Privacy Pass
Internet-Draft
Intended status: Informational
Expires: 23 April 2026

T. Meunier
Cloudflare Inc.

20 October 2025

# Privacy Pass Reverse Flow
## draft-meunier-privacypass-reverse-flow-02

## Abstract

This document specifies an instantiation of Privacy Pass Architecture [RFC9576] that allows for a "reverse" flow from the Origin to the Client. It describes a method for an Origin to issue new tokens in response to a request in which a token is redeemed.

## About This Document

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

The latest revision of this draft can be found at https://thibmeu.github.io/draft-meunier-privacypass-reverse-flow-informational/draft-meunier-privacypass-reverse-flow.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-meunier-privacypass-reverse-flow/.

Discussion of this document takes place on the Privacy Pass Working Group 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/thibmeu/draft-meunier-privacypass-reverse-flow-informational.

## Status of This Memo

This Internet-Draft will expire on 23 April 2026.

## Copyright Notice



## Table of Contents

## 1. Introduction

This document specifies an instantiation of Privacy Pass Architecture [RFC9576] that allows for a reverse flow from the Origin to the Client.

In other words, it specifies a way for an Origin to act as an Attester + Issuer.

## 2. Motivation

With Privacy Pass issuance as described in [RFC9576], once a token is sent by a Client, it is considered spent and cannot be reused in order to guarantee unlinkability. If a token were to be spent twice, the two requests would be linkable by the Origin.

However, requiring that all tokens are spent only once means that Clients might need to request more tokens for new requests, even if the request that included the original token doesn’t need to "spend" that token from the Origin’s perspective (due to the request ending up being insignificant to the Origin). This draft provides a mechanism for an Origin to provide tokens, allowing reuse without reaching out to the initial Attester and Issuer.

### 2.1. Refunding tokens

Certain Origin use Privacy Pass tokens to rate-limit requests they receive over a certain time window because of resource constraints. If a Client sends a request that can be served without utilising that resource, the Origin would like to authorise them to do a second request. This is the case for request requiring compute and the compute is low, or when the request leads to a redirection instead of content generation for instance.

With a reverse flow, a Client that has already been authorised by an Origin can maintain that authorization, without losing the unlinkability property provided by Privacy Pass.

### 2.2. Bootstraping issuer

An Origin wants to grant 30 access for Clients that solved a CAPTCHA. To do so, it consumes a type 0x0002 public veriable token from an initial issuer that guarantees a CAPTCHA has been solved, and use it to issue 30 type 0x0001 private tokens. Without a reverse flow, the Origin would have to require 30 0x0002 issuer tokens, which have lower performance and a higher number of requests going to the issuer.

### 2.3. Attester feedback loop

In [RFC9576], a Client gets a token from an Issuer and redeems it at an Origin. However, if the Client's request is deemed unwanted by the Origin at redemption time, there is no mechanism that prevents the Client from going back to the initial Issuer to get a new token and be authorized again.

With a reverse flow, the initial Issuer may require Clients to present an Origin-issued token before providing them with a second token. This allows for a feedback loop between the Origin and the initial Issuer, without breaking Client unlinkability.

## 3. Terminology

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.

We reuse terminology from [RFC9576].

The following terms are used throughout this document:

*Flow:* Direction from PrivateToken issuance to its redemption. The entity starting the flow acts as an Issuer, while the end of the flow acts as an Origin. The Client is always included, as it finalises the TokenResponse, and coordinate interactions.

*Initial Flow:* Issuer -> Attester -> Client -> Origin. This flow produces a PrivateToken that is used by the Origin to kickstart a Reverse Flow.

