Workgroup: TLS
Internet-Draft:
draft-davidben-tls-key-share-prediction-00
Updates: 8446 (if approved)
Published: 25 September 2023
Intended Status: Standards Track
Expires: 28 March 2024
Authors: D. Benjamin
Google LLC

TLS Key Share Prediction

Abstract

This document clarifies an ambiguity in the TLS 1.3 key share selection, to avoid a downgrade when server assumptions do not match client behavior. It additionally defines a mechanism for servers to communicate key share preferences in DNS. Clients may use this information to reduce TLS handshake round-trips.

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 28 March 2024.

Table of Contents

1. Introduction

Most TLS [RFC8446] parameters are negotiated as follows: The client sends a list of supported options in preference order. Then, the server evaluates this against its own preferences to make a selection. The aim is to arrive at the best common option, without permitting attackers to downgrade to a weaker one. Newer clients and servers often support legacy options for compatibility with older peers. Downgrade-protected parameter selection reduces the security risk of those legacy options when both sides of a connection are newer.

Named groups in TLS 1.3 instead use two client lists. The client sends its full preferences in the supported_groups extension, but also generates key shares for a subset in the key_share extension. If the server selects a named group in this subset, the handshake may complete in one round trip. Otherwise, the handshake requires a HelloRetryRequest and two round trips. Unused key shares consume network and computational resources, so clients only predict a

subset of supported groups, balancing round-trip reduction against other concerns. This adds another dimension to server group selection.

[RFC8446] is ambiguous on the semantics of the key_share subset. Some existing servers assume it reflects client preferences, selecting named groups in key_share above all others. However, the concerns above mean clients may need to predict based on other factors. Where these interpretations conflict, the selection may be downgraded, potentially even under attacker influence.

This document resolves the ambiguity in three ways:

- It updates server behavior to clarify that key shares may not reflect client preferences
- For existing named groups, it recommends clients to predict key shares that reflect their preferences, for compatibility with servers that predate this document
- For future named groups, it mandates the updated server behavior, so that clients may predict key shares more flexibly

It is expected that all post-quantum key encapsulation methods (KEMs) will fall in the last category. Post-quantum KEMs have large keys and ciphertexts, so bandwidth concerns are particularly pronounced.

This document additionally defines a method for servers to declare their named group preferences in DNS, using SVCB or HTTPS resource records [I-D.ietf-dnsop-svcb-https]. This allows the client to predict key shares more accurately.

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. Predictions vs Preferences in TLS

3.1. Downgrades

Some existing TLS 1.3 servers implement the following named group selection algorithm:

1. Select a common named group in key_share. If found, select it and send ServerHello.

2. Otherwise, select a common named group in supported_groups. If found, select it and send HelloRetryRequest.

3. Otherwise, terminate the handshake with a handshake_failure alert.

While this algorithm avoids HelloRetryRequest whenever possible, it implicitly assumes the client prefers the values sent in key_share, and that the server has no preferences between any groups. If these assumptions do not hold, the server's selection may be downgraded.

The following sections describe example downgrade scenarios with this algorithm. postquantum1 and postquantum2 refer to future post-quantum named groups, which both client and server prefer over x25519.

3.1.1. Uncommon Groups

Consider a client which implements, in preference order, postquantum2, postquantum1, and x25519. Sending keys for both postquantum2 and postquantum1 is expensive, so the client only predicts one of them. postquantum2 is preferred (e.g. more efficient or more commonly deployed), and older x25519-only servers still exist, so the client predicts postquantum2, x25519 in key_share.

If the server predates postquantum2 and only implements postquantum1 and x25519, it will select x25519, although postquantum1 is available in supported_groups.

3.1.2. Predictions

The client may predict key shares based on prior knowledge about the server, such as a DNS hint (see Section 4). For example, during a transition from postquantum1 to postquantum2, both options will be available in the ecosystem. The client may use a DNS hint to avoid needing HelloRetryRequest with both existing and upgraded servers.

If the client's prior knowledge is outdated or under attacker influence, this can lead to a downgrade. Suppose the server implements postquantum1 and x25519, but the client believed it only implemented x25519. The client may then predict x25519 in key_share, leading the server to select x25519 over the preferred postquantum1.

