Privacy Pass
Internet-Draft
Intended status: Informational
Expires: 4 September 2025

T. Pauly
Apple
S. Hendrickson
Google

3 March 2025

Including Privacy Pass Tokens in TLS Handshakes
draft-pauly-privacypass-for-tls-00

Abstract

This document defines a mechanism for TLS servers to request, and TLS clients to provide, Privacy Pass tokens as part of the Encrypted Client Hello in the TLS handshake. This creates a way to add support for anonymous attestation and rate-limiting to servers that are enforcing denial-of-service protections as part of processing TLS handshakes.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-pauly-privacypass-for-tls/.

Discussion of this document takes place on the Privacy Pass mailing list (mailto:privacy-pass@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/privacy-pass/. Subscribe at https://www.ietf.org/mailman/listinfo/privacy-pass/.

Source for this draft and an issue tracker can be found at https://github.com/tfpauly/draft-privacypass-for-tls.

Status of This Memo

This Internet-Draft will expire on 4 September 2025.

Table of Contents

1. Introduction

Privacy Pass Tokens [PPARCH] are cryptographic authentication messages that can be used to verify properties of a network entity, such as proving that a client passed some attestation check, without being linkable to other tokens or revealing identities.

[PPAUTH] defines how Privacy Pass Tokens can be requested by HTTP servers (via an authentication challenge) and provided by HTTP clients. This is useful for providing privacy-preserving authentication or attestation in HTTP workflows. However, Privacy Pass Tokens can also be used in other contexts and protocols. For example, [I-D.sawant-eap-ppt] defines how to include tokens in EAP (Extensible Authentication Protocol) exchanges.

Some server deployments enforce rate-limiting on TLS [TLS13] handshakes to prevent denial-of-service (DoS) attacks, particularly by rate limiting the number of connections allowed from individual client IP addresses or IP address subnets. This is common in scenarios where the cost of handling a terminated TLS connection is significantly higher than handling the initial handshake, like in L7 loadbalancers with heavy-weight protocol conversions after termination.

This enforcement can particularly impact cases where many clients are using a particular IP subnet due to using a privacy-preserving proxy (some examples are described in [PRIVACYPARTITIONING]). For such cases, even if clients are able to provide Privacy Pass Tokens or similar proofs at the HTTP layer, their connections might be denied or rate-limiting during TLS session establishment.

In order to signal that clients meet certain criteria (rate-limiting, etc), without disclosing individual client identities or pseudonyms, this document defines a way to include Privacy Pass Tokens within the TLS handshake. Specifically, these tokens are sent within the TLS Encrypted Client Hello [ECH]. This prevents network observers from being able to directly observe the tokens, while still allowing the TLS server to observe the token early in the handshake.

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. Overview

Clients can include Privacy Pass tokens as part of the TLS Client Hello via the privacy_pass_token extension. This is described in Section 5. Clients MAY be configured to always present tokens when performing a TLS handshake with a particular server. However, in general, clients SHOULD NOT automatically include Privacy Pass tokens; without an explicit challenge, clients won’t know the relevant token type or issuer to use.

Servers can request tokens by adding the privacy_pass_challenge extension to a TLS Hello Retry Request. This is described in Section 4. Servers that want to receive Privacy Pass tokens as a way to enforce DoS protection SHOULD send challenges to clients when these clients would otherwise be blocked or rate-limited in some fashion.

4. Requesting Privacy Pass Tokens

In order to request that a client sends a Privacy Pass token, a server can send a Hello Retry Request ([TLS13], Section 4.1.4) that includes a privacy_pass_challenge extension.

The privacy_pass_challenge extension has the following format:

```
struct {
    opaque challenge<1..2^16-1>;
    opaque token_key<0..2^16-1>;
} PrivacyPassChallenge;
```

The fields are defined as follows:

* challenge contains a TokenChallenge structure, as defined in [PPAUTH], Section 2.1.1.

* token_key contains a public key for use with the issuance protocol, where applicable. This is equivalent to the token-key parameter used in HTTP authentication challenges discussed in [PPAUTH], Section 2.1.1. The token_key may be empty (have a zero length), in which case clients are expected to fetch the token key for a particular issuer name in another way.

If a client does not include the token in the Client Hello (or subsequent Client Hello after being challenged), the server MAY reject the request or apply rate-limiting.

Clients SHOULD apply some form of consistency check on the token challenge to avoid (malicious) anonymity set partitioning by the server; see Section 6.2 of [PPARCH] for more details.

Servers sending challenges can use a non-empty redemption_context in order to bind the token challenge to a particular context (such as the client IP address, or a time window) to aid in token replay prevention. Servers MAY combine sending privacy_pass_challenge extensions with a cookie extension ([TLS13], Section 4.2.2). For example, servers that cannot statefully persist the token challenge presented to the client in the privacy_pass_challenge extension can use the cookie extension to encode this challenge.