*Reverse Flow:* Issuer <- Attester <- Client <- Origin. This flow allows Origin to issues PrivateToken. In the reverse flow, the Origin operates one or more Issuer, and the Client MAY provide these tokens either to the Initial Attester/Issuer, or use them against the Origin

*Initial Attester/Issuer:* Attester/Issuer part of the Initial Flow

*Origin Issuer:* Issuer operated by the Origin

*Origin PrivateToken:* PrivateToken issued by the Origin

*Reverse Origin:* An entity that consumes the Origin PrivateToken. It can be the Origin, or the Initial Attester/Issuer

## 4. Architecture overview

Along with sending their PrivateToken for authentication (as specified in [RFC9576]), Client sends TokenRequest

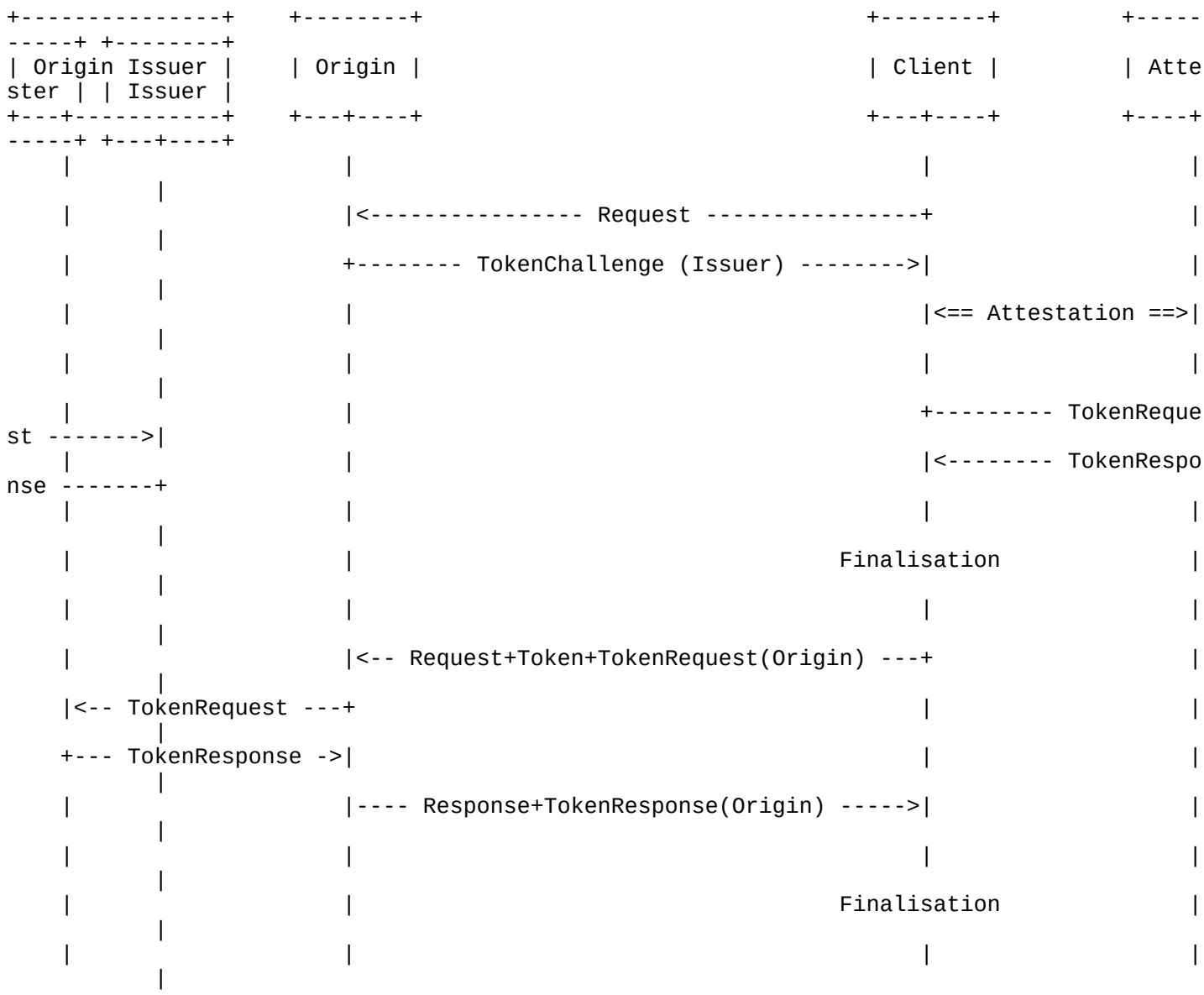


Figure 1: Architec ture of Privacy Pass with a Reverse Flow

The initial flow matches the one defined by [RFC9576]. A Client gets challenged when accessing a resource on an Origin. The Client goes to the Attester to get issue a Token.

Through configuration mechanism not defined in this document, the Client is aware the Origin acts as a Reverse Flow issuer.

This is an extension of [RFC9576]. The redemption flow of a Privacy Pass token is defined in Section 3.6.4 of [RFC9576]. Reverse flow extends this so that redemption flow is interleaved with the issuance flow described in Section 3.6.3 of [RFC9576]. This is denoted in the diagram above by the Client sending Request+Token+TokenRequest(Origin Issuer). The Origin runs the issuance protocol, and returns Response+TokenResponse(Origin Issuer).

Such flow can be performed through various means. This document introduces one to serve as example and first basis.

## 5. TokenRequest, TokenResponse, and Finalisation

In Figure 1, the Client sends an TokenRequest and receives an TokenResponse. These are meant to abstract request from different protocol to the Issuer.

As specified in Section 3.5 of [RFC9576],