3.1.3. Compatibility

Software bugs in existing TLS servers may prevent them from processing larger ClientHellos. During an early rollout of post-quantum KEMs, a client may prefer postquantum1, but sometimes only predict x25519 to reduce compatibility risk, expecting that newer servers can still select it with HelloRetryRequest.

However, a server implementing the above algorithm would instead select x25519 over the preferred postquantum1.

3.2. Server Behavior

TLS 1.3 servers implementing this document MUST NOT assume the client's key_share extension reflects client preferences. Instead, servers SHOULD select the best common named group based on supported_groups, without reference to key_share. The server then looks for the selected named group in key_share to decide whether to send HelloRetryRequest or ServerHello.

If choosing between two named groups which the server equally prefers, and for which the server is willing to ignore the client's supported_groups preference order, the server MAY use presence in the client's key_share extension to select one which will avoid HelloRetryRequest. However, attackers may then influence which of the two is chosen.

Note the algorithm in Section 3.1 is permitted if the above applies to all of a server's supported groups. However, this is unlikely to apply if the server implements a combination of post-quantum and legacy named groups, or if the server software's configuration specifies a preference order.

3.3. Prediction-Safe Named Groups

Although Section 3.2 defines new rules for TLS 1.3 servers, TLS 1.3 has already been deployed. Clients that assume a server implements the new rules may introduce a downgrade attack on a pre-existing server. To avoid this, this document uses named group codepoints to distinguish the old and new behavior.

A named group is considered prediction-safe if the value in the "Prediction-Safe" column of the registry (see Section 6) is "Y". Otherwise, it is considered prediction-unsafe. Any TLS server which implements a prediction-safe named group MUST follow the guidelines in Section 3.2. To be a prediction-safe named group, the defining specification MUST cite this document and include such a requirement. For example:

> TLS servers which support this named group MUST select parameters as described in Section 3.2 of [this-RFC].

3.4. Client Behavior

When sending the initial ClientHello, clients SHOULD ensure the prediction-unsafe groups in the key_share extension are consistent with its preferences. This is determined by the following procedure:

1. Let key_share_pred_unsafe be the list of prediction-unsafe named groups in the key_share extension

2. Let supported_groups_pred_unsafe be the list of prediction-unsafe named groups in the supported_groups extension

3. The key_share extension is consistent if and only if key_share_pred_unsafe is a prefix of supported_groups_pred_unsafe

This procedure ignores all prediction-safe named groups. Clients MAY freely vary whether a prediction-safe named group is included, including using untrusted signals.

For example, suppose safe1 and safe2 are prediction-safe, while unsafe1 and unsafe2 are prediction-unsafe. If the client's supported_groups extension contains, in order, safe1, unsafe1, safe2, unsafe2, the following key_share predictions would meet this criteria:

*No key shares

*safe1, safe2

*safe2

*unsafe1, unsafe2

*unsafe1, safe2

The following would not:

*unsafe2

*safe1, unsafe2

If the client has trusted, prior knowledge that the server implements a selection algorithm consistent with Section 3.2, it MAY disregard the above and freely vary both prediction-safe and prediction-unsafe groups.

4. DNS Service Parameter

This section defines the tls-supported-groups SvcParamKey [I-D.ietf-dnsop-svcb-https], which specifies the endpoint's TLS supported group preferences, as a sequence of TLS NamedGroup codepoints in order of decreasing preference.

4.1. Format

The presentation value of the SvcParamValue is a comma-separated list (Appendix A.1 of [I-D.ietf-dnsop-svcb-https]) of decimal integers between 0 and 65536 in ASCII. Any other value is a syntax error. To enable simpler parsing, this SvcParam MUST NOT contain escape sequences.

The wire format of the SvcParamValue is a sequence of 2-octet numeric values in network byte order. An empty list of values is invalid.

4.2. Configuring Services

Services SHOULD include supported TLS named groups, in order of decreasing preference in the tls-supported-groups parameter of their HTTPS or SVCB endpoints. As TLS preferences are updated, services SHOULD update the DNS record to match. Services MAY include GREASE values [RFC8701] in this list.

A service MUST NOT configure this service parameter if any of the corresponding TLS servers do not implement the TLS server guidance in Section 3.2.