5. Presenting Privacy Pass Tokens in Encrypted Client Hello

Clients can include Privacy Pass tokens in TLS handshakes using the privacy_pass_token extension. This extension MUST be sent in the Inner Client Hello, using [ECH]. If ECH is not supported, clients SHOULD NOT use Privacy Pass tokens in TLS in order to avoid adding more tracking entropy visible on the wire, and making it easier to trivially replay tokens to a server.

The privacy_pass_token extension has the following format:

```
struct {
    opaque token<1..2^16-1>;
} PrivacyPassToken;
```

The token field uses the Token structure defined in [PPAUTH], Section 2.1.1.

Tokens are generally presented after receiving a challenge, but a client MAY include a token without having received a challenge if it has other out-of-band configuration to do so.

5.1. Handling Inability to Present Tokens

Servers need to be able to detect when clients are unable to present a token after receiving a challenge. A client might be unable to present tokens because it has reached a token rate limit, because it does not have a way to generate tokens for the required token issuer, or simply because it does not support this specification.

The RECOMMENDED approach to handle such cases is for the server to include a cookie extension ([TLS13], Section 4.2.2) along with the challenge, and for clients to retry the handshake including the cookie extension, but not including the privacy_pass_token extension. Servers can then assume that the client received the challenge and was unable to generate a valid token. The policy for what servers do in such cases will be specific to the overall use case, and beyond the scope of this document.

6. Applicable Token Types

This document is defined such that any Privacy Pass token type would be possible to use in the TLS handshake. However, different token types will have different properties for latency, replay protection, and privacy.

Ideally, deployments can use token types that allow for unique redemption contexts (to prevent replay attacks) that also do not require communicating with a token attester or issuer for each token creation (thus improving latency, and not creating new activity that can be used to fingerprint clients). Some proposed token types like [ARC] and [BBS] have these properties.

7. Security Considerations

Servers redeeming Privacy Pass tokens in TLS handshakes need to take care to avoid replay attacks. Using a fresh redemption context in the challenge ensures that tokens are equally fresh and unique.

As discussed in Section 6, token issuance types that don’t require clients talking to an issuance server with a new network request for every token generation will have better properties for privacy, since the client won’t make a new request after each TLS handshake challenge.

8. IANA Considerations

8.1. Update of the TLS ExtensionType Registry

IANA is requested to create the following entries in the existing registry for ExtensionType (defined in [TLS13]):

1. privacy_pass_challenge(0xfd00), with the "TLS 1.3" column values set to "HRR", "DTLS-Only" column set to "N", "Recommended" column set to "Yes".

2. privacy_pass_token(TBD), with "TLS 1.3" column values set to "CH", "DTLS-Only" column set to "N", and "Recommended" column set to "Yes", and the "Comment" column set to "Only appears in inner CH."

9. References

9.1. Normative References

[ECH] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft, draft-ietf-tls-esni-23, 19 February 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-23>.

[PPARCH] Davidson, A., Iyengar, J., and C. A. Wood, "The Privacy Pass Architecture", RFC 9576, DOI 10.17487/RFC9576, June 2024, <https://www.rfc-editor.org/rfc/rfc9576>.

[PPAUTH] Davidson, A., Iyengar, J., and C. A. Wood, "The Privacy Pass Architecture", RFC 9576, DOI 10.17487/RFC9576, June 2024, <https://www.rfc-editor.org/rfc/rfc9576>.

[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>.

[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>.

[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>.

9.2. Informative References

[ARC] Yun, C. and C. A. Wood, "Privacy Pass Issuance Protocol for Anonymous Rate-Limited Credentials", Work in Progress, Internet-Draft, draft-yun-privacypass-arc-00, 5 February 2025, <https://datatracker.ietf.org/doc/html/draft-yun-privacypass-arc-00>.

[BBS] Ladd, W., "BBS for PrivacyPass", Work in Progress, Internet-Draft, draft-ladd-privacypass-bbs-01, 26 February 2024, <https://datatracker.ietf.org/doc/html/draft-ladd-privacypass-bbs-01>.

[I-D.sawant-eap-ppt] Sawant, P. and B. Brinckman, "Extensible Authentication Protocol (EAP) Using Privacy Pass Token", Work in Progress, Internet-Draft, draft-sawant-eap-ppt-01, 20 October 2024, <https://datatracker.ietf.org/doc/html/draft-sawant-eap-ppt-01>.

[PRIVACYPARTITIONING] K端hlewind, M., Pauly, T., and C. A. Wood, "Partitioning as an Architecture for Privacy", RFC 9614, DOI 10.17487/RFC9614, July 2024, <https://www.rfc-editor.org/rfc/rfc9614>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Tommy Pauly
Apple
Email: tpauly@apple.com

Scott Hendrickson
Google
Email: scott@shendrickson.com