> The structure and semantics of the TokenRequest and TokenResponse messages depend on the issuance protocol and token type being used.

The introducion of Privacy Pass issuance protocol based on Anonymous credential, such as [PRIVACYPASS-ARC] or [PRIVACYPASS-ACT], modifies TokenRequest to use CredentialRequest instead.

Upon receiving an TokenResponse, the Client has to finalise the Token so it can be presented to an origin. This may be a Finalisation for type 0x0002 as defined in Section 7 of [RFC9578], a presentation for Section 7 of [PRIVACYPASS-ARC], or even a refund for [PRIVACYPASS-ACT].

## 6. Reverse flow with an HTTP header

This section defines a Reverse Flow, as presented in Section 4, leveraging a new HTTP headers.

TokenRequest(Origin) and TokenResponse(Origin) happen through a new HTTP Header PrivacyPass-Reverse. PrivacyPass-Reverse is a base64url ([RFC4648]) encoded GenericBatchTokenRequest as defined in Section 6 of [BATCHED-TOKENS].

> The use of generic batch tokens as defined in Section 6 of [BATCHED-TOKENS] is because this already provides encoding for request and response, error wrapping, and a concise format. One could use binary http or a new format

### 6.1. Client behaviour

Along with sending PrivateToken from the Initial Issuer to the Origin, the Client sends a TokenRequest as defined in [RFC9578], [BATCHED-TOKENS], or [PRIVACYPASS-ARC]. In all these definitions, TokenRequest MUST prepended by a uint16_t representing the token type. The Client SHOULD consider Privacy Pass Reverse Flow like the initial flow. The Client is responsible to coordinate between the different entities. Specifically, if the Reverse Origin is the Initial Attester/Issuer, the Client SHOULD account for possible privacy leakage.

### 6.2. Origin/Issuer/Attester deployment

In this model, the Origin, Attester, and Issuer are all operated by the same entity, as shown in Figure 2. The Reverse Flow is the same as the Initial Flow, except for the request/response encapsulation. The Origin is the Reverse Origin.

