are application scenarios that may rely on sharing the same PSK among multiple nodes. [I-D.ietf-tls-external-psk-importer] helps in mitigating rerouting and Selfie style reflection attacks when the PSK is shared among multiple nodes. This is achieved by correctly using the node identifiers in the ImportedIdentity.context construct specified in [I-D.ietf-tls-external-psk-importer]. It is RECOMMENDED that each endpoint selects one globally unique identifier and uses it in all PSK handshakes. The unique identifier can, for example, be one of its MAC addresses, a 32-byte random number, or its Universally Unique IDentifier (UUID) [RFC4122]. Each endpoint SHOULD know the identifier of the other endpoint with which its wants to connect and SHOULD compare it with the other endpoint's identifier used in ImportedIdentity.context. It is however important to remember that endpoints sharing the same group PSK can always impersonate each other.

8. IANA Considerations

This document makes no IANA requests.

9. References

9.1. Normative References

[I-D.ietf-tls-dtls13]
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-dtls13-37, 9 March 2020, <http://www.ietf.org/internet-drafts/draft-ietf-tls-dtls13-37.txt>.

[I-D.ietf-tls-external-psk-importer]
Benjamin, D. and C. Wood, "Importing External PSKs for TLS", Work in Progress, Internet-Draft, draft-ietf-tls-external-psk-importer-03, 15 February 2020, <http://www.ietf.org/internet-drafts/draft-ietf-tls-external-psk-importer-03.txt>.

[I-D.rescorla-tls-ctls]
Rescorla, E., Barnes, R., and H. Tschofenig, "Compact TLS 1.3", Work in Progress, Internet-Draft, draft-rescorla-tls-ctls-04, 9 March 2020, <http://www.ietf.org/internet-drafts/draft-rescorla-tls-ctls-04.txt>.

[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>.

9.2. Informative References

[Akhmetzyanova] Akhmetzyanova, L., Alekseev, E., Smyshlyaeva, E., and A. Sokolov, "Continuing to reflect on TLS 1.3 with external PSK", 2019, <https://eprint.iacr.org/2019/421.pdf>.

[GAA] "TR33.919 version 12.0.0 Release 12", n.d., <https://www.etsi.org/deliver/etsi_tr/133900_133999/133919/12.00.00_60/tr_133919v120000p.pdf>.

[Krawczyk] Krawczyk, H., "SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols", Annual International Cryptology Conference. Springer, Berlin, Heidelberg , 2003, <https://link.springer.com/content/pdf/10.1007/978-3-540-45146-4_24.pdf>.

[LwM2M] "Lightweight Machine to Machine Technical Specification", n.d., <http://www.openmobilealliance.org/release/LightweightM2M/V1_0-20170208-A/OMA-TS-LightweightM2M-V1_0-20170208-A.pdf>.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <https://www.rfc-editor.org/info/rfc2865>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[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>.

[RFC6614] Winter, S., McCauley, M., Venaas, S., and K. Wierenga, "Transport Layer Security (TLS) Encryption for RADIUS", RFC 6614, DOI 10.17487/RFC6614, May 2012, <https://www.rfc-editor.org/info/rfc6614>.

[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/info/rfc7925>.

[Selfie] Drucker, N. and S. Gueron, "Selfie: reflections on TLS 1.3 with PSK", 2019, <https://eprint.iacr.org/2019/347.pdf>.

[Sethi] Sethi, M., Peltonen, A., and T. Aura, "Misbinding Attacks on Secure Device Pairing and Bootstrapping", Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security , 2019, <https://arxiv.org/pdf/1902.07550>.

[SmartCard] "Technical Guideline TR-03112-7 eCard-API-Framework - Protocols", 2015, <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03112/TR-03112-api_teil7.pdf?__blob=publicationFile&v=1>.

Appendix A. Acknowledgements

This document is the output of the TLS External PSK Design Team, comprised of the following members: Benjamin Beurdouche, Bjoern Haase, Christopher Wood, Colm MacCarthaigh, Eric Rescorla, Jonathan Hoyland, Martin Thomson, Mohamad Badra, Mohit Sethi, Oleg Pekar, Owen Friel, and Russ Housley.

Authors' Addresses

Russ Housley

Vigil Security

Email: housley@vigilsec.com

Jonathan Hoyland
Cloudflare Ltd.

Email: jonathan.hoyland@gmail.com

Mohit Sethi
Ericsson

Email: mohit@piuha.net

Christopher A. Wood

Email: caw@heapingbits.net