4.3. Client Behavior

When connecting to a service endpoint whose HTTPS or SVCB record contains the tls-supported-groups parameter, the client evaluates the server preferences against its own and predicts named groups to send in the key_share extension. In evaluating the server preferences, the client MUST ignore any codepoints that it does not support or recognize.

If one of the following hold, the client MAY use the result to predict key shares in the initial ClientHello:

- The HTTPS or SVCB record was authenticated by the origin server, or
- The resulting prediction is consistent with client preferences, as described in Section 3.4

Otherwise, the client SHOULD ignore the parameter and compute key_share via its usual logic. Note a secure connection to the DNS resolver, such as DNS over TLS [RFC7858] or DNS over HTTPS [RFC8484], is not sufficient to authenticate the record by the origin server.

4.4. Misprediction

Although this service parameter is intended to reduce key share mispredictions, mispredictions may still occur. For example, HelloRetryRequest may be required in the following cases:

- The client has fetched a stale HTTPS or SVCB record that no longer reflects the server preferences
- The server is in the process of deploying a change to named group preferences, and different server instances temporary evaluate different preferences
- The client was unable to fetch the HTTPS or SVCB record
- The client and server implement incompatible selection algorithms, such that client's evaluation of the service parameter did not match the server's final selection
- The server preferred a prediction-unsafe named group for this client, so the client was unable to safely act on the service parameter

Clients and servers MUST correctly handle mispredictions by responding to or sending HelloRetryRequest, respectively.

5. Security Considerations

This document updates TLS server behavior and introduces a notion of prediction-safe named groups to avoid the downgrades in Section 3.1, for both new and existing TLS 1.3 implementations:

- New servers that implement Section 3.2 have selection algorithms that permit arbitrary client key_share prediction criteria, even under attacker influence.
- Existing servers are assumed to only implement prediction-unsafe named groups. Section 3.4 ensures that, for all named groups they implement, the client's predicted list will be compatible with possible server assumptions.

If a TLS server implements a prediction-safe named group but does not follow the guidelines in Section 3.2, downgrades are possible.

Thus Section 3.3 requires all prediction-safe named groups to include text referencing this document.

6. IANA Considerations

6.1. Updates to the TLS Supported Groups Registry

This document updates the TLS Supported Groups registry [RFC8422] to add a "Prediction-Safe" column immediately following the "Recommended" column. The "Prediction-Safe" column is set to a value of "N" for all existing allocations except for X25519Kyber768Draft00 and SecP256r1Kyber768Draft00. Those two values should be set to "Y".

[[TODO: As of writing, neither of the Kyber768 hybrids above include the necessary text. But, as Kyber is a large post-quantum KEM, it's desirable for them to be prediction-safe. If this document is adopted, the respective Kyber drafts can be updated to incorporate the necessary sentence.]]

This document additional adds the following note to the registry:

> Note: Section 3.3 of [this-RFC] defines the procedure for a group to be considered prediction-safe and thus set the corresponding column to a value of "Y". All new allocations to this registry are expected to be prediction-safe, unless some interoperability consideration prevents it. For example, if the new allocation is documenting a pre-existing deployment with the older server behavior, it may be allocated with a value of "N".

6.2. Updates to the Service Parameter Keys Registry

This document updates the Service Parameter Keys registry [I-D.ietf-dnsop-svcb-https] with the following entry:

Number	Name	Meaning	Format Reference	Change Controller
TBD	tls-supported-groups	Supported groups in TLS	(this document) Section 4.1	IETF

Table 1

7. References

7.1. Normative References

[I-D.ietf-dnsop-svcb-https] Schwartz, B. M., 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-12, 11 March

2023, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-svcb-https-12>.

[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>.

[RFC8422] Nir, Y., Josefsson, S., and M. Pegourie-Gonnard, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier", RFC 8422, DOI 10.17487/RFC8422, August 2018, <https://www.rfc-editor.org/info/rfc8422>.

[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>.

[RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <https://www.rfc-editor.org/info/rfc8701>.

7.2. Informative References

[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/info/rfc7858>.

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

Acknowledgments

The author would like to thank David Adrian, Bob Beck, Sophie Schmieg, and Bas Westerbaan for discussions and review on initial iterations of this document.

Author's Address

David Benjamin
Google LLC

Email: davidben@google.com