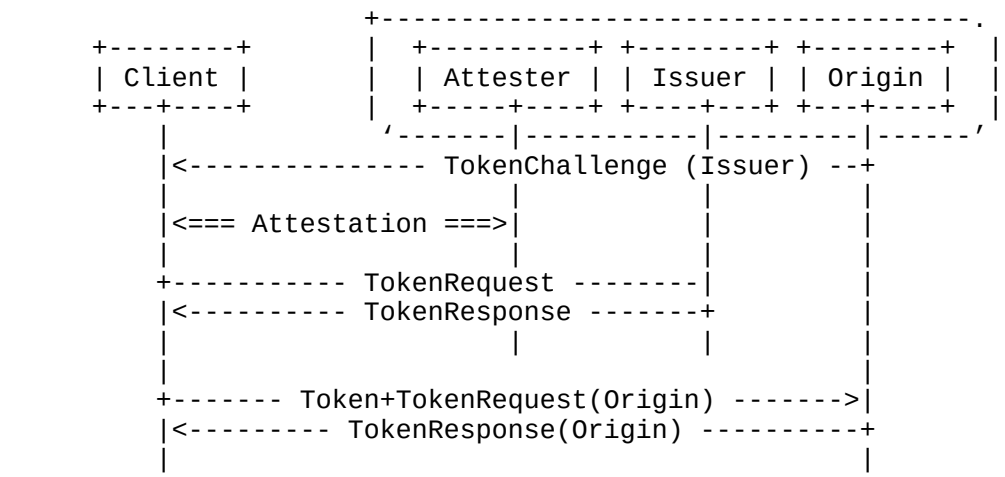


Figure 2: Shared Deployment Model

Similar to the original Shared Deployment Model, the Attester, Issuer, and Origin share the attestation, issuance, and redemption contexts. Even if this context changes between the Initial and Reverse Flow, attestation mechanism that can uniquely identify a Client are not appropriate as they could lead to unlinkability violations.

This model allows for private verifiability. Even though no optimised scheme is available at the time of writting, the author recommends to follow advances of anonymous credential within the Privacy Pass group.

Specifically

1. [PRIVACYPASS-ARC]

2. [PRIVACYPASS-BBS]

3. [PRIVACYPASS-ACT]

These scheme allow for optimisation of Token finalisation by the Client.

### 6.3. Split Origin-Attester deployment

In this model, the Attester and Issuer are operated by the same entity that is separate from the Origin. The Origin trusts the joint Attester and Issuer to perform attestation and issue Tokens. Origin Tokens can then be sent by Client on new requests, as long as the Reverse Origin trusts the Origin to perform attestation and issue Tokens.

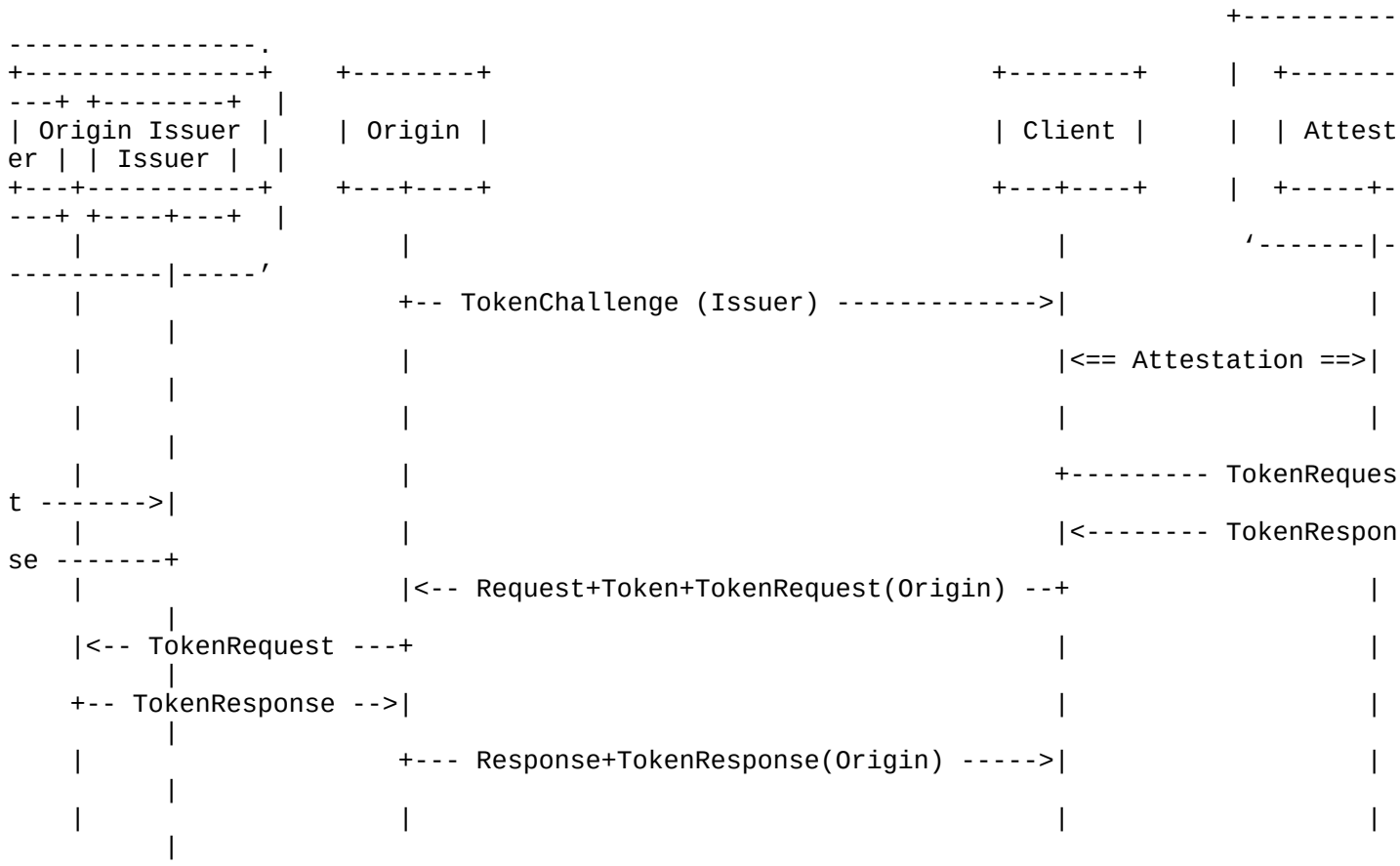


Figure 3: Joint Attester and Issuer Deployment Model

The Origin Issuer MUST NOT issue privately verifiable tokens, as this would lead to secret material being shared between the Origin and the Reverse Origin.

A particular deployment model is when the Reverse Origin is the Attester/Issuer. This model is described in Figure 4

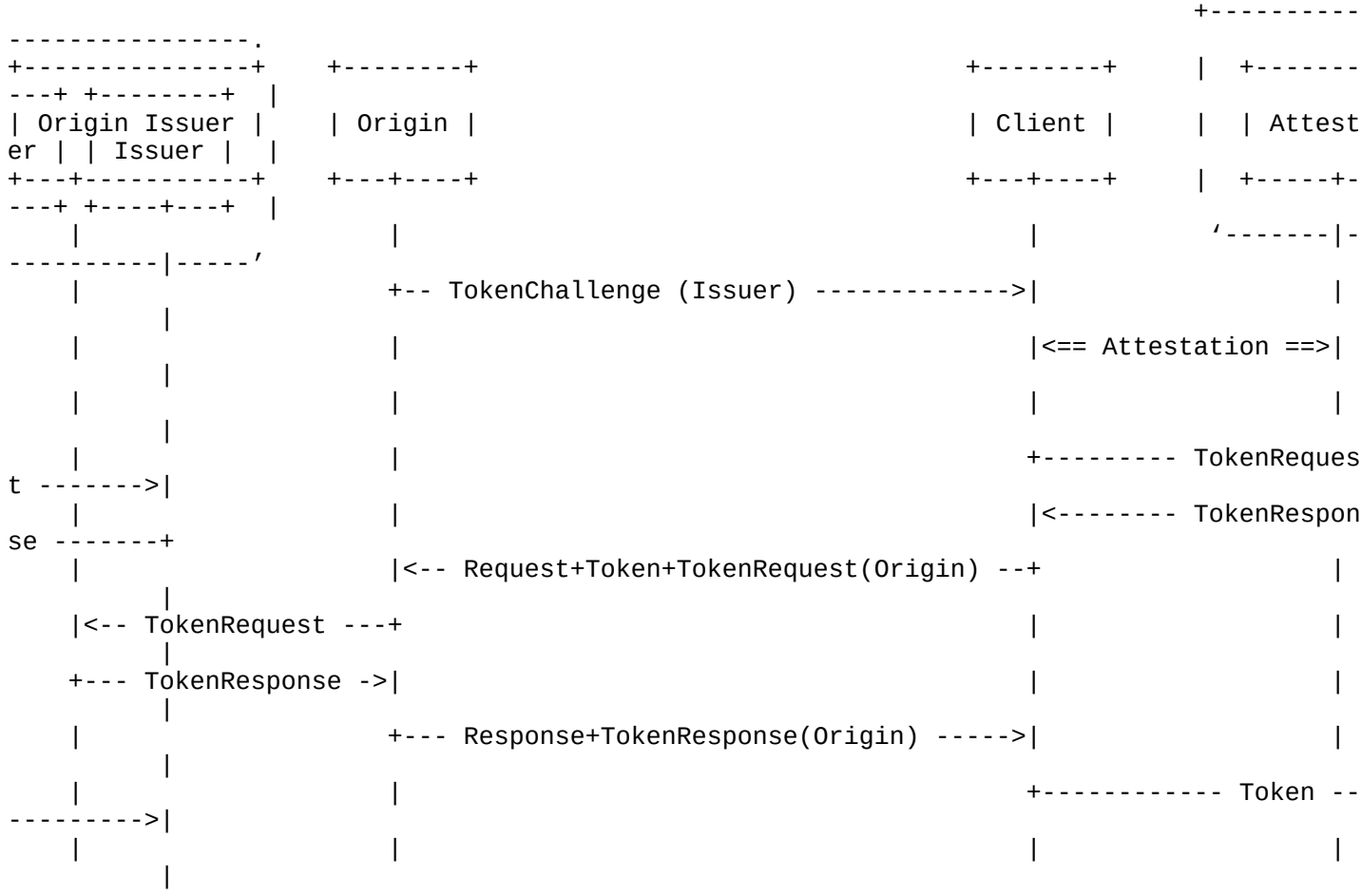


Figure 4: Joint Attester and Issuer Deployment Model with reverse

This deployment SHOULD not allow the Reverse Origin to infer the request made to the Origin, as it would break unlinkability.

## 7. Privacy Considerations

Privacy Pass [RFC9576] states

> In general, limiting the amount of metadata permitted helps limit the extent to which metadata can uniquely identify individual Clients. Failure to bound the number of possible metadata values can therefore lead to a reduction in Client privacy. Most token types do not admit any metadata, so this bound is implicitly enforced.

In Privacy Pass with a reverse flow, Clients are provided with new PrivateTokens depending on their request. They can spend these tokens to continue making further requests.

While the token are still unlinkable, the token_key_id associated to them represent metadata. It leaks some information about the Client. The following subsections discuss the issues that influence the anonymity set, and possible mitigations/safeguards to protect against this underlying problem.

### 7.1. Issuer face values

When setting up a reverse flow deployment, an Origin MAY operate

multiple Issuers, and assign them some metadata to them. The amount of possible metadata grows as 2^(origin_issuers).

We RECOMMEND that:

1. Origin defines their anonimity sets, and deploy no more than log2(#anonimity_sets). This bounds the possible anonimity sets by design.

2. Client to only send 1 PrivateToken per request. This is inline with RFC9577 and RFC (Web Authentication) which only allows one challenge response to be provided as part of Authorization HTTP header.

3. Issuers metadata to be publicly disclosed via an origin endpoint, and externally monitored

### 7.2. Token for specific Clients

In Privacy Pass with a reverse flow, an Origin MAY operate multiple Issuers, with arbitrary metadata associated to them. A malicious Origin MAY uses this opportunity to associate certain token values to a specific set of Clients.

Let’s consider the following deployment: the Origin operates two issuers A and B. The Client sends Token_A, and (TokenRequest_A, TokenRequest_B). Issuer B is associated to croissant aficionados.

If a Client requests croissant, or sends Token_B, the origin provides TokenResponse_B. If not, it provides TokenResponse_A.

Over time, this means the Origin is able to track croissants aficionados.

To mitigate this, we RECOMMEND:

1. The initial PrivateToken to be provided by an Issuer not in control of the Origin. The joint Origin/Attester/Issuer model SHOULD NOT be used.

2. Clients to reset their state regularly with the initial Issuer.

### 7.3. Sending more than one token

While that’s not part of Privacy Pass with a reverse flow, some deployment might consider allowing Clients to send multiple PrivateToken, similar to how normal Privacy Pass deployment allow two distinct PrivateToken to be sent.

In Privacy Pass with a reverse flow deployment, there are as many bits as Issuers; each token is one bit. We RECOMMEND to have a maximum of 6 Origin operated Issuers, bounding Client information to 2^6 = 64. Accounting for the initial Issuer, this means a total of log2(64)+1=7 issuers.

Origin should have sufficient traffic to not single-out particular Client based on timings of requests.

### 7.4. Swap endpoint and its privacy implication

With multiple Issuers, a Client MAY end up with a bunch of tokens, for various Issuers. Origins MAY propose a swap endpoint at which a Client can exchange one or more Origin tokens against one or more new Origin tokens.

The Origin SHOULD ensure this endpoint receives enough traffic to not reduce the anonymity sets.

## 8. IANA Considerations

This document has no IANA actions.

## 9. References

### 9.1. Normative References

[BATCHED-TOKENS] Robert, R., Wood, C. A., and T. Meunier, "Batched Token Issuance Protocol", Work in Progress, Internet-Draft, draft-ietf-privacypass-batched-tokens-05, 3 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-privacypass-batched-tokens-05>.

[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>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[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>.

[RFC9576] 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>.

[RFC9578] Celi, S., Davidson, A., Valdez, S., and C. A. Wood, "Privacy Pass Issuance Protocols", RFC 9578, DOI 10.17487/RFC9578, June 2024, <https://www.rfc-editor.org/rfc/rfc9578>.

### 9.2. Informative References

[PRIVACYPASS-ACT]
"Anonymous Credit Tokens", n.d., <https://samuelschlesinger.github.io/draft-act/draft-schlesinger-privacypass-act.html>.

[PRIVACYPASS-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-01, 6 August 2025, <https://datatracker.ietf.org/doc/html/draft-yun-privacypass-arc-01>.

[PRIVACYPASS-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>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

## Acknowledgments

The author would like to thank Tommy Pauly, Chris Wood, Raphael Robert, and Armando Faz Hernandez for helpful discussion on Privacy Pass architecture and its considerations.

## Changelog

v01

* Editorial pass on the introduction
* Add a motivation section: refunding tokens, bootstraping issuer, attester feeback loop

* Split protocol overview via HTTP headers in its own section

* Add consideration about anonymous credentials in joint origin/ issuer deployment

v00

* Initial draft

* Possibility of a new HTTP request for inlining request

* Privacy considerations about additional metadata

## Author’s Address

Thibault Meunier
Cloudflare Inc.
Email: ot-ietf@thibault